研究成果報告書

実用的符号及び通信路に対する並列反復事後確率計算アルゴリズムの応用と解析

（課題番号：15560338）

平成15年度～平成17年度科学研究費補助金（基盤研究（C））研究成果報告書

平成18年5月

研究代表者　松嶋　敏泰

（早稲田大学 理工学術院　教授）

はしがき

研究組織

研究代表者：松嶋　敏泰　（早稲田大学理工学術院教授）

研究分担者：平澤　茂一　（早稲田大学理工学術院教授）

交付決定額 (配分額)

	直接経費	間接経費	合計
平成15年度	1,600	0	1,600
平成16年度	1,300	0	1,300
平成17年度	800	0	800
総計	3,700	0	3,700

（金額単位：千円）

研究発表

(1) 学会誌等

浮田善文，松嶋敏泰，平澤茂一，”直交計画を用いたブール関数の学習に関する一考察，”電子情報通信学会論文誌（A） Vol.J86-A，No.4, 2003.

Hideki Yagi, Manabu Kobayashi, Toshiyasu Matsushima and Shigeichi Hirasawa, "An improved method of reliability-based maximum likelihood decoding algorithm using an order relation among binary vectors, "IEICE Trans on Fundamentals(A), Vol.E87-A, No.10, 2004

Hideki Yagi, Manabu Kobayashi and Shigeichi Hirasawa, "A heuristic search method with the reduced list of test error patterns for maximum likelihood decoding, "IEICE Trans on Fundamentals(A), Vol.E88-A, No.10, 2005.

松嶋敏泰,”ターボ符号・LDPC符号とその復号法の概要，”電子情報通信学会誌Vol.88, No.4, 2005.

Naoto Kobayashi, Toshiyasu Matsushima, Shigeichi Hirasawa, "Transformation of a Parity-Check Matrix for a Message-Passing Algorithm over the BEC", IEICE Trans

on Fundamentals(A), Vol.E89-A, No.5, 2006

Tomohiko Saito, Toshiyasu Matsushima, Shigeichi Hirasawa, "A Note on Construction of Orthogonal Arrays with Unequal Strength from Error-Correcting Codes", IEICE Trans on Fundamentals(A), Vol.E89-A, No.5, 2006

細渕智史，斉藤友彦，松嶋敏泰，"ストリーム暗号への攻撃法の改良に関する一考察－多次元の相関を利用した攻撃－，"電子情報通信学会論文誌（A） Vol.J89-A, No.2,2006.

(2) 口頭発表

Toshiyasu Matsushima, Tomoko K. Matsushima and Shigeichi Hirasawa, "A Parallel Iterative Decoding Algorithm for Zero-Tail and Tail-Biting Convolutional Codes" IEEE International Symposium on Information Theory, 2003.

Toshiyasu Matsushima, Tomoko K.Matsushima and Shigeichi Hirasawa, "Parallel Propagation Algorithms for Tailbiting Convolutional Codes, "第26回情報理論とその応用シンポジウム, 2003.

Hideki Yagi, Toshiyasu Matsushima and Shigeichi Hirasawa, "A Method for Reducing Space Complexity of Reliability based Heuristic Search Maximum Likelihood Decoding Algorithms, "第26回情報理論とその応用シンポジウム, 2003.

Hideki Yagi, Toshiyasu Matsushima and Shigeichi Hirasawa, "A heuristic search algorithm with the reduced list of test error patterns for maximum likelihood decoding, "第27回情報理論とその応用シンポジウム, 2004.

Hideki Yagi, Toshiyasu Matsushima and Shigeichi Hirasawa, "Efficient reliability-based soft decision decoding algorithm over Markov modulated channel, "ISITA2004, 2004.

Toshiyasu Matsushima and Shigeichi Hirasawa, "Universal Coding Algorithm for Side Information Context Tree Models, "第27回情報理論とその応用シンポジウム, 2004

Tomohiko Saito, Toshiyasu Matsushima, Shigeichi Hirasawa, "A Note on Construction of Nonlinear Unequal Orthogonal Arrays from Error-Correcting Codes, " 電子情報通信学会技術研究報告 IT2005-15, pp.13-18, 2005.

Daiki Koizumi, Toshiyasu Matsushima, Shigeichi Hirasawa, "A Study of Reliabil-

ity Based Hybrid ARQ Scheme with Bitwise Posterior Probability Evaluation from Message Passing Algorithm, " 電子情報通信学会技術研究報告 IT, 2005-24, 2005.
N. Kobayashi, T. Matsushima, S. Hirasawa, " A Note on a Decoding Algorithm of Codes on Graphs with Small Loops, " IEEE ISOC ITW2005, 2005.
G. Hosoya, T. Matsushima and S. Hirasawa, "A decoding algorithm of low-density parity-check codes using decisions of erasure correcting, "第 28 回情報理論とその応用シンポジウム, 2005
Daiki Koizumi, Toshiyasu Matsushima, Shigeichi Hirasawa, " A Note on HTTP Traffic Analysis of the Time Series Model with a Time Varying Density Parameter, " 第 28 回情報理論とその応用シンポジウム, 2005.
Toshiyasu Matsushima, Shigeich Hirasawa, " Bayes Universal Coding Algorithm for Side Information Context Tree, " IEEE International Symposium of Information Theory, 2005.
大澤雅之, 野村亮, 松嶋敏泰, "畳み込み符号の並列復号アルゴリズムの性能評価に関する一考察, "電子情報通信学会技術報告 IT2003-33, 2003.
須子統太, 松嶋敏泰, 平澤茂一, " 区間で一定なパラメータを持つ情報源におけるベイズ符号化法について, " 第 26 回情報理論とその応用シンポジウム, 2003.
渋谷知成, 斉藤友彦, 松嶋敏泰, " 衝突困難ハッシュ関数の安全性について, " 第 26 回情報理論とその応用シンポジウム, 2003.
江口公盛, 須子統太, 松嶋敏泰, " ベイズ決定理論に基づく予測における近似手法について, " 第 26 回情報理論とその応用シンポジウム, 2003.
岡野洋平, 小泉大城, 松嶋敏泰, "Tailbiting 畳込み符号の復号アルゴリズムに関する一考察, "電子情報通信学会技術報告 IT2004-26, 2004.
小林 直人, 松嶋 敏泰, 平澤 茂一, " Small-Loop を含む低密度パリティ検査 (LDPC) 符 号の復号に関する研究, " 第 27 回情報理論とその応用シンポジウム, 2004.
斉藤友彦, 松嶋敏泰, 平澤茂一, " 誤り訂正符号を利用した直交計画の構成法に関する一考察 ～逐次実験に適した直交計画について～, " 第 27 回情報理論とその応用シンポジウム, 2004.
須子統太, 松嶋敏泰, 平澤茂一, "区間で一定なパラメータを持つ非定常情報源におけるベイズ符号の冗長度について, "第 27 回情報理論とその応用シンポジウム,

2004.
宅味丈夫，須子統太，松嶋敏泰，”階層モデルにおけるベイズ予測の漸近評価に関する一考察，”第27回情報理論とその応用シンポジウム，2004
細渕智史，斎藤友彦，松嶋敏泰，” Fast Correlation Attackの改良法に関する一考察，” 第27回情報理論とその応用シンポジウム，2004.
吉田隆弘，松嶋敏泰，平澤茂一，”ID情報に基づくランプ型分散鍵配送方式について，”第27回情報理論とその応用シンポジウム，2004.
小林直人，小泉大城，松嶋敏泰，平澤茂一，“再帰的畳み込み符号を利用したReliability Based Hybrid ARQについての研究，”電子情報通信学会技術研究報告IT, 2005-52，2005.
小林直人，松嶋敏泰，平澤茂一，“帰還通信路を用いた誤り制御方式に関しての研究，”第28回情報理論とその応用シンポジウム，2005.
雨宮康二，小林直人，松嶋敏泰，”統計的決定理論に基づく有限受信バッファSR ARQ方式，”第28回情報理論とその応用シンポジウム，2005.
岡野洋平，小泉大城，松嶋敏泰，“単一ループをもつグラフィカルモデルにおける確率伝播型アルゴリズムに関する一考察，”第28回情報理論とその応用シンポジウム，2005.
堀井俊佑，須子統太，松嶋敏泰，”使用ユーザが変化するDS/CDMAシステムにおけるベイズ最適なマルチユーザ検出について，”第28回情報理論とその応用シンポジウム，2005.
須子統太，松嶋敏泰，平澤茂一，”BW変換を用いたユニバーサル符号化アルゴリズムに関する研究，”第28回情報理論とその応用シンポジウム，2005.
海上勇二，斎藤友彦，松嶋敏泰，”ID情報に基づくランプ型非対称鍵配送方式について，” 暗号と情報セキュリティシンポジウム, 2006.

目 次

第1章　研究目的

近年，誤り訂正符号の分野で発見あるいは再発見されたTurbo符号やLDPC符号はその誤り訂正能力の高さから21世紀の高信頼度情報通信，蓄積技術の重要な技術として注目を集めている．これらの符号の復号法として考案された，Turbo復号やsum-product復号法などの反復復号アルゴリズムが，知識情報処理の分野で不確実推論アルゴリズムとして用いられている確信度伝搬(BP: Belief Propagation)アルゴリズムの応用例として解釈されることが明らかとなってきた．

BPアルゴリズムは対象とするモデルの確率構造が簡単（グラフ表現では木構造）な場合においては性能の保証がなされている．しかし，残念ながら複雑な確率構造（グラフ表現ではループのある構造）にBPアルゴリズムを適応した場合の性能は保証されていない．Turbo符号やLDPC符号といった符号も複雑な確率構造をもっているため，いまだに理論的裏付けは不完全と言わざるを得ない．

平成12〜14年度科学研究費補助「不確実推論を用いた高信頼度反復復号アルゴリズムの設計と解析」の研究成果として得られた拡張ジャンクショングラフ（EJG）と並列反復事後確率計算アルゴリズムは，従来法に比べより一般的な確率モデル（符号）を扱えると共にループのあるグラフ（ほとんどの実用的符号ははループがある）においても性能の劣化が少ないことが確認された．そのためこのグラフとアルゴリズムは実用的な符号に対し優れた性能が期待される．

本研究では様々な符号や通信路に対しその応用を試み，性能の実験的及び解析的に検討を行う事を目的とする．

第2章　一般化確率推論

2.1　確率推論と統計的決定理論

統計的決定理論において事後確率が重要な役割を果たすことは良く知られている．例として確率分布 $P(y|x)$ 上での統計的決定問題を考えよう．$x \in S_x$ は未知で推定したい対象と仮定する．x は統計パラメータ推定問題ではパラメータに対応し，復号問題では送信シンボルとも考えられる．y は既知で，観測されたデータや受信シンボルなどと考えても良い．

決定問題は決定関数 $d(y)$ と損失関数 $L(x, d(y))$ により定式化される．ここで決定関数 $d(y)$ は x の推定値とすると，例えば以下のような損失関数が考えれらる．

$$Loss_1 = \begin{cases} 0, & x = d(y) \\ 1, & x \neq d(y) \end{cases} . \tag{2.1}$$

$$Loss_2 = (x - d(y))^2. \tag{2.2}$$

これらの損失関数に対してベイズ基準の下で最適な決定はそれぞれ以下のようになる．

$$d_1(y) = \arg\max_{x \in S_x} P(x|y). \tag{2.3}$$

$$d_2(y) = \sum_{x \in S_x} xP(x|y). \tag{2.4}$$

$d_1(y)$ の決定では事後確率最大となる x を選択することになり，$d_2(y)$ の決定では事後確率 $P(x|y)$ で x の期待値をとったものが最適決定となっている．いずれの決定においても事後確率 $P(X|Y = y)$ が重要であることが分かる．$d_1(y)$ の決定において，

x の事前分布 $P(x)$ が一様分布の場合，最適な決定は $d_1(y)$ の尤度 $P(y|x)$ を最大とする x を選択する推定，つまり最尤推定となっており，最尤推定と事後確率最大の決定は本質的に同じものといえる．

また，統計力学や，人工知能の確率推論の分野では事後確率自体を求めることが主要なテーマとも言える．BN（Bayesian network）などを用いた確率推論 [?][?] において，証拠（evidence）が与えられたもとで各命題の確信度（belief）[?] の更新を行うことは，ある確率変数の値のみが与えられたもとで，その他の確率変数の周辺事後確率を求めていることに他ならない．確率推論で事後確率をもとめることが目的になっている理由は，上記で述べてきたように事後確率が統計的決定/判断において重要な情報/値となっていることによる．このように様々な分野において，事後確率は重要な役目を持ち，それを効率よく計算することは重要な研究課題のひとつとなっている．

2.2　一般化確率推論と一般化事後確率

BP（belief propagation）等の通常の確率推論では証拠（evidence）として $X = x$ が与えられたもとで，目的とする確率変数 Y の事後確率 $P(Y|X = x)$ を求めている．この場合の証拠は確率変数 X の確定値 x が情報として与えられている．もし，ある確率変数の分布 $P(X = x) = p_x$ が情報として与えられた場合の確率推論はどのようにかんがえればよいのだろうか．このような証拠を従来研究では distribution-evidence または soft-evidence と呼んでいる．

証拠として distribution-evidence が与えれた場合の推論法は従来からいくつか研究されているが，ほとんどの研究がその推論手続きのみを提案しており，推論結果がどのような意味をもつのか，その理論的裏づけを明確にしたものは見当たらない．従来研究のいくつかでは Jeffry's Rule と呼ばれる確率計算法を用いて推論を行うことを推奨している．Jeffry's Rule を用いた計算法は残念ながら一つの確率変数に対する distribution-evidence が与えられた場合のみ推論が可能で，複数の確率変数に対して distribution-evidence が与えられた場合には計算が行えない．また，そもそもなぜこの計算法を用いればよいのか，ほとんどの従来研究では定性的概念からしか正当性が説明されていない．

以降，複数の distribution-evidence を扱える一般化確率推論について説明する．これは従来の確率推論や Jeffry's Rule を含む自然な拡張となっている．

確率変数 $X_i, i \in I = 1, ..., n$ と証拠（evidence）と呼ばれる確率変数 $E_j, j \in I_C \subset I$ を定義する．簡単のためこれらの確率変数は離散としておく．一つの証拠 E_j はそれに対応する一つの確率変数 X_j についてのみ情報をもっており，その他の確率変数 $X_i, i \in I - j$ についての情報は一切含まないと考える．これを数式で示すと以下のようになる．

仮定 2.1 確率変数 X_j が与えられたもとで，それに対応する証拠 E_j とそれ以外の確率変数 $X_i, i \in I - j$ は条件付独立になっている．

$$P(X_1, ..., X_n, E_j) = \frac{P(X_1, ..., X_n)P(X_j, E_j)}{P(X_j)}. \tag{2.5}$$

□

以下で一般化確率推論を定義する．

定義 2.1 証拠 e_j により "X_j の分布は $P^*(X_j)$ である" が与えられたとする．証拠 e_j により与えられる情報は $P^*(X_j) = P(X_j|E_j = e_j)$ とする．この情報が与えられたもとで，事前分布 $P(X_1, ..., X_n)$ から $P(X_1, ..., X_n|E^{I_C} = e^{I_C})$ を求める推論を一般化確率推論と定義する． □

もし与えられた分布 $P(X_j|E_j = e_j)$ が一点に確率をもつ，すなわち $P(X_j = x_j|E_j = e_j) = 1$ となる場合，証拠 e_j から与えられる情報は "x_j が世紀した"，つまり，$X_j = x_j$ と同値である．この場合上記で定義された一般化確率推論は $P(X_1, ..., X_n|X^{I_C} = x^{I_C})$ を求めることと同値になる．ここで $X^{I_C} = x^{I_C}$ は $(X_{j_1} = x_{j_1}, ..., X_{j_{|I_C|}} = x_{j_{|I_C|}})$ を表す．これは通常の確率推論と一致し，一般化確率推論は通常の確率推論を含み，その自然な拡張になっていることが分る．

定義 2.2 一般化確率推論により求められた分布を，X_j の周辺分布 $P^*(X_j)$ が与えられたもとでの一般化事後分布と定義する． □

この一般化事後分布がその特殊な場合として通常の事後分布を含むことは明らかであるが，この一般化事後分布が統計的決定理論において，通常の事後分布と全く

同様の役割を果たすことに注意されたい．つまり，先に述べた幾つかの決定関数と損失関数に対して，一般化事後確率を最大化する確率変数の値や確率変数の一般化事後確率による期待値などが最適な決定となっている．

次に一般化事後分布と事前分布の重要な関係を示す．

定理 2.2.1 M_C を X_j の周辺分布が $P^*(X_j) = P(X_j|E_j = e_j), j \in I_C$ を満たす分布 $P(X_1, ..., X_n)$ の集合とする．事前分布 P_{pr}から仮定*2.1*のもとで，定義 *2.1* の一般化事後確率推論により求められてた一般化事後確率は以下の式を満たす．

$$P_{po} = \arg \min_{P \in M_C} I(P \| P_{pr}). \tag{2.6}$$

ここで I は *Kullback-Leibler(K-L)* 情報量を表す． □

一般化事後確率は，周辺分布の制約のもとでK-L 情報量にｙり測られた距離が事前分布に最も近い分布となっていることを，この定理は示している．

2.3 Itrative Proportional Fitting Procedure（IPFP）

定理 2.2.1 で示した一般化確率推論の問題は，一種の制約付最適化問題とみなせる．一般に制約付最適化問題において最適会を求めるためには多くの計算量が必要になる．しかし，この周辺分布制約のもとでのK-L 情報量最小化問題については従来からいくつかの研究があり，Itrative Proportional Fitting Procedure （IPFP）またはIterative Scaling Procedure（ISP）と呼ばれる効率的な繰り返しアルゴリズムが提案されている．IPFP は離散データの統計解析において，分割表の周辺和が与えれているもとで，ある種の独立性を仮定したモデルのBAN 推定量や最尤推定量の計算に用いられている．

このIPFP は一般化確率推論においても利用可能である．以下で $P(X_1, ..., X_n|E^{I_C} = e^{I_C})$ を求める一般化確率推論のIPFP を適用した手続きを示す．

【Procedure 1A:IPFP】
begin
$P(X_1, ..., X_n) := P_{pr}(X_1, ..., X_n)$;

while $\exists_{j \in I_C} P(X_i) \neq P^*(X_j)$ do
　　begin
　　Pick up X_j from $\{X_j | P(X_j) \neq P^*(X_j), j \in I_C\}$;
　　$P(X_1, ..., X_n) := P(X_1, ..., X_n) P^*(X_j)/P(X_j)$;
　　end
$P_{po}(X_1, ..., X_n) := P(X_1, ..., X_n)$;
end　□

上記の手続きの 6 行目では，周辺分布 $P(X_j)$ が制約条件の周辺分布 $P^*(X_j)$ に合うように毎ステップごとに比例計算により分布の更新を行っている．IPFP はこれを周辺確率が制約条件に収束するまで繰り返す簡潔なアルゴリズムである．

このアルゴリズムの性質を幾何学的に解釈してみる．一つの確率分布を一つの点に対応させた確率分布の空間を考える．周辺分布制約のもとでの事前分布との K-L 情報量を最小化する分布は，事前分布から制約条件を満たす分布の多様体上へのある種の射影とみなせる．この射影は e-射影または I-射影と呼ばれている．

補題 2.3.1 M_C を X_j の周辺分布が $P^*(X_j) = P(X_j | E_j = e_j), j \in I_C$ をみたす分布 $P(X_1, ..., X_n)$ の多様体とする．この多様体は m-平坦となり，一般化事後確率 $P_{po} = \arg\max_{P \in M} I(P \| P_{pr})$ は事前分布 P_{pr} から多様体 M_C 上への e-射影で与えられる．

また，任意の分布 $p \in M_C$ は以下を満たす．

$$I(p \| P_{pr}) = I(p \| P_{po}) + I(P_{po} \| P_{pr}). \tag{2.7}$$

□

複数の周辺制約を満たす多様体への e-射影は，ここの周辺制約への e-射影の繰り返しにより求めることができる．

補題 2.3.2 複数の周辺制約 $C = \bigcap_{j=1}^{m} C_j \neq \phi$ を構成する一つの周辺制約 C_j を満たす分布の多様体を M_j で表す．分布 P_t を分布 P_{t-1} から多様体 $M_t, t = 1, 2, ...$ への e-

射影とする．ここで，$P_0 = P_{pr}$ であり，$M_t = M_j, (\mod m)$ とする．以上により再帰的に定義された $P_1, P_2, ...$ の系列は P_{pr} から周辺制約 C を満たす多様体 M_C 上への e-射影に収束する．□

事前分布 P_{pr} から複数の周辺制約を満たす多様体上への e-射影は以下のような周辺パラメータを用いて表すことが可能である．

補題 2.3.3 事前分布 P_{pr} から周辺制約 $P(x_j) = P^*(x_j), j \in I_C$ を満たす多様体上への e-射影である一般化事後分布 P_{po} は以下のように表すことが出来る．

$$P_{po}(x_1, ..., x_n) = P_{pr}(x_1, ..., x_n) \prod_{j \in I_C} \beta(x_j), \tag{2.8}$$

ここで

$$P^*(x_j) = \sum_{\{x_i | i \neq j\}} P_{pr}(x_1, ..., x_n) \prod_{j \in I_C} \beta(x_j). \tag{2.9}$$

□

系 2.3.1 $P(X_1, ..., X_n)$ から周辺制約 $P^*(X_j)$ を満たす多様体上への e-射影 $P^*(X_1, ..., X_n)$ は以下のように計算される．

$$P^*(x_1, ..., x_n) = P(x_1, ..., x_n) \frac{P^*(x_j)}{P(x_j)}. \tag{2.10}$$

□

Procedure 1A つまり IPFP の正当性は補題 2.3.2 と系 2.3.1 より以下のように示される．

補題 2.3.4 *Procedure 1A* は停止し，求められた分布は $P(X_1, ..., X_n | E^{I_C} = e^{I_C})$ へ収束する．□

2.4　周辺パラメータを用いたアルゴリズムと並列アルゴリズム

Procedure 1A においては，計算過程の分布を結合分布 $P(X_1,...,X_n)$ で表現し保持していたが，補題 2.3.3 で示したように，目的とする事後分布は事前分布 P_{pr} と周辺パラメータ $\beta(X_j), j \in I_C$ を用いても表現可能である．

Procedure 1A を事前分布 P_{pr} と周辺パラメータ $\beta(X_j)$ を用いた表現で書き直すことで，以下のような Procedure 1B が提案されている．

【Procedure 1B】
begin
$\beta(X_j) := 0, j \in I_C$
while $\exists_{j \in I_C} P(X_i) \neq P^*(X_j)$ do
　　begin
　　Pick up X_j from $\{X_j | P(X_j) \neq P^*(X_j), j \in I_C\}$;
　　$\gamma(X_j) := \sum_{i \neq j} P_{pr}(X_1,...,X_n) \prod_{j \in I_C} \beta(X_j)$;
　　$\beta(X_j) := P^*(X_j)/\gamma(X_j)$;
　　end
$P_{po}(X_1,...,X_n) := P_{pr}(X_1,...,X_n) \prod_{j \in I_C} \beta(X_j)$;
end　□

このアルゴリズムを Procedure 1A と比較すると，収束するまでの反復回数は同じであるが，メモリ量は少なくて済む．また，後で述べる分解可能モデルに対して Procedure 1B を利用した確率伝播アルゴリズムは Procedure 1A を利用した場合に比べ計算量が少ないという特長がある．

Procedure 1A と Procedure 1B は逐次アルゴリズムであるが，以下のような並列アルゴリズムも提案されている．

【Procedure 2B】
begin

$\beta(X_j) := 0, j \in I_C$
while $\exists_{j \in I_C} P(X_i) \neq P^*(X_j)$ do
　　begin
　　Pick up X_j from $\{X_j | P(X_j) \neq P^*(X_j), j \in I_C\}$;
　　$\gamma(X_j) := \sum_{i \neq j} P_{pr}(X_1, ..., X_n) \prod_{j \in I_C} \beta(X_j), j \in I_C$;
　　$\beta(X_j) := P^*(X_j)/\gamma(X_j)$;
　　end
$P_{po}(X_1, ..., X_n) := P_{pr}(X_1, ..., X_n) \prod_{j \in I_C} \beta(X_j), j \in I_C$;
end　□

Procedure 1A と Procedure 1B は一回の繰り返しにおいて，一つの周辺分布制約についてe-射影を行うことで分布の更新を行っているが，この並列アルゴリズムでは全ての周辺分布制約に対して同時に更新を行っていることになる．Procedure 2B が行っている計算を幾何学的に解釈してみよう．1時点前のステップで計算された分布から，各周辺制約を満たす多様体へのe-射影を考える．この分布から各e-射影へのベクトルを用い，それらの合成ベクトルを作る．この合成ベクトルにより移される分布がこのステップで計算された分布となっていることが分る．この解釈を厳密化することで以下の定理が証明される．

定理 2.4.1 *Procedure 2B* は停止し，*Procedure 1A* および *Procedure 1B* と同様の分布に収束する．□

2.5 Junction graphの拡張

確率モデルが次式のようにクリークの積で記述される場合，従来の Junction graph(JG) を拡張した extended junction graph（EJG）が提案されている．

$$P(x_1, ..., x_n) = \alpha q(N_1) q(N_2) \cdots q(N_{n_N}), \tag{2.11}$$

ここで，$N_l = (X_{i_1(l)}, ..., X_{i_{n(l)}(l)})$.
EJG は JG と同様に交差節（intersection node）とクリーク節（clique node）の 2 種

類の節で記述されるが，交差節の結合法が少々異なっている．従来の JG では交差節は必ず 2 つのクリーク節と枝で結ばれているが，EJG の場合は 2 つ以上のクリーク節と結合されても構わない．JG の場合，全ての交差節を削除しても残されたグラフは通常のグラフであるが，EJG の場合，残されたグラフが hypre graph となる場合もありえる．

EJG や JG は BN や factor graph を含む，より広い範囲の確率モデルを表現できるグラフと考えられる．

2.6　EGJ 上の一般化事後確率の効率的計算法

一般の確率モデルにおける一般化事後確率の計算は確率変数の数に対して指数オーダの計算量が一回の繰り返しにおいて必要であり，通常の事後確率計算以上に膨大な計算量が必要である．しかし，通常の事後確率の計算と同様に，確率モデルが式 (2.11) のようにクリークの積で記述される場合，効率的アルゴリズムが提案されている．従来の周辺事後確率計算と同様にグラフで確率モデルを表現しその上でメッセージを伝播させる方法が有効である．

ループのない EJG 上での周辺一般化事後確率の計算は，Procedure 1B を各クリーク節ごとに行い，その結果を隣接するノードに伝播させることで基本的には計算が行える．これは逐次的確率伝播アルゴリズムと位置付けられる．

他方，Procedure 2B を応用することで，一般化事後確率を計算する並列確率伝播アルゴリズムも構成することができる．この並列アルゴリズムはループのある EJG にもそのまま適用することが可能となり，正確な一般化事後確率計算のみならず近似計算にも利用可能である．

ここでは並列伝播アルゴリズムを示す．このアルゴリズムでは交差節からクリーク節へのメッセージと，その逆にクリーク節から交差節へのメッセージが交互に交換される．

【Algorithm 1B】

各交差節 D_m からそれに隣接するクリーク節 N_i へのメッセージの伝播は以下となる．$D_m = \{X_j\}$ かつ $j \in I_C$ のとき，D_m を制約節と呼ぶ．また，$S^N(D_m)$ は D_m に

隣接するクリーク節の集合とする.

$$q_{t+1}^{N_l}(D_m) := \begin{cases} P^*(D_m)/\gamma_t^{N_l}(D_m), & D_m\text{は制約節} \\ \prod_{\{N_k \in S^N(D_m)|k\neq l\}} \gamma_t^{N_k} D(_m), & \text{その他} \end{cases} . \tag{2.12}$$

各クリーク節 N_l からそれに隣接する交差節 D_m へのメッセージ伝播は以下となる. $S^D(N_l)$ は N_l に隣接する交差節の集合とする.

$$\gamma_t^{N_k} D(_m) := \sum_{x \notin D_m} q(N_l) \prod_{\{D_h \in S^D(N_l)|h\neq m\}} q_t^{N_l}(D_h). \tag{2.13}$$

それぞれのクリーク節の周辺一般化事後確率は以下のように計算される.

$$P_{t+1}(N_l) = q(N_l) \prod_{D_h \in S^D(N_l)} q_t^{N_l}(D_h). \tag{2.14}$$

□

定理 2.6.1 もし *EJG* がループを含まなければ, *Algorithm 1B* は停止し, 正しい一般が事後確率を計算する. □

一般化事後確率は通常の事後確率も含んでいるため, このアルゴリズムにより通常の事後確率も計算可能である. その意味で, このアルゴリズムは BP や HUGIN アルゴリズムの一般化になっている. さらに EJG が BN や factor graph を含むことから, 表現できる確率モデルのクラスの意味でも拡張されている.

この並列伝播アルゴリズムは確率推論の分野における distribution-evidence を含む推論への応用はもちろんのこと, 復号問題など通常の事後確率計算を行っている分野への応用も可能である. 次章ではこのアルゴリズムを復号問題へ適用した場合について述べる.

第3章　一般化確率推論の応用

3.1　通信路符号化のモデルと復号問題

本章では，通信路符号化について考える．通信路符号化とは，雑音のある通信路を介して通信を行う際に、送信された情報を受信先で推定する技術であり，一般には以下のような数理モデルとして扱われている．

送信すべき情報を u とする．送信元では，この情報をブロック長 n の符号語 x^n に符号化したものを送信する．通信路のモデルは条件付確率 $P(y^n|x^n)$ で表現され，受信側では受信系列 y^n より元の情報 u を推定する．このモデルは，第 2 章の統計的決定問題と等価なモデルとなっていることがわかる．推定の評価には一般に，各シンボルの平均誤り率（シンボル誤り率）又は各ブロックの通信回数に対する平均誤り率（ブロック誤り率）が用いられる．これが先に述べた損失関数に対応していることは明らかである．推定方法としては，各シンボルの（近似）周辺事後確率を計算し，その確率が最大となるシンボルの値をそれぞれの推定シンボルとする方法 (MAP(Maximum a posteriori probability) 復号) や，受信系列に対して最も尤度が高くなる符号語を，推定符号語とする方法（最尤復号）等がある．

MAP 復号を考えた場合，通信路符号化における復号問題は各シンボルごとの（近似）周辺事後確率の計算問題へ帰着される．そのため，前述の一般化確率推論における応用問題の一つであると見なすことができる．

また，復号の際に用いる確率モデルは，符号の種類や通信路のモデルに依存する．多くの実用的な符号や通信路において，この確率モデルはグラフィカルモデルで表現した場合，多数のループ構造を持つことが知られている．そのため，前述のアルゴリズムを適用することでその性能の向上が期待される．以下，様々な符号や通信路に対して前述のアルゴリズムを適用した結果について述べる．

3.2　実用的符号への応用

3.2.1　tail-biting 畳み込み符号への応用

tail-biting 畳み込み符号は EJG で表現した場合，単一のループを持つ事が知られている．ループを持つものの，グラフの構造が単純であるため比較的解析が行いやすいと考えられる．そこで，tail-biting 畳み込み符号に Algorithm 1B を適用した場合についていくつかの実験によってその性能を確かめた．その結果，BCJR アルゴリズム（tail-biting BCJR アルゴリズム）と同じ誤り率をより短い時間で得られること，復号に必要な時間は畳み込み符号の拘束長に比例することなどが分かった．

この実験を踏まえ，畳み込み符号の場合の性能の理論値を部分的にモンテカルロ法も用いることにより数値計算によって示した．

また Algorithm 1B における sum 演算を max 演算に変換することによって Viterbi アルゴリズムの並列計算版を構成可能である．これを tail-biting 畳み込み符号に適用してブロック誤り率を調べる実験を行った．結果 tail-biting Viterbi アルゴリズム，tail-biting BCJR アルゴリズムを用いるよりも計算時間は大幅に減少し，ブロック誤り率も向上することが実験によって確かめられている．

前述の通り，tail-biting 畳み込み符号は EGJ で表記した場合単一ループをもつ構造をしている．その為，Algorithm 1B で復号を行っても正確な事後確率計算はできず近似計算となる．厳密に事後確率を計算する場合，多くの計算量が必要となるが，単一ループしか持たないため，他の多くのループを持つ符号よりは少ない計算量で済み，実用的な計算量ではないが実験的に計算をすることは可能である．そこで，厳密に事後確率を計算した場合と Algorithm 1B で計算した場合とについて KL 情報量を比べることにより，この近似精度について実験的に評価を行った．この結果を発展させることにより，ループを多く持つような複雑な確率構造で表現される LDPC 符号や Turbo 符号などの復号性能の評価，および事後確率計算アルゴリズムの改良などが期待される．

3.2.2　LDPC 符号への応用

EJG や JG は BN や factor graph を含む，より広い範囲の確率モデルを表現できるグラフと考えれらる．例えばLDPC 符号は factor graph で記述されることが一般的であるが，多くのループを含んだグラフとなる．特に長さ 4 のループが factor fraph に含まれると sum-product アルゴリズムは極端に性能が劣化することが知られており，通常は長さ 4 のループを含まないようにLDPC 符号は構成される．しかしこのような構成法は符号のクラスを限定していると言え，長さ 4 のループを持つ符号により性能の高い符号が存在する可能性も考えられる，

そこで，ランダムに構成されたLDPC 符号に対し，EJG を用いて確率モデルを表現し長さ 4 のループ部分を単一のノードとして扱うことによって，従来長さ 4 のループ部分に起きていた確率伝播アルゴリズムの近似精度の劣化に対する影響を軽減することを行った．その結果，エラーフロア領域の改善が実験的に認められた．また，EJG を用いる事で sum-product アルゴリズムに比べ部分的に計算量が増大してしまうことから，符号制約を利用することで計算量の削減を行う新たなアルゴリズムを提案した．

3.2.3　帰還通信路を用いた通信方式への応用

帰還通信路を用いた通信方式の研究については，従来から ARQ 方式などの研究が盛んに行われている．ARQ 方式では，誤り検出符号を用いて受信者側が誤り検出を行い，誤りが検出された場合送信者側に帰還通信路を用いて再送要求を送信する．再送要求を受けた送信者は受信者が誤りを検出しなくなるまで同じ信号を受信者へ繰り返し送信する．更にこの ARQ 方式と誤り訂正符号を組み合わせた Hybrid-ARQ 方式など数多くの研究がなされている．従来の ARQ や Hybrid-ARQ 方式の研究では，帰還通信路を通す情報としては，受信者が信号を正しく受信したという ACK（肯定応答）信号，もしくは正しく受信しなかったという NACK（否定応答）信号のみであった．近年，この帰還通信路により多くの情報を通すことで効率よく通信を行う研究がなされている．

本研究では，畳み込み符号をベースとした誤り訂正符号を送信し，帰還通信路を

用いて各ビットの信頼度情報を送信する方式についての研究を行った．帰還通信路で信頼度情報を送信することで，送信者は信頼度の低いビットの情報を重点的に再送することができる．また，再送を行うことにより復号に用いる確率モデルが変わる．前述の通り復号における事後確率計算の精度は確率モデルに依存する．そのためどのような情報を再送するのか，またその情報を用いてどのように事後確率計算を行うのかが重要になってくる．本研究では再送情報の取り扱いについて様々なパターンを実験的に検証を行うことで，あらたな帰還通信路を用いた通信方式の提案を行った．その結果，従来よりもスループットの向上を図ることができた．

3.3　その他の分野への応用

近年，暗号分野におけるストリーム暗号の攻撃法に対して事後確率計算アルゴリズムを用いた研究が行われている．ストリーム暗号の攻撃は非線形コンバイナ型乱数生成機における多入力一出力の非線形関数部分の攻撃に帰着される．従来，この多入力一出力の非線形関数部分を一入力一出力の通信路モデルで近似させることにより，BP を用いて事後確率計算を行い攻撃をする手法が提案されていた．

本研究では，多入力一出力の非線形関数を二入力一出力の確率モデルで近似することで攻撃を行った．確率モデルを複雑にすることにより関数の近似精度が上がった結果，従来よりも解読成功率を向上させることができた．また，より入力数を増やしていくことで関数の近似精度が上がるが，確率モデルが複雑になるため事後確率の近似精度は低下する．このトレードオフについても実験的に検証を行った．

論　文

直交計画を用いたブール関数の学習に関する一考察

浮田　善文[†a)]　　松嶋　敏泰[††]　　平澤　茂一[††]

A Note on Learning Boolean Functions by Using Orthogonal Design

Yoshifumi UKITA[†a)], Toshiyasu MATSUSHIMA[††], and Shigeichi HIRASAWA[††]

あらまし　本論文では，質問する入力集合があらかじめ固定されている一括型所属性質問によりブール関数を学習する問題を扱う．まず，実験計画法と質問からのブール関数の学習との関係を明らかにし，ブールドメイン上の実数値関数に対するサンプリング定理が直交計画の特別な場合として与えられることを示す．次に，今までの計算論的学習理論では扱われていない概念クラス（どの変数同士の値の組合せが関数値に影響を与える可能性があるかといった細かい情報も制約に加えたクラス）を提案する．このクラスに真のブール関数が含まれる場合，質問する入力集合を直交計画とすることで真のブール関数を必ず出力できることを示す．更に，多次元高速フーリエ変換アルゴリズムを用いることで，質問後にブール関数を決定する際の計算量を削減できることを示す．

キーワード　所属性質問からの学習，ブール関数，実験計画法，直交計画，高速フーリエ変換

1. まえがき

質問からのブール関数の学習問題は計算論的学習理論の立場から多くの研究が行われている[5]～[7]．ここでは，ブール変数の個数 n，サイズを表すパラメータ s により規定されたブール関数のクラス（概念クラス）に対し，どの種類の質問を用いた場合にパラメータ n, s の多項式時間で学習可能かどうかが明らかにされている．例えば，単調 DNF と呼ばれる概念クラスに対し，所属性質問と等価性質問を用いて真のブール関数に含まれる項を一つずつ見つけることで，n, s と質問回数の多項式時間で学習するアルゴリズムが示されている[5], [6]．

ところで，クラスが与えられたもとで，質問を行い真のブール関数を見つける問題と類似の問題は，信号処理や実験計画法の分野でも研究されている．

信号が周波数 t 以上の高次の周波数成分をもたない帯域制限信号クラスに含まれる場合，ナイキストのサンプリング定理[19]により，周期が $1/(2t)$ より小さい標本点集合から，もとの信号を完全に復元することができる．このため，信号は周波数成分の直交基底を用いて表現されている．近年では，直交基底により表現された実数値関数やブール関数の学習について研究が行われている[1]～[4]．瀧本ら[1]は，ブールドメイン上の実数値関数に対してもサンプリング定理と同様な定理が成り立つことを示し，計算論的な立場から学習可能性について論じている．

一方，統計学の実験計画法でも，関数は直交基底により表現されており，高次の成分をもたない関数クラスを対象に古くから多くの研究が行われている[11], [12]．ただし，実験計画法では，どの変数同士の値の組合せが関数値に影響を与える可能性があり，どの変数同士の値の組合せがそうでないかが，実験前にわかる場合が多く，この情報も制約に加えたクラス(注1)を扱う．実際に実験を行うことを考えた場合，関数値に影響する可能性のある変数（変数同士）の効果を調べることが可能で実験回数が最小となる実験計画を求めることが要求される．このため，同じ実験回数の実験計画の中で最適（各係数の不偏推定量の分散の最大値が最小）な実験計画である直交計画に関する研究が多く行われている．しかし，変数の個数などの多項式時間で学

† 横浜商科大学商学部経営情報学科，横浜市
Yokohama College of Commerce, 4-11-1 Higashi-terao, Tsurumi-ku, Yokohama-shi, 230-8577 Japan
†† 早稲田大学理工学部経営システム工学科，東京都
School of Science and Engineering, Waseda University, 3-4-1 Ohkubo, Shinjuku-ku, Tokyo, 169-8555 Japan
a) E-mail: ukita@shodai.ac.jp

(注1)：制約する条件に関する情報は細かいものが必要だが，実験計画法や統計解析ではこのような制約をおくことが通常であり，実用上多くの対象や場面に応用されている重要なクラスとなっている．

習可能かといった計算論的な立場からの研究は見られない．

このように，3分野（質問学習，信号処理，実験計画法）の問題は，クラスに制約をおくことで，ドメインのある部分集合に対してのみ，質問やサンプリングや実験を行うだけで，真の関数の同定が可能になるという点では共通である．しかし，計算論的学習理論では多項式時間学習可能なクラスの制約を明らかにすることが主な目的であり，実験計画法で扱われている実用的で有用なクラス（細かい情報も制約に加えたクラス）を効率的に学習するという考えはこれまで見られなかった．また，これら3分野間の関係さえも明らかにされていなかった．

本論文では，質問する入力集合があらかじめ固定されている一括型所属性質問によりブール関数を学習する問題を扱う．まず，実験計画法と質問からのブール関数の学習との関係を明らかにし，ブールドメイン上の実数値関数に対するサンプリング定理が直交計画の特別な場合として与えられることを示す．次に，今までの計算論的学習理論では扱われていない概念クラス（どの変数同士の値の組合せが関数値に影響を与える可能性があるかといった細かい情報も制約に加えたクラス）を提案する．このクラスに真のブール関数が含まれる場合，質問する入力集合を直交計画とすることで真のブール関数を必ず出力できることを示す．

ここで質問回数を考えた場合，実験計画法での実験回数(注2)に比べ，より大きい質問回数を用いる場合が考えられる．このとき，質問結果からブール関数を決定するときの計算量が問題となる．そこで，多次元高速フーリエ変換 (Fast Fourier Transform:FFT) アルゴリズムであるベクトルラディックス FFT [19] を用いることで，質問後にブール関数を決定する際の計算量を削減できることを示す．

以下で，本論文の流れを説明しておく．まず **2.** で準備として，必要な記号の定義と直交基底による関数の表現を与えておく．**3.** では，実験計画法について，モデルと直交計画を中心に説明する．**4.** では，本論文と関連が深いブールドメイン上の実数値関数を実数体上の直交基底により表現した場合のサンプリング定理 [1] が実験計画法の結果から得られる定理の特殊な場合であることを明らかにする．また，ベクトルラディックス FFT を用いることで，質問後に関数を決定する際の計算量を削減できることを示す．そして，**5.** で，質問からの学習モデルを定義し，直交計画を質問するときの性質を述べる．真のブール関数が概念クラスに含まれる場合，直交計画を質問することで，真のブール関数を必ず出力できることを示す．**6.** では，真のブール関数が概念クラスに含まれない場合について触れ，**7.** でまとめを行う．

2. 準　　備

2.1 記号の定義

はじめに，必要な用語を与えておく．長さ n のベクトルを $\underline{a} = (a_1, a_2, \cdots, a_n), \underline{b} = (b_1, b_2, \cdots, b_n)$ と表し，$\underline{a}, \underline{b} \in \{0,1\}^n$ とする．ここで，ベクトル間の加法 $+$ を，

$$\underline{a} + \underline{b} = (a_1 \oplus b_1, a_2 \oplus b_2, \cdots, a_n \oplus b_n), \qquad (1)$$

とする．ただし，$\oplus$ は排他的論理和である．ベクトル間の内積 $\cdot$ を，

$$\underline{a} \cdot \underline{b}^T = a_1 b_1 \oplus a_2 b_2 \oplus \cdots \oplus a_n b_n, \qquad (2)$$

とする．ただし，記号 T は転置を表す．

ベクトル $\underline{a}$ のハミング重み $w(\underline{a})$ を，$w(\underline{a}) = \sum_{i=1}^{n} a_i$ とする．また，すべての要素が0であるベクトルを $\underline{0}$ とする(注3)．

[定義 1]　1次独立

$u_i \in \{0,1\}, (1 \le i \le l)$ とする．$u_1\underline{c_1} + u_2\underline{c_2} + \cdots + u_l\underline{c_l} = 0$ が成立するのは，$u_1 = u_2 = \cdots = u_l = 0$ のみであるとき，$\{\underline{c_1}, \underline{c_2}, \cdots, \underline{c_l}\}$ は1次独立という．　□

[定義 2]　行列 G

$k \times n$ 行列 G を次式で定義する．

$$G = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{21} & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ g_{k1} & g_{k2} & \cdots & g_{kn} \end{bmatrix}, \qquad (3)$$

ただし，$g_{ij} \in \{0,1\}, (1 \le i \le k, 1 \le j \le n)$ である．

G の第 i 行を $\underline{g_{i\cdot}} = [g_{i1}\ g_{i2}\ \cdots\ g_{in}]$，$G$ の第 j 列を $\underline{g_{\cdot j}} = [g_{1j}\ g_{2j}\ \cdots\ g_{kj}]^T$ と表記する．行列 G の行ベクトルは1次独立，すなわち，$\{\underline{g_{1\cdot}}, \underline{g_{2\cdot}}, \cdots, \underline{g_{k\cdot}}\}$ は1次独立とする．　□

(注2)：実験計画法ではランダムな順序で実験を行う必要があり，実験回数が多いとランダム化が困難になる [14]．
(注3)：すなわち，$\underline{0} = (0, 0, \cdots, 0)$ である．

[定義 3] $X, X_{\underline{s}}^{\perp}$

行列 G を用い，集合 X，集合 $X_{\underline{s}}^{\perp}, (\underline{s} \in \{0,1\}^k)$ を次式で定義する．

$$X = \{\underline{x} | \underline{x} = \underline{r} \cdot G,\ \underline{r} \in \{0,1\}^k\}. \tag{4}$$

$$X_{\underline{s}}^{\perp} = \{\underline{a} | G \cdot \underline{a}^T = \underline{s}^T, \underline{a} \in \{0,1\}^n\}. \tag{5}$$

このとき，$|X| = 2^k, |X_{\underline{s}}^{\perp}| = 2^{n-k}$ である． □

[例 1] 3×4 行列 G が次式で与えられる場合を考える．

$$G = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}. \tag{6}$$

このとき，集合 X，集合 $X_{\underline{s}}^{\perp}, (\underline{s} \in \{0,1\}^3)$ は次式で与えられる．

$$X = \{0000, 1000, 0101, 1101, 0011, 1011, 0110, 1110\}. \tag{7}$$

$X_{000}^{\perp} = \{0000, 0111\}, X_{100}^{\perp} = \{0010, 0101\},$
$X_{010}^{\perp} = \{0011, 0100\}, X_{110}^{\perp} = \{0001, 0110\},$
$X_{001}^{\perp} = \{1000, 1111\}, X_{101}^{\perp} = \{1010, 1101\},$
$X_{011}^{\perp} = \{1011, 1100\}, X_{111}^{\perp} = \{1001, 1110\}.$ □

2.2 直交基底による関数の表現

$\underline{x} = (x_1, x_2, \cdots, x_n) \in \{0,1\}^n$ とする．任意のベクトル $\underline{a} = (a_1, a_2, \cdots, a_n) \in \{0,1\}^n$ に対して以下に定義される関数 $\mathcal{X}_{\underline{a}}: \{0,1\}^n \to \{-1,+1\}$ を，基底関数と呼ぶ．

$$\mathcal{X}_{\underline{a}}(\underline{x}) = (-1)^{\underline{a} \cdot \underline{x}^T}. \tag{8}$$

このとき，基底関数のクラス $\{\mathcal{X}_{\underline{a}} | \underline{a} \in \{0,1\}^n\}$ は正規直交系をなす．すなわち，

$$\frac{1}{2^n} \sum_{\underline{x} \in \{0,1\}^n} \mathcal{X}_{\underline{a}}(\underline{x}) \mathcal{X}_{\underline{b}}(\underline{x}) = \begin{cases} 1; & \underline{a} = \underline{b}, \\ 0; & \underline{a} \neq \underline{b}, \end{cases} \tag{9}$$

が成立する．このことから，任意の実数値関数 $f: \{0,1\}^n \to \mathcal{R}$ は次式により，基底関数の線形結合として一意に表すことができる．

$$f(\underline{x}) = \sum_{\underline{a} \in \{0,1\}^n} f_{\underline{a}} \mathcal{X}_{\underline{a}}(\underline{x}). \tag{10}$$

すべての関数値 $f(\underline{x}), (\underline{x} \in \{0,1\}^n)$ が与えられる場合，次式により係数 $f_{\underline{a}}$ を計算することができる．

$$f_{\underline{a}} = \frac{1}{2^n} \sum_{\underline{x} \in \{0,1\}^n} f(\underline{x}) \mathcal{X}_{\underline{a}}(\underline{x}). \tag{11}$$

式 (11) は多次元高速フーリエ変換 (Fast Fourier Transform:FFT) アルゴリズムであるベクトルラディックス FFT を用いて計算することができる．ベクトルラディックス FFT を用いると，$n2^n$ 回の加算で，すべての $\underline{a} \in \{0,1\}^n$ に対する $f_{\underline{a}}$ を求めることができる [19, p.127]．

一方，一部の関数値 $f(\underline{x}), (\underline{x} \in X)$ のみ与えられる場合に関する補題を以下に載せる．

[補題 1] 任意の $\underline{a} \in X_{\underline{s}}^{\perp}$ に対し，次式が成立する(注4)．

$$\sum_{\underline{c} \in X_{\underline{s}}^{\perp}} f_{\underline{c}} = \frac{1}{2^k} \sum_{\underline{x} \in X} f(\underline{x}) \mathcal{X}_{\underline{a}}(\underline{x}). \tag{12}$$

(証明) $f(\underline{x}) = \sum_{\underline{c} \in \{0,1\}^n} f_{\underline{c}} \mathcal{X}_{\underline{c}}(\underline{x})$ より，次式が成立する．

式 (12) の右辺

$$= \frac{1}{2^k} \sum_{\underline{x} \in X} \sum_{\underline{c} \in \{0,1\}^n} f_{\underline{c}} \mathcal{X}_{\underline{c}}(\underline{x}) \mathcal{X}_{\underline{a}}(\underline{x})$$

$$= \frac{1}{2^k} \sum_{\underline{c} \in \{0,1\}^n} \sum_{\underline{r} \in \{0,1\}^k} f_{\underline{c}} \mathcal{X}_{\underline{c}}(\underline{r}G) \mathcal{X}_{\underline{a}}(\underline{r}G). \tag{13}$$

ここで，$\mathcal{X}_{\underline{c}}(\underline{r}G) = (-1)^{\underline{r}G\underline{c}^T} = \mathcal{X}_{\underline{c}G^T}(\underline{r})$，$\mathcal{X}_{\underline{a}}(\underline{r}G) = (-1)^{\underline{r}G\underline{a}^T} = (-1)^{\underline{r} \cdot \underline{s}^T} = \mathcal{X}_{\underline{s}}(\underline{r})$ を用いると，式 (12) の右辺は更に以下のように変形できる．

式 (12) の右辺

$$= \sum_{\underline{c} \in \{0,1\}^n} f_{\underline{c}} \frac{1}{2^k} \sum_{\underline{r} \in \{0,1\}^k} \mathcal{X}_{\underline{c}G^T}(\underline{r}) \mathcal{X}_{\underline{s}}(\underline{r}). \tag{14}$$

ここで，$\underline{c} \in X_{\underline{s}}^{\perp}$ であれば $\underline{c}G^T = \underline{s}$ が成立するので，式 (9) を用いると次式が得られる．

$$\frac{1}{2^k} \sum_{\underline{r} \in \{0,1\}^k} \mathcal{X}_{\underline{c}G^T}(\underline{r}) \mathcal{X}_{\underline{s}}(\underline{r}) = \begin{cases} 1; & \underline{c} \in X_{\underline{s}}^{\perp}, \\ 0; & \underline{c} \notin X_{\underline{s}}^{\perp}. \end{cases} \tag{15}$$

式 (14) と式 (15) より，式 (12) が証明される． □

[補題 2] $\underline{a} \in X_{\underline{s}}^{\perp}$ とする．このとき，$\forall \underline{b}, (\underline{b} \in X_{\underline{s}}^{\perp}, \underline{b} \neq \underline{a})$ に対し，$f_{\underline{b}} = 0$ が成立するならば，次式により $f_{\underline{a}}$ を求めることができる．

(注4)：ここで，$f_{\underline{a}}, f_{\underline{b}}, (\underline{a}, \underline{b} \in X_{\underline{s}}^{\perp})$ を区別して求めることはできない．実験計画法では，このことを交絡と呼んでいる [12]．

$$f_{\underline{a}} = \frac{1}{2^k} \sum_{\underline{x} \in X} f(\underline{x}) \mathcal{X}_{\underline{a}}(\underline{x}). \quad (16)$$

（証明）補題 1 より明らか. □

3. 実験計画法

はじめに，実験計画法のモデルを載せる．

3.1 実験計画法のモデル

3.1.1 実験形式

真の構造式を $f^*(\underline{x})$ とする．実験を行うとは，入力 $\underline{x} \in \{0,1\}^n$ に対し，出力 $f^*(\underline{x}) \in \mathcal{R}$ を得ることである．実験する $\underline{x}$ の集合を実験計画 X と呼ぶ．

なお，実験計画 X は実験前に決定され，X を一括して実験する一括型実験である．すなわち，他の $\underline{x}$ の実験結果により，実験計画 X が変わることはない．また，実験回数 K は $K = |X| = 2^k$（k は自然数）とする．

3.1.2 構造式クラス

構造式クラスを定義するのに必要な仮定を与えておく．

[仮定 1] 集合 $A, (A \subseteq \{0,1\}^n)$ は，$\underline{a} \in A$ ならば，任意の $\underline{b}, (\underline{b} \in \{0,1\}^n, \underline{b} \sqsubseteq \underline{a})$ に対して $\underline{b} \in A$ が成立する．ただし，$\underline{b} \sqsubseteq \underline{a}$ は，すべての $i, (1 \leq i \leq n)$ において $b_i \leq a_i$ であることを表す. □

[例 2] 集合 A の例

$A = \{0000, 1000, 0100, 0010, 0001, 1100, 1010, 1001\}$. □

A を用い，構造式クラス $\mathcal{F}_A$ を以下で定義する．ただし，本論文ではブール関数と対応(注5)させて考えるので，$x_i \in \{0,1\}, (1 \leq i \leq n)$ としておく(注6).

[定義 4] 構造式クラス $\mathcal{F}_A$

通常の実験計画法では，仮定 1 を満たす，ある A を用いて次式で表現可能な構造式の集合を $\mathcal{F}_A$ とする(注7).

$$f(\underline{x}) = \sum_{\underline{a} \in A} f_{\underline{a}} \mathcal{X}_{\underline{a}}(\underline{x}) + e_{\underline{x}}. \quad (17)$$

ただし，構造式の定義域と値域は $\underline{x} \in \{0,1\}^n, f(\underline{x}) \in \mathcal{R}$ である．また，$e_{\underline{x}}$ は偶然誤差であり，期待値 0，分散 σ^2，独立性を満たす確率変数である [12, p.69]．また一般に，$3 \leq w(\underline{a})$ となる $\underline{a}$ は A に含まれないと前提がおかれている [11, p.156]. □

定義 4 は，構造式クラスが集合 A に依存することを示している．このことから，実験計画は A に依存して決めなければならないことがわかる．

3.1.3 アルゴリズムの入出力

仮定 1 を満たす集合 A と $e_{\underline{x}}$ に関する情報が入力されたもとで，アルゴリズムは，実験計画 X を決定し一括実験を行い，得られた実験結果 $\{(\underline{x}, f^*(\underline{x}))|\underline{x} \in X\}$ から，係数 $f_{\underline{a}}, (\underline{a} \in A)$ を求める．係数が求まれば構造式 $f \in \mathcal{F}_A$ が決まるので，この構造式を出力する．

3.1.4 実験計画の評価基準

実験計画の良さを測る評価基準として，実験後に得られる各係数の不偏推定量の分散の最大値が挙げられる．

後述するように，同じ実験回数のすべての実験計画の中でこの評価基準を最小にする実験計画は直交計画である．このため，統計の実験計画法では直交計画を中心に研究が行われており，これまでに多くの研究成果が得られている [11]〜[14]．次節で直交計画を定義する．

3.2 直交計画

[定義 5] 直交計画 X_A

$v(\underline{a}, \underline{b}) = \{i|a_i \neq 0, 1 \leq i \leq n\} \triangle \{i|b_i \neq 0, 1 \leq i \leq n\}$ とする．ただし，$A \triangle B$ は，集合 A, B の対称差を表す．

$\forall \underline{a}, \underline{b} \in A$ に対して，$\{\underline{g_j}|j \in v(\underline{a}, \underline{b})\}$ が 1 次独立であり，かつ，行ベクトルが 1 次独立となる行列を G_A とする．なお，G_A の行数は集合 A に依存するため，k_A で表す．

このとき，直交計画 X_A は，$k_A \times n$ 行列 G_A を用い，次式で与えられる．

$$X_A = \{\underline{x}|\underline{x} = \underline{r} G_A, \underline{r} \in \{0,1\}^{k_A}\}. \quad (18)$$

ここで，$|X_A| = 2^{k_A}$ である. □

直交計画の例を例 3 に載せる．

[例 3] 直交計画

$n = 4$, $A = \{0000, 1000, 0100, 0010, 0001, 1100, 1010, 1001\}$ の場合を考える．このとき，定義 5 の条件を満たす G_A として，

(注5)：ブール関数の学習問題と実験計画法の用語の関係を説明しておく．学習問題における質問する入力集合 X，関数値 $f(\underline{x})$，インデックス i，ブール変数 x_i，係数 $f_{\underline{a}}$ はそれぞれ，実験計画法における実験計画 X，特性値 $f(\underline{x})$，因子 i，因子 i の水準 x_i，効果 $f_{\underline{a}}$ に対応する．なお，効果 $f_{\underline{a}}$ は，$w(\underline{a}) = 0$ のとき中心効果，$w(\underline{a}) = 1 \wedge a_i = 1$ のとき因子 i の主効果，$w(\underline{a}) = 2 \wedge a_i = 1 \wedge a_j = 1$ のとき因子 i と因子 j の 2 因子交互作用とそれぞれ呼ばれる．また，3 因子以上の交互作用についても 2 因子と同様に定義される．

(注6)：実験計画法においては，これはすべての因子の水準数が 2 の場合に相当するが，一般に各因子の水準数は任意である．

(注7)：仮定 1 は，因子間に低次の交互作用が存在しない場合，それらの因子を含む高次の交互作用は存在しないことを意味している [11, p.206].

$$G_A = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \qquad (19)$$

が挙げられる．このとき，直交計画 $X_A = \{0000, 1000, 0101, 1101, 0011, 1011, 0110, 1110\}$ となる． □

次に直交計画の最適性に関する定理を載せる．

[定理 1] 直交計画の最適性 [12]

定義 4 で与えられる構造式クラスに対し，実験回数 2^{k_A} のすべての実験計画の中で，各係数の不偏推定量の分散の最大値を最小にする実験計画は直交計画 X_A である．このとき，係数 $f_{\underline{a}}$ の不偏推定量 $\hat{f}_{\underline{a}}$ は次式で与えられる．

$$\hat{f}_{\underline{a}} = \frac{1}{2^{k_A}} \sum_{\underline{x} \in X_A} f^*(\underline{x}) \mathcal{X}_{\underline{a}}(\underline{x}). \qquad (20)$$

□

3.3 G_A の構成方法

G_A を決定することは，列（列番号）の集合 $\{1, 2, \cdots, n\}$ から k_A 次元列ベクトルの集合 $\{0,1\}^{k_A}$ への写像 $\xi : \{1, 2, \cdots, n\} \to \{0,1\}^{k_A}$ を決めることと等価である．この写像 ξ を割付けと呼ぶ．

割付けを決定するアルゴリズムは，以下の原則をもとに実行される [13](注8)．

（1） 一列ずつ割付けを行う．

（2） $\underline{\xi(l_1)}, \underline{\xi(l_2)}, \cdots, \underline{\xi(l_m)}$ が既に決定しているとき，$v(\underline{a}, \underline{b}) = \{l_1, l_2, \cdots, l_m, i\}$ となる $\underline{a}, \underline{b} \in A$ が存在すれば，$u_1\underline{\xi(l_1)} + u_2\underline{\xi(l_2)} + \cdots + u_m\underline{\xi(l_m)}, (\forall u_1, u_2, \cdots, u_m \in \{0,1\})$ を列 i に割り付けることはできない(注9)．

原則 (2) は，定義 5 の $\{\underline{g_{\cdot j}} | j \in v(\underline{a}, \underline{b})\}$ が 1 次独立という条件に対応している．

ただし，この原則を用いても，G_A を決定するには多くの計算量を必要とする(注10)．このため，上記の原則をもとに G_A の近似解 G_A^{ap} を探索するアルゴリズムの一例を図 1 に載せる．

図 1 のアルゴリズムについて，以下で説明を行う．

1〜3 行目では割付けを行う列の順序を求めている．これは，$w(\underline{a})$ が大きい $\underline{a}$ の $\{i | a_i \neq 0\}$ に含まれる列から先に決めた方が解が見つかる可能性が高いためである．

次に 4〜15 行目の for 文において $i = s$ であるときを考える．$\underline{\xi(t_1)}, \underline{\xi(t_2)}, \cdots, \underline{\xi(t_{s-1})}$ は既に決定されており，このもとで $\underline{\xi(t_s)}$ を決めることになる．ここで，

入力 A, 出力 G_A^{ap}
$|A| \leq 2^k$ を満たす最小の k を k_A とする．
初期設定：
配列 $S[\underline{z}], (\underline{z} \in \{0,1\}^{k_A} \setminus \{\underline{0}\})$ を用意する．
$A_0 := \{\underline{0}\}$.

```
1:for i := 1 to n do
2:    W(i) := max_{c∈{a|a_i=1, a∈A}} w(c);
3:W(i) が大きい i から順に，t_1, t_2, ··· , t_n とする．
4:for i := 1 to n do
5:    A_i := {a | a_{t_i} = 1, a_{t_{i+1}} = ··· = a_{t_n} = 0, a ∈ A};
6:    for z ∈ {0,1}^{k_A} \ {0} do
7:        S[z] := 0;
8:    for all a ∈ A_i, b ∈ ∪_{0≤j<i} A_j do
9:        v(a, b)\{t_i} に含まれる要素を l_1, l_2, ··· , l_m
10:       とする．
11:       for all (u_1, u_2, ··· , u_m) ∈ {0,1}^m do
12:           z := u_1 ξ(l_1) + u_2 ξ(l_2) + ··· + u_m ξ(l_m);
13:           S[z] := 1;
14:   if S[z] = 0 となる z が存在する．
15:       then ξ(t_i) := z;
16:   else k_A := k_A + 1 として，初期設定へ．
17:行列 [ξ(1) ξ(2) ··· ξ(n)] を出力し，終了する．
```

図 1 G_A^{ap} を出力するアルゴリズム
Fig. 1 Algorithm to output G_A^{ap}.

原則 (2) の条件を満たすように割付けを行う必要があるため，$t_s \in v(\underline{a}, \underline{b})$ かつ $v(\underline{a}, \underline{b}) \subseteq \{t_1, t_2, \cdots, t_s\}$ となる $\underline{a}, \underline{b}$ を見つける必要がある．8 行目で，このような $v(\underline{a}, \underline{b})$ を列挙している．そして，11〜12 行目で原則 (2) より t_s を割り付けることができない k_A ビット 2 進数の列ベクトルを探している．t_s を割り付けることができない列ベクトル $\underline{z}$ に対して，12 行目で $S[\underline{z}] = 1$ としている．これより，13〜14 行目で t_s を割り付ける $\underline{z}$ は原則 (2) の条件を満たしていることがわかる．

15 行目について説明する．$|A| \leq 2^k$ を満たす最小の k を k_A としても，定義 5 の条件を満たす行列がいつも構成できるとは限らない．このような場合，実験計画法 [14, p.119] と同様に行列の行数を 15 行目で増やしている．最後に，16 行目で行列を出力し，終了している．

(注8)：文献 [13] では有限射影幾何を用いた割付けを行っている．詳細は参考文献を参照されたい．
(注9)：ここでの + は式 (1) で定義したベクトルの要素間の排他的論理和である．
(注10)：実験計画の分野では，効率の良い探索の研究はあまり見られず，上記の原則をもとに人間が関与した探索で G_A を求めることが多い．これは，統計学の実務家の場合，実験のしやすい順序や誤差項の独立性などを考慮して決めるため，探索条件が複雑になるためと思われる．

なお，図 1 のアルゴリズムで出力される行列 G_A^{ap} は，行ベクトルが 1 次独立とならない可能性がある．この場合，G_A^{ap} を用い式 (18) から構成される実験計画の大きさは $2^{rank(G_A^{ap})}$ となる．ここで，$rank(G_A^{ap})$ は G_A^{ap} の階数である．

4. 質問する入力集合に直交計画を用いたブールドメイン上の実数値関数の学習

まず，瀧本らにより得られたブールドメイン上の関数に対するサンプリング定理を載せる．

[命題 1] サンプリング定理 [1]

真の実数値関数 $f^*(\underline{x})$ が次式で与えられる場合を考える．

$$f^*(\underline{x}) = \sum_{\underline{a}\in A'} f_{\underline{a}}^* \mathcal{X}_{\underline{a}}(\underline{x}). \tag{21}$$

ただし，$A' = \{\underline{a} | w(\underline{a}) \leq t, \underline{a} \in \{0,1\}^n\}$ である．ここで，双対符号(注11)の最小距離が $2t+1$ 以上となる線形符号 Y を用いると，係数 $f_{\underline{a}}^*$ は次式で計算される．

$$f_{\underline{a}}^* = \frac{1}{2^k} \sum_{\underline{x}\in Y} f^*(\underline{x}) \mathcal{X}_{\underline{a}}(\underline{x}). \tag{22}$$

□

一方，直交計画に関する以下の定理が得られる．

[定理 2] 真の実数値関数 $f^*(\underline{x})$ が次式で与えられる場合を考える．

$$f^*(\underline{x}) = \sum_{\underline{a}\in A} f_{\underline{a}}^* \mathcal{X}_{\underline{a}}(\underline{x}). \tag{23}$$

このとき，係数 $f_{\underline{a}}^*$ は次式で計算される．

$$f_{\underline{a}}^* = \frac{1}{2^{k_A}} \sum_{\underline{x}\in X_A} f^*(\underline{x}) \mathcal{X}_{\underline{a}}(\underline{x}). \tag{24}$$

(証明) $\underline{a} \in X_{\underline{s}}^{\perp}, (\underline{a} \in A)$ とする．定義 5 より，$\forall \underline{c} \in A, (\underline{a} \neq \underline{c})$ に対し，$\{\underline{g_{\cdot j}} | j \in v(\underline{a}, \underline{c})\}$ が 1 次独立である．これより，

$$G_A(\underline{a}+\underline{c})^T \neq \underline{0}, \tag{25}$$

すなわち，

$$G_A\underline{a}^T \neq G_A\underline{c}^T, \tag{26}$$

が成立する．

式 (5) と式 (26) により，$\underline{c} \notin X_{\underline{s}}^{\perp}$ が成立する．これより，$\forall \underline{b}, (\underline{b} \in X_{\underline{s}}^{\perp}, \underline{b} \neq \underline{a})$ に対し，$f_{\underline{b}} = 0$ が成立する．

以上と補題 2 より，定理 2 が証明される． □

式 (23) で表現可能な実数値関数の集合を $\mathcal{F}_A^{\mathcal{R}}$ としておく．

次に，定理 2 と命題 1 の関係に関する系 1 を与えるために必要な補題を与えておく．

[補題 3] [12, p.156, 定理 1]

G の任意の $2t$ 列が 1 次独立で 1 次従属な $2t+1$ 列が存在するための必要十分条件は，G を検査行列としてもつ線形符号の最小距離が $2t+1$ に等しいことである．

(証明) $G = [\underline{g_{\cdot 1}}\ \underline{g_{\cdot 2}}\ \cdots\ \underline{g_{\cdot n}}]$ と書き，G を検査行列としてもつ線形符号を Y とする．$\underline{x} = (x_1, x_2, \cdots, x_n)$ に対して，$G\underline{x}^T = \underline{0}$ は，

$$\underline{g_{\cdot 1}}x_1 + \underline{g_{\cdot 2}}x_2 + \cdots + \underline{g_{\cdot n}}x_n = \underline{0}, \tag{27}$$

と書くことができるので，$\underline{x} \in Y$ と式 (27) は同値である．したがって，Y の中にハミング重み $w(\underline{x}) = 2t+1$ なる $\underline{x}$ が存在することと，$\underline{g_{\cdot 1}}, \underline{g_{\cdot 2}}, \cdots, \underline{g_{\cdot n}}$ のうち 1 次従属な $2t+1$ 列が存在することは同値である．また，$Y \setminus \{\underline{0}\}$ の中にハミング重み $w(\underline{x}) = 2t$ なる $\underline{x}$ が存在しないことと，$\underline{g_{\cdot 1}}, \underline{g_{\cdot 2}}, \cdots, \underline{g_{\cdot n}}$ の任意の $2t$ 列が 1 次独立であることは同値である．

以上の結果と，線形符号 Y の最小距離と $w(\underline{x}), (\underline{x} \in Y \setminus \{\underline{0}\})$ の最小値が等しいことより，補題 3 は証明される． □

このとき，以下の系が成立する．

[系 1] 定理 2 は命題 1 を特殊な場合として含む．

(証明) $A = \{\underline{c} | \underline{c} \in \{0,1\}^n, w(\underline{c}) \leq t\}$ である場合について，定理 2 を考える．このとき，定義 5 より，G_A の任意の $2t$ 列が 1 次独立でなければならない．更に，補題 3 より，G_A を検査行列としてもつ線形符号 (X_A の双対符号) の最小距離が $2t+1$ 以上でなければならないことがわかる．これより，定理 2 は命題 1 を特殊な場合として含むことがわかる． □

命題 1 に対して定理 2 の優位性を示す例を載せる．

[例 4] $n = 15$, $A = \{\underline{a} | w(\underline{a}) \leq 1\} \cup \{1100000000000000, 001100000000000\}$ である場合を考える．このとき，$t = \max_{\underline{a}\in A} w(\underline{a}) = 2$ より，$A' = \{\underline{a} | w(\underline{a}) \leq 2\}$ に対して命題 1 を用いると，Y(注12)は次式で与えられる．

$$Y = \{\underline{x} | \underline{x} = \underline{r}G_{A'},\ \underline{r} \in \{0,1\}^8\}, \tag{28}$$

(注11)：線形符号 Y の生成行列を G とする．このとき，G を検査行列としてもつ線形符号を Y の双対符号と呼ぶ [20].

(注12)：Y の双対符号は最小距離が 5 となる 2 重誤り訂正 BCH 符号である．

ただし，$G_{A'}$ は次式で与えられる．

$$G_{A'} = \begin{bmatrix} 1&0&0&0&0&0&0&0&1&0&0&0&1&0&1 \\ 0&1&0&0&0&0&0&0&1&1&0&0&1&1&1 \\ 0&0&1&0&0&0&0&0&1&1&1&0&1&1&0 \\ 0&0&0&1&0&0&0&0&0&1&1&1&0&1&1 \\ 0&0&0&0&1&0&0&0&1&0&1&1&0&0&0 \\ 0&0&0&0&0&1&0&0&0&1&0&1&1&0&0 \\ 0&0&0&0&0&0&1&0&0&0&1&0&1&1&0 \\ 0&0&0&0&0&0&0&1&0&0&0&1&0&1&1 \end{bmatrix}. \tag{29}$$

一方，定理 2 を用いると，X_A は次式で与えられる．

$$X_A = \{\underline{x} | \underline{x} = \underline{r} G_A,\ \underline{r} \in \{0,1\}^5\}, \tag{30}$$

ただし，G_A は次式で与えられる．

$$G_A = \begin{bmatrix} 1&0&0&0&0&1&1&1&0&0&0&0&0&1&1 \\ 0&1&0&0&0&0&0&0&1&1&1&0&0&1&1 \\ 0&0&1&0&0&1&0&0&1&0&0&1&0&1&0 \\ 0&0&0&1&0&0&1&0&0&1&0&0&1&0&1 \\ 0&0&0&0&1&0&0&1&0&0&1&1&1&0&0 \end{bmatrix}. \tag{31}$$

以上より，$|X_A| = 2^5, |Y| = 2^8$ であり，$|X_A| < |Y|$ が成立する．これより，命題 1 よりも定理 2 を用いた方がよい場合が存在することがわかる．

また，G_A と $G_{A'}$ を比べてみると，$G_{A'}$ は任意の 4 列が 1 次独立であるのに対し，G_A は一部の 4 列が 1 次独立であることがわかる． □

次に，式 (24) の計算について考える．$\underline{x} = \underline{r} G_A$ より，$\underline{x}$ は $\underline{r}$ の関数なので，特に $\underline{x_r}$ と表記する．また，$\underline{s_a} = G_A \underline{a}^T$ とおく．このとき，式 (24) は以下のように展開できる．

$$\begin{aligned} f^*_{\underline{a}} &= \frac{1}{2^{k_A}} \sum_{\underline{x_r} \in X_A} f^*(\underline{x_r}) \mathcal{X}_{\underline{a}}(\underline{x_r}) \\ &= \frac{1}{2^{k_A}} \sum_{\underline{r} \in \{0,1\}^{k_A}} f^*(\underline{x_r}) (-1)^{\underline{x_r} \cdot \underline{a}^T} \\ &= \frac{1}{2^{k_A}} \sum_{\underline{r} \in \{0,1\}^{k_A}} f^*(\underline{x_r}) (-1)^{\underline{r} \cdot \underline{s_a}^T} \\ &= \frac{1}{2^{k_A}} \sum_{\underline{r} \in \{0,1\}^{k_A}} f^*(\underline{x_r}) \mathcal{X}_{\underline{s_a}}(\underline{r}). \end{aligned} \tag{32}$$

このとき，式 (32) を式 (11) と比べてみると，同じ計算であることがわかる．質問する入力の個数を $K_A = |X_A| = 2^{k_A}$ としておく．このとき，式 (32) の計算にベクトルラディックス FFT を用いると，$K_A \log_2 K_A$ 回の加算で，すべての $\underline{a} \in A$ に対する $f^*_{\underline{a}}$ を求めることができる．

5. 質問する入力集合に直交計画を用いたブール関数の学習

5.1 質問からの学習モデル

5.1.1 質問形式

本論文では所属性質問からの学習問題を対象とするので，所属性質問を定義しておく．

真のブール関数を $f^*(\underline{x})$ とする．所属性質問をするとは，オラクルへの入力 $\underline{x} \in \{0,1\}^n$ に対し，出力 $f^*(\underline{x}) \in \{-1,+1\}$ を得ることである．以後，所属性質問を略して質問と呼ぶ．

質問の形式は，質問する入力 $\underline{x}$ の集合 X を一括して質問する一括型質問である．すなわち，他の $\underline{x}$ の質問結果により，X が変わることはない．このため，X を一括質問入力集合と呼ぶこととする．また，質問する入力の個数 K は $K = |X| = 2^k$ (k は自然数) とする．

5.1.2 概念クラス(注13)

概念クラスを以下で与える．

[定義 6] 概念クラス $\mathcal{F}_A$(注14)

仮定 1 を満たす，ある A を用い次式で表現可能なブール関数の集合を $\mathcal{F}_A$ とする．

$$f(\underline{x}) = \sum_{\underline{a} \in A} f_{\underline{a}} \mathcal{X}_{\underline{a}}(\underline{x}). \tag{33}$$

ただし，真のブール関数 f^* は，$f^* \in \mathcal{F}_A$ とする． □

5.1.3 アルゴリズムの入出力

本論文で対象とする質問からの学習では，入力として，仮定 1 を満たす集合 A が与えられる．そして，一括質問入力集合 X を決定し，一括して質問する．得られた質問結果 $\{(\underline{x}, f^*(\underline{x})) | \underline{x} \in X\}$ から，概念クラス $\mathcal{F}_A$ に含まれるブール関数の中から一つ選択し，出力する(注15)．

本論文では一括質問入力集合に直交計画 X_A を用い

(注13)：本論文ではブール関数の学習を扱うため，対象クラスをブール関数のクラスに限定するが，5. の結果は実数値関数のクラスに対しても成立する．

(注14)：人間が用いる規則の集合を考える場合，一般には単純な概念がよく用いられるといわれている．例えば，情報検索において一般のユーザは複雑な論理式を理解できないことが指摘されている [18]．このとき，概念の複雑さを測る基準にも様々なものが考えられる．例えば，オッカムアルゴリズム [15] では概念のサイズにより複雑さを測り，Quinlan ら [16] は記述長で複雑さを測っている．$\mathcal{F}_A$ は関連変数の個数が小さいブール関数ほど単純なブール関数であるととらえたときによく用いられるブール関数の集合となっている．

(注15)：本論文では，仮説の表現形については考えない．

入力　直交計画 X_A，出力　ブール関数 $f(\underline{x})$

1: X_A を一括して質問し，質問結果
　$\{(\underline{x}, f^*(\underline{x}))|\underline{x} \in X_A\}$ を得る．
2: 式 (34) を計算することで係数 $f^*_{\underline{a}}, (\underline{a} \in A)$ を求める．
3: $f(\underline{x}) = \sum_{\underline{a}\in A} f^*_{\underline{a}} \mathcal{X}_{\underline{a}}(\underline{x})$.
4: ブール関数 $f(\underline{x})$ を出力し，終了する．

図 2　提案アルゴリズム
Fig. 2　Proposed algorithm.

る．そこで次節で，ブール関数の学習に直交計画を用いるときの性質を与えておく．

5.2 直交計画 X_A を質問することでブール関数を学習するときの性質

[定理 3]　$f^*(\underline{x}) \in \mathcal{F}_A$ であれば，係数 $f^*_{\underline{a}}$ は次式により計算される．

$$f^*_{\underline{a}} = \frac{1}{2^{k_A}} \sum_{\underline{x}\in X_A} f^*(\underline{x}) \mathcal{X}_{\underline{a}}(\underline{x}). \qquad (34)$$

(証明) 定義 6 で与えられる概念クラス $\mathcal{F}_A$ は $\mathcal{F}_A^{\mathcal{R}}$ の部分クラスである．このため，定理 2 より，式 (34) が成立する．　□

5.3 提案アルゴリズム

直交計画 X_A が与えられたもとで，ブール関数を出力する提案アルゴリズムを図 2 に載せる．

提案アルゴリズムの出力に関する定理を以下に載せる．

[定理 4]　$f^*(\underline{x}) \in \mathcal{F}_A$ である場合，提案アルゴリズムの出力するブール関数 $f(\underline{x})$ は $f^*(\underline{x})$ である．

(証明)　定理 3 より明らか．　□

5.4 アルゴリズムの計算量

1 行目の計算量は，$|X_A| = K_A$ より，$O(K_A)$ である．2 行目であるが，式 (34) の計算も式 (32) と同様にベクトルラディックス FFT を用いることができる．これより，$K_A \log_2 K_A$ 回の加算で，すべての $\underline{a} \in A$ に対する $f^*_{\underline{a}}$ を求めることができる．これより，2 行目の計算量は $O(K_A \log_2 K_A)$ であることがわかる．また，3 行目の計算量は $O(K_A)$ である．以上より，提案アルゴリズムに必要な計算量は，$O(K_A \log_2 K_A)$ となる．

6. 真のブール関数が概念クラスに含まれない場合

前章までは，真のブール関数が概念クラスに含まれる仮定のもとで話を進めてきたが，常にこの仮定を置くことができるとは限らない．そこで，真のブール関数が概念クラスに含まれなかった場合に図 2 の提案アルゴリズムを適用するとどうなるかを説明する．このとき，一括質問入力集合に関して真のブール関数と同じ出力をするブール関数が概念クラスに含まれれば提案アルゴリズムはこのブール関数を出力し，含まれなければ 4 行目で出力される関数 $f(\underline{x})$ はブール関数とならない．ブール関数が出力されない場合，概念クラス以外から質問結果に無矛盾なブール関数を探索することが考えられるが，このとき，どのブール関数を選ぶかの基準が必要となる．

ところで近年，概念の事前確率が与えられる決定理論による学習モデルについて盛んに研究が行われている [7]〜[10]．この学習モデルで質問からのブール関数の学習を考えた場合，無矛盾なブール関数の中で事前確率が最大なブール関数を選択することが最適な決定となる．

決定理論による学習モデルに提案アルゴリズムを適用し，出力がブール関数とならない場合，無矛盾で事前確率が最大のブール関数の探索には式 (12) を用いることで効率良く探索できる．この詳細については別稿で発表予定である．

7. むすび

本論文ではまず，実験計画法と質問からのブール関数の学習との関係を明らかにし，ブールドメイン上の実数値関数に対するサンプリング定理が直交計画の特別な場合として与えられることを示した．次に，今までの計算論的学習理論では扱われていない概念クラスを提案し，このクラスに真のブール関数が含まれる場合，質問する入力集合を直交計画とすることで真のブール関数を必ず出力できることを示した．更に，多次元高速フーリエ変換アルゴリズムを用いることで，質問後にブール関数を決定する際の計算量を削減できることを示した．

本論文では明らかにできなかったが，k_A と集合 A の関係を明らかにすることは重要である．また，G_A の構成については近似解を探索するアルゴリズムの一例を載せることにとどめたが，G_A を見つける問題は重要であり，今後の課題である．

謝辞　本研究を進めるにあたり，有益な御討論や御指摘を頂いた早稲田大学の中澤真氏並びに松嶋・平澤両研究室各位に深く感謝致します．また，直交計画の

構成法に関する貴重な御助言を頂いた群馬大学の関庸一教授，非常に有益な御示唆と御指摘を頂いた査読者の方々に深く感謝致します．本研究の一部は，文部科学省科学研究費基盤研究（C）(2)（課題番号 12650400），横浜商科大学学術研究会研究助成金，早稲田大学特定課題研究助成費（2001A-570）の援助による．

文　献

[1] 瀧本英二，丸岡　章，“ブールドメイン上の関数に対するサンプリングの定理，” 信学技報，COMP97-56, 1997.
[2] 天野一幸，瀧本英二，“ブール関数のフーリエ変換とその応用，” 信学誌，vol.82, no.12, pp.1270–1272, 1999.
[3] E. Kushilevitz and Y. Mansour, “Learning decision trees using the Fourier spectrum,” SIAM J. Comput., vol.22, no.6, pp.1331–1348, 1993.
[4] N. Linial, Y. Mansour, and N. Nisan, “Constant depth circuits, Fourier transform and learnability,” J. ACM., vol.40, no.3, pp.607–620, 1993.
[5] L.G. Valiant, “A theory of the learnable,” Commun. ACM, vol.27, pp.1134–1142, 1984.
[6] D. Angluin, “Queries and concept learning,” Machine Learning, vol.2, no.4, pp.319–342, 1988.
[7] D. Haussler, “Decision theoretic generalizations of the PAC learning model,” 1st International Workshop, ALT'90, pp.21–41, 1990.
[8] G. Paass and J. Kindermann, “Bayesian query construction for neural network models,” Advances in Neural Information Processing Systems 7, pp.443–450, MIT Press, 1995.
[9] 松嶋敏泰，“帰納・演繹推論と予測—決定理論による学習モデル，” 1998 年情報論的学習理論ワークショップ予稿集，pp.1–8, 1998.
[10] 浮田善文，松嶋敏泰，平澤茂一，“質問からの学習問題の決定理論による定式化に関する一考察，” 情処学論，vol.39, no.11, pp.2937–2948, Nov. 1998.
[11] 奥野忠一，芳賀敏郎，実験計画法，培風館，東京，1969.
[12] 高橋磐郎，組合せ理論とその応用，岩波全書 316，東京，1979.
[13] R. Fuji-Hara, “On automatical construction for orthogonal designs of experiments,” Rep. Stat. Appl. Res., JUSE, vol.25, no.1, pp.13–25, 1978.
[14] 永田　靖，入門実験計画法，日科技連，東京，2000.
[15] A. Blumer, A. Ehrenfeucht, D. Haussler, and M.K. Warmuth, “Learnability and the Vapnik-Chervonenkis dimension,” J. Assoc. Comput. Mach., vol.36, no.4, pp.929–965, 1989.
[16] J.R. Quinlan and R.L. Rivest, “Inferring decision tree using the minimum description length principle,” Inf. Comput, vol.80, no.1, pp.227–248, 1989.
[17] 松嶋敏泰，平澤茂一，“MDL の帰納推論への応用，” 人工知能誌，vol.7, no.4, pp.615–621, 1992.
[18] 徳永健伸，情報検索と言語処理，東京大学出版会，東京，1999.
[19] 佐川雅彦，貴家仁志，高速フーリエ変換とその応用，昭晃堂，1992.
[20] 平澤茂一，西島利尚，符号理論入門，培風館，東京，1999.

（平成 13 年 11 月 26 日受付，14 年 6 月 10 日再受付，12 月 12 日最終原稿受付）

浮田　善文　（正員）

平 6 早大・理工・工業経営卒．平 8 同大大学院修士課程了．同年，同大大学院理工学研究科博士後期課程入学．平 10 同大同学部経営システム工学科助手．平 13 横浜商科大学講師，現在に至る．機械学習，特に質問からの学習に関する研究に従事．情報処理学会等各会員．

松嶋　敏泰　（正員）

昭 53 早大・理工・工業経営卒．昭 55 同大大学院修士課程了．同年，日本電気（株）入社．昭 61 早大・理工学研究科・博士後期課程入学．平元横浜商科大学講師．平 3 同大助教授．平 4 早大・理工学部・工業経営学科（現在経営システム工学科）助教授，平 9 同大教授，現在に至る．知識情報処理及び情報理論とその応用に関する研究に従事．工博．平 13 ハワイ大学客員研究員．IEEE，情報理論とその応用学会，人工知能学会，情報処理学会，OR 学会，日本経営工学会等各会員．

平澤　茂一　（正員：フェロー）

昭 36 早大・理工・数学卒．昭 38 同電気通信卒．同年三菱電機（株）入社．昭 56 早大・理工・工業経営学科（現在経営システム工学科）教授，現在に至る．情報理論とその応用及びデータ伝送方式の研究，並びに計算機応用システムの開発などに従事．工博．昭 54，平 14 UCLA 計算機科学科客員研究員．昭 60 ハンガリー科学アカデミー，昭 61 伊トリエステ大学客員研究員．平 5 本会小林記念特別賞，業績賞受賞．IEEE Fellow，情報理論とその応用学会，人工知能学会，情報処理学会，OR 学会，日本経営工学会等各会員．

PAPER Special Section on Information Theory and Its Applications

An Improved Method of Reliability-Based Maximum Likelihood Decoding Algorithms Using an Order Relation among Binary Vectors*

Hideki YAGI[†a)], *Student Member*, Manabu KOBAYASHI[††], Toshiyasu MATSUSHIMA[†], *Members*, *and* Shigeichi HIRASAWA[†], *Fellow*

SUMMARY Reliability-based maximum likelihood decoding (MLD) algorithms of linear block codes have been widely studied. These algorithms efficiently search the most likely codeword using the generator matrix whose most reliable and linearly independent k (dimension of the code) columns form the identity matrix. In this paper, conditions for omitting unnecessary metrics computation of candidate codewords are derived in reliability-based MLD algorithms. The proposed conditions utilize an order relation of binary vectors. A simple method for testing if the proposed conditions are satisfied is devised. The method for testing proposed conditions requires no real number operations and, consequently, the MLD algorithm employing this method reduces the number of real number operations, compared to known reliability-based MLD algorithms.

key words: *maximum likelihood decoding, soft decision decoding, reliability measure, linear block codes, order relation*

1. Introduction

Maximum likelihood decoding (MLD) minimizes the block error probability of decoding when each codeword is equally likely to be transmitted. Since the complexity for performing MLD of block codes becomes impractically large as the code length becomes larger, many researchers have been devoted to reduce the complexity of MLD algorithms.

There are, in general, two types of efficient MLD algorithms. The first type is trellis-based MLD algorithms such as the Viterbi algorithm [11] or the recursive MLD algorithm [5]. Trellis-based MLD algorithms are "breadth-first" search algorithm [7] which mainly reduces the maximum number of computations. The latter type of efficient MLD algorithms is reliability-based MLD algorithms which iteratively generate candidate codewords. Reliability-based MLD algorithms are "depth-first" search algorithm which reduces the average number of computations and they are known to be efficient at moderate or high signal to noise ratio (SNR). One of well-known reliability-based MLD algorithms uses the bounded distance decoder (BDD) [1], [8], [10], [16] to generate candidate codewords. Objects of MLD algorithms using the BDD are codes with algebraic structure such as the BCH codes or the Goppa codes. Another reliability-based MLD algorithm uses the permuted generator matrix (PGM) of the code [6], [7], [12], [14], [17] to generate candidate codewords (Sub-optimum versions are found in [2]–[4], [13], [15]). MLD algorithms using the PGM are applicable to any binary linear block codes [15]. In this paper, we will focus on the reliability-based MLD algorithms using the PGM and we will call them, simply, the reliability-based MLD algorithms.

In the reliability-based MLD algorithms, test error patterns are iteratively generated to construct candidate codewords. Each time a new candidate codeword is constructed, metrics computation of it is carried out. In these algorithms, implicitly or explicitly, a sufficient condition for the optimality is tested. A sufficient condition for eliminating unnecessary test error patterns is also applied before they are encoded with the PGM. As a result, the reliability-based MLD algorithms require the relatively small number of candidate codewords and of their metrics computations. At low to moderate SNRs and for long codes, however, the number of candidate codewords for which the algorithm searches is still large. Therefore, the number of real number additions, subtractions and comparisons (hereafter, they will be called real number operations) is impractically large as the number of computing metrics of candidate codewords increases. We note that the total number of real number operations is one of the typical measures to evaluate the efficiency of MLD algorithms [4], [7].

In order to reduce the complexity of the reliability-based decoding algorithm where a large number of iterations (one iteration step consists of constructing candidate codewords and computing their metrics) are processed, we can consider the following two approaches: (1) reducing the number of iterations and (2) reducing the complexity for each iteration step. In this paper, we will concentrate ourselves on the latter approach. First, we define an order relation among binary vectors. Then we derive a sufficient condition for omitting unnecessary metrics computations of candidate codewords by using the defined order relation. A simple method for testing if the proposed conditions are satisfied is devised so that the test of proposed

Manuscript received January 19, 2004.
Manuscript revised June 3, 2004.
Final manuscript received June 28, 2004.

[†]The authors are with the Department of Industrial and Management Systems Engineering, School of Science and Engineering, Waseda University, Tokyo, 169-8555 Japan.
[††]The author is with the Department of Information Science, School of Engineering, Shonan Institute of Technology, Kanagawa, 251-8511 Japan.
a) E-mail: yagi@hirasa.mgmt.waseda.ac.jp

*The content of this work is partially based on [18].

conditions is implemented with increments of an integer or shift operations. In accordance with more likely codewords obtained, an *adaptive procedure* of the proposed condition, in which the codeword referenced by it is adaptively altered, is considered to make the proposed condition more effective. Testing the proposed conditions requires no real number operations and, as a result, the total number of real number operations for MLD is reduced. Finally, we show the effectiveness of the proposed conditions while the decoding algorithm employing the proposed conditions has no degradation of the error performance.

This paper is organized as follows. In Sect. 2, the general framework of the reliability-based MLD algorithm is briefly reviewed as a preliminary. In Sect. 3, a sufficient condition for omitting unnecessary metrics computation of candidate codewords is derived. Then, an adaptive procedure for implementing the proposed method is presented. Some simulation results are shown in Sect. 4 to demonstrate the effectiveness of the proposed conditions and concluding remarks are stated in Sect. 5

2. The Reliability-Based MLD Algorithm

2.1 Preliminary

For integers j_1 and j_2 such that $j_1 \leq j_2$, let $[j_1, j_2]$ denote the set of positive integers from j_1 to j_2. For binary vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_\alpha)$ of finite length α, let $w_H(\boldsymbol{x})$ and $supp(\boldsymbol{x})$ be, respectively, the Hamming weight of $\boldsymbol{x}$ and the support of $\boldsymbol{x}$ defined as $supp(\boldsymbol{x}) = \{j \,|\, x_j = 1\}$. For a set X, let $|X|$ be the cardinality of X.

Let $\mathcal{V}^n$ denote a set of all binary n-dimensional vectors. Let $C \subseteq \mathcal{V}^n$ be a binary linear (n, k, d) block code with length n, dimension k and minimum distance d. Let G be a generator matrix of C. Assume that each codeword $\boldsymbol{c} = (c_1, c_2, \ldots, c_n) \in C$ has equal probability to be transmitted over the Additive White Gaussian Noise (AWGN) channel with the signal to noise ratio (SNR) E_b/N_0 [dB]. The detector projects the received sequence $\boldsymbol{r} = (r_1, r_2, \ldots, r_n) \in \mathcal{R}^n$ into a sequence $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_n) \in \mathcal{R}^n$ such that $\theta_j = \ln \frac{P(r_j|c_j=0)}{P(r_j|c_j=1)}$, $j \in [1, n]$, and delivers $\boldsymbol{\theta}$ into the decoder. Let $\boldsymbol{z} = (z_1, z_2, \ldots, z_n) \in \mathcal{V}^n$ be the hard decision received sequence of $\boldsymbol{\theta}$ such that

$$z_j = \begin{cases} 0, & \text{if } \theta_j \geq 0; \\ 1, & \text{otherwise.} \end{cases} \tag{1}$$

The decoder estimates a transmitted codeword from both $\boldsymbol{\theta}$ and $\boldsymbol{z}$. For $j \in [1, n]$, an error probability of the symbol z_j, $P(z_j \neq c_j|r_j)$, is smaller as the value $|\theta_j|$ becomes larger. Therefore, we call $|\theta_j|$ *reliability measure* of j-th symbol.

For any $\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \mathcal{V}^n$, let $L(\boldsymbol{x})$ be the *reliability loss* with respect to $\boldsymbol{z}$ defined as

$$L(\boldsymbol{x}) = \sum_{j=1}^{n} (x_j \oplus z_j)|\theta_j|, \tag{2}$$

where $\oplus$ represents the exclusive OR operation. For $\boldsymbol{x} \in \mathcal{V}^n$, $L(\boldsymbol{x})$ is also known as *correlation discrepancy* [9], [10], [15]. For a subspace X of $\mathcal{V}^n$, let $\underline{L}[X]$ be defined as

$$\underline{L}[X] = \min_{\boldsymbol{x} \in X} L(\boldsymbol{x}). \tag{3}$$

Then $L(\boldsymbol{c}_{ML}) = \underline{L}[C]$ if and only if $\boldsymbol{c}_{ML} \in C$ is the most likely (ML) codeword [9], [10], [12]. i.e., a codeword which has the smallest reliability loss is the closest codeword from $\boldsymbol{\theta}$.

2.2 General Framework of the Reliability-Based MLD Algorithm

After receiving $\boldsymbol{\theta}$, the decoder reorders positions of $\boldsymbol{\theta}$ in the non-increasing order of reliability measure. We denote the resultant sequence with $\overline{\boldsymbol{\theta}} = \lambda(\boldsymbol{\theta})$ where λ is the permutation function from $\boldsymbol{\theta}$ to $\overline{\boldsymbol{\theta}}$. i.e., $|\overline{\theta}_{j_1}| \geq |\overline{\theta}_{j_2}|$, $1 \leq j_1 < j_2 \leq n$. Let G' be the column-permuted generator matrix given by the same ordering of $\overline{\boldsymbol{\theta}}$.

For a location set $X \subseteq [1, n]$, let G'_X be the $k \times |X|$ matrix which consists of columns of G' over X. Define

$$\mathcal{M} = \arg\max_{X} \Big\{ \sum_{j \in X} |\overline{\theta}_j| \,\Big|\, |X| = k, rank(G'_X) = k \Big\}. \tag{4}$$

Then $\mathcal{M}$ is called the k *most reliable and linearly independent* (MRI) positions, i.e., the sum of reliability measures of the k MRI positions are the largest among that of any other k linearly independent positions. For G', the elementary row operations are carried out so that the k MRI columns form the identity matrix. The resultant generator matrix is denoted with $\overline{G}$. Let $\overline{C}$ be the code given by $\overline{G}$ which is equivalent to C. Furthermore, let $\overline{\boldsymbol{z}} = \lambda(\boldsymbol{z})$. Let $\overline{\mathcal{V}}^n$ denote the set of binary vectors such that $\overline{\mathcal{V}}^n = \{\overline{\boldsymbol{x}} = \lambda(\boldsymbol{x}) \,|\, \boldsymbol{x} \in \mathcal{V}^n\}$, i.e., any $\overline{\boldsymbol{x}} \in \overline{\mathcal{V}}^n$ is permuted in the non-increasing order of reliability.

Define that $\boldsymbol{u} = (u_1, u_2, \ldots, u_k) \in \{0, 1\}^k$ consists of the k MRI symbols of $\overline{\boldsymbol{z}}$ in non-increasing order of reliability. The sequence $\boldsymbol{u}$ is regarded as an information sequence and the decoder constructs the initial codeword $\overline{\boldsymbol{c}}_0$ by $\overline{\boldsymbol{c}}_0 = \boldsymbol{u}\overline{G}$. Remark that $\overline{\boldsymbol{c}}_0$ is the ML codeword if $\overline{\boldsymbol{c}}_0 = \overline{\boldsymbol{z}}$ [10], [12]. If $L(\overline{\boldsymbol{c}}_0) > 0$, the decoder iteratively constructs candidate codewords by $\overline{G}$ and searches the ML codeword which minimizes Eq. (2).

Definition 1: For $0 \leq i \leq 2^k$, k-dimensional vector $\boldsymbol{t}_i \in \{0, 1\}^k$ is called i-th *test error pattern*. A codeword $\overline{\boldsymbol{w}}_i = (\overline{w}_{i,1}, \overline{w}_{i,2}, \ldots, \overline{w}_{i,n}) = \boldsymbol{t}_i\overline{G}$ is called a *test error codeword* which gives a candidate codeword $\overline{\boldsymbol{c}}_i = \overline{\boldsymbol{c}}_0 \oplus \overline{\boldsymbol{w}}_i$. A candidate codeword $\overline{\boldsymbol{c}}_i$ (or a test error codeword $\overline{\boldsymbol{w}}_i$) is said to be *better* than $\overline{\boldsymbol{c}}_{i'}$ (or $\overline{\boldsymbol{w}}_{i'}$), if and only if $L(\overline{\boldsymbol{c}}_i) < L(\overline{\boldsymbol{c}}_{i'})$. For a subset $\overline{C}'$ of $\overline{C}$, a candidate codeword $\overline{\boldsymbol{c}}_i$ and a test error codeword $\overline{\boldsymbol{w}}_i$ are said to be the *best* in $\overline{C}'$ if and only if $L(\overline{\boldsymbol{c}}_i) = \underline{L}[\overline{C}']$.

Let $\boldsymbol{t}_0 = 0^k$ where 0^k is k-dimensional all zero vector. Then $\boldsymbol{t}_0$ can be regarded as the test error pattern corresponding to the initial codeword $\overline{\boldsymbol{c}}_0$ since $\overline{\boldsymbol{c}}_0 = \overline{\boldsymbol{c}}_0 \oplus \boldsymbol{t}_0\overline{G} = \overline{\boldsymbol{c}}_0 \oplus 0^n$. For given $\overline{G}$ and $\overline{\boldsymbol{c}}_0$, it is obvious that there is one to one correspondence between $\boldsymbol{t}_i$ and $\overline{\boldsymbol{c}}_i$. Then the order of searching

candidate codewords depends on that of generating test error patterns. Efficient orders of generating test error patterns have been devised [6], [7], [14].

Let $\overline{C}_s$ be a set of codewords which includes all candidate codewords $\overline{c}_i = \overline{c}_0 \oplus \overline{w}_i$ such that $0 \le i < s$ at a decoding stage of generating t_s, i.e.,

$$\overline{C}_s = \{\overline{c}_i = \overline{c}_0 \oplus \overline{w}_i \,|\, \overline{w}_i = t_i\overline{G}, 0 \le i < s\}. \tag{5}$$

For a test error pattern t_i, let $F(t_i)$ express arbitrary evaluation function of t_i satisfying

$$0 \le F(t_i) \le L(\overline{c}_i), \tag{6}$$

where $\overline{c}_i = \overline{c}_0 \oplus t_i\overline{G}$. Several evaluation functions have been proposed [2], [6], [7], [12].

At a decoding stage of generating t_i, we need not to encode t_i if

$$\underline{L}[\overline{C}_i] \le F(t_i), \tag{7}$$

since $\overline{c}_i$ cannot be better than the best candidate codeword obtained so far. i.e., if Eq. (7) holds, t_i cannot give the best candidate codeword.

For $0 \le \forall i \le 2^k$, let $\overline{e}_i = (\overline{e}_{i,1}, \overline{e}_{i,2}, \ldots, \overline{e}_{i,n})$ be such that $\overline{e}_i = \overline{z} \oplus \overline{c}_i$. For $\overline{w}_i \in \overline{C}$, let $\Lambda(\overline{w}_i)$ be defined as

$$\Lambda(\overline{w}_i) = \sum_{j \in supp(\overline{w}_i)} (1 - 2\overline{e}_{0,j})|\overline{\theta}_j|. \tag{8}$$

Then, for $\overline{c}_i (= \overline{c}_0 \oplus \overline{w}_i)$, we can compute $L(\overline{c}_i)$ by

$$L(\overline{c}_i) = L(\overline{c}_0) + \Lambda(\overline{w}_i), \tag{9}$$

since from $\overline{z} \oplus \overline{c}_i = \overline{e}_0 \oplus \overline{w}_i$,

$$\begin{aligned} L(\overline{c}_i) &= \sum_{j=1}^{n} (\overline{e}_{0,j} \oplus \overline{w}_{i,j})|\overline{\theta}_j| \\ &= \sum_{j=1}^{n} \overline{e}_{0,j}|\overline{\theta}_j| + \sum_{j=1}^{n} (1 - 2\overline{e}_{0,j})\overline{w}_{i,j}|\overline{\theta}_j| \\ &= L(\overline{c}_0) + \sum_{j \in supp(\overline{w}_i)} (1 - 2\overline{e}_{0,j})|\overline{\theta}_j|. \end{aligned} \tag{10}$$

By Eq. (9), for a fixed $\overline{c}_0$, searching $\overline{c}_i$ which minimizes $L(\overline{c}_i)$ is equivalent to searching $\overline{w}_i$ which minimizes $\Lambda(\overline{w}_i)$.

We describe a general version of the reliability-based MLD algorithm below. For an integer α, let $\alpha{+}{+}$ denote the increment operation of α.

[The reliability-based MLD Algorithm]

1) Generate $\overline{c}_0 := u\overline{G}$, and set $\underline{L} := L(\overline{c}_0)$, $\overline{w}^* := 0^n$, $\underline{\Lambda} := 0$ and $i := 1$.
2) Generate t_i and compute $F(t_i)$. If $\underline{L} \le F(t_i)$, then go to 4).
3) Generate $\overline{w}_i := t_i\overline{G}$ and compute $\Lambda(\overline{w}_i)$. If $\Lambda(\overline{w}_i) < \underline{\Lambda}$, then $\underline{\Lambda} := \Lambda(\overline{w}_i)$, $\underline{L} := L(\overline{c}_0) + \underline{\Lambda}$ and $\overline{w}^* := \overline{w}_i$.
4) Set $i{+}{+}$. If $i \le 2^k$ and a certain terminating criterion does not hold, then go to 2), otherwise output $\overline{c}_{ML} := \overline{c}_0 \oplus \overline{w}^*$ and stop. □

As for a terminating criterion of the decoding algorithm at step 4), several criteria have been proposed [2], [3], [6], [7], [12].

We here state the complexity of the reliability-based decoding algorithm. The time complexity of permuting θ in the non-increasing order is $O(n \log n)$ and of constructing $\overline{G}$ is $O(n \times \kappa^2)$ where $\kappa = \min\{k, n-k\}$ [2], [6], [7]. These steps are carried out only once in a decoding procedure. Contrary to the above steps, generating t_i and constructing $\overline{w}_i = t_i\overline{G}$ are carried out iteratively, where each encoding requires binary operations of $O(kn)$ by conventional encoding method [2], [12]. For each test error codeword constructed, computing Eq. (8) costs real number operations of $O(n)$. Therefore, both encoding test error patterns and the real number operations of step 3) dominate mainly the whole decoding complexity [4], [7], [12]. As for the space complexity, storing $\overline{G}$ requires $O(kn)$. In some MLD algorithms [7], [12], [14], the test error patterns are stored in a list before encoded by $\overline{G}$. In these algorithms, denoting the maximum list size for decoding r by $N(r)$, the space complexity is $O(\gamma)$ where $\gamma = \max\{kn, N(r)\}$.

3. Proposed Methods Using an Order Relation

3.1 Conditions for Omitting Unnecessary Metrics Computations

We will develop the method for reducing the complexity of the reliability-based decoding algorithms by exploiting the following two properties of the decoding algorithm: (1) Every n-dimensional sequence is permuted in the non-increasing order of reliability measure, (2) at least one codeword (the initial codeword) is obtained before generating each test error codeword.

We consider the case in which t_i does not satisfy Eq. (7) and is encoded to $\overline{w}_i$ in a decoding procedure. If we find out $\overline{w}_i$ cannot give the best codeword, then the computation of Eq. (8) (which is the metrics computation of $\overline{w}_i$) can be omitted. Roughly speaking, we measure a distance† (defined over $\overline{\mathcal{V}}^n$ like the Hamming distance) between $\overline{c}_i$ and $\overline{\theta}$ and that between $\overline{c}_0$ and $\overline{\theta}$. If $\overline{c}_i$ is obviously farther from $\overline{\theta}$ than $\overline{c}_0$, we eliminate $\overline{c}_i$ from consideration without computing its metrics. We will derive a condition that guarantees a test error codeword $\overline{w}_i$ which cannot give the ML codeword.

We define the following order relations:

Definition 2: (The Order Relation for Supports) For two location sets $X = \{j_1, j_2, \ldots, j_m\}$ and $X' = \{j'_1, j'_2, \ldots, j'_{m'}\}$ such that $j_1 < j_2 < \cdots < j_m$ and $j'_1 < j'_2 < \cdots < j'_{m'}$, we write "$X' <_S X$" if $m' \le m$ and $j_h \le j'_h$, $\forall h \in [1, m']$.

Definition 3: (The Order Relation for Binary Vectors) For two vectors x and x', we write "$x' <_V x$" if and only if $supp(x') <_S supp(x)$.

For two vectors $\overline{x} = (\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n) \in \overline{\mathcal{V}}^n$ and $\overline{x}' =$

†It will be defined in Definition 2 and 3, although it does not satisfy an axiom of the distance measure.

$(\overline{x}'_1, \overline{x}'_2, \ldots, \overline{x}'_n) \in \overline{\mathcal{V}}^n$, define two location sets as

$$\mathcal{D}_0(\overline{x}, \overline{x}') = \{j \,|\, \overline{x}_j = 1 \text{ and } \overline{x}'_j = 0\}, \tag{11}$$

$$\mathcal{D}_1(\overline{x}, \overline{x}') = \{j \,|\, \overline{x}_j = 1 \text{ and } \overline{x}'_j = 1\}. \tag{12}$$

Theorem 1: For a test error codeword $\overline{w}_i$, assume that there is an order relation such that

$$\mathcal{D}_1(\overline{w}_i, \overline{e}_0) <_S \mathcal{D}_0(\overline{w}_i, \overline{e}_0). \tag{13}$$

Then $\overline{c}_i (= \overline{c}_0 \oplus \overline{w}_i)$ cannot be better than $\overline{c}_0$.

Proof: Assume that $\mathcal{D}_0(\overline{w}_i, \overline{e}_0) = \{j_1, j_2, \ldots, j_m\}$ and $\mathcal{D}_1(\overline{w}_i, \overline{e}_0) = \{j'_1, j'_2, \ldots, j'_{m'}\}$ such that $j_1 < j_2 < \cdots < j_m$ and $j'_1 < j'_2 < \cdots < j'_{m'}$. For each element of $\mathcal{D}_\alpha(\overline{w}_i, \overline{e}_0)$, $\alpha \in \{0, 1\}$ satisfies

$$|\overline{\theta}_{j_1}| \geq |\overline{\theta}_{j_2}| \geq \cdots \geq |\overline{\theta}_{j_m}|, \tag{14}$$

$$|\overline{\theta}_{j'_1}| \geq |\overline{\theta}_{j'_2}| \geq \cdots \geq |\overline{\theta}_{j'_{m'}}|. \tag{15}$$

By the assumption of Eq. (13), $m' \leq m$ and

$$|\overline{\theta}_{j_h}| \geq |\overline{\theta}_{j'_h}|, \quad \text{for } h \in [1, m']. \tag{16}$$

Equation (8) is now

$$\begin{aligned} \Lambda(\overline{w}_i) &= 6 \sum_{j=1}^{n} (1 - 2\overline{e}_{0,j}) \overline{w}_{i,j} |\overline{\theta}_j| \\ &= \sum_{j|\overline{e}_{0,j}=0} \overline{w}_{i,j} |\overline{\theta}_j| - \sum_{j|\overline{e}_{0,j}=1} \overline{w}_{i,j} |\overline{\theta}_j| \\ &= \sum_{j \in \mathcal{D}_0(\overline{w}_i, \overline{e}_0)} |\overline{\theta}_j| - \sum_{j \in \mathcal{D}_1(\overline{w}_i, \overline{e}_0)} |\overline{\theta}_j|. \end{aligned} \tag{17}$$

Therefore $\Lambda(\overline{w}_i) \geq 0$ by Eq. (16). Hence $L(\overline{c}_i) = L(\overline{c}_0) + \Lambda(\overline{w}_i) \geq L(\overline{c}_0)$ and $\overline{c}_i$ cannot be better than $\overline{c}_0$. □

If the order relation of Eq. (13) is satisfied, $\overline{c}_i$ given by $\overline{w}_i$ is farther from $\overline{\theta}$ than $\overline{c}_0$. i.e., $\overline{c}_i$ cannot be the ML codeword. Equation (13) can be used for judging if the metrics of a candidate codeword need not to be computed. Hereafter, we call this order relation of Eq. (13) *Omitting Criterion A*.

We now present a method for testing if an order relation

$$\mathcal{D}_1(\overline{x}, \overline{x}') <_S \mathcal{D}_0(\overline{x}, \overline{x}'), \tag{18}$$

holds for $\overline{x} = (\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n) \in \overline{\mathcal{V}}^n$ and $\overline{x}' = (\overline{x}'_1, \overline{x}'_2, \ldots, \overline{x}'_n) \in \overline{\mathcal{V}}^n$. For $\boldsymbol{a} = (a_1, a_2, \ldots, a_n) \in \{0, 1\}^n$, let $\boldsymbol{a} \gg$ and $\boldsymbol{a} \ll$ be the right and the left shift operation by one bit, respectively. The algorithm can be performed by increments of an integer and shift operations of a binary array.

[Procedure $OT(\overline{x}, \overline{x}')$]

1) Set $\boldsymbol{a} := (0, 1, 0, \ldots, 0)$ and $\tau := 1$.
2) If $\overline{x}_\tau = 1$ and $\overline{x}'_\tau = 0$, then $\boldsymbol{a} \gg$. If $\overline{x}_\tau = 1$ and $\overline{x}'_\tau = 1$, then $\boldsymbol{a} \ll$.
3) If $a_1 = 1$, then $OT(\overline{x}, \overline{x}') := 1$ and stop. If $\tau = n$, then $OT(\overline{x}, \overline{x}') := 0$ and stop. Otherwise, set $\tau{+}{+}$ and go to 2). □

Note that for an integer α, $\alpha{+}{+}$ denotes the increment of α. In the above algorithm, we denote a returned value with $OT(\overline{x}, \overline{x}') \in \{0, 1\}$. If Eq. (18) is satisfied, the algorithm returns $OT(\overline{x}, \overline{x}') = 0$ (the validity of the algorithm will be given below). Otherwise, it returns $OT(\overline{x}, \overline{x}') = 1$. Remark that $OT(\overline{x}, \overline{x}')$ denotes either testing Eq. (18) or a returned value of the test.

At step 2) in the above algorithm, τ such that $\overline{x}_\tau = 1$ and $\overline{x}'_\tau = 0$ is an element of $\mathcal{D}_0(\overline{x}, \overline{x}')$. Similarly, τ such that $\overline{x}_\tau = 1$ and $\overline{x}'_\tau = 1$ is an element of $\mathcal{D}_1(\overline{x}, \overline{x}')$. i.e., the procedure of step 2) means:

(1) if we find $\tau \in \mathcal{D}_0(\overline{x}, \overline{x}')$, then we set $\boldsymbol{a} \gg$,

(2) if we find $\tau \in \mathcal{D}_1(\overline{x}, \overline{x}')$, then we set $\boldsymbol{a} \ll$.

Remark that we can realize shift operations of $\boldsymbol{a} \gg$ and $\boldsymbol{a} \ll$ much easier than ordinary shift operations of binary array of size n since $w_H(\boldsymbol{a}) = 1$. It is enough that the element one is moved by one bit and this shift can be accomplished by two exclusive OR operations. We also remark that we can describe Procedure $OT(\overline{x}, \overline{x}')$ using a variable ρ, which keeps the position of the element 1 in $\boldsymbol{a}$, instead of using the vector $\boldsymbol{a}$. In that case, we initially set $\rho = 2$. The left and right shift operation in Procedure $OT(\overline{x}, \overline{x}')$ can be expressed by the decrement and increment of ρ, respectively.

We will show the validity of the above algorithm.

Theorem 2: For $\overline{x}, \overline{x}' \in \overline{\mathcal{V}}^n$, the returned value is $OT(\overline{x}, \overline{x}') = 0$ if and only if Eq. (18) holds.

Proof: First, we will prove if part. We assume that Eq. (18) holds. Let $\mathcal{D}_0(\overline{x}, \overline{x}') = \{j_1, j_2, \ldots, j_m\}$ and $\mathcal{D}_1(\overline{x}, \overline{x}') = \{j'_1, j'_2, \ldots, j'_{m'}\}$ such that $j_1 < j_2 < \cdots < j_m$ and $j'_1 < j'_2 < \cdots < j'_{m'}$. The algorithm searches $j_h \in \mathcal{D}_0(\overline{x}, \overline{x}')$ or $j'_h \in \mathcal{D}_1(\overline{x}, \overline{x}')$ from left position to right one. For any $h \in [1, m']$, before we encounter $\tau = j'_h \in \mathcal{D}_1(\overline{x}, \overline{x}')$, we have already found $j_h \in \mathcal{D}_0(\overline{x}, \overline{x}')$ since $j_h < j'_h$, $\forall h \in [1, m']$, from Eq. (18). Therefore, after we encounter $\tau = j'_h \in \mathcal{D}_1(\overline{x}, \overline{x}')$ for $h \in [1, m']$, j such that $a_j = 1$ is necessarily greater than one, i.e., $j > 1$. The condition $a_1 = 1$ does not hold for all positions τ, $\forall \tau \in [1, n]$. Since τ is incremented up to n, the returned value is $OT(\overline{x}, \overline{x}') = 0$.

Next, we will prove only if part. We assume $OT(\overline{x}, \overline{x}') = 0$. We here assume there exist a certain h^* such that $j'_{h^*} < j_{h^*}$ and we will prove the theorem by contradiction. If $h^* = 1$, then we set $\boldsymbol{a} \ll$ and $a_1 = 1$ holds when encountering $\tau = j'_1 \in \mathcal{D}_1(\overline{x}, \overline{x}')$ at step 2). Then h^* should be greater than one. If $h^* > 1$, then $j_{h^*-1} < j'_{h^*-1} < j'_{h^*} < j_{h^*}$. When we encounter $\tau = j'_{h^*-1} \in \mathcal{D}_1(\overline{x}, \overline{x}')$ at step 2), we set $\boldsymbol{a} \ll$ and $a_2 = 1$ holds since we have already found exactly $h^* - 1$ elements of each $\mathcal{D}_0(\overline{x}, \overline{x}')$ and $\mathcal{D}_1(\overline{x}, \overline{x}')$. Therefore, when encountering $\tau = j'_{h^*} \in \mathcal{D}_1(\overline{x}, \overline{x}')$ at step 2), we set $\boldsymbol{a} \ll$ and $a_1 = 1$ holds. At step 3), the algorithm returns $OT(\overline{x}, \overline{x}') = 1$. This contradicts the assumption, $OT(\overline{x}, \overline{x}') = 0$. Hence $j_h < j'_h$ must be satisfied for $\forall h \in [1, m']$ and Eq. (18) holds. □

Testing Omitting Criterion A is denoted with $OT(\overline{w}_i, \overline{e}_0)$.

Corollary 1: For $\overline{w}_i$ and $\overline{e}_0$, the returned value is $OT(\overline{w}_i, \overline{e}_0) = 0$ if and only if Eq. (13) holds. When $OT(\overline{w}_i, \overline{e}_0) = 0$, $\overline{c}_i (= \overline{c}_0 \oplus \overline{w}_i)$ cannot be better than $\overline{c}_0$.

By Corollary 1, if $OT(\overline{w}_i, \overline{e}_0) = 0$ for $\overline{w}_i$, we can omit the computation of Eq. (17) for $\overline{w}_i$. It is obvious that the time complexity for performing $OT(\overline{w}_i, \overline{e}_0)$ is increments of an integer and shift operations of $O(n)$. Note that increments of an integer are also necessary for encoding and computing metrics if we realize them serially by software. Since the computation of Eq. (17) costs real number operations of $O(n)$, the time complexity of testing Omitting Criterion A is fairly small. As for the space complexity, we need to store $\overline{e}_0$ and this requires a binary array of size $O(n)$. In the reliability-based MLD algorithm, this number is negligible since storing only $\overline{G}$ requires a binary array of size $O(kn)$.

Example 1: Let $\overline{w}_i$ = (00011011) and $\overline{e}_0$ = (00001010). Then $\mathcal{D}_0(\overline{w}_i, \overline{e}_0) = \{4, 8\}$ and $\mathcal{D}_1(\overline{w}_i, \overline{e}_0) = \{5, 7\}$. First, we find $\tau = 4 \in \mathcal{D}_0(\overline{w}_i, \overline{e}_0)$, then we set $a_2 := 0$ and $a_3 := 1$ by $a >>$. Next, we find $\tau = 5 \in \mathcal{D}_1(\overline{w}_i, \overline{e}_0)$, then we set $a_2 := 1$ and $a_3 := 0$ by $a <<$. When we find $\tau = 7 \in \mathcal{D}_1(\overline{w}_i, \overline{e}_0)$, we set $a_1 := 1$ and $a_2 := 0$ by $a <<$. Since $a_1 = 1$, $OT(\overline{w}_i, \overline{e}_0) = 1$ is returned.

Example 2: Let $\overline{w}_i$ = (00101110) and $\overline{e}_0$ = (00001010). Then $\mathcal{D}_0(\overline{w}_i, \overline{e}_0) = \{j_1 = 3, j_2 = 6\}$ and $\mathcal{D}_1(\overline{w}_i, \overline{e}_0) = \{j'_1 = 5, j'_2 = 7\}$. Therefore, from $j_1 < j'_1$ and $j_2 < j'_2$, Eq. (13) holds. When we increment τ up to $n = 8$, the algorithm returns $OT(\overline{w}_i, \overline{e}_0) = 0$.

We describe the reliability-based MLD algorithm employing the test of Omitting Criterion A. We will call this decoding algorithm the proposed decoding algorithm A.

[The Proposed Decoding Algorithm A]

1) Generate $\overline{c}_0 := \boldsymbol{u}\overline{G}$, and set $\underline{L} := L(\overline{c}_0)$, $\overline{e}_0 := \overline{z} \oplus \overline{c}_0$, $\overline{w}^* := 0^n$, $\underline{\Lambda} := 0$ and $i := 1$.
2) Generate $\boldsymbol{t}_i$ and compute $F(\boldsymbol{t}_i)$. If $\underline{L} \le F(\boldsymbol{t}_i)$, then go to 4).
3) a) Generate $\overline{w}_i := \boldsymbol{t}_i\overline{G}$. If $OT(\overline{w}_i, \overline{e}_0) = 0$, then go to 4).
   b) Compute $\Lambda(\overline{w}_i)$. If $\Lambda(\overline{w}_i) < \underline{\Lambda}$, then $\underline{\Lambda} := \Lambda(\overline{w}_i)$, $\underline{L} := L(\overline{c}_0) + \underline{\Lambda}$ and $\overline{w}^* := \overline{w}_i$.
4) Set i++. If $i \le 2^k$ and a certain terminating criterion does not hold, then go to 2), otherwise output $\overline{c}_{ML} := \overline{c}_0 \oplus \overline{w}^*$ and stop. □

In the above algorithm, step 1) and 3) is modified to the original reliability-based decoding algorithm. At step 3)a), if $OT(\overline{w}_i) = 0$, real number operations at step 3)b), which include the computation of Eq. (17) and one addition and comparison, are omitted.

3.2 Adaptive Procedure of the Proposed Conditions

Theorem 1 implies that Omitting Criterion A compares $\overline{c}_i$ with $\overline{c}_0$ and judges if $\overline{c}_i$ is farther from $\overline{\theta}$ than $\overline{c}_0$. In a decoding procedure, let $\overline{c}^*$ denote the best candidate codeword obtained so far such that $\overline{c}^* = \overline{c}_0 \oplus \overline{w}^*$. At a decoding stage of constructing $\overline{w}_i$, the best candidate codeword $\overline{c}^*$ is not necessarily equal to $\overline{c}_0$ and such $\overline{c}^*$ is closer to $\overline{\theta}$ than $\overline{c}_0$. If we can compare $\overline{c}_i$ with $\overline{c}^*$ (not with $\overline{c}_0$) and we test whether $\overline{c}_i$ is farther from $\overline{\theta}$ than $\overline{c}^*$, a sufficient condition for omitting unnecessary metrics computation may be more effective. We will consider an *adaptive procedure* in which $\overline{c}^*$ referenced by the proposed condition is adaptively altered.

At a decoding stage of constructing $\overline{w}_i$, let $\overline{e}^* = (\overline{e}^*_1, \overline{e}^*_2, \ldots, \overline{e}^*_n)$ be such that $\overline{e}^* = \overline{z} \oplus \overline{c}^*$. Furthermore, let $\overline{v}_i = \overline{w}^* \oplus \overline{w}_i$.

Lemma 1: Using $\overline{c}^*$ and $\overline{v}_i = \overline{w}^* \oplus \overline{w}_i$, $L(\overline{c}_i)$ is expressed as follows:

$$L(\overline{c}_i) = L(\overline{c}^*) + \sum_{j \in supp(\overline{v}_i)} (1 - 2\overline{e}^*_j)|\overline{\theta}_j|. \tag{19}$$

Equation (19) shows the relation between $L(\overline{c}^*)$ and $L(\overline{c}_i)$.

Proof: Since $\overline{c}_0 = \overline{c}^* \oplus \overline{w}^*$ and $\overline{e}^* = \overline{z} \oplus \overline{c}^*$, the left hand side (l.h.s.) of Eq. (19) expands in the following way:

$$\begin{aligned} L(\overline{c}_i) &= L(\overline{c}_0 \oplus \overline{w}_i) = L(\overline{c}^* \oplus \overline{w}^* \oplus \overline{w}_i) \\ &= \sum_{j=1}^{n} (\overline{e}^*_j \oplus \overline{w}^*_j \oplus \overline{w}_{i,j})|\overline{\theta}_j| \\ &= \sum_{j=1}^{n} \overline{e}^*_j|\overline{\theta}_j| + \sum_{j=1}^{n} (1 - 2\overline{e}^*_j)(\overline{w}^*_j \oplus \overline{w}_{i,j})|\overline{\theta}_j| \\ &= L(\overline{c}^*) + \sum_{j | \overline{w}^*_j \oplus \overline{w}_{i,j} = 1} (1 - 2\overline{e}^*_j)|\overline{\theta}_j|. \end{aligned} \tag{20}$$

Hence we have Eq. (19). □

Theorem 3: For a test error codeword $\overline{w}_i$, assume that there is an order relation such that

$$\mathcal{D}_1(\overline{v}_i, \overline{e}^*) <_S \mathcal{D}_0(\overline{v}_i, \overline{e}^*). \tag{21}$$

Then $\overline{c}_i (= \overline{c}_0 \oplus \overline{w}_i)$ cannot be better than $\overline{c}^*$.

Proof: We can prove the theorem in a similar way of proving Theorem 1 by using Lemma 1. □

Theorem 3 implies that $\overline{c}_i$, given by $\overline{w}_i$, is farther from $\overline{\theta}$ than $\overline{c}^*$ if Eq. (21) holds for $\overline{w}_i$. Then $\overline{w}_i$ cannot give the ML codeword. Therefore we need not compute metrics of $\overline{w}_i$ which satisfies Eq. (21).

In general, for $\overline{e}^* \neq \overline{e}_0$, $w_H(\overline{e}^*)$ tends to be smaller than $w_H(\overline{e}_0)$ [3]. This implies that $|\mathcal{D}_1(\overline{v}_i, \overline{e}^*)|$ tends to be smaller than $|\mathcal{D}_1(\overline{w}_i, \overline{e}_0)|$. Consequently, Eq. (21) can be a more effective condition than Omitting Criterion A since Eq. (13) or Eq. (21) is satisfied more often, as the cardinality of its l.h.s. is smaller. We will call the order relation of Eq. (21) *Omitting Criterion B*. For $\overline{w}_i$ and $\overline{w}^*$, testing Omitting Criterion B is denoted with $OT(\overline{v}_i, \overline{e}^*)$.

We describe the reliability-based MLD algorithm employing Omitting Criterion B in which step 1) and 3) is modified to the original MLD algorithm. We will call this decoding algorithm the proposed decoding algorithm B.

[The Proposed Decoding Algorithm B]

1) Generate $\overline{c}_0 := u\overline{G}$, and set $\underline{L} := L(\overline{c}_0)$, $\overline{e}_0 := \overline{z} \oplus \overline{c}_0$, $\overline{w}^* := 0^n$, $\overline{e}^* := 0^n$, $\underline{\Lambda} := 0$ and $i := 1$.
2) Generate t_i and compute $F(t_i)$. If $\underline{L} \leq F(t_i)$, then go to 4).
3) a) Generate $\overline{w}_i := t_i\overline{G}$ and set $\overline{v}_i := \overline{w}^* \oplus \overline{w}_i$. If $OT(\overline{v}_i, \overline{e}^*) = 0$, then go to 4).
   b) Compute $\Lambda(\overline{w}_i)$. If $\Lambda(\overline{w}_i) < \underline{\Lambda}$, then $\underline{\Lambda} := \Lambda(\overline{w}_i)$, $\underline{L} := L(\overline{c}_0) + \underline{\Lambda}$, $\overline{w}^* := \overline{w}_i$ and $\overline{e}^* := \overline{e}_0 \oplus \overline{w}^*$.
4) Set i++. If $i \leq 2^k$ and a certain terminating criterion does not hold, then go to 2), otherwise output $\overline{c}_{ML} := \overline{c}_0 \oplus \overline{w}^*$ and stop. □

In the proposed decoding algorithm B, we construct $\overline{v}_i$ each time $\overline{w}_i$ is obtained at step 3)a). For $\overline{w}^* \neq 0^n$, constructing $\overline{v}_i$ costs just n binary operations which is smaller than the complexity of encoding each test error pattern (ordinarily that costs binary operations of $O(kn)$). We also update $\overline{e}^*$ in order to implement $OT(\overline{v}_i, \overline{e}^*)$ each time a new best candidate codeword is obtained at step 3)b). Since $\overline{e}^* = \overline{z} \oplus \overline{c}^*$ and $\overline{e}_0 = \overline{z} \oplus \overline{c}_0$, we can obtain $\overline{e}^*$ by

$$\overline{e}^* = \overline{z} \oplus \overline{c}_0 \oplus \overline{w}^* = \overline{e}_0 \oplus \overline{w}^*. \tag{22}$$

For $\overline{w}^* \neq 0^n$, the computation of the right hand side (r.h.s.) of Eq. (22) requires just n binary operations. Furthermore, the space complexity for storing $\overline{v}_i$ and $\overline{e}^*$ is two binary arrays of size $O(n)$. We remark again that storing $\overline{G}$ requires $O(kn)$ and the increased space complexity is small.

We can also consider the following modification: Either Omitting Criterion A and B is selectively tested for $\overline{w}_i$ in accordance with the order relation between t_i and t^* such that $\overline{w}^* = t^*\overline{G}$. Remark that, for $\overline{w}_i$ and $\overline{w}^*$, Omitting Criterion B holds only if $t^* <_V t_i$. Therefore, for $\overline{w}_i$, we adopt Omitting Criterion A if $t^* \not<_V t_i$, and adopt Omitting Criterion B otherwise[†]. In this case, testing if $t^* <_V t_i$ can be carried out in a similar way of the test $OT(\overline{x}, \overline{x}')$ for $\overline{x}, \overline{x}' \in \overline{\mathcal{V}}^n$. The time complexity for testing the order relation of test error patterns of length k is smaller than the test $OT(\overline{x}, \overline{x}')$ for $\overline{x}, \overline{x}'$ of length n.

3.3 Performance of the Proposed Decoding Algorithms

In this subsection, we state the performance of the proposed decoding algorithms.

Theorem 4: The both proposed decoding algorithms A and B achieve MLD.

Proof: Test error codewords constructed in the both proposed decoding algorithms A and B are the same as that constructed in the original MLD algorithm. The proposed decoding algorithms eliminate codewords which cannot be the ML codeword. For the ML codeword, its metrics is necessarily computed. □

We summarize the additional complexity of the proposed decoding algorithms A and B to the original reliability-based MLD algorithm. First, we state the additional complexity of the proposed decoding algorithm A.

(1) **Time complexity:** For testing $OT(\overline{w}_i, \overline{e}_0)$, at most $n-1$ increments of an integer and at most n shift operations are required.
(2) **Space complexity:** For storing $\overline{e}_0$, we allocate a binary array of size n.

Next, we state the additional complexity of the proposed decoding algorithm B.

(1) **Time complexity:** For testing $OT(\overline{v}_i, \overline{e}^*)$, we construct $\overline{v}_i$ each time $\overline{w}_i$ is constructed. Furthermore, we construct $\overline{e}^*$ each time $\overline{w}^*$ is obtained. Constructing $\overline{v}_i$ or $\overline{e}^*$ costs n binary operations. For implementation of $OT(\overline{v}_i, \overline{e}^*)$, at most $n-1$ increments of an integer and at most n shift operations are required.
(2) **Space complexity:** For storing $\overline{v}_i$ and $\overline{e}_0$, we allocate two binary arrays of size n.

Theorem 5: The numbers of real number operations for both proposed decoding algorithms A and B are smaller than that for the original reliability-based MLD algorithm.

Proof: The total number of generating candidate codewords is the same for each decoding algorithm. The proposed decoding algorithm A (B) omits the computation of Eq. (17) if Omitting Criterion A (B) is satisfied for a test error codeword. Therefore, in proposed decoding algorithms A and B, the numbers of computation of Eq. (17) are reduced or at most the same as that in the original MLD algorithm. □

4. Simulation Results

4.1 Conditions for Simulations

In this section, we present simulation results for the binary (63,30,13) BCH code and the binary (127,64,21) BCH code in order to evaluate effectiveness of the proposed conditions. We assume each codeword is transmitted over the AWGN channel with the SNR E_b/N_0 [dB]. Although the proposed conditions can be applicable to any reliability-based decoding algorithms, we adopt the Gazelle and Snyders (GS) decoding algorithm [6] which is well-known for its efficiency with small space complexity.

The GS decoding algorithm [6] employs a simple evaluation function of t_i defined as

$$\Delta(t_i) = \sum_{j=1}^{k} t_{i,j}|\tilde{\theta}_j|, \tag{23}$$

where $\tilde{\theta} = (\tilde{\theta}_1, \tilde{\theta}_2, \ldots, \tilde{\theta}_n)$ is permuted sequence of $\overline{\theta}$ such that the leftmost k positions are the k MRI positions in the non-increasing order of reliability measure. The function

[†]For $\overline{x}$ and $\overline{x}'$, the order relation $\overline{x} \not<_V \overline{x}'$ means that the order relation $\overline{x}' <_V \overline{x}$ never holds.

$\Delta(t_i)$ satisfies Eq. (6) (i.e., $\Delta(t_i) \leq L(\overline{c}_i)$) since

$$L(\overline{c}_i) = \Delta(t_i) + \sum_{j \in [1,n] \setminus \mathcal{M}} (\overline{z}_j \oplus \overline{c}_{i,j}) |\bar{\theta}_j|. \tag{24}$$

Another evaluation function of t_i is used by MLD algorithms in [2], [6], [7]. The evaluation function $f(t_i)$ gives a tighter lower bound of $L(\overline{c}_i)$ than the function $\Delta(t_i)$. The function $f(t_i)$ uses the fact that the Hamming distance between some codeword $\overline{c}_{\text{seed}}$ and any codeword $\overline{c}_i \neq \overline{c}_{\text{seed}}$ is no less than d, which is the minimum distance of the code C. Here, as in [2], [6], we consider the case $\overline{c}_{\text{seed}} = \overline{c}_0$[†]. We define $\mathcal{B}(\overline{c}_0)$ as the set of positions where element of $\overline{e}_0 (= \overline{z} \oplus \overline{c}_0)$ is 0. Furthermore, for t_i, let $\mathcal{A}(\overline{c}_0, t_i)$ be the set of $d - w_H(\overline{e}_0) - w_H(t_i)$ least reliable positions in $\mathcal{B}(\overline{c}_0)$[††]. Then, the function $f(t_i)$ is defined as

$$f(t_i) = \Delta(t_i) + \sum_{j \in \mathcal{A}(\overline{c}_0, t_i)} |\bar{\theta}_j|, \tag{25}$$

where $\Delta(t_i)$ is given by Eq. (23). The second term of r.h.s. of Eq. (25) is non-negative, so the function $f(t_i)$ is a tighter lower bound of $L(\overline{c}_i)$ than $\Delta(t_i)$. The function $f(t_i)$ is the same as the *heuristic function* of the A* decoding algorithm [7], if we set $\overline{c}_{\text{seed}} = \overline{c}_0$.

The main difference between $\Delta(\cdot)$ and $f(\cdot)$ is the second term of r.h.s. of Eq. (25). The second term of Eq. (25) depends only on the Hamming weight of a test error pattern so it can be computed beforehand for each Hamming weight $1, 2, \ldots, d - w_H(\overline{e}_0) - 1$ and be stored in memory. Furthermore, the second term of Eq. (25) for the larger Hamming weight than one can be computed during the computation of the second term for the Hamming weight one.

We consider the two GS decoding algorithms using (i) the evaluation function $\Delta(\cdot)$ (denoted as the algorithm GS(Δ)) and (ii) the evaluation function $f(\cdot)$ (denoted as the algorithm GS(f)). For each original GS decoding algorithm, we consider the following two modifications:

(1) [The algorithm A(Δ) and A(f)]: In algorithms GS(Δ) and GS(f), respectively, each time a test error codeword $\overline{w}_i$ is constructed, Omitting Criterion A is tested if $\overline{c}_i (= \overline{c}_0 \oplus \overline{w}_i)$ cannot be better than $\overline{c}_0$.

(2) [The algorithm B(Δ) and B(f)]: In algorithms GS(Δ) and GS(f), respectively, each time the best candidate codeword $\overline{c}^*$ is obtained, the codeword referenced by Omitting Criterion B is updated. After each test error codeword $\overline{w}_i$ is constructed, Omitting Criterion B is tested if $\overline{c}_i$ cannot be better than $\overline{c}^*$.

Note that the numbers of real number operations for modified algorithms A(Δ) and B(Δ) are no more than that for the algorithm GS(Δ) by Theorem 5. As for algorithms using $f(\cdot)$, the same relation holds.

The results are obtained by decoding 10000 codewords for each SNR and the average values are shown in tables. In tables, we show the results of the following simulations.

(1) In order to evaluate the effectiveness of modified algorithms employing Omitting Criterion A or B, we compare the average number of real number operations for each decoding algorithm[†††]. For computation of $L(\overline{c}_0)$, we count $w_H(\overline{e}_0) - 1$ real number operations. Similarly, for each computation of $\Lambda(\overline{w}_i)$, we count $w_H(\overline{w}_i) - 1$ real number operations. The results are shown in Tables 1 and 2.

(2) In order to evaluate the effectiveness of Omitting Criterion A and B, we compare the average number of computations of Eq. (17) in six decoding algorithms. The results are shown in Tables 3 and 4.

4.2 Results about the Number of Real Number Operations

First we describe results at low to medium SNRs. By Tables 1 and 2, we can see that the numbers of real number operations for algorithms GS(Δ) and GS(f) are almost the same. These results imply that there is almost no difference between the effects of two evaluation functions. The number of real number operations for the algorithm GS(Δ) is the largest and that for the algorithm GS(f) is the second largest. The number of real number operations for the algorithm A(Δ) is the third largest (and the largest among modified algorithms A(Δ), B(Δ), A(f) and B(f)). Even the algorithm A(Δ) reduces the number of real number operations less than 1/3 that for the (63,30,13) and the (127,64,21) codes, compared with the algorithm GS(Δ). The algorithm B(Δ) requires less number of real number operations than that of the algorithm A(Δ) for the (63,30,13) and the (127,64,21) codes. The similar results are obtained for algorithms with the function $f(\cdot)$ for both codes.

Next we describe results at high SNRs. Contrary to the case at low to medium SNRs, the algorithm A(f) (or B(f)) requires more number of real number operations than that of the algorithm A(Δ) (or B(Δ)). The reason is that the number of candidate codewords are relatively small and computations of the second term of Eq. (25) dominate for the whole decoding complexity of A(f) and B(f). Note that the complexity for computing the second term of Eq. (25) is independent of the number of candidate codewords. The numbers of real number operations for the algorithm B(Δ) were the least among six algorithms and the values for B(Δ) were less than 1/3 that for GS(f), which required less real number operations between two conventional algorithms. These results indicate that we should select the evaluation function depending on SNRs if we adopt proposed conditions.

4.3 Results about the Number of Metrics Computations

First we describe results at low SNRs. By Tables 3 and 4, the numbers of metrics computations for algorithms GS(Δ) and

[†]Note that before generating any t_i, the initial codeword $\overline{c}_0$ is already obtained.

[††]We define $\mathcal{A}(\overline{c}_0, t_i) = \emptyset$ if $d - w_H(\overline{e}_0) - w_H(t_i) \leq 0$.

[†††]We do not include the number of real number operations for permuting from θ to $\bar{\theta}$ because it depends on sorting method. For the (63,30,13) and the (127,64,21) codes at each SNR, on average 251 and 582 real number operations are required, respectively, by the quick sort technique.

Table 1 The number of real number operations for the (63, 30, 13) BCH code with the function $\Delta(\cdot)$ and $f(\cdot)$.

E_b/N_0 [dB]	original	proposed		original	proposed	
	GS(Δ)	A(Δ)	B(Δ)	GS(f)	A(f)	B(f)
1.00	$6.94 \cdot 10^4$	$2.44 \cdot 10^4$	$2.17 \cdot 10^4$	$6.89 \cdot 10^4$	$2.43 \cdot 10^4$	$2.16 \cdot 10^4$
1.50	$3.97 \cdot 10^4$	$1.39 \cdot 10^4$	$1.21 \cdot 10^4$	$3.91 \cdot 10^4$	$1.38 \cdot 10^4$	$1.20 \cdot 10^4$
2.00	$2.04 \cdot 10^4$	$6.92 \cdot 10^3$	$6.03 \cdot 10^3$	$1.98 \cdot 10^4$	$6.86 \cdot 10^3$	$5.98 \cdot 10^3$
2.50	$9.36 \cdot 10^3$	$3.08 \cdot 10^3$	$2.62 \cdot 10^3$	$8.82 \cdot 10^3$	$3.03 \cdot 10^3$	$2.57 \cdot 10^3$
3.00	$3.88 \cdot 10^3$	$1.21 \cdot 10^3$	$8.93 \cdot 10^2$	$3.44 \cdot 10^3$	$1.17 \cdot 10^3$	$8.61 \cdot 10^2$
3.50	$1.43 \cdot 10^3$	$4.26 \cdot 10^2$	$3.00 \cdot 10^2$	$1.12 \cdot 10^3$	$4.07 \cdot 10^2$	$2.81 \cdot 10^2$
4.00	$5.49 \cdot 10^2$	$1.42 \cdot 10^2$	$9.21 \cdot 10^1$	$3.50 \cdot 10^2$	$1.34 \cdot 10^2$	$8.40 \cdot 10^1$
4.50	$2.26 \cdot 10^2$	$5.71 \cdot 10^1$	$4.02 \cdot 10^1$	$1.21 \cdot 10^2$	$5.73 \cdot 10^1$	$4.04 \cdot 10^1$
5.00	$8.39 \cdot 10^1$	$2.03 \cdot 10^1$	$1.27 \cdot 10^1$	$3.81 \cdot 10^1$	$2.52 \cdot 10^1$	$1.76 \cdot 10^1$
5.50	$3.31 \cdot 10^1$	7.63	5.79	$1.70 \cdot 10^1$	$1.45 \cdot 10^1$	$1.27 \cdot 10^1$

Table 2 The number of real number operations for the (127, 64, 21) BCH code with the function $\Delta(\cdot)$ and $f(\cdot)$.

E_b/N_0 [dB]	original	proposed		original	proposed	
	GS(Δ)	A(Δ)	B(Δ)	GS(f)	A(f)	B(f)
2.50	$3.16 \cdot 10^7$	$1.04 \cdot 10^7$	$9.23 \cdot 10^6$	$3.15 \cdot 10^7$	$1.04 \cdot 10^7$	$9.21 \cdot 10^6$
3.00	$8.04 \cdot 10^6$	$2.38 \cdot 10^6$	$1.64 \cdot 10^6$	$7.93 \cdot 10^6$	$2.37 \cdot 10^6$	$1.63 \cdot 10^6$
3.50	$8.80 \cdot 10^5$	$2.69 \cdot 10^5$	$1.55 \cdot 10^5$	$8.13 \cdot 10^5$	$2.63 \cdot 10^5$	$1.49 \cdot 10^5$
4.00	$1.10 \cdot 10^5$	$3.38 \cdot 10^4$	$1.59 \cdot 10^4$	$8.40 \cdot 10^4$	$3.18 \cdot 10^4$	$1.40 \cdot 10^4$
4.50	$1.62 \cdot 10^4$	$4.03 \cdot 10^3$	$1.85 \cdot 10^3$	$8.34 \cdot 10^3$	$3.52 \cdot 10^3$	$1.34 \cdot 10^3$
5.00	$2.99 \cdot 10^3$	$5.83 \cdot 10^2$	$2.89 \cdot 10^2$	$9.58 \cdot 10^2$	$4.89 \cdot 10^2$	$1.95 \cdot 10^2$
5.50	$6.86 \cdot 10^2$	$8.25 \cdot 10^1$	$4.73 \cdot 10^1$	$1.23 \cdot 10^2$	$7.31 \cdot 10^1$	$3.79 \cdot 10^1$
6.00	$1.82 \cdot 10^2$	$1.93 \cdot 10^1$	$1.33 \cdot 10^1$	$3.51 \cdot 10^1$	$2.93 \cdot 10^1$	$2.33 \cdot 10^1$
6.50	$4.94 \cdot 10^1$	5.59	4.80	$2.02 \cdot 10^1$	$1.97 \cdot 10^1$	$1.89 \cdot 10^1$

Table 3 The number of computations of Eq. (17) for the (63,30,13) BCH code with the function $\Delta(\cdot)$ and $f(\cdot)$.

E_b/N_0 [dB]	original	proposed		original	proposed	
	GS(Δ)	A(Δ)	B(Δ)	GS(f)	A(f)	B(f)
1.00	$2.68 \cdot 10^3$	$6.66 \cdot 10^2$	$5.39 \cdot 10^2$	$2.66 \cdot 10^3$	$6.65 \cdot 10^2$	$5.38 \cdot 10^2$
1.50	$1.55 \cdot 10^3$	$3.87 \cdot 10^2$	$3.03 \cdot 10^2$	$1.52 \cdot 10^3$	$3.86 \cdot 10^2$	$3.02 \cdot 10^2$
2.00	$8.08 \cdot 10^2$	$1.95 \cdot 10^2$	$1.52 \cdot 10^2$	$7.81 \cdot 10^2$	$1.95 \cdot 10^2$	$1.52 \cdot 10^2$
2.50	$3.77 \cdot 10^2$	$8.89 \cdot 10^1$	$6.60 \cdot 10^1$	$3.52 \cdot 10^2$	$8.89 \cdot 10^1$	$6.60 \cdot 10^1$
3.00	$1.60 \cdot 10^2$	$3.56 \cdot 10^1$	$2.03 \cdot 10^1$	$1.40 \cdot 10^2$	$3.56 \cdot 10^1$	$2.03 \cdot 10^1$
3.50	$6.10 \cdot 10^1$	$1.32 \cdot 10^1$	6.93	$4.64 \cdot 10^1$	$1.32 \cdot 10^1$	6.92
4.00	$2.39 \cdot 10^1$	4.24	1.76	$1.44 \cdot 10^1$	4.24	1.76
4.50	9.91	1.65	$8.00 \cdot 10^{-1}$	4.64	1.65	$7.98 \cdot 10^{-1}$
5.00	3.71	$5.42 \cdot 10^{-1}$	$1.57 \cdot 10^{-1}$	1.15	$5.41 \cdot 10^{-1}$	$1.57 \cdot 10^{-1}$
5.50	1.42	$1.36 \cdot 10^{-1}$	$4.16 \cdot 10^{-2}$	$2.54 \cdot 10^{-1}$	$1.36 \cdot 10^{-1}$	$4.14 \cdot 10^{-2}$

Table 4 The number of computations of Eq. (17) for the (127,64,21) BCH code with the function $\Delta(\cdot)$ and $f(\cdot)$.

E_b/N_0 [dB]	original	proposed		original	proposed	
	GS(Δ)	A(Δ)	B(Δ)	GS(f)	A(f)	B(f)
2.50	$6.98 \cdot 10^5$	$1.61 \cdot 10^5$	$1.30 \cdot 10^5$	$6.94 \cdot 10^5$	$1.61 \cdot 10^5$	$1.30 \cdot 10^5$
3.00	$1.78 \cdot 10^5$	$3.47 \cdot 10^4$	$1.55 \cdot 10^4$	$1.76 \cdot 10^5$	$3.47 \cdot 10^4$	$1.55 \cdot 10^4$
3.50	$2.05 \cdot 10^4$	$4.61 \cdot 10^3$	$1.55 \cdot 10^3$	$1.89 \cdot 10^4$	$4.61 \cdot 10^3$	$1.55 \cdot 10^3$
4.00	$2.69 \cdot 10^3$	$6.52 \cdot 10^2$	$1.63 \cdot 10^2$	$2.04 \cdot 10^3$	$6.52 \cdot 10^2$	$1.63 \cdot 10^2$
4.50	$4.15 \cdot 10^2$	$7.97 \cdot 10^1$	$1.87 \cdot 10^1$	$2.10 \cdot 10^2$	$7.97 \cdot 10^1$	$1.87 \cdot 10^1$
5.00	$7.92 \cdot 10^1$	$1.16 \cdot 10^1$	3.21	$2.44 \cdot 10^1$	$1.16 \cdot 10^1$	3.21
5.50	$1.86 \cdot 10^1$	1.37	$3.51 \cdot 10^{-1}$	2.73	1.37	$3.52 \cdot 10^{-1}$
6.00	4.95	$2.58 \cdot 10^{-1}$	$8.10 \cdot 10^{-2}$	$4.14 \cdot 10^{-1}$	$2.58 \cdot 10^{-1}$	$8.10 \cdot 10^{-2}$
6.50	1.30	$3.16 \cdot 10^{-2}$	$8.40 \cdot 10^{-3}$	$4.56 \cdot 10^{-2}$	$3.16 \cdot 10^{-2}$	$8.40 \cdot 10^{-3}$

GS(f) are almost the same. This relationship holds between A(Δ) and A(f) and between B(Δ) and B(f). By Table 3, the algorithm A(Δ) (or A(f)) computes metrics of less than 1/4 test error codewords for the (63,30,13) code compared with

the algorithm GS(Δ) (or GS(f)). i.e., Omitting Criterion A holds for more than 3/4 test error codewords. By Table 4, the value which Omitting Criterion A holds is also more than 3/4 for the (127,64,21) code. These results indicate that Omitting Criterion A works well and $\overline{c}_0$ is a good candidate as the initial codeword in the reliability-based decoding algorithms. The values for the algorithm B(Δ) (or B(f)) is less than that for the algorithm A(Δ) (or A(f)) at each SNR for both codes. This implies that Omitting Criterion B is, in general, more effective than Omitting Criterion A. If we use Omitting Criterion B, the number of computing Eq. (17) is less than 1/5 test error codewords for the (63,30,13) and the (127,64,21) codes.

Next we describe results at medium to high SNRs. The numbers of computing Eq. (17) for algorithms A(Δ), A(f), B(Δ) and B(f) decrease as the SNR increases for both codes. The difference between GS(Δ) and GS(f) becomes large at high SNRs. Nevertheless, the numbers of computing Eq. (17) in the algorithm A(Δ) and A(f) are almost the same and similar results hold for algorithm B(Δ) and B(f). These results indicate the effects of proposed conditions are independent of evaluation functions. It is noteworthy that, at high SNR, the numbers of computing Eq. (17) in the algorithm B(Δ) and B(f) are almost negligible for both (63,30,13) and (127,64,21) codes.

5. Concluding Remarks

In this paper, we have derived two sufficient conditions for omitting unnecessary metrics computations of candidate codewords in the reliability-based MLD algorithms. A simple method for testing the proposed conditions is presented. For implementation of this method, we need no real number operations. The results of computer simulations show the effectiveness of the proposed conditions for the (63,30,13) and the (127,64,21) BCH codes. As a result, we can reduce the number of real number operations which is one of the typical measures for evaluating the efficiency of MLD algorithms. The proposed conditions are applicable to any reliability-based MLD algorithms such as one in [6], [7], [12], [14], [17].

As future improvements, a rest of decoding complexity such as for encoding test error patterns should be reduced. A method that quantitatively reduces the number of metrics computations of candidate codewords is also to be developed.

Acknowledgement

The authors wish to thank the anonymous reviewers for their valuable comments. One of the authors, H. Yagi would like to thank Mr. T. Ishida and Mr. G. Hosoya at Waseda University for their supports.

This work is supported by Japan Society for the Promotion of Science under Grants-in-Aid for Scientific Research No. 1556-0338 and No. 1576-0281 and Waseda University Grant for Special Research Project No. 2001A-566.

References

[1] D. Chase, "A new class for decoding block codes with channel measurement information," IEEE Trans. Inf. Theory, vol.IT-18, no.1, pp.170–182, Jan. 1972.

[2] M.P.C. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," IEEE Trans. Inf. Theory, vol.41, no.5, pp.1379–1396, Sept. 1995.

[3] M.P.C. Fossorier, S. Lin, and J. Snyders, "Reliability-based syndrome decoding of linear block codes," IEEE Trans. Inf. Theory, vol.44, no.1, pp.388–398, Jan. 1998.

[4] M.P.C. Fossorier and S. Lin, "Reliability-based information set decoding of binary linear block codes," IEICE Trans. Fundamentals, vol.E82-A, no.10, pp.2034–2042, Oct. 1999.

[5] T. Fujiwara, H. Yamamoto, T. Kasami, and S. Lin, "A trellis-based recursive maximum likelihood decoding algorithm for binary linear block codes," IEEE Trans. Inf. Theory, vol.44, no.2, pp.714–729, March 1998.

[6] D. Gazelle and J. Snyders, "Reliability-based code-search algorithm for maximum likelihood decoding of block codes," IEEE Trans. Inf. Theory, vol.43, no.1, pp.239–249, Jan. 1997.

[7] Y.S. Han, C.P.R. Hartman, and C.C. Chen, "Efficient priority-first search maximum-likelihood soft decision decoding of linear block codes," IEEE Trans. Inf. Theory, vol.39, no.5, pp.1514–1523, Sept. 1993.

[8] T. Kaneko, T. Nishijima, H. Inazumi, and S. Hirasawa, "An efficient maximum-likelihood-decoding algorithm for linear block codes with algebraic decoder," IEEE Trans. Inf. Theory, vol.40, no.2, pp.320–327, March 1994.

[9] T. Kasami, Y. Tang, T. Koumoto, and T. Fujiwara, "Sufficient conditions for ruling-out useless iteration steps in a class of iterative decoding algorithms," IEICE Trans. Fundamentals, vol.E82-A, no.10, pp.2061–2073, Oct. 1999.

[10] T. Koumoto, T. Kasami, and S. Lin, "A sufficient condition for ruling out some useless test error patterns in iterative decoding algorithms," IEICE Trans. Fundamentals, vol.E81-A, no.2, pp.321–326, Feb. 1998.

[11] A. Lafourcade and A. Vardy, "Optimal sectionalization of a trellis," IEEE Trans. Inf. Theory, vol.42, no.3, pp.689–703, May 1996.

[12] T. Okada, M. Kobayashi, and S. Hirasawa, "An efficient heuristic search method for maximum likelihood decoding of linear block codes using dual codes," IEICE Trans. Fundamentals, vol.E85-A, no.2, pp.485–489, Feb. 2002.

[13] C.C. Shih, C.R. Wulff, C.R.P. Hartmann, and C.K. Mohan, "Efficient heuristic search algorithms for soft-decision decoding of linear block codes," IEEE Trans. Inf. Theory, vol.44, no.6, pp.3023–3038, Nov. 1998.

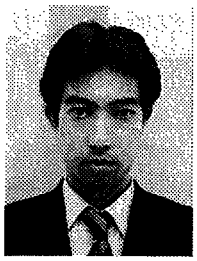
[14] A. Valembois and M.P.C. Fossorier, "An improved method to compute lists of binary vectors that optimize a given weight function with application of soft-decision decoding," IEEE Commun. Lett., vol.5, no.11, pp.456–458, Nov. 2001.

[15] A. Valembois and M.P.C. Fossorier, "A comparison between "most-reliable-basis reprocessing" strategies," IEICE Trans. Fundamentals, vol.E85-A, no.7, pp.1727–1741, July 2002.

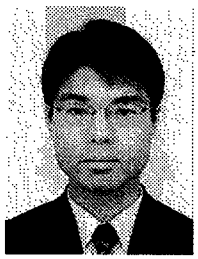
[16] Y. Wu and D.A. Pados, "An adaptive two-stage algorithm for ML and sub-ML decoding of binary linear block codes," IEEE Trans. Inf. Theory, vol.49, no.1, pp.261–269, Jan. 2003.

[17] H. Yagi, M. Kobayashi, and S. Hirasawa, "Complexity reduction of the Gazelle and Snyders decoding algorithm for maximum likelihood decoding," IEICE Trans. Fundamentals, vol.E86-A, no.10, pp.2461–2471, Oct. 2003.

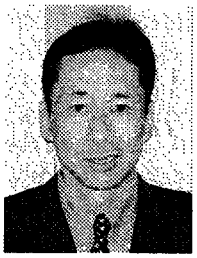
[18] H. Yagi, M. Kobayashi, and S. Hirasawa, "An improved method of maximum likelihood decoding algorithms using the most reliable basis based on an order relation among binary vectors," IEICE Technical Report, IT2003-6, May 2003.



Hideki Yagi was born in Yokohama, Japan, on Oct. 14, 1975. He received the B.E. degree and M.E. degree in Industrial and Management Systems Engineering from Waseda University, Tokyo, Japan, in 2001 and 2003, respectively. He is currently a doctorial student in Industrial and Management Systems Engineering at Graduate School of Waseda University. His research interests are coding and information theory.



Manabu Kobayashi was born in Yokohama, Japan, on Oct. 30, 1971. He received the B.E. degree, M.E. degree and Dr.E. degree in Industrial and Management Systems Engineering form Waseda University, Tokyo, Japan, in 1994, 1996 and 2000, respectively. From 1998 to 2001, he was a research associate in Industrial and Management Systems Engineering at Waseda University. He is currently a full-time lecturer of the Department of Information Science at Shonan Institute of Technology, Kanagawa, Japan. His research interests are coding and information theory and data mining. He is a member of the Society of Information Theory and Its Applications, Information Processing Society of Japan and IEEE.



Toshiyasu Matsushima was born in Tokyo, Japan, on Nov. 26, 1955. He received the B.E. degree, M.E. degree and Dr.E. degree in Industrial and Management Systems Engineering form Waseda University, Tokyo, Japan, in 1978, 1980 and 1991, respectively. From 1980 to 1986, he was with Nippon Electric Corporation, Kanagawa, Japan. From 1986 to 1992, he was a lecturer at Department of Management Information, Yokohama College of Commerce. From 1993, he was an associate professor and since 1996 has been a professor of School of Science and Engineering, Waseda University, Tokyo, Japan. His research interests are information theory and its application, statistics and artificial intelligence. He is a member of the Society of Information Theory and Its Applications, the Japan Society for Quality Control, the Japan Industrial Management Association, the Japan Society for Artificial Intelligence and IEEE.

Shigeichi Hirasawa was born in Kobe, Japan, on Oct. 2, 1938. He received the B.S. degree in mathematics and the B.E. degree in electrical communication engineering from Waseda University, Tokyo, Japan, in 1961 and 1963, respectively, and the Dr.E. degree in electrical communication engineering from Osaka University, Osaka, Japan, in 1975. From 1963 to 1981, he was with the Mitsubishi Electric Corporation, Hyogo, Japan. Since 1981, he has been a professor of School of Science and Engineering, Waseda University, Tokyo, Japan. In 1979, he was a Visiting Scholar in the Computer Science Department at the University of California, Los Angels (CSD, UCLA), CA. He was a Visiting Researcher at the Hungarian Academy of Science, Hungary, in 1985, and at the University of Trieste, Italy, in 1986. In 2002, he was also a Visiting Faculty at CSD, UCLA. From 1987 to 1989, he was the Chairman of Technical Group on Information Theory of IEICE. He received the 1993 Achievement Award, and the 1993 Kobayashi-Memorial Achievement Award from IEICE. In 1996, he was the President of the Society of Information Theory and Its Applications (Soc. of ITA). His research interests are information theory and its applications, and information processing systems. He is an IEEE Fellow, and a member of Soc. of ITA, the Operations Research Society of Japan, the Information Processing Society of Japan, the Japan Industrial Management Association, and Informs.

PAPER Special Section on Information Theory and Its Applications

A Heuristic Search Method with the Reduced List of Test Error Patterns for Maximum Likelihood Decoding*

Hideki YAGI[†,††a)], *Student Member*, Toshiyasu MATSUSHIMA[†], *Member, and* Shigeichi HIRASAWA[†], *Fellow*

SUMMARY The reliability-based heuristic search methods for maximum likelihood decoding (MLD) generate test error patterns (or, equivalently, candidate codewords) according to their heuristic values. Test error patterns are stored in lists and its space complexity is crucially large for MLD of long block codes. Based on the decoding algorithms both of Battail and Fang and of its generalized version suggested by Valembois and Fossorier, we propose a new method for reducing the space complexity of the heuristic search methods for MLD including the well-known decoding algorithm of Han et al. If the heuristic function satisfies a certain condition, the proposed method guarantees to reduce the space complexity of both the Battail-Fang and Han et al. decoding algorithms. Simulation results show the high efficiency of the proposed method.

key words: *maximum likelihood decoding, binary block codes, heuristic search, most reliable basis, reliability*

1. Introduction

Maximum likelihood decoding (MLD) of block codes minimizes the probability of decoding error when we assume that each codeword has the equal probability to be transmitted. Since the complexity of searching the most likely codeword is significantly large, many researchers have devoted to develop efficient algorithms for MLD of long block codes. One of the most efficient MLD algorithms is the reliability-based decoding algorithm that uses the column permuted generator matrix in non-increasing order of reliability.

In general, the reliability-based decoding algorithms are divided into two types due to the generation rule of candidate codewords. The first type of them generates the candidate codewords according to a predetermined generation rule [4], [5], [10]. The latter one is called the *heuristic search MLD algorithms* where candidate codewords are generated in increasing value of the heuristic function (also called the evaluation function) [1]–[3], [6]–[9]. Test error patterns (information sequences corresponding to candidate codewords) are generated and stored in lists before they are tested to be the most likely codeword. In this paper, we will consider the latter one. As known to the authors, G. Battail and J. Fang first proposed a heuristic search method for MLD over the additive white Gaussian noise (AWGN) channel [1] (we will call this method the BF decoding algorithm). Recently, in [8], [9], A. Valembois and M. Fossorier have indicated that a generalized version of the BF decoding algorithm is equivalent to the well-known A* decoding algorithm proposed by Y.S. Han et al. [2]. The generalized BF (GBF) decoding algorithm is a prominent and effective algorithm which can deal with almost all heuristic functions ever proposed.

For heuristic search MLD algorithms, their memory management is the critical issue since the maximum list size of test error patterns (TEPs), which dominates the space complexity, becomes quite large as the signal to noise ratio (SNR) of the channel decreases. There are roughly three approaches to reduce the maximum list size of TEPs in heuristic search MLD algorithms: (i) Some studies have proposed effective heuristic functions of TEPs to early terminate decoding procedure before the list of TEPs becomes very large [6], [7]. (ii) Some studies have proposed techniques for reducing the maximum list size of TEPs employing conventional heuristic functions [8]. (iii) Some studies have discarded the optimality of decoding while the complexity of decoding is drastically reduced [3].

Valembois et al. have taken the second approach. They have proposed a technique which considerably reduces the maximum list size of TEPs of the original BF decoding algorithm which imposes some condition for heuristic functions [8]. However, their technique cannot be adopted to the GBF decoding algorithm in which the search is guided by more effective heuristic functions than that considered in [1].

In this paper, we also consider the second approach and propose a method for reducing the maximum list size of TEPs of the GBF decoding algorithm. Similarly to the Valembois' approach, we first define a condition of heuristic functions. We show that the defined condition is satisfied by most of well-known heuristic functions. Then, we propose the improved method for the GBF decoding algorithm when the heuristic function satisfies the defined condition. We also devise the adaptive procedure of the proposed method where the heuristic function is updated as decoding proceeds. Proposed methods guarantee to reduce the maximum list size of TEPs of the GBF decoding algorithm. The number of TEPs generated and stored in lists are reduced and so they also reduce the time complexity of the GBF decoding algorithm. We also show by computer simulations that the space complexity of the GBF decoding (or equiva-

Manuscript received January 24, 2005.
Manuscript revised April 22, 2005.
Final manuscript received June 13, 2005.

[†]The authors are with the Department of Industrial and Management Systems Engineering, Waseda University, Tokyo, 169-8555 Japan.
[††]The author is with Media Network Center, Waseda University, Tokyo, 169-0051 Japan.

*The content of this paper is partly based on [11], [12].

a) E-mail: yagi@hirasa.mgmt.waseda.ac.jp

DOI: 10.1093/ietfec/e88–a.10.2721

lently, the A* decoding) algorithm is significantly reduced.

This paper is organized as follows. In Sect. 2, we briefly review general reliability-based MLD algorithms. In Sect. 3, we describe some heuristic functions and the GBF decoding algorithm. In Sect. 4 and 5, we propose new methods for reducing the space complexity of the GBF decoding algorithms. In Sect. 6, we show some simulation results and finally we state the concluding remarks in Sect. 7.

2. Reliability-Based MLD Algorithm

Let C be a binary linear (n, k, d) block code of the code length n, the number of information symbols k and the minimum distance d. We denote a generator matrix of C by G and the weight profile of C by $W(C)$. We assume any codewords $\boldsymbol{c} = (c_1, c_2, \ldots, c_n) \in \{0,1\}^n$ of C are transmitted over the AWGN channel. The receiver maps the received sequence $\boldsymbol{r} = (r_1, r_2, \ldots, r_n) \in \mathcal{R}^n$ into the reliability sequence $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_n)$, $\theta_j = \ln \frac{P(r_j|c_j=0)}{P(r_j|c_j=1)}$, where $P(r_j|c_j)$ represents the likelihood of the symbol† c_j. Furthermore, the hard-decision received sequence $\boldsymbol{z} = (z_1, z_2, \ldots, z_n) \in \{0,1\}^n$ is obtained by setting $z_j = 0$ if $\theta_j \geq 0$ and $z_j = 1$ otherwise. The decoder estimates the transmitted codeword both from $\boldsymbol{\theta}$ and $\boldsymbol{z}$.

In reliability-based decoding algorithms, we permute columns of a generator matrix in non-increasing order of reliability so that the leftmost k positions are the *most reliable and linearly independent* (MRI) [2], [6], [8], [9]. The other columns outside the k MRI positions are also reordered in non-increasing order of reliability, i.e., $|\theta_{j_1}| \geq |\theta_{j_2}|$ for $1 \leq j_1 < j_2 \leq k$ and for $k+1 \leq j_1 < j_2 \leq n$. We perform the standard row operations with respect to the permuted matrix to make the leftmost k columns the identity matrix. We denote the resultant matrix by $\tilde{G}$.

Let $\tilde{\boldsymbol{\theta}} = (\tilde{\theta}_1, \tilde{\theta}_2, \ldots, \tilde{\theta}_n)$ and $\tilde{\boldsymbol{z}} = (\tilde{z}_1, \tilde{z}_2, \ldots, \tilde{z}_n)$ be permuted sequences of $\boldsymbol{\theta}$ and $\boldsymbol{z}$, respectively, in the same ordering of columns of $\tilde{G}$. Let $\tilde{C}$ be the code generated by $\tilde{G}$ which is equivalent to C. Define $\boldsymbol{u} = (u_1, u_2, \ldots, u_k) \in \{0,1\}^k$ as the leftmost k symbols of $\tilde{\boldsymbol{z}}$, i.e., $u_j = \tilde{z}_j, \forall j \in [1, k]$. The decoder first encodes $\boldsymbol{u}$ by $\tilde{G}$ to obtain the initial codeword $\tilde{\boldsymbol{c}}_{\emptyset} (= \boldsymbol{u}\tilde{G})$. Afterwards, k-dimensional vectors, called *test error patterns* $\boldsymbol{t} \in \{0,1\}^k$, are iteratively generated and encoded by $\tilde{G}$. Then, $\tilde{\boldsymbol{c}} = \tilde{\boldsymbol{c}}_{\emptyset} \oplus \boldsymbol{t}\tilde{G}$ is a candidate codeword††. This procedure is repeated until a sufficient condition for the optimality is satisfied.

Definition 1: For a position set $J \subseteq [1, k]$, the test error pattern (TEP) $\boldsymbol{t}(J) = \left(t_1, t_2, \ldots, t_k\right) \in \{0,1\}^k$ has element one in J and element zero in the complement of J. Such J is called the *support* of $\boldsymbol{t}(J)$. Define that $\mu(J)$ be the rightmost position in J, i.e., $\mu(J) = \max J$. For $j > \mu(J)$, the TEP $\boldsymbol{t}(J \cup \{j\})$ (or simply $\boldsymbol{t}(J \cup j)$) is called an *extended pattern* of $\boldsymbol{t}(J)$. For any J, define $J^{\mathrm{a}} = J \setminus \mu(J)$. For J and $\mu(J^{\mathrm{a}}) < j < \mu(J)$, the TEP $\boldsymbol{t}(J^{\mathrm{a}} \cup j)$ is called an *adjacent pattern* of $\boldsymbol{t}(J)$ in j. □

Example 1: Assuming $k = 7$ and $J = \{2, 5\}$, then the TEP $\boldsymbol{t}(J)$ with the support J is $\boldsymbol{t}(J) = (0,1,0,0,1,0,0)$. Since $\mu(J) = 5$, there exist two extended patterns of $\boldsymbol{t}(J)$: $\boldsymbol{t}(J \cup 6)$ and $\boldsymbol{t}(J \cup 7)$. We find $J^{\mathrm{a}} = \{2\}$ and there also exist two adjacent patterns of $\boldsymbol{t}(J)$ in the position $j = 3, 4$: $\boldsymbol{t}(J^{\mathrm{a}} \cup 3) = (0,1,1,0,0,0,0)$ and $\boldsymbol{t}(J^{\mathrm{a}} \cup 4) = (0,1,0,1,0,0,0)$. □

For a binary vector $\boldsymbol{v} = (v_1, v_2, \ldots, v_n) \in \{0,1\}^n$, we define the *correlation discrepancy* [8], [9] of $\boldsymbol{v}$ as

$$L(\boldsymbol{v}) = \sum_{j|\tilde{z}_j \neq v_j} |\tilde{\theta}_j|. \tag{1}$$

It is well-known that $\tilde{\boldsymbol{c}}_{\mathrm{best}}$ is the most likely codeword if and only if $L(\tilde{\boldsymbol{c}}_{\mathrm{best}}) = \min_{\tilde{\boldsymbol{c}} \in \tilde{C}} L(\tilde{\boldsymbol{c}})$ [8], [10].

3. The Generalized BF Decoding Algorithm

3.1 Heuristic Functions of the Search

The methods considered in this paper generate TEPs according to their *heuristic values* (or *heuristics*). Here, we review heuristic functions which are used for searching the most likely codeword in [1]–[4], [8], [10].

Definition 2: For a TEP $\boldsymbol{t}(J)$, any function $F\left(\boldsymbol{t}(J)\right)$ satisfying

$$0 \leq F\left(\boldsymbol{t}(J)\right) \leq L(\tilde{\boldsymbol{c}}_J), \tag{2}$$

where $\tilde{\boldsymbol{c}}_J = (\tilde{c}_{J,1}, \tilde{c}_{J,2}, \ldots, \tilde{c}_{J,n}) = \tilde{\boldsymbol{c}}_{\emptyset} \oplus \boldsymbol{t}(J)\tilde{G}$, is called the *heuristic function* of the TEP. i.e., the heuristic value of $\boldsymbol{t}(J)$ is a lower bound of the discrepancy of $\tilde{\boldsymbol{c}}_J$. □

For a TEP $\boldsymbol{t}(J)$, the most simple heuristic function may be the correlation discrepancy over the k MRI positions defined as

$$\Delta\left(\boldsymbol{t}(J)\right) = \sum_{j \in J} |\tilde{\theta}_j|. \tag{3}$$

The function $\Delta(\cdot)$ actually satisfies Eq. (2), since $L(\tilde{\boldsymbol{c}}_J) = \Delta\left(\boldsymbol{t}(J)\right) + \sum_{j=k+1}^{n} (\tilde{z}_j \oplus \tilde{c}_{J,j}) |\tilde{\theta}_j|$. This heuristic function is used in [1], [4], [8], [10].

The heuristic function in [2], [3] utilizes the fact that any codeword in $\tilde{C}$ is at a distance $i \in W(\tilde{C})$ from a given codeword $\tilde{\boldsymbol{c}}_{\mathrm{ref}} \in \tilde{C}$. For $\tilde{\boldsymbol{c}}_{\mathrm{ref}} \in \tilde{C}$ and $\boldsymbol{t} = (t_1, t_2, \ldots, t_k)$, we define†††

$$T\left(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}\right) = \Big\{ \boldsymbol{v} = (\boldsymbol{u} \oplus \boldsymbol{t}) \| (v_{k+1}, v_{k+2}, \ldots, v_n) \Big| d_H(\boldsymbol{v}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}) \in W(\tilde{C}) \Big\}, \tag{4}$$

where $d_H(\cdot, \cdot)$ represents the Hamming distance. If we do not know the exact weight profile $W(\tilde{C})$, which case is often occurred for long block codes, then we can substitute it by its superset. Then the heuristic function in [2], [3] is defined

†Since the probability of decision error of z_j becomes smaller as the value of $|\theta_j|$ is larger, $|\theta_j|$ is called reliability.

††The symbol $\oplus$ represents Exclusive OR operation.

†††The symbol $\|$ represents concatenation of vectors.

as

$$f(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}) = \sum_{j|t_j=1} |\tilde{\theta}_j| + \min_{\boldsymbol{v}\in T(\boldsymbol{t},\tilde{\boldsymbol{c}}_{\mathrm{ref}})} \left\{ \sum_{j|\tilde{z}_j\neq v_j} |\tilde{\theta}_j| \right\}. \tag{5}$$

Such $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$ is called the *referenced codeword* [4], [5], [8] or the *seed* [2], [3], [6], [7].

We describe another heuristic function proposed by Fossorier and Lin [5]. For $\tilde{\boldsymbol{c}}_{\mathrm{ref}} \in \tilde{C}$ and $\boldsymbol{t}$, we define

$$T_F\left(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}\right) = \Big\{ \boldsymbol{v} = (\boldsymbol{u} \oplus \boldsymbol{t}) \| (v_{k+1}, v_{k+2}, \ldots, v_n) \Big| \; d_H(\boldsymbol{v}, \tilde{\boldsymbol{z}}) + d_H(\tilde{\boldsymbol{c}}_{\mathrm{ref}}, \tilde{\boldsymbol{z}}) \in W'(\tilde{C}) \Big\}, \tag{6}$$

where $W'(\tilde{C}) = \{0, d, d+1, \ldots, n\}$ is the superset of the weight profile $W(\tilde{C})$. Then the heuristic function in [5] is expressed as

$$g\left(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}\right) = \sum_{j|t_j=1} |\tilde{\theta}_j| + \min_{\boldsymbol{v}\in T_F(\boldsymbol{t},\tilde{\boldsymbol{c}}_{\mathrm{ref}})} \left\{ \sum_{j|\tilde{z}_j\neq v_j} |\tilde{\theta}_j| \right\}. \tag{7}$$

Note that since

$$d_H(\boldsymbol{v}, \tilde{\boldsymbol{z}}) = w_H(\boldsymbol{t}) + \#\{j|\tilde{z}_j \neq v_j\}, \tag{8}$$

the sequence $\boldsymbol{v} \in T_F\left(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}\right)$ which minimizes the second term of r.h.s. of Eq. (7) is determined only by the Hamming weight $w_H(\boldsymbol{t})$ [5], [9] where w_H and $\#\{\cdot\}$ represent the Hamming weight and the cardinality, respectively. In [5], Fossorier et al. have devised a method for making the function $g(\cdot)$ more effective according to updating the referenced codeword. For details, see [5].

The heuristic function is also used for reducing the time complexity of decoding procedure. Denote a currently best candidate codeword by $\tilde{\boldsymbol{c}}^*$. We note that if $F\left(\boldsymbol{t}(J)\right) \geq L(\tilde{\boldsymbol{c}}^*)$ for a TEP $\boldsymbol{t}(J)$, the candidate codeword $\tilde{\boldsymbol{c}}_J$ cannot be the most likely codeword because of Eq. (2). Hence, if all TEPs not encoded so far satisfy $F\left(\boldsymbol{t}(J)\right) \geq L(\tilde{\boldsymbol{c}}^*)$, then the sufficient condition for the optimality holds and we can terminate the decoding procedure. The tighter the lower bound of $L(\tilde{\boldsymbol{c}}_J)$ is, the more effective a sufficient condition for the optimality is. Since $f\left(\boldsymbol{t}(J), \tilde{\boldsymbol{c}}_{\mathrm{ref}}\right) \geq \Delta\left(\boldsymbol{t}(J)\right)$ for any $\boldsymbol{t}(J)$, $f(\cdot)$ can give a tighter sufficient condition for the optimality than $\Delta(\cdot)$.

3.2 Generation Method of TEPs

We state how to dynamically generate TEPs according to their heuristic values. We will call the search strategies which process TEPs in the increasing order of their heuristic values the *priority-first search* [2], [3], [7].

The well-known MLD algorithm via the priority-first search is the A* decoding algorithm [2] in which the search is conducted by the A* algorithm through trellis or binary tree of the code. Although the A* decoding algorithm employs the function $f(\cdot)$, it performs the priority-first search with any heuristic functions $F(\cdot)$ satisfying the following condition:

(C1) $F\left(\boldsymbol{t}(J)\right) \leq F\left(\boldsymbol{t}(J \cup j)\right)$ for $j \notin J$.

The heuristic functions $\Delta(\cdot)$, $f(\cdot)$ and $g(\cdot)$ actually satisfy the condition (C1) [9].

Other well-known MLD algorithm via the priority-first search is the BF decoding algorithm [1] which requires heuristic functions to satisfy not only the condition (C1) but also the condition:

$$F\left(\boldsymbol{t}(J)\right) \leq F\left(\boldsymbol{t}(J')\right) \Rightarrow F\left(\boldsymbol{t}(J \cup j)\right) \leq F\left(\boldsymbol{t}(J' \cup j)\right) \quad \text{for } j \notin J \cup J'. \tag{9}$$

It is readily shown that the function $\Delta(\cdot)$ satisfies Eq. (9) while the functions $f(\cdot)$ and $g(\cdot)$ do not necessarily satisfy it [8].

In [8], [9], Valembois et al. have shown that we can easily generalize the BF decoding algorithm to perform the priority-first search when the heuristic function satisfies only the condition (C1).

Hereafter, we assume heuristic functions satisfy (C1) and we will describe the GBF decoding algorithm. Let $M^{(1)}, M^{(2)}, \ldots, M^{(k)}$ represent k lists of TEPs. The TEP $\boldsymbol{t}(J)$ is supposed to be in $M^{(\mu(J))}$ where $\mu(J) = \max J$. Then the list for storing any TEP is uniquely determined. In a list $M^{(j)}, \forall j \in [1, k]$, TEPs are ordered in increasing order of their heuristic values. We call the TEP with the minimum heuristic value among all TEPs in lists the *best pattern*. The algorithm iteratively selects the best pattern, encodes it by $\tilde{G}$ and deletes it from lists. If there needs to generate new TEPs which have been not processed yet, the algorithm generates them. The basic strategy of generating TEPs is such that any TEP $\boldsymbol{t}(J)$ is not generated while we know that better patterns than $\boldsymbol{t}(J)$ are stored in lists or not generated so far.

In the initial stage of the algorithm, we construct the initial list of TEPs as follows: By the condition (C1), the TEPs with the minimum heuristic value in $M^{(j)}$, $j \in [1, k]$, is $\boldsymbol{t}(j)$ whose Hamming weight is one. i.e.,

$$\boldsymbol{t}(j) = \arg\min_{j=\mu(J)} \left\{ F\left(\boldsymbol{t}(J)\right) \right\} \tag{10}$$

for all $j \in [1, k]$. Therefore, we just need to set the initial lists as $M^{(j)} = \left\{\boldsymbol{t}(j)\right\}$ for $j \in [1, k]$. Thereafter, the algorithm selects the best patterns among TEPs that have not been processed†.

We here describe the GBF decoding algorithm.

[The generalized BF decoding algorithm]

S1) Set $\tilde{\boldsymbol{c}}_{\emptyset} := \boldsymbol{u}\tilde{G}$, $\tilde{\boldsymbol{c}}^* := \tilde{\boldsymbol{c}}_{\emptyset}$ and $\underline{L} := L(\tilde{\boldsymbol{c}}_{\emptyset})$. Construct the initial lists of TEPs.

S2) Select the best pattern $\boldsymbol{t}(J) \in M^{(\mu(J))}$ among the topmost TEPs in non-empty lists $M^{(j)}$. If $F(\boldsymbol{t}(J)) \geq \underline{L}$, then output $\tilde{\boldsymbol{c}}^*$ and halt the algorithm.

†In [8], Valembois et al. have devised the technique for selecting the best pattern by $O(\lceil \log k \rceil)$ comparisons.

S3) Generate the next candidate codeword by $\tilde{\boldsymbol{c}}_J := \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}(J)\tilde{G}$. If $L(\tilde{\boldsymbol{c}}_J) < \underline{L}$, then set $\underline{L} := L(\tilde{\boldsymbol{c}}_J)$ and $\tilde{\boldsymbol{c}}^* := \tilde{\boldsymbol{c}}_J$.

S4) For all lists $M^{(j)}$ such that $j > \mu(J)$, insert the extended patterns $\boldsymbol{t}(J \cup j)$ at the position such that the list remains increasing order of heuristic values. Delete $\boldsymbol{t}(J)$ from $M^{(\mu(J))}$.

S5) If $M^{(j)} = \emptyset$ for all $j \in [1, k]$, then output $\tilde{\boldsymbol{c}}^*$ and halt the algorithm. Otherwise, go to S2). □

In the above algorithm, S4) is the step of generating new TEPs which are extended patterns of $\boldsymbol{t}(J)$. We need to sort the generated TEP $\boldsymbol{t}(J \cup j)$ so that the list $M^{(j)}$ remains increasing order of the heuristic values. By sorting, the priority-first search is maintained.

The original BF decoding algorithm requires the heuristic functions to satisfy Eq. (9) as well as (C1). There we need not sort the new generated TEPs since Eq. (9) guarantees it is not better than any TEPs already stored in lists.

In the A* decoding algorithm, the only one list of TEPs is used. If we combine the k lists into the united list and order TEPs increasing order of heuristic values in it, then the above algorithm becomes identical to the A* decoding algorithm although the behaviors of the two algorithms seem different [8], [9]. Note that the essential properties are independent of the number of lists.

We here state the complexity of the GBF decoding algorithm. As for the space complexity, storing $\tilde{G}$ requires $O(kn)$ binary arrays. Denoting the maximum list size for decoding $\boldsymbol{r}$ by $M(\boldsymbol{r})$, the space complexity for lists is $O(k \times M(\boldsymbol{r}))$ binary arrays and $O(M(\boldsymbol{r}))$ arrays of real numbers. Therefore the overall space complexity is $O(\gamma)$ where $\gamma = \max\{kn, k \times M(\boldsymbol{r})\}$. If the maximum list size $M(\boldsymbol{r})$ is larger than n (which situations are usual from low to medium SNRs), the value $M(\boldsymbol{r})$ is dominant in the space complexity. It has been shown that the value $M(\boldsymbol{r})$ drastically increases as the SNR decreases.

As for the time complexity, permuting $\boldsymbol{\theta}$ in the non-increasing order of reliability costs $O(n \log n)$ comparisons and constructing $\tilde{G}$ costs $O(n \times \kappa^2)$ binary operations where $\kappa = \min\{k, n-k\}$ [2], [4], [5]. These steps are carried out only once in decoding of $\boldsymbol{r}$. Contrary to the above steps, generating $\boldsymbol{t}(J)$ and encoding them by $\tilde{G}$ are carried out iteratively, where each encoding requires $O(kn)$ binary operations by conventional encoding method [5], [6], [10]. For each TEP, computing its heuristic value costs real number operations of $O(n)$. Since a large number of TEPs are generated, both generating TEPs and the real number operations of heuristic values dominate mainly the whole decoding complexity [2], [6], [8] as well as encoding TEPs.

4. Proposed Decoding Algorithm

In this section, we propose a method for reducing the list size of TEPs in the GBF decoding algorithm. Before deriving the proposed method, we show some properties of conventional heuristic functions. These properties will be exploited by the proposed method.

4.1 Some Properties of Heuristic Functions

We here define the following condition for a heuristic function $F(\cdot)$.

Definition 3: Let $S^{(0)}$ be a certain subset of $[1, k]$ and $S^{(1)}$ be the complement of $S^{(0)}$. For $J \subseteq [1, k]$, assume $j_1, j_2 \notin J$ and $j_1 < j_2$. If $j_1, j_2 \in S^{(\alpha)}$ with $\alpha \in \{0, 1\}$, then a function $F(\cdot)$ satisfies

$$\textbf{(C2)} \qquad F\big(\boldsymbol{t}(J \cup j_1)\big) \geq F\big(\boldsymbol{t}(J \cup j_2)\big). \tag{11}$$

We will call this condition the condition (C2). □

The following propositions play important roles in deriving the improved method.

Proposition 1: Assume that $S^{(0)} = [1, k]$. Then the function $\Delta(\cdot)$ satisfies the condition (C2).

(**Proof**) Note that by Eq. (3), an extended pattern $\boldsymbol{t}(J \cup j)$ of $\boldsymbol{t}(J)$ such that $j \notin J$ satisfies

$$\Delta\big(\boldsymbol{t}(J \cup j)\big) = \Delta\big(\boldsymbol{t}(J)\big) + |\tilde{\theta}_j|. \tag{12}$$

If $1 \leq j_1 < j_2 \leq k$ and $j_2 \notin J$, then we have

$$\begin{aligned} &\Delta\big(\boldsymbol{t}(J \cup j_1)\big) - \Delta\big(\boldsymbol{t}(J \cup j_2)\big) \\ &\quad = \Delta\big(\boldsymbol{t}(J)\big) + |\tilde{\theta}_{j_1}| - \Delta\big(\boldsymbol{t}(J)\big) - |\tilde{\theta}_{j_2}| \geq 0 \end{aligned} \tag{13}$$

since $|\tilde{\theta}_{j_1}| \geq |\tilde{\theta}_{j_2}|$. By the assumption, both j_1 and j_2 are in $S^{(0)}(= [1, k])$ and thus the function $\Delta(\cdot)$ satisfies (C2). □

Proposition 2: For a given referenced codeword $\tilde{\boldsymbol{c}}_{\mathrm{ref}} \in \tilde{C}$, let $\boldsymbol{t}_{\mathrm{ref}}$ be the TEP of $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$ (i.e., $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}_{\mathrm{ref}}\tilde{G}$). Assuming that $S^{(1)}$ and $S^{(0)}$ be the support of $\boldsymbol{t}_{\mathrm{ref}}$ and its complement, respectively. Then the heuristic function $f(\cdot)$ satisfies the condition (C2).

In order to prove Proposition 2, we first show the following lemma.

Lemma 1: Denote the second term of the r.h.s. of Eq. (5) by $A(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}})$, i.e.,

$$A(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}) = \min_{\boldsymbol{v} \in T(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}})} \Big\{ \sum_{j | \tilde{z}_j \neq v_j} |\tilde{\theta}_j| \Big\}. \tag{14}$$

Then for a given $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$, any pairs $(\boldsymbol{t}, \boldsymbol{t}')$ such that $d_H(\boldsymbol{t}, \boldsymbol{t}_{\mathrm{ref}}) = d_H(\boldsymbol{t}', \boldsymbol{t}_{\mathrm{ref}})$ satisfy

$$A(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}) = A(\boldsymbol{t}', \tilde{\boldsymbol{c}}_{\mathrm{ref}}) \tag{15}$$

i.e., the vector $\boldsymbol{v}$ which gives the minimum value of Eq. (14) is determined only by the Hamming distance $d_H(\boldsymbol{t}, \boldsymbol{t}_{\mathrm{ref}})$.

(**Proof**) By the definition, $\boldsymbol{v} \in T\big(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}\big)$ satisfies

$$d_H(\boldsymbol{v}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}) = d_H(\boldsymbol{t}, \boldsymbol{t}_{\mathrm{ref}}) + \#\{j | v_j \neq \tilde{c}_{\mathrm{r},j}\} \tag{16}$$

where $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = (\tilde{c}_{\mathrm{r},1}, \tilde{c}_{\mathrm{r},2}, \ldots, \tilde{c}_{\mathrm{r},n})$. In r.h.s., the first term expresses the distance over the left k positions and the second

term does the distance over the rest of $n-k$ positions.

For t, if we assume that $v = (u \oplus t)\|(v_{k+1}^*, v_{k+2}^*, \ldots, v_n^*)$ minimizes the r.h.s. of Eq. (14), then v must satisfy $d_H(v, \tilde{c}_{\text{ref}}) \in W(\tilde{C})$ from the condition in Eq. (4). Note that we have

$$A(t, \tilde{c}_{\text{ref}}) = \sum_{j|v_j^* \neq \tilde{z}_j} |\tilde{\theta}_j|. \tag{17}$$

For another TEP t' such that $d_H(t', t_{\text{ref}}) = d_H(t, t_{\text{ref}})$, the vector $v' = (u \oplus t')\|(v_{k+1}^*, v_{k+2}^*, \ldots, v_n^*)$ satisfies

$$d_H(v', \tilde{c}_{\text{ref}}) = d_H(v, \tilde{c}_{\text{ref}}) \in W(\tilde{C}) \tag{18}$$

by Eq. (16). This equation implies $v' \in T(t', \tilde{c}_{\text{ref}})$ and

$$A(t', \tilde{c}_{\text{ref}}) = \sum_{j|v_j^* \neq \tilde{z}_j} |\tilde{\theta}_j|. \tag{19}$$

in which v' minimizes the r.h.s. of Eq. (14). Equations (17) and (19) complete the proof. □

(Proof of Proposition 2) Assuming that $j_1, j_2 \in S^{(0)}$ and $j_1, j_2 \notin J$, then the Hamming distance between TEPs $t(J \cup j_1)$ and t_{ref} and that between $t(J \cup j_2)$ and t_{ref} are the same since $S^{(0)}$ is the complement of the support of t_{ref}. i.e.,

$$d_H\big(t(J \cup j_1), t_{\text{ref}}\big) = d_H\big(t(J \cup j_2), t_{\text{ref}}\big). \tag{20}$$

Therefore, we have

$$A\big(t(J \cup j_1), \tilde{c}_{\text{ref}}\big) = A\big(t(J \cup j_2), \tilde{c}_{\text{ref}}\big) \tag{21}$$

from Lemma 1. Furthermore if $j_1 < j_2$, we have

$$\begin{aligned} &f\big(t(J \cup j_1), \tilde{c}_{\text{ref}}\big) - f\big(t(J \cup j_2), \tilde{c}_{\text{ref}}\big) \\ &\quad = \sum_{j \in J \cup j_1} |\tilde{\theta}_j| - \sum_{j \in J \cup j_2} |\tilde{\theta}_j| = |\tilde{\theta}_{j_1}| - |\tilde{\theta}_{j_2}| \geq 0. \end{aligned}$$

This inequality implies that the function $f(\cdot)$ satisfies (C2) if $j_1, j_2 \in S^{(0)}$. In the case of $j_1, j_2 \in S^{(1)}$, Eq. (20) also holds and we can prove the proposition similarly. □

Proposition 3: Assume that $S^{(0)} = [1, k]$. Then for a given $\tilde{c}_{\text{ref}}$, the function $g(\cdot)$ satisfies the condition (C2).

(Proof) Denote the second terms of r.h.s. of Eq. (7) by $B(t, \tilde{c}_{\text{ref}})$ for a given t. Recall that the vector $v \in T_F(t, \tilde{c}_{\text{ref}})$ which takes the value $B(t, \tilde{c}_{\text{ref}})$ is determined only by $w_H(t)$ (see Sect. 3.1). i.e., arbitrary pairs (t, t') with $w_H(t) = w_H(t')$ satisfy

$$B(t, \tilde{c}_{\text{ref}}) = B(t', \tilde{c}_{\text{ref}}). \tag{22}$$

Since $t(J \cup j_1)$ and $t(J \cup j_2)$ with $j_1, j_2 \notin J$ have the same Hamming weight, if $1 \leq j_1 < j_2 \leq k$, then

$$\begin{aligned} &g\big(t(J \cup j_1)\big) - g\big(t(J \cup j_2)\big) \\ &\quad = \Delta\big(t(J \cup j_1)\big) - \Delta\big(t(J \cup j_2)\big) \geq 0 \end{aligned} \tag{23}$$

where the last inequality is obtained from Eq. (13). By the assumption, both j_1 and j_2 must be in $S^{(0)}(= [1, k])$ and thus the function $g(\cdot)$ satisfies (C2). □

Propositions 1, 2 and 3 show that the heuristic functions $\Delta(\cdot)$, $f(\cdot)$ and $g(\cdot)$ satisfy the condition (C2) as well as (C1). In the following, we consider heuristic functions satisfying both (C1) and (C2).

4.2 Improved Generation Method of TEPs

In this section, we propose an improved method for reducing the list size of TEPs in the GBF decoding algorithm. For our purpose, we utilize the condition (C2) as well as (C1) to judge unnecessary TEPs and such unnecessary TEPs will not be generated as long as possible. More precisely, we regard a TEP t as *unnecessary* if it is clear that there is a TEP t' whose heuristic value is smaller than that of t in the lists. In the improved method, such an unnecessary TEP t is generated after the TEP t' is chosen as the best pattern at S2). This approach is similar to the improved technique for the original BF decoding algorithm† [8].

We also arrange k lists $M^{(j)}$ as in the GBF decoding algorithm. Hereafter, we denote $S^{(0)} = \{i_1, i_2, \ldots, i_s\}$ with $s \geq 1$ and $S^{(1)} = \{i'_1, i'_2, \ldots, i'_p\}$ with $p \geq 0$.

By the condition (C1), the TEP with the minimum heuristic value in a list $M^{(j)}$, $j \in [1, k]$, is $t(j)$ whose Hamming weight is one. Furthermore, we can see that the best pattern among s TEPs $t(j), j \in S^{(0)}$, is $t(i_s)$ by the condition (C2). Similarly, the best pattern among p TEPs $t(j), j \in S^{(1)}$, is $t(i'_p)$. Therefore, we can construct the initial lists as

$$M^{(j)} = \begin{cases} \{t(j)\}, & \text{if } j \in \{i_s, i'_p\}; \\ \emptyset, & \text{otherwise.} \end{cases} \tag{24}$$

Note that we generate at most two TEPs at this stage.

At S2) of the GBF decoding algorithm, if $t(J) \in M^{(\mu(J))}$ is selected as the best pattern, $k - \mu(J)$ extended patterns of $t(J)$ will be stored at S4). However, it is enough to store only its extended patterns $t(J \cup i_s)$ and $t(J \cup i'_p)$ in the list $M^{(i_s)}$ and $M^{(i'_p)}$, respectively. This is guaranteed by (C2), since

$$F\big(t(J \cup j)\big) \geq F\big(t(J \cup i_s)\big), \quad \text{for } \forall j \in S^{(0)} \tag{25}$$

$$F\big(t(J \cup j)\big) \geq F\big(t(J \cup i'_p)\big), \quad \text{for } \forall j \in S^{(1)} \tag{26}$$

where $t(J \cup j)$ represents extended patterns of $t(J)$.

Following this modification, we need to determine when to insert other extended patterns $t(J \cup j), j \notin \{i_s, i'_p\}$, into lists. Consider that a TEP $t(J \cup i_q)$ such that $i_q \in S^{(0)}$ and $i_q > \mu(J)$ is stored in the list $M^{(i_q)}$. Since adjacent patterns $t(J \cup j)$ such that $j \in S^{(0)}$ and $j < i_q$ cannot be the best pattern by (C2), we just need to store these adjacent patterns after $t(J \cup i_q)$ is selected as the best pattern at S2). If $i_{q-1} > \mu(J)$, $t(J \cup i_{q-1})$ has the smallest heuristic value

†Note again that the original BF decoding algorithm requires Eq. (9) for heuristic functions which is not satisfied by the function $f(\cdot)$ and $g(\cdot)$.

among all adjacent patterns of $\boldsymbol{t}(J \cup i_q)$ in $S^{(0)}$ from the condition (C2), i.e.,

$$\boldsymbol{t}(J \cup i_{q-1}) = \arg\min_{j \in S^{(0)}} \left\{ F\big(\boldsymbol{t}(J \cup j)\big) \,\middle|\, j < i_q, j \notin J \right\}. \quad (27)$$

Therefore, after $\boldsymbol{t}(J \cup i_q)$ is selected as the best pattern at S2), only $\boldsymbol{t}(J \cup i_{q-1})$ is inserted into the list $M^{(i_{q-1})}$. This modification significantly reduces the maximum list size. Similar arguments also hold when $\boldsymbol{t}(J \cup i'_q), i'_q \in S^{(1)}$, is selected as the best pattern at S2).

We describe a proposed decoding algorithm employing the above method.

[The proposed decoding algorithm]

P1) Set $\tilde{\boldsymbol{c}}_\emptyset := \boldsymbol{u}\tilde{G}$, $\tilde{\boldsymbol{c}}^* := \tilde{\boldsymbol{c}}_\emptyset$ and $\underline{L} := L(\tilde{\boldsymbol{c}}_\emptyset)$. Construct the initial lists of TEPs by Eq. (24).

P2) Select the best pattern $\boldsymbol{t}(J) \in M^{(\mu(J))}$ among non-empty lists. If $F(\boldsymbol{t}(J)) \geq \underline{L}$, then output $\tilde{\boldsymbol{c}}^*$ and halt the algorithm.

P3) Generate the next candidate codeword by $\tilde{\boldsymbol{c}}_J := \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}(J)\tilde{G}$. If $L(\tilde{\boldsymbol{c}}_J) < \underline{L}$, then set $\underline{L} := L(\tilde{\boldsymbol{c}}_J)$ and $\tilde{\boldsymbol{c}}^* := \tilde{\boldsymbol{c}}_J$.

P4) a) If $\mu(J) = i_q$ (i.e., $\mu(J) \in S^{(0)}$) and the adjacent pattern $\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1})$ exists where $J^{\mathrm{a}} = J \setminus \mu(J)$, then insert it into the list $M^{(i_{q-1})}$.

b) If $\mu(J) = i'_q$ (i.e., $\mu(J) \in S^{(1)}$) and the adjacent pattern $\boldsymbol{t}(J^{\mathrm{a}} \cup i'_{q-1})$ exists, then insert it into the list $M^{(i'_{q-1})}$.

c) If $\mu(J) < i_s$, then insert $\boldsymbol{t}(J \cup i_s)$ into $M^{(i_s)}$. If $\mu(J) < i'_p$, then insert $\boldsymbol{t}(J \cup i'_p)$ into $M^{(i'_p)}$. Delete $\boldsymbol{t}(J)$ from $M^{(\mu(J))}$.

P5) If $M^{(j)} = \emptyset$ for all $j \in [1, k]$, then output $\tilde{\boldsymbol{c}}^*$ and halt the algorithm. Otherwise, go to P2). □

The step P4) corresponds to the modification. Note that we need to store at most three TEPs at P4), while we need to store at most $k - \mu(J)$ TEPs at S4) of the GBF decoding algorithm.

Remark that we set $S^{(0)} = [1, k]$ by Propositions 1 and 3 if we employ either the function $\Delta(\cdot)$ or $g(\cdot)$. Since $S^{(1)} = \emptyset$, we can skip P4-b) and at most two TEPs (one is an adjacent pattern and the other is an extended pattern) are generated for each iteration (one iteration consists of selecting the best pattern, encoding it and generating new TEPs).

Example 2: Assuming $k = 7$ and $S^{(0)} = \{2, 4, 5\}$, let $\boldsymbol{t}(J) = (1, 0, 0, 1, 0, 0, 0)$ be the best pattern selected at P2). Since $\mu(J) = 4 \in S^{(0)}$, the adjacent pattern $\boldsymbol{t}(J^{\mathrm{a}} \cup 2) = (1, 1, 0, 0, 0, 0, 0)$ at the position $j = 2$ is inserted into the list $M^{(2)}$ at P4-a). Since $\mu(J) < i_s = 5$ and $\mu(J) < i'_p = 7$, extended patterns $\boldsymbol{t}(J \cup 5) = (1, 1, 0, 0, 1, 0, 0)$ and $\boldsymbol{t}(J \cup 7) = (1, 1, 0, 0, 0, 0, 1)$ of $\boldsymbol{t}(J)$ are inserted into the lists $M^{(5)}$ and $M^{(7)}$, respectively, at P4-c). □

We note that the next generated TEP $\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1})$ at P4-a) can be easily computed from the selected best pattern $\boldsymbol{t}(J)(= \boldsymbol{t}(J^{\mathrm{a}} \cup i_q))$. Furthermore its heuristic value $F\big(\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1})\big)$ may be easily calculated from that of $\boldsymbol{t}(J)$. For example, if we adopt the function $\Delta(\cdot)$, the heuristic values of $\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1})$ is calculated as

$$\Delta\big(\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1})\big) = \Delta\big(\boldsymbol{t}(J^{\mathrm{a}} \cup i_q)\big) - |\tilde{\theta}_{i_q}| + |\tilde{\theta}_{i_{q-1}}|. \quad (28)$$

Similarly, if we adopt the function $f(\cdot)$, the heuristic values of $\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1}), i_{q-1} \in S^{(0)}$, is calculated as

$$f\big(\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1}), \tilde{\boldsymbol{c}}_{\mathrm{ref}}\big) = f\big(\boldsymbol{t}(J^{\mathrm{a}} \cup i_q), \tilde{\boldsymbol{c}}_{\mathrm{ref}}\big) - |\tilde{\theta}_{i_q}| + |\tilde{\theta}_{i_{q-1}}| \quad (29)$$

from Eq. (5) and Lemma 1. The heuristic value of $\boldsymbol{t}(J^{\mathrm{a}} \cup i'_{q-1}), i'_{q-1} \in S^{(1)}$, can also be calculated by that of $\boldsymbol{t}(J^{\mathrm{a}} \cup i'_q), i'_q \in S^{(1)}$, since the similar relationship as Eq. (29) holds between $\boldsymbol{t}(J^{\mathrm{a}} \cup i'_{q-1})$ and $\boldsymbol{t}(J^{\mathrm{a}} \cup i'_q)$. As for the function $g(\cdot)$, if two TEPs have the same Hamming weight, the values $B(\cdot)$ of them take the same value. We can also calculate $f(\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1}), \tilde{\boldsymbol{c}}_{\mathrm{ref}})$ by the similar way of Eq. (29).

We show the validity of the proposed decoding algorithm.

Theorem 1: Assume that a heuristic function $F(\cdot)$ satisfies both (C1) and (C2). The ν-th iteration of the proposed decoding algorithm selects the TEP with the ν-th least heuristic value among all TEPs. i.e., it performs the priority-first search.

(**Proof**) See Appendix A. □

The foregoing theorem also implies that the proposed decoding algorithm performs MLD by Eq. (2).

Corollary 1: If we employ either $\Delta(\cdot)$, $f(\cdot)$ or $g(\cdot)$ as the heuristic function, the proposed decoding algorithm performs the priority-first search and achieves MLD. □

In terms of the space complexity of the proposed decoding algorithm, we show a lemma and a theorem.

Lemma 2: In each iteration of the proposed decoding algorithm, the list size of TEPs is no more than that in the GBF decoding algorithm if both decoding algorithms employ the same heuristic function satisfying (C1) and (C2).

(**Proof**) From Theorem 1, both the GBF and the proposed decoding algorithms perform the priority-first search. Therefore, if we employ the same heuristic function, the numbers of TEPs selected as the best pattern at S2) and S4) (these numbers are equal to those of encoding TEPs) are identical.

Based on the above fact, we will prove the lemma by the mathematical induction. We denote the iteration number by ν.

(i) The case of $\nu = 1$:
The initial list constructed by Eq. (24) guarantees the list size of the proposed decoding algorithm is less than that in the GBF decoding algorithm.

(ii) The case of $\nu \geq 2$:
Assume that the list size in the $(\nu-1)$-th iteration of the proposed decoding algorithm is less than or equal to that

of the GBF decoding algorithm. When $t(J)$ is selected as the best pattern in the ν-th iteration, all of its $k - \mu(J)$ extended patterns will be stored in lists at the step S4) in the GBF decoding algorithm, while at most two extended patterns of $t(J)$ will be stored in lists at P4) of the same iteration in the proposed decoding algorithm. Furthermore, if proposed decoding algorithm needs to store the adjacent pattern of $t(J)$, such adjacent pattern has been already stored in lists in the GBF decoding algorithm. Therefore, the list size in the ν-th iteration of the proposed decoding algorithm is less than or equal to that of the GBF decoding algorithm.

From the arguments (i) and (ii), we can prove the lemma. □

Theorem 2: The maximum list size of TEPs in the proposed decoding algorithm is no more than that in the GBF decoding algorithm if both decoding algorithms employ the same heuristic function satisfying (C1) and (C2).

(**Proof**) We can readily prove the theorem by Lemma 2. □

We show the following theorem on the time complexity. Note that the number of TEPs generated in a decoding procedure is in general greater than the maximum list size and it is one of the indices to evaluate the time complexity of heuristic search MLD algorithms [2], [8].

Theorem 3: The number of generated TEPs in the proposed decoding algorithm is no more than that in the GBF decoding algorithm if both decoding algorithms employ the same heuristic function satisfying (C1) and (C2).

(**Proof**) We note again that if we employ the same heuristic function, the numbers of encoding TEPs for both decoding algorithms are identical. So both algorithms perform priority-first search and the number of iterations in which a sufficient condition for the optimality holds is the same. Furthermore, the TEPs generated in the ν-th iteration of the proposed decoding algorithm are obtained in the λ-th ($\lambda \leq \nu$) iteration of the GBF decoding algorithm. These facts guarantee that the number of generated TEPs in the proposed decoding algorithm is no more than that in the GBF decoding algorithm. □

5. Adaptive Procedures

5.1 Method for Updating Referenced Codeword

Some of the heuristic functions such as the functions $f(\cdot)$ and $g(\cdot)$ use a referenced codeword $\tilde{\boldsymbol{c}}_{\text{ref}}$. In this case, we need not fix the referenced codeword throughout the decoding procedure. We call the decoding procedure in which the referenced codewords are updated the *adaptive procedure* [2], [6].

In [2], [5], [6], the currently best codeword $\tilde{\boldsymbol{c}}^*$ is set as referenced codeword. When a new (currently) best codeword $\tilde{\boldsymbol{c}}^*$ is obtained, the referenced codeword is updated by $\tilde{\boldsymbol{c}}_{\text{ref}} = \tilde{\boldsymbol{c}}^*$. Note that we initially set $\tilde{\boldsymbol{c}}_{\text{ref}} = \tilde{\boldsymbol{c}}_{\boldsymbol{0}}$ in the first iteration since the first best codeword is necessarily $\tilde{\boldsymbol{c}}_{\boldsymbol{0}}$. Since the number of TEPs stored in lists may be so large that we do not recalculate their heuristic values even when a referenced codeword is updated. If Eq. (2) holds for arbitrary pair of $\tilde{\boldsymbol{c}}_{\text{ref}}$ and $t(J)$, the GBF decoding algorithm still achieves MLD.

Let $\tilde{\boldsymbol{c}}_{\text{ref}}$ denote the current referenced codeword in the ν-th iteration and $\tilde{\boldsymbol{c}}'_{\text{ref}}$ be the referenced codeword for the selected best pattern $t(J)$ at λ-th ($\lambda \leq \nu$) iteration. At S4) in the ν-th iteration of the GBF decoding algorithm, the heuristic value of $t(J \cup j)$, an extended pattern of $t(J)$, is calculated referring $\tilde{\boldsymbol{c}}_{\text{ref}}$. This $\tilde{\boldsymbol{c}}_{\text{ref}}$ may differ from the referenced codeword $\tilde{\boldsymbol{c}}'_{\text{ref}}$ which is referenced by $t(J)$.

Meanwhile, in the ν-th iteration of the proposed decoding algorithm, we modify P4) as follows:

(a) When we obtain the adjacent pattern $t(J^{\text{a}} \cup i_{q-1})$ or $t(J^{\text{a}} \cup i'_{q-1})$ in P4-a) or P4-b) where $J^{\text{a}} = J \setminus \mu(J)$, we calculate their heuristic values using $\tilde{\boldsymbol{c}}'_{\text{ref}}$ which is the referenced codeword for the selected best pattern $t(J)$.

(b) When we obtain the extended pattern $t(J \cup i_s)$ or $t(J \cup i'_p)$ in P4-c), we calculate their heuristic values using $\tilde{\boldsymbol{c}}_{\text{ref}}$ which is the current referenced codeword.

Thus we need to store past referenced codewords $\tilde{\boldsymbol{c}}'_{\text{ref}}$ in memory by above (a). However the increased space complexity may not be so large since the number of past referenced codewords is not so great compared to the list size of TEPs as we will see in the next section.

We have the following lemma on the proposed decoding algorithm with the adaptive procedure.

Lemma 3: For any TEP $t(J)$, its heuristic value calculated in the proposed decoding algorithm is the same as that in the GBF decoding algorithm when we adopt the adaptive procedure.

(**Proof**) See Appendix B. □

Lemma 3 implies that the search order in the GBF and the proposed decoding algorithms with the adaptive procedures are strictly identical and they carry out MLD. Lemma 3 leads to the following theorems which are the counterparts of Theorems 2 and 3, respectively. The proofs are straightforward and hence we omit them.

Theorem 4: Assume that both the GBF and the proposed decoding algorithms employ the same heuristic function satisfying (C1) and (C2). Then the maximum list size of TEPs in the proposed decoding algorithm is less than that in the GBF decoding algorithm when they adopt the adaptive procedure.

Theorem 5: Assume that both the GBF and the proposed decoding algorithms employ the same heuristic function satisfying (C1) and (C2). Then the number of generated TEPs in the proposed decoding algorithm is no more than that in

the GBF decoding algorithm when they adopt the adaptive procedure.

5.2 The Case of Specific Heuristic Functions

If we use the heuristic function $f(\cdot)$, we can reduce the increased (time and space) complexity in the proposed decoding algorithm with the adaptive procedure (the increased space complexity is required to store past referenced codewords). For the TEP $t(J^{\mathrm{a}} \cup i_{q-1})$ generated at P4-a), since its referenced codeword $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$ is the same as that for the selected best pattern $t(J)(= t(J^{\mathrm{a}} \cup i_q))$, its heuristic value can be calculated by Eq. (29). To calculate r.h.s. of Eq. (29), we just need to know the value $f\big(t(J), \tilde{\boldsymbol{c}}_{\mathrm{ref}}\big)$ and the position $i_{q-1} \in S^{(0)}$ which is the adjacent to $i_q \in S^{(0)}$. For this reason, we need to store TEPs (not codewords themselves) corresponding to old referenced codewords in memory. We call these TEPs *referenced TEPs*. The similar argument holds for the TEP $t(J^{\mathrm{a}} \cup i'_{q-1})$ generated at P4-b) where $i'_{q-1} \in S^{(1)}$. We remark that the space complexity for storing referenced TEPs are smaller than that for storing ordinary TEPs since we need not store heuristic values of referenced TEPs.

If we use the heuristic function $g(\cdot)$, we can further save the space complexity for storing the past referenced codeword. As we see in Sect. 3.1, the value $B(\cdot)$ of a TEP defined in Eq. (22) depends only on its Hamming weight. Therefore even if the referenced codeword is updated in the adaptive procedure, we need not hold the past referenced codewords since we can calculate the heuristic value of the generated adjacent pattern by its Hamming weight at P4-a). There is no increased space complexity compared to the GBF decoding algorithm.

6. Simulation Results

In this section, we evaluate the effectiveness of the proposed decoding algorithm by computer simulations.

6.1 Conditions of Simulations

For the binary (63, 30, 13) BCH code and the binary (104, 52, 20) quadratic residue (QR) code, we perform MLD by the GBF decoding algorithm (we denote it by "GBF" in tables) and the proposed decoding algorithm (we denote it by "Proposed" in tables). At each SNR E_{b}/S_0 [dB], both decoding algorithms are carried out 10,000 times.

We adopt the function $f(\cdot)$ as the heuristic function in both decoding algorithms. We remark again that this GBF decoding algorithm is identical to the well-known A* decoding algorithm [2]. We assume that the weight profiles $W(C)$ of these two codes are unknown and we use their supersets $W'(C) = \{0, d, d{+}1, \cdots, n\}$. We compare two versions according to the way of arranging the referenced codeword: (i) We fix the referenced codeword as $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_{\emptyset}$ and we set $S^{(0)} = [1, k]$ and $S^{(1)} = \emptyset$. So we can skip P4-b) of the proposed decoding algorithm and the new extended pattern which is generated at P4) is only one. (ii) We consider the adaptive procedure in which the referenced codeword is initially set as $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_{\emptyset}$ and then updated as $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}^*$ each time a new (currently) best codeword is obtained.

In tables, we use the following notations:

- $N(\boldsymbol{r})$: the number of generated TEPs in decoding of $\boldsymbol{r}$
- $M(\boldsymbol{r})$: the maximum list size in decoding of $\boldsymbol{r}$
- $R(\boldsymbol{r})$: the number of updating referenced codeword in the proposed decoding algorithm
- Ave : the average value among 10,000 times of decoding
- Max : the maximum value among 10,000 times of decoding

6.2 Results and Discussion

(The results on the space complexity for algorithms fixing $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_{\emptyset}$)

We show the results of decoding with fixed $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_{\emptyset}$ for the (63, 30, 13) BCH code and the (104, 52, 20) QR code in Tables 1 and 2, respectively.

By Table 1, the maximum list size Max $M(\boldsymbol{r})$ in the proposed decoding algorithm is less than 1/3 of that in the GBF decoding algorithm at each SNR. Furthermore, the average

Table 1 The results of decoding with fixed $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_{\emptyset}$ for (63, 30, 13) BCH code.

E_{b}/N_0 [dB]			GBF	Proposed
5.0	Ave	$N(\boldsymbol{r})$	$3.54 \cdot 10^{1}$	2.10
		$M(\boldsymbol{r})$	1.46	$1.92 \cdot 10^{-1}$
	Max	$M(\boldsymbol{r})$	$2.073 \cdot 10^{3}$	$4.110 \cdot 10^{2}$
4.0	Ave	$N(\boldsymbol{r})$	$1.09 \cdot 10^{2}$	$2.50 \cdot 10^{1}$
		$M(\boldsymbol{r})$	$1.40 \cdot 10^{1}$	2.28
	Max	$M(\boldsymbol{r})$	$9.586 \cdot 10^{3}$	$2.406 \cdot 10^{3}$
3.0	Ave	$N(\boldsymbol{r})$	$6.74 \cdot 10^{2}$	$2.33 \cdot 10^{2}$
		$M(\boldsymbol{r})$	$1.13 \cdot 10^{2}$	$2.31 \cdot 10^{1}$
	Max	$M(\boldsymbol{r})$	$1.582 \cdot 10^{4}$	$4.600 \cdot 10^{3}$
2.0	Ave	$N(\boldsymbol{r})$	$3.15 \cdot 10^{3}$	$1.26 \cdot 10^{3}$
		$M(\boldsymbol{r})$	$5.82 \cdot 10^{2}$	$1.36 \cdot 10^{2}$
	Max	$M(\boldsymbol{r})$	$7.052 \cdot 10^{4}$	$2.055 \cdot 10^{4}$

Table 2 The results of decoding with fixed $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_{\emptyset}$ for (104, 52, 20) QR code.

E_{b}/N_0 [dB]			GBF	Proposed
6.0	Ave	$N(\boldsymbol{r})$	$4.94 \cdot 10^{1}$	$4.18 \cdot 10^{-1}$
		$M(\boldsymbol{r})$	$3.68 \cdot 10^{-1}$	$3.26 \cdot 10^{-2}$
	Max	$M(\boldsymbol{r})$	$5.160 \cdot 10^{2}$	$4.400 \cdot 10^{1}$
5.0	Ave	$N(\boldsymbol{r})$	$1.18 \cdot 10^{2}$	$1.27 \cdot 10^{1}$
		$M(\boldsymbol{r})$	7.46	$7.65 \cdot 10^{-1}$
	Max	$M(\boldsymbol{r})$	$1.093 \cdot 10^{4}$	$1.264 \cdot 10^{3}$
4.0	Ave	$N(\boldsymbol{r})$	$2.24 \cdot 10^{3}$	$5.98 \cdot 10^{2}$
		$M(\boldsymbol{r})$	$2.89 \cdot 10^{2}$	$4.68 \cdot 10^{1}$
	Max	$M(\boldsymbol{r})$	$4.634 \cdot 10^{5}$	$9.875 \cdot 10^{4}$
3.0	Ave	$N(\boldsymbol{r})$	$4.70 \cdot 10^{4}$	$1.32 \cdot 10^{4}$
		$M(\boldsymbol{r})$	$7.38 \cdot 10^{3}$	$1.16 \cdot 10^{3}$
	Max	$M(\boldsymbol{r})$	$1.145 \cdot 10^{7}$	$2.674 \cdot 10^{6}$

Table 3 The results of decoding with adaptive procedure for (63, 30, 13) BCH code.

E_b/N_0 [dB]			GBF	Proposed
5.0	Ave	$N(\boldsymbol{r})$	$3.37 \cdot 10^{1}$	1.78
		$M(\boldsymbol{r})$	1.19	$1.76 \cdot 10^{-1}$
	Max	$M(\boldsymbol{r})$	$2.064 \cdot 10^{3}$	$4.370 \cdot 10^{2}$
4.0	Ave	$N(\boldsymbol{r})$	$9.36 \cdot 10^{1}$	$2.19 \cdot 10^{1}$
		$M(\boldsymbol{r})$	$1.14 \cdot 10^{1}$	2.19
	Max	$M(\boldsymbol{r})$	$9.579 \cdot 10^{3}$	$3.272 \cdot 10^{3}$
3.0	Ave	$N(\boldsymbol{r})$	$5.83 \cdot 10^{2}$	$2.25 \cdot 10^{2}$
		$M(\boldsymbol{r})$	$9.66 \cdot 10^{1}$	$2.44 \cdot 10^{1}$
	Max	$M(\boldsymbol{r})$	$1.432 \cdot 10^{4}$	$5.103 \cdot 10^{3}$
2.0	Ave	$N(\boldsymbol{r})$	$2.91 \cdot 10^{3}$	$1.31 \cdot 10^{3}$
		$M(\boldsymbol{r})$	$5.53 \cdot 10^{2}$	$1.56 \cdot 10^{2}$
	Max	$M(\boldsymbol{r})$	$7.052 \cdot 10^{4}$	$2.801 \cdot 10^{4}$

Table 4 The results of decoding with adaptive procedure for (104, 52, 20) QR code.

E_b/N_0 [dB]			GBF	Proposed
6.0	Ave	$N(\boldsymbol{r})$	$4.85 \cdot 10^{1}$	$2.39 \cdot 10^{-1}$
		$M(\boldsymbol{r})$	$2.48 \cdot 10^{-1}$	$3.42 \cdot 10^{-2}$
	Max	$M(\boldsymbol{r})$	$1.87 \cdot 10^{2}$	$3.20 \cdot 10^{1}$
5.0	Ave	$N(\boldsymbol{r})$	$8.24 \cdot 10^{1}$	5.79
		$M(\boldsymbol{r})$	3.60	$4.71 \cdot 10^{-1}$
	Max	$M(\boldsymbol{r})$	$4.932 \cdot 10^{3}$	$7.930 \cdot 10^{2}$
4.0	Ave	$N(\boldsymbol{r})$	$1.23 \cdot 10^{3}$	$3.57 \cdot 10^{2}$
		$M(\boldsymbol{r})$	$1.66 \cdot 10^{2}$	$3.01 \cdot 10^{1}$
	Max	$M(\boldsymbol{r})$	$4.630 \cdot 10^{5}$	$9.862 \cdot 10^{4}$
3.0	Ave	$N(\boldsymbol{r})$	$3.38 \cdot 10^{4}$	$1.30 \cdot 10^{4}$
		$M(\boldsymbol{r})$	$5.46 \cdot 10^{3}$	$1.30 \cdot 10^{3}$
	Max	$M(\boldsymbol{r})$	$1.145 \cdot 10^{7}$	$2.681 \cdot 10^{6}$

value of the maximum list size Ave $M(\boldsymbol{r})$ in the proposed decoding algorithm is less than 1/4 of that in the GBF decoding algorithm. These results show that the effectiveness of the proposed decoding algorithm. By Table 2, the values Max $M(\boldsymbol{r})$ and Ave $M(\boldsymbol{r})$ in the proposed decoding algorithm are less than 1/4 and 1/6 of those in the GBF decoding algorithm, respectively. These results indicate that the proposed method also works well for the (104, 52, 20) QR code.

(The results on the space complexity for the adaptive procedure)

We show the results of decoding with the adaptive procedure (in which we update as $\tilde{\boldsymbol{c}}_{\text{ref}} = \tilde{\boldsymbol{c}}^{*}$) for the (63, 30, 13) BCH code and the (104, 52, 20) QR code in Tables 3 and 4, respectively. We also show the number of updating referenced codeword (which is equal to the number of the past referenced TEPs stored in memory) in the proposed decoding algorithm with the adaptive procedure in Tables 5 and 6.

By Table 3 for the (63, 30, 13) BCH code, the maximum list size Max $M(\boldsymbol{r})$ in the proposed decoding algorithm is less than 2/5 of that in the GBF decoding algorithm at each SNR. Furthermore, the average values of the maximum list size Ave $M(\boldsymbol{r})$ in the proposed decoding algorithm are less

Table 5 The number of past referenced codewords in the proposed decoding algorithm for the (63, 30, 13) BCH code.

E_b/N_0	Ave $R(\boldsymbol{r})$	Max $R(\boldsymbol{r})$
5.5	1.025	10
5.0	1.056	12
4.5	1.117	12
4.0	1.215	14
3.5	1.394	16
3.0	1.638	18
2.5	1.984	16
2.0	2.425	24

Table 6 The number of past referenced codewords in the proposed decoding algorithm for the (104, 52, 20) QR code.

E_b/N_0	Ave $R(\boldsymbol{r})$	Max $R(\boldsymbol{r})$
6.5	1.004	8
6.0	1.013	8
5.5	1.039	10
5.0	1.101	12
4.5	1.209	14
4.0	1.393	16
3.5	1.714	20
3.0	2.189	22

than 1/3 of those in the GBF decoding algorithm. By Table 4 for the (104, 52, 20) QR code, the values Max $M(\boldsymbol{r})$ and Ave $M(\boldsymbol{r})$ in the proposed decoding algorithm are less than 1/4 of those in the GBF decoding algorithm†. By Tables 5 and 6, the average values of $R(\boldsymbol{r})$ are fairly small and the maximum value of $R(\boldsymbol{r})$ is only 24 at 2.0 [dB] for the (63, 30, 13) BCH code. On the other hand, the average and the maximum values of $M(\boldsymbol{r})$ are 156 and 2,801 at 2.0 [dB], respectively, so the values $R(\boldsymbol{r})$ seem to be negligible. These values demonstrate that there are almost no increases of the space complexity for the proposed decoding algorithm even though we store the past referenced TEPs. Note that the average and maximum values $R(\boldsymbol{r})$ are hardly increased even for the long (104, 52, 20) QR code.

(The results on the number of generating TEPs)

The number of generating TEPs $N(\boldsymbol{r})$ is one of indices to evaluate time complexity in heuristic search MLD algorithms [2], [8] although the reduction of the time complexity led by reducing $N(\boldsymbol{r})$ may not be so large in the whole decoding complexity.

By Tables 1 and 2 for decoding with fixed $\tilde{\boldsymbol{c}}_{\text{ref}}$, $N(\boldsymbol{r})$ in the proposed decoding algorithm are less than 2/5 of $N(\boldsymbol{r})$ in the GBF decoding algorithm even at low SNRs. These results demonstrate the proposed decoding algorithm reduces the time complexity of the GBF decoding algorithm as well as the space complexity.

By Tables 3 and 4 for decoding algorithms with adap-

†The ratio of the value Ave $M(\boldsymbol{r})$ of the method in [6] to that of the GBF decoding algorithm is about 2/5 at 5.0 [dB] for the (104, 52, 20) QR code. By Table 4, the ratio of the value Ave $M(\boldsymbol{r})$ of the proposed decoding algorithm to that of the GBF decoding algorithm is about 1/8 at 5.0 [dB] for the (104, 52, 20) QR code .

tive procedure, $N(r)$ in the proposed decoding algorithm is less than 2/5 of $N(r)$ in the GBF decoding algorithm even at 3.0 [dB]. These results demonstrate the proposed method also reduces the time complexity of the GBF decoding algorithm even when we adopt the adaptive procedure.

7. Concluding Remarks

In this paper, we propose a new heuristic search method for reducing the space complexity of the GBF decoding algorithm. The GBF decoding algorithm is identical to the well-known A* decoding algorithm and includes the original BF decoding algorithm. As a result, the proposed method reduces the space complexity of the well-known A* and the original BF decoding algorithms. Though heuristic functions considered here are restricted by a condition, we show this class of heuristic functions includes some well-known functions. The proposed decoding algorithm guarantees to perform MLD since the set of generated candidate codewords is identical to that in the GBF decoding algorithm. Since the proposed decoding algorithm also reduces the number of generated TEPs which tends to vastly increase from low to medium SNRs, the proposed decoding algorithm reduces not only the space complexity but the time one in the GBF decoding algorithm.

As future works, we need to develop a method for heuristic search MLD algorithm with powerful heuristic functions such as in [6], [7]. More detailed comparisons between the proposed decoding algorithm and methods in [6], [7] are to be evaluated.

Acknowledgments

H. Yagi wishes to thank Dr. M. Kobayashi at Shonan Institute of Technology and Mr. T. Ishida and Mr. G. Hosoya at Waseda University for their valuable comments.

References

[1] G. Battail and J. Fang, "Décodage pondéré optimal descodes linéaires en blocs," Annales des télé-communications, vol.41, nos.11–12, pp.580–604, Nov.–Dec. 1986.
[2] Y.S. Han, C.R.P. Hartmann, and C.C. Chan, "Efficient priority-first search maximum likelihood soft decision decoding of linear block codes," IEEE Trans. Inf. Theory, vol.39, no.5, pp.1514–1523, Sept. 1993.
[3] Y.S. Han, C.R.P. Hartmann, and K.G. Mehrotra, "Decoding linear block codes using a priority-first search: Performance analysis and suboptimal version," IEEE Trans. Inf. Theory, vol.44, no.3, pp.1233–1246, May 1998.
[4] D. Gazelle and J. Snyders, "Reliability-based code-search algorithm for maximum-likelihood decoding of block codes," IEEE Trans. Inf. Theory, vol.43, no.1, pp.239–249, Jan. 1997.
[5] M.P.C. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," IEEE Trans. Inf. Theory, vol.41, no.5, pp.1379–1396, Sept. 1995.
[6] T. Okada, M. Kobayashi, and S. Hirasawa, "An efficient heuristic search method for maximum likelihood decoding of linear block codes using dual codes," IEICE Trans. Fundamentals, vol.E85-A, no.2, pp.485–489, Feb. 2002.
[7] C.C. Shih, C.R. Wulff, C.R.P. Hartmann, and C.K. Mohan, "Efficient heuristic search algorithms for soft-decision decoding of linear block codes," IEEE Trans. Inf. Theory, vol.44, no.7, pp.3023–3038, Nov. 1998.
[8] A. Valembois and M. Fossorier, "An improved method to compute lists of binary vectors that optimize a given weight function with application to soft decision decoding," IEEE Commun. Lett., vol.5, no.3, pp.456–458, Nov. 2001.
[9] A. Valembois and M. Fossorier, "A comparison between "most-reliable-basis reprocessing" strategies," IEICE Trans. Fundamentals, vol.E85-A, no.7, pp.1727–1741, July 2002.
[10] H. Yagi, M. Kobayashi, T. Matsushima, and S. Hirasawa, "Complexity reduction of the Gazelle and Snyders decoding algorithm for maximum likelihood decoding," IEICE Trans. Fundamentals, vol.E86-A, no.10, pp.2461–2472, Oct. 2003.
[11] H. Yagi, T. Matsushima, and S. Hirasawa, "A method for reducing space complexity of reliability-based heuristic search maximum likelihood decoding algorithms," Proc. 26th Symposium on Information Theory and its Applications (SITA2003), pp.185–188, Hyogo, Japan, Dec. 2003.
[12] H. Yagi, T. Matsushima, and S. Hirasawa, "A heuristic search algorithm with the reduced list of test error patterns for maximum likelihood decoding algorithms," Proc. 27th Symposium on Information Theory and its Applications (SITA2004), pp.571–574, Gifu, Japan, Dec. 2004.

Appendix A: The Proof of Theorem 1

It is sufficient to show that the TEP with the ν-th smallest heuristic value among all TEPs has been already generated and stored in the list at the beginning of ν-th iteration of the decoding algorithm. We will prove it by the mathematical induction. Let $\boldsymbol{t}(J_q)$ with the support J_q be the TEP with the q-th smallest heuristic value among all possible TEPs.

(i) The case of $\nu = 1$:
By the conditions (C1) and (C2), the best pattern $\boldsymbol{t}(J_1)$ is either $\boldsymbol{t}(i_s)$ or $\boldsymbol{t}(i'_p)$. The initial list of TEPs is constructed by Eq. (24) so $\boldsymbol{t}(J_1)$ has been already stored in the list $M^{(\mu(J_1))}$ in the first iteration.

(ii) The case of $\nu > 1$:
Let $\mathcal{T}_\nu$ denote the set of all $\nu - 1$ best patterns before the ν-th iteration. i.e.,

$$\mathcal{T}_\nu = \left\{\boldsymbol{t}(J_q) \mid q = 1, 2, \ldots, \nu - 1\right\}. \tag{A·1}$$

Assume that we have exactly selected all $\nu - 1$ TEPs in $\mathcal{T}_\nu$ before the ν-th iteration.

(a) If $\mu(J_\nu) = i_q \in S^{(0)}$ such that $i_q < i_s$, there is $\boldsymbol{t}(J)$ such that both $\mu(J) = i_{q+1}$ and $\boldsymbol{t}(J_\nu)$ is the adjacent pattern of $\boldsymbol{t}(J)$. We have $F(\boldsymbol{t}(J)) \leq F(\boldsymbol{t}(J_\nu))$ and $\boldsymbol{t}(J) \in \mathcal{T}_\nu$ by the condition (C2). Therefore $\boldsymbol{t}(J)$ was selected as the best pattern at P2) in a previous iteration. When such $\boldsymbol{t}(J)$ was selected as the best pattern, $\boldsymbol{t}(J_\nu)$ has inserted into the list $M^{(i_q)}$ at P4-a) in the same iteration.

(b) Arguing similarly to the case of $\mu(J_\nu) \in S^{(0)}$, when $\mu(J_\nu) = i'_q \in S^{(1)}$ and $i'_q < i'_p$, $\boldsymbol{t}(J_\nu)$ has inserted into the list $M^{(i'_q)}$ at P4-b) in a previous iteration.

(c) If $\mu(J_\nu) = i_s \in S^{(0)}$, the TEP $\boldsymbol{t}(J)$ such that $J = J_\nu \backslash \{i_s\}$ satisfies $F(\boldsymbol{t}(J)) \leq F(\boldsymbol{t}(J_\nu))$ and $\boldsymbol{t}(J) \in \mathcal{T}_\nu$ by the condition (C1). Therefore such $\boldsymbol{t}(J) \in \mathcal{T}_\nu$ was selected

as the best pattern at P2) in a previous iteration and then $t(J_\nu)$ has inserted into the list $M^{(i_s)}$ at P4-c) in the same iteration.

Similarly, when $\mu(J_\nu) = i'_q \in S^{(1)}$, we can show $t(J_\nu)$ has inserted into the list $M^{(i'_p)}$ at P4-c) in the λ-th iteration such that $\lambda \leq \nu$.

As we mentioned in (i), since the first best pattern $t(J_1)$ has been generated when initial lists has been constructed by Eq. (24), the assumptions of (ii) are satisfied and this completes the proof. □

Appendix B: The Proof of Lemma 3

We will prove the lemma by the mathematical induction. Let ν represent the iteration number.

(i) The case of $\nu = 1$:

The first referenced codeword is the same ($\tilde{c}_{\text{ref}} = \tilde{c}_\emptyset$) in both decoding algorithm. So the heuristic values of the first best pattern selected at S2) and P2) are identical.

(ii) The case of $\nu \geq 2$:

Assume that heuristic values of all TEPs generated before the ν-th iteration of the proposed decoding algorithm are the same as those of the GBF decoding algorithm. So the heuristic values of the best pattern $t(J)$ selected at S2) and P4) in the ν-the iteration are identical.

We first consider the heuristic value of $t(J^a \cup i_q), i_q \in S^{(0)}$, which is obtained at P4-a) after selecting $t(J)$ as the best pattern. This TEP $t(J^a \cup i_q)$ is generated at the same iteration of generating $t(J)(= t(J^a \cup i_{q+1})), i_{q+1} \in S^{(0)}$, in the GBF decoding algorithm since both of them are extended patterns of $t(J^a)$. Then their heuristic values are calculated by referring the same codeword, say $\tilde{c}'_{\text{ref}}$. On the other hand, by the step (a) of the proposed decoding algorithm with the adaptive procedure, if we calculate the heuristic value of $t(J^a \cup i_q)$ by referring $\tilde{c}'_{\text{ref}}$, it is identical to that in the GBF decoding algorithm. Similarly, the heuristic value of $t(J^a \cup i'_q), i'_q \in S^{(1)}$, which is obtained at P4-b) is calculated by the same referenced codeword and thus it has the same heuristic value in both decoding algorithm.

Next we consider the heuristic values of $t(J \cup i_s)$ and $t(J \cup i'_p)$ which are obtained at P4-c) after selecting $t(J)$ as the best pattern. These TEPs $t(J \cup i_s)$ and $t(J \cup i'_p)$ are also generated in the same iteration of the GBF decoding algorithm and hence the heuristic values of them are identical. Therefore, heuristic values of TEPs generated in the ν-th iteration of the proposed decoding algorithm are the same as those of the GBF decoding algorithm.

The arguments (i) and (ii) complete the proof. □

Appendix C: Comparison with the Improved Method of [8]

Valembois et al. have proposed an improved method of the original BF decoding algorithm (we will call this method the improved BF (IBF) decoding algorithm) [8]. In this section, we compare the proposed decoding algorithm with the IBF decoding algorithm.

The IBF decoding algorithm exploits the property of the function $\Delta(\cdot)$ satisfying Eq. (9) as well as the condition (C1). We will consider any heuristic functions $F(\cdot)$ satisfying both the condition (C1) and Eq. (9).

In the IBF decoding algorithm, each list $M^{(1)}, M^{(2)}, \ldots, M^{(k)}$ contains at most one TEP while we arrange another list A which stores TEPs already selected as the best pattern at S2). After a TEP $t(J)$ is selected as the best pattern, it is added to the end of the list A.

The initial lists of TEPs are constructed by

$$M^{(j)} = \{t(j)\} \qquad \text{for } j \in [1, k], \tag{A·2}$$

as in the original BF decoding algorithm. In the initial step of the algorithm, the list A is set as $A = \emptyset$.

When a TEP $t(J), \mu(J) \neq k$, is selected as the best pattern, we delete it from the list $M^{(\mu(J))}$ and added it to the end of the list A. It is readily shown by Eq. (9) that the next TEP to be stored in the list $M^{(\mu(J))}$ is $t(J' \cup \mu(J))$ where $t(J')$ is given by

$$t(J') = \arg\min_{t(I)} \left\{ F(t(I)) \geq F(t(J^a)) \middle| \mu(I) < \mu(J) \right\}. \tag{A·3}$$

We can show that the $t(J')$ is the first TEP with $\mu(J') < \mu(J)$ following $t(J^a)$ in the list A if it has been already stored in the list A. Note that if a selected best pattern has no extended patterns or if all its extended patterns have been already generated, we do not need to possess it in the list A.

[The improved BF decoding algorithm [8]]

S'1) Set $\tilde{c}_\emptyset := u\tilde{G}$, $\tilde{c}^* := \tilde{c}_\emptyset$ and $\underline{L} := L(\tilde{c}_\emptyset)$. Construct the initial lists of TEPs by Eq. (A·2).

S'2) Select the best pattern $t(J) \in M^{(\mu(J))}$ among TEPs in non-empty lists $M^{(j)}$. If $F(t(J)) \geq \underline{L}$, then output $\tilde{c}^*$ and halt the algorithm.

S'3) Generate the next candidate codeword by $\tilde{c}_J := \tilde{c}_\emptyset \oplus t(J)\tilde{G}$. If $L(\tilde{c}_J) < \underline{L}$, then set $\underline{L} := L(\tilde{c}_J)$ and $\tilde{c}^* := \tilde{c}_J$.

S'4) a) Delete $t(J)$ from the list $M^{(\mu(J))}$. If $\mu(J) \neq k$, store $t(J)$ into the list A.

b) Find a TEP $t(J')$ which satisfies Eq. (A·3) in the list A. If such $t(J')$ exists, then generate $t(J' \cup \mu(J))$ and store it in the list $M^{(\mu(J))}$.

c) If there is an empty list $M^{(j)}$ with $\mu(J^a) < j, j \neq \mu(J)$, store $t(J^a \cup j)$ in the list $M^{(j)}$.

d) If all extended patterns of the TEP $t(J^a)$ have been already generated, delete it from the list A.

S'5) If $M^{(j)} = \emptyset$ for all $j \in [1, k]$, then output $\tilde{c}^*$ and halt the algorithm. Otherwise, go to S2). □

Note that for a selected best pattern $t(J)$, at most one TEP is newly stored in a list (the selected best pattern is only moved from the list $M^{(\mu(J))}$ to the list A and the new generated TEP is stored in the list $M^{(\mu(J))}$). Therefore the

list size increases at most by one in each iteration.

We can easily show that the function $\Delta(\cdot)$ satisfies conditions both (C1) and Eq. (9). Since the function $\Delta(\cdot)$ also satisfies the condition (C2), the proposed decoding algorithm can also employ it. By Proposition 1, we set $S^{(0)} = [1, k]$ and the number of TEPs newly stored in lists in each iteration of the proposed decoding algorithm is at most two (one is an extended pattern and the other is an adjacent pattern). The list size of the proposed decoding algorithm also increases at most by one in each iteration since the selected best pattern is deleted from the list.

We have the following proposition on the relationship between the IBF decoding algorithm and the proposed decoding algorithm.

Proposition 4: Assume that both the IBF and the proposed decoding algorithms employ a heuristic function satisfying the condition (C1) and Eq. (9). We further assume that the heuristic function satisfies the condition (C2) with $S^{(0)} = [1, k]$ such as the function $\Delta(\cdot)$. Then the maximum list size of TEPs in the proposed decoding algorithm is no more than that in the IBF decoding algorithm.

(**Proof**) We will prove the proposition by mathematical induction. We here denote the iteration number by ν.

(i) The case of $\nu = 1$:

The initial list constructed by Eqs. (24) and (A· 2) indicate the list size of the proposed decoding algorithm is no more than that in the IBF decoding algorithm. Note that Eq. (9) cannot tell that we only need to store $\boldsymbol{t}(k)$ in lists so we need to store other TEPs with Hamming weight one in the IBF decoding algorithm.

(ii) The case of $\nu \geq 2$:

We first remark that the selected best pattern $\boldsymbol{t}(J)$ in the ν-th iteration of the IBF and the proposed decoding algorithms is identical since both algorithms perform the priority-first search.

Denote the set of TEPs stored in lists in the proposed and the IBF decoding algorithms by $\mathcal{T}_p$ and $\mathcal{T}_{IBF}$, respectively. Then there exists a one-to-one mapping ϕ from each element of $\mathcal{T}_p$ to that of $\mathcal{T}_{IBF}$ given by

$$\phi : \mathcal{T}_p \to \mathcal{T}_{IBF}, \tag{A·4}$$

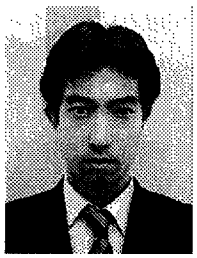
and

$$\phi(\boldsymbol{t}) \neq \phi(\boldsymbol{t}') \quad \text{if} \quad \boldsymbol{t} \neq \boldsymbol{t}'. \tag{A·5}$$

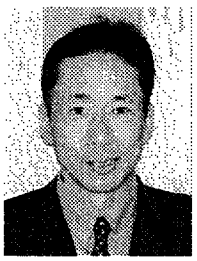
Actually $\phi(\boldsymbol{t}(J)) = \boldsymbol{t}(J^{\mathrm{a}})$ or $\phi(\boldsymbol{t}(J)) = \boldsymbol{t}(J)$. i.e., when a TEP $\boldsymbol{t}(J)$ is newly generated in the ν-th iteration of the proposed decoding algorithm, then the same iteration of the IBF decoding algorithm possesses the corresponding TEP of the form of either its adjacent pattern $\boldsymbol{t}(J^{\mathrm{a}})$ in the list A or $\boldsymbol{t}(J)$ itself in the list $M^{(\mu(J))}$. Based on this mapping, we can see that there is a correspondence between the TEP newly generated in the proposed decoding algorithm and a TEP stored in list in the IBF decoding algorithm. Therefore, the increased list sizes in the ν-th iteration of both algorithms are the same.

From the arguments (i) and (ii), we can prove the theorem. □

Unfortunately, as known to the authors, there are no heuristic functions which satisfy both conditions (C1) and Eq. (9) except for the function $\Delta(\cdot)$. The function $\Delta(\cdot)$ is ineffective heuristic function compared with the function $f(\cdot)$ or $g(\cdot)$ since it only utilizes the information over the k MRI positions while functions $f(\cdot)$ and $g(\cdot)$ utilize one over the remaining $n - k$ positions as well as one over the k MRI positions. Therefore we may say that the proposed decoding algorithm is more effective than the method in [8].



Hideki Yagi was born in Yokohama, Japan, on Oct. 14, 1975. He received the B.E. degree and M.E. degree in Industrial and Management Systems Engineering from Waseda University, Tokyo, Japan, in 2001 and 2003, respectively. He is currently a doctoral student in Industrial and Management Systems Engineering at Graduate School of Waseda University. Since 2005, he has been a research associate at Media Network Center of Waseda University. His research interests are coding theory and information security. He is a student member of the Society of Information Theory and its Applications.



Toshiyasu Matsushima was born in Tokyo, Japan, on Nov. 26, 1955. He received the B.E. degree, M.E. degree and Dr.E. degree in Industrial and Management Systems Engineering from Waseda University, Tokyo, Japan, in 1978, 1980 and 1991, respectively. From 1980 to 1986, he was with Nippon Electric Corporation, Kanagawa, Japan. From 1986 to 1992, he was a lecturer at Department of Management Information, Yokohama College of Commerce. From 1993, he was an associate professor and since 1996 has been a professor of School of Science and Engineering, Waseda University, Tokyo, Japan. His research interests are information theory and its application, statistics and artificial intelligence. He is a member of the Society of Information Theory and Its Applications, the Japan Society for Quality Control, the Japan Industrial Management Association, the Japan Society for Artificial Intelligence and IEEE.

Shigeichi Hirasawa was born in Kobe, Japan, on Oct. 2, 1938. He received the B.S. degree in mathematics and the B.E. degree in electrical communication engineering from Waseda University, Tokyo, Japan, in 1961 and 1963, respectively, and the Dr.E. degree in electrical communication engineering from Osaka University, Osaka, Japan, in 1975. From 1963 to 1981, he was with the Mitsubishi Electric Corporation, Hyogo, Japan. Since 1981, he has been a professor of School of Science and Engineering, Waseda University, Tokyo, Japan. In 1979, he was a Visiting Scholar in the Computer Science Department at the University of California, Los Angels (CSD, UCLA), CA. He was a Visiting Researcher at the Hungarian Academy of Science, Hungary, in 1985, and at the University of Trieste, Italy, in 1986. In 2002, he was also a Visiting Faculty at CSD, UCLA. From 1987 to 1989, he was the Chairman of Technical Group on Information Theory of IEICE. He received the 1993 Achievement Award, and the 1993 Kobayashi-Memorial Achievement Award from IEICE. In 1996, he was the President of the Society of Information Theory and Its Applications (Soc. of ITA). His research interests are information theory and its applications, and information processing systems. He is an IEEE Fellow, and a member of Soc. of ITA, the Operations Research Society of Japan, the Information Processing Society of Japan, the Japan Industrial Management Association, and Informs.

ターボ符号・LDPC符号とその復号法の概要

Turbo Codes, LDPC Codes and Their Decoding Methods Using Belief

松嶋敏泰

Abstract

近年まで困難と思われてきた通信路符号化定理の限界をほとんど達成するターボ符号やLDPC符号が注目を集めている．これらの符号で高い性能が得られる仕組みについて，グラフ上でメッセージを伝搬させることにより効率的に確率計算を行うアルゴリズムを利用した復号法を中心に概説する．

キーワード：ターボ符号，LDPC符号，メッセージ伝搬アルゴリズム，BPアルゴリズム，sum-productアルゴリズム，反復復号法，ターボ復号法

1. はじめに

シャノンによって示された情報理論の通信路符号化定理において，通信路容量を達成する符号の存在は証明されたが，それを達成する実用的符号と復号法の構成は長年の夢となっていた．近年まで，線形符号と限界距離復号により多くの研究成果が得られ，実用的にもある程度満足のいく性能が得られたものの，その性能は符号化定理の限界に遠く及ばなかった．ところが，ターボ符号[1]の発見と低密度パリティ検査（LDPC：Low-Density Parity-Check）符号（以下，LDPC符号[2]）の再発見により状況は一転した．それらの符号が符号化定理の限界をほとんど達成することが実験的に確かめられたことにより，次世代通信方式の符号の主流として一躍注目され，幾つかの次期通信システムにも既に採用が決定している．本稿では，まず符号化定理を達成する難しさについて説明し，それをどのような仕組みでターボ符号とLDPC符号が解決したのか，特にグラフを用いたメッセージ伝搬アルゴリズムから解説する．

松嶋敏泰　正員　早稲田大学理工学術院
E-mail toshi@matsu.mgmt.waseda.ac.jp
Toshiyasu MATSUSHIMA, Member (School of Science and Engineering, Waseda University, Tokyo, 169-8555 Japan).
電子情報通信学会誌 Vol.88 No.4 pp.244-248 2005年4月

2. 誤り訂正符号の限界と限界達成の難しさ

誤り訂正符号の問題を簡単に復習してみたい．送信側ではまず送信したい情報系列（ベクトル）$\boldsymbol{u}=(u_1,\cdots,u_K)$[注1]から符号化（関数）$C$により符号語$C(\boldsymbol{u})=\boldsymbol{x}=(x_1,x_2,\cdots,x_N)$を生成し，通信路を通して受信側へ送る．受信側では符号語が雑音などで変化した受信系列$\boldsymbol{y}=(y_1,y_2,\cdots,y_N)$を受け取り，復号$D(\boldsymbol{y})=\hat{\boldsymbol{x}}$により元の符号語または情報系列に戻す．

通信路として$P(\boldsymbol{y}\mid\boldsymbol{x})$の確率モデルを仮定すると，結局この復号の問題は，受信系列$\boldsymbol{y}$から符号語$\boldsymbol{x}$または情報系列$\boldsymbol{u}$を推定する問題に帰着される．推定は情報系列全体$\boldsymbol{u}$をブロックとして推定する場合と各情報シンボルu_kを推定する場合に分けられる．一般に推定の評価は，前者に対してブロック誤り率，後者に対してシンボル誤り率が用いられる．

ブロック誤り率を最小化する復号は受信系列を与えられたもとでの事後確率$P(\boldsymbol{u}\mid\boldsymbol{y})$を最大化する情報系列$\hat{\boldsymbol{u}}$を推定値とすることになる．これは$\boldsymbol{u}$の事前分布が一様な場合は，ゆう度$P(\boldsymbol{y}\mid\boldsymbol{u})$を最大化する情報系列と一致し，通常これを最ゆう復号と呼んでいる．一方，シンボル誤り率を最小化する復号は事後確率$P(u_k\mid\boldsymbol{y})$を最大化する情報シンボル$\hat{u}_k$を推定値とすることになり，通常これをMPM（Maximum Posterior Marginal）復

(注1)　ここでは，簡単のためu_k, x_nなどは2値{0, 1}とする．

号[注2]と呼んでいる．

次に，誤りを任意に少なくする条件のもと，どのくらいの情報（伝送率 $R=K/N$ で測る）を送信側から受信側に送ることができるのであろうか．

定理1（シャノンの通信路符号化定理）通信路容量[注3]が $C>0$ のとき，ある正数 $\delta>0$ が存在して伝送率が $R<C-\delta$ となるならば，ブロック誤り率を任意に小さくできる符号化がブロック長 N を大きくすることにより可能である．

この定理により伝送率の限界とその限界を達成する符号の存在は示されたが，その具体的符号の構成法や実用的復号法については次の課題として残されてしまった[注4]．

課題の難しさをもう少し詳しく説明しよう．符号化には，定理の証明法から類推し，何らかのランダム性を導入する必要があると思われるが，完全にランダムな符号を構成すれば，符号化のための計算量，メモリ量は現実的ではない．また，復号においても誤り率を最小化する最ゆう復号や MPM 復号を実現するには，一般的には符号長 N の指数オーダの計算量[注5]が必要でこれも現実的ではない．

そこで，近年までの符号理論の分野では，通信路符号化定理の証明と類似したアプローチによる具体的な符号の構成と復号をあきらめ，少し違った方向からアプローチが続けられてきた．それは符号化と復号を代数的に行うことであった[注6]．

3. グラフ上のメッセージ伝搬アルゴリズムによる事後確率計算

復号の主問題は，周辺（事後）確率の計算または最大結合確率をとるベクトルを探索する問題であったが，これらの問題が主要な課題となっている分野は統計理論，制御理論，信号処理，統計力学等多岐にわたっている．例えば，人工知能の確率推論[3]では，命題間の確率的依存性（結合確率）を事前知識として，幾つかの evidence：証拠（確率変数の実現値）が与えられたもとで各命題の belief：確信度（周辺事後確率）や composite belief：信頼度の高い命題の組（最大結合確率をとるベクトル）を求めている[注7]．

これらの問題は一般的には指数オーダの計算量を必要とする困難な問題であることを 2. で述べた．しかし，確率モデルに条件付独立が成り立つとき，つまり結合確率が確率変数の部分集合 $V_j \subset \{X_1,\cdots,X_N\}$ に関する関数の積の形 $P(X_1,\cdots,X_N)=\alpha q(V_1)q(V_2)\cdots q(V_J)$ で表現できる場合，これらの問題に対する効率的アルゴリズムが幾つか提案されている．積の表現に用いられる関数としては，条件付確率やポテンシャル関数が挙げられ，この条件が当てはまる確率モデルとしてマルコフモデルなどがある．条件を満たす確率モデルの簡単な例を示そう．

$$P(X_1,\cdots,X_5)= P(X_5\mid X_4)P(X_4\mid X_3,X_2)P(X_3)P(X_2\mid X_1)P(X_1). \quad (1)$$

それらの効率的アルゴリズムでは，確率モデルの上記のような条件付独立性の確率構造をグラフで表現し，そのグラフ上をメッセージと呼ばれる一種の確率情報を伝搬させることにより計算を行っている．

確率推論の分野では，BN (Bayesian Network) または DAG (Directed Acyclic Graph) と呼ばれる有向グラフを用い，節に各命題（確率変数）を対応させ，親節から子節への有向枝に条件付確率を対応させて確率モデルの確率構造をグラフ化している．式（1）の例は，BN では図1 のように表される．BN における確信度更新，つまり周辺事後確率の計算のための効率的アルゴリズムとしては BP (Belief Propagation) と呼ばれているアルゴリズムがある．BP アルゴリズムでは有向グラフの順

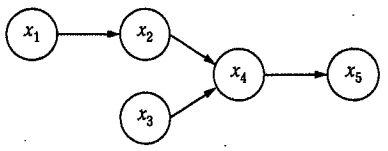


図1 式(1)の結合確率の構造を表現した Bayesian Network の例

(注2) MAP (Maximum a Posteriori Probability) 復号と呼ぶこともあるが，正確には情報シンボルに対する最大事後確率復号と呼ぶべきであり，先の最ゆう復号も情報系列（ブロック）に対する最大事後確率復号と呼ぶべきであろうが，ここでは慣例に従う．

(注3) 通信路が $P(\boldsymbol{y}\mid\boldsymbol{x})=\prod_{i=1}^{n}P(y_i\mid x_i)$ を満たすとき無記憶であると呼ぶ．無記憶通信路の通信路容量 C は以下で定義される．

$$C=\sup_{P(X)}\sum_X\sum_Y P(y\mid x)P(x)\log\frac{P(y\mid x)}{P(y)}.$$

(注4) この定理の証明には，ランダム符号化というテクニックが用いられており，ランダムに構成した符号のクラス全体での平均的性質から，上記の性質を満たす符号の存在を証明しているだけで，上記の性能を満たすある一つの符号を構成して証明を行っているわけではない．

(注5) 情報系列全体の事後確率 $P(\boldsymbol{u}\mid\boldsymbol{y})$ は比較的容易に計算可能であるが，最大値をとる $\hat{\boldsymbol{u}}$ を候補の中から探索する問題は符号長 N（あるいは情報系列長 K）の指数オーダの計算量が必要とされる．また，個々の情報シンボル u_k を推定する場合においては，個々のシンボルの周辺事後確率 $P(u_k\mid\boldsymbol{y})$ の最大値の比較は容易であるが，周辺事後確率自体の計算は事後結合確率 $P(\boldsymbol{u}\mid\boldsymbol{y})$ から周辺確率を計算することに対応し，こちらも一般には符号長 N の指数オーダの計算量が必要とされる．

(注6) 具体的には，シンボル長 K の情報系列 $\boldsymbol{u}$ に $K\times N$ 生成行列 G をかけてシンボル長 N の符号語 $\boldsymbol{x}=\boldsymbol{u}G$ を生成する線形符号のクラスである．特に生成行列の左側 $K\times K$ が単位行列であった場合，符号語は $\boldsymbol{x}=(\boldsymbol{u},\boldsymbol{c})$ となる．$\boldsymbol{c}$ をパリティと呼び，このような符号を組織符号と呼ぶ．

生成行列 G に対して $GH^T=0$ を満たす行列 H を検査行列と呼び，すべての符号 $\boldsymbol{x}$ が $\boldsymbol{x}H^T=0$ の性質を満たすことは符号語の生成過程より明らかであろう．受信系列が符号語と異なってしまった場合，受信系列 $\boldsymbol{y}$ は検査行列 H をかけても 0 ベクトルにはならず，通信路で誤りが生じたことが受信側で検出される．このベクトルをシンドロームと呼び，これを手がかりにある範囲内の誤りに対しては，ある種の方程式を解くことによって，誤りが生じたシンボルを特定することができる．このような代数的復号法の誤り率は一般に最ゆう復号のものより悪くなるが，計算量は符号長の多項式オーダであり，訂正可能な領域が理論的に保証されているなどの利点がある．

(注7) 前者が MPM 復号，後者が最ゆう復号の問題に対応している．

方向へのメッセージπと逆方向へのメッセージλを伝搬させることにより各節の周辺事後確率を求めている．この仕組みを式(1)の例で示そう．例えば，$X_1=x_1$が観測されたもとでX_5の周辺事後確率$P(X_5 \mid x_1)$を求めてみよう，

$$P(X_5 \mid x_1)= \sum_{X_4} P(X_5 \mid X_4) \sum_{X_3} \sum_{X_2} P(X_4 \mid X_3, X_2) P(X_3) P(X_2 \mid x_1) \quad (2)$$

結合確率から加算を単純に繰り返して周辺確率を求めるのに比べ，式（2）の右側の加算から順番に行っていくと計算量が少なくなることが分かる．またこの計算手順を図1のグラフと対応させて考えると，節X_4が節X_2から$P(X_2 \mid X_1=x_1)$の情報，節X_3から$P(X_3)$の情報を受け，$\sum_{X_3}\sum_{X_2}$によりX_4の周辺確率の計算を行い，その結果の情報をX_5に送っているように解釈できる．この情報がメッセージπに対応している．逆方向の周辺事後確率$P(X_1 \mid X_5=x_5)$の場合にもほとんど同様な考え方で，逆方向にメッセージを伝搬させればよいことが示せ，これがメッセージλに対応している．全体の周辺事後確率の計算にはπとλ両方向の伝搬を用いればよく，一般的に確率変数の数（節の数）の指数オーダ必要な計算量が，多項式オーダに低減されることになる．ただし，BPアルゴリズムで正しく周辺事後確率を求めることができるグラフはループのないDAGのクラス，つまり木に限定されている．

その他のグラフ表現として，2種類の節を用いるファクターグラフが挙げられる．ファクターグラフでは白丸の節（変数節）で確率変数を表し，黒丸の節（関数節）でその節と結ばれる白丸節の確率変数間に関連があることを表している．図2に例を示す．ファクターグラフ上の周辺事後確率計算アルゴリズムであるsum-productアルゴリズムはグラフ上の変数節と関数節で交互にメッセージを交換することで計算を行う．変数節では隣接する関数節から伝搬してきたメッセージを送り先の関数節からきたメッセージを除いて積をとりその関数節へ返す．関数節では隣接する変数節から伝搬してきたメッセージを，送り先の変数節のメッセージを除いて積と和で計算しその変数節に返す．

計算が和と積であるためsum-productアルゴリズム

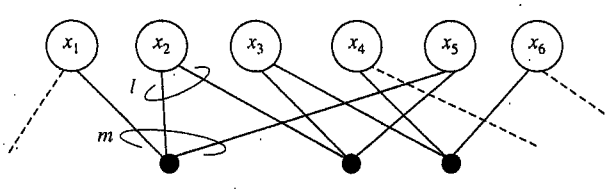


図2　LDPC符号のファクターグラフによる表現例

と呼ばれる．このアルゴリズムもグラフにループがない場合は正しい周辺事後確率を計算する保証があり，BPアルゴリズムと本質的には同じアルゴリズムであることが示されている．

4. ターボ符号の登場とLDPC符号の再発見

ほとんどの実用上重要な線形符号をグラフ表現すると木で表現できないため，効率良いメッセージ伝搬アルゴリズムを復号に使うことはできない．しかし，従来から用いられていた線形符号のクラスの中にも，少ない計算量で最ゆう復号を可能にする畳込み符号のようなクラスも存在していた．畳込み符号はシフトレジスタを用い符号化が行われ，符号長の多項式オーダでの符号化が可能である．畳込み符号と通信路モデルをBNで表現すると図3(A)のような木(注8)となるためBPアルゴリズムで正確な周辺事後確率が計算できる．符号理論の分野では，トレリスを用いて正確なMPM復号を符号長の多項式オーダで実現するBCJRアルゴリズムが有名であるが，これはこのBPアルゴリズムと同値のアルゴリズム[4]となっている．また，トレリスを用い最ゆう復号を行うアルゴリズムとしてビタビアルゴリズムが知られているが，これもcomposite beliefを求めるアルゴリズムと等価であることが示せる．

このように畳込み符号はBNの木で表現可能なため効率良い復号法が存在する．しかし，符号のランダム性が高くないため，最ゆう復号をしてもシャノン限界に近づくような性能は残念ながら得られない．そこで，もう少しランダム性を高めた符号を構成しようとすると，BNで木表現できなくなってしまう．やはりシャノン限界達成は困難と思われたが，この問題をうまく解決したのが正にターボ符号とLDPC符号であった．

4.1 ターボ符号と反復復号法

約10年前に登場したターボ符号は，情報系列を一つの組織的畳込み符号で符号化した符号語（情報系列とパリティ系列）を送信し，それと同時に，その情報系列をインタリーバにより置換した新たな系列を同様の畳込み符号で符号化し，そのパリティ系列も送信するもので，並列連接畳込み符号と考えられる．このインタリーバの部分により符号にランダム性が加味されたと考えることができる．

ランダム性が加味された代償にターボ符号全体をBNでグラフ表現すると，図3のようにループが存在し，そのままでは効率良いメッセージ伝搬アルゴリズムによる

(注8)　このBNではシフトレジスタの内部状態，情報シンボル，パリティシンボル等を節で表現しており，符号理論で従来から用いられてきたグラフ表現である内部状態を節で表現したトレリスと同等なグラフとなっている．

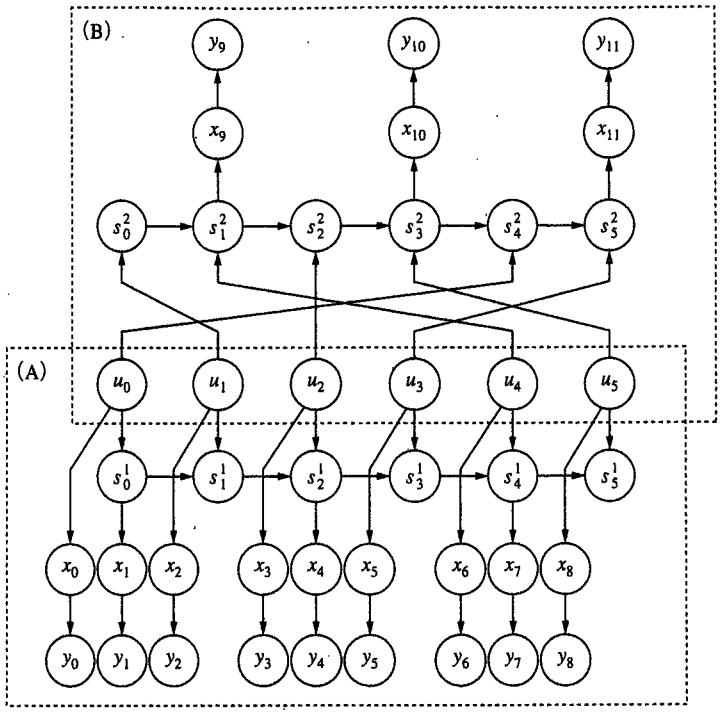


図3　ターボ符号の Bayesian Network による表現例

MPM 復号ができない．そこで次のような工夫を行った．ターボ符号全体を表現した BN を二つの畳込み符号の BN に分割してみると，それぞれは木となる．一方の木（図3 (A)）つまり通常の畳込み符号に対して BP アルゴリズムと等価な BCJR アルゴリズムを用いて効率良くグラフ(A)上のみの事後周辺確率(注9)を求め，(A)，(B)両方の木に共通な節つまり情報系列 $\boldsymbol{u}$ により(A)で計算された仮の事後確率を受け渡し，もう一方の木（図3 (B)）上でまた BCJR アルゴリズムによりグラフ(B)上での事後確率を求めるということを交互に反復して，グラフ全体における周辺事後確率の近似値を求めている．これがターボ復号法で，ループのあるグラフを二つの木に分割することで，効率の良いメッセージ伝搬アルゴリズムである BCJR アルゴリズムを交互に使用して近似値を求めている．このような反復計算により，本来指数オーダの計算量が必要な周辺確率の計算における計算量の問題を回避していると考えられる．

4.2　LDPC 符号と sum-product 復号法

ガラガーにより提案され最近再発見された LDPC 符号は，二元符号で1の密度が大変疎な検査行列 H で定義される符号である．正則 LDPC 符号では各列の1の数 l と各行の1の数 m をそれぞれ一定の小さな数に制限したもとで，なるべくランダムに1を配置するように検査行列が構成される．

このように LDPC 符号はランダム性を加味した符号となっているが，ファクターグラフ(注10)で表現すると図3のようになり，たくさんのループ構造を持っている．ループがあるグラフには逐次的な BP アルゴリズムは直接的に利用できない．しかし，並列の sum-product アルゴリズム(注11)は利用可能である．もちろん，ループのあるグラフに対して sum-product アルゴリズムが正確な周辺事後確率を計算する保証はないが，小ループを含まない LDPC 符号に対しては良好な結果が報告されている．復号の計算量は検査行列の1の数と関係があり(注12)，符号長の多項式オーダとなっている．

5.　符号の評価，設計

このように近似 MPM 復号を行っているにもかかわらず，ターボ符号と LDPC 符号はそれまでの符号では到達できなかったシャノン限界へ限りなく近づくことが実験的に確認されている．しかし，ループのあるグラフに対

(注 9)　これはグラフ全体の確率モデルの正しい事後周辺確率ではない．

(注 10)　符号語のシンボル X_i を変数節，シンドロームが0となるための検査行列の行の制約が関数節となる．

(注 11)　つまりグラフ上のすべての関数節からそれぞれに隣接するすべての変数節へメッセージを伝搬させ，次にすべての変数節から隣接するすべての関数節にメッセージを伝搬させる．このアルゴリズムが統計力学の平均場近似計算のための代表的アルゴリズムと等価であることも知られている．

(注 12)　符号を表現したファクターグラフの枝の数は検査行列の1の数となり，列の1の数 l は変数節に，行の1の数 m は関数節に，接続する枝の数となっている．sum-product アルゴリズムによる各節での計算量は l と m のオーダとなるため，検査行列の密度が低く，行や列の1の数が制限された LDPC 符号の復号計算量は少なくて済む．

してメッセージ伝搬アルゴリズムを用いた場合の近似性能や収束性について理論的に議論することは難しく，まだ明らかになっていない部分も多い．そのため，密度発展法（Density Evolution）やEXITチャート法などの数値計算的手法で性能の評価が行われている．

復号法の変化は符号の設計法にも影響を与える．従来は代数的復号（限界距離復号）が主に用いられていたため，符号の最小距離を大きくすることが設計の中心的指標であった．しかし，メッセージ伝搬アルゴリズムを復号に用いる符号の設計においては，メッセージ伝搬アルゴリズムの計算量を低く抑えかつ近似性能を悪化させない条件のもとで，性能の良い符号を考える必要がある．例えばLDPC符号の場合，小ループがないように構成する必要があり，射影幾何を用いてある長さ以下のループが存在しないLDPC符号を構成する方法なども提案されている．

6. ま と め

このように，ランダム性の高い符号に対しても近似的MPM復号可能なメッセージ伝搬アルゴリズムは，新たな流れを符号理論に作り出している．この流れは符号理論のみならず周辺の関連分野に大きな影響を与えつつある．またこの流れは，ある意味でシャノンによる情報理論の原点に立ち返る流れでもあり，今までの代数的復号や代数からの符号の評価や設計を原点から見直し，符号理論や情報理論の新たな発展につながっていくことが期待される．

文 献

(1) C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding", Proc. IEEE Int. Conf. Commun., pp.1064-1070, 1993.
(2) R.G. Gallager, Low-Density Parity-Check Codes, MIT Press, Cambridge, 1963.
(3) J. Pearl, Probabilistic reasoning in intelligent systems, Morgan Kaufmann, 1988.
(4) R.J. McElice, D.J.C. MacKay, and J. Cheng, "Turbo decording as an instance of Pearl's "Belief Propagation", IEEE J. Sel. Areas Commun., vol.16, no.2, pp.140-152, 1998.

松嶋 敏泰（正員）

昭55早大大学院修士課程了．同年，日本電気(株)入社．昭61早大大学院理工学研究科博士後期課程入学．平元横浜商大講師．平3同大学助教授．平4早大・理工・工業経営(現経営システム)助教授，平9同大学教授，現在に至る．平13ハワイ大客員研究員．知識情報処理及び情報理論とその応用に関する研究に従事．工博．

PAPER Special Section on Discrete Mathematics and Its Applications

Transformation of a Parity-Check Matrix for a Message-Passing Algorithm over the BEC

Naoto KOBAYASHI[†a)], *Student Member*, Toshiyasu MATSUSHIMA[†], *Member*, *and* Shigeichi HIRASAWA[†], *Fellow*

SUMMARY We propose transformation of a parity-check matrix of any low-density parity-check code. A code with transformed parity-check matrix is an equivalent of a code with the original parity-check matrix. For the binary erasure channel, performance of a message-passing algorithm with a transformed parity-check matrix is better than that with the original matrix.

key words: *low-density parity-check code, sum-product algorithm, binary erasure channel, generalized parity-check matrix*

1. Introduction

The best-known algorithm for the decoding of low-density parity-check (LDPC) codes is the sum-product algorithm [2]. The sum-product algorithm is a message-passing algorithm on a graphical model called a factor graph (FG). This algorithm is applied to various channels. Decoding performance of LDPC codes with the sum-product algorithm closely approaches the capacity of many communication channels.

The binary erasure channel (BEC) model was proposed in [8]. Decoding of linear codes over the BEC is equivalent to solving simultaneous equations. The maximum likelihood decoding (MLD) over the BEC is utilized by Gaussian elimination, however, the complexity is $O(N^3)$ where N is codeword length. An efficient message-passing decoding algorithm for LDPC codes over the BEC was proposed in [5]. The complexity of this algorithm is $O(N)$. A bit erasure rate (BER) of the message-passing algorithm are very low, but higher than that of MLD. Analyzing performance of this algorithm for LDPC codes has been studied using stopping sets [6]. Decoding errors of this algorithm occur if and only if all bits that are included in a stopping set are erased symbols in a received word.

In this paper, we propose transformation of a parity-check matrix of any LDPC code. A transformed parity-check matrix is a generalized parity-check (GPC) matrix [7] of the original parity-check matrix. Namely, the codes defined by both parity-check matrices have equivalent codewords. Transformation of a parity-check matrix to a GPC matrix is also proposed in [7]. However, it is transformation from a not sparse matrix to a sparse matrix; the decoding performance can be investigated only by simulations. Using a transformed parity-check matrix by our proposed transformation for decoding, the message-passing algorithm can correct the erased symbols of some bits that the algorithm cannot correct using the original parity-check matrix. And we prove it.

This paper is organized as follows: In Sect. 2, we introduce some notations and terminologies. In Sect. 3, we explain iterative decoding algorithms over the BEC. In Sect. 4, we propose the transformation of a parity-check matrix of any LDPC codes and show conditions that the transformation improves the performance of a message-passing algorithm. In Sect. 5, we discuss relationships between the loop elimination methods in a FG and our method. In Sect. 6, we show an example for decoding of LDPC codes with a transformed matrix by simulations. In Sect. 7, we summarize the paper.

2. Preliminaries

2.1 Notations

Let $X := \{X_1, X_2, \ldots, X_N\} \in \{0,1\}^N$ be a codeword that is encoded with a LDPC code. We assume the BEC with an erasure probability ϵ. Let $Y := \{Y_1, Y_2, \ldots, Y_N\} \in \{0,1,e\}^N$ be a received word, where e means an erased symbol. On receiving Y, we estimate a codeword $\hat{X} := \{\hat{X}_1, \hat{X}_2, \ldots, \hat{X}_N\} \in \{0,1,e\}^N$ with the message-passing decoding algorithm in Sect. 3. Let H be a parity-check matrix whose row and column length are M and N respectively, and let H_{mn} be the value of mth rows and nth columns of H ($1 \leq m \leq M$). We define $N(m) := \{n | H_{mn} = 1\}$ and $M(n) := \{m | H_{mn} = 1\}$ as sets of indices to indicate the position of symbol "1" for each rows and columns. For simple descriptions, let $A[i]$ be ith column of a matrix A. $A[i] = 0$ implies that all the elements of ith column of a matrix A are zero. For an arbitrary set S, let $|S|$ be the number of elements and let S_l be the lth element.

2.2 Stopping Sets

With $N(m)$ and $M(n)$, we can define a factor graph (FG) that represent any linear code [4]. Stopping sets in a FG are defined as follows [6]:

Manuscript received August 22, 2005.
Manuscript revised November 14, 2005.
Final manuscript received December 26, 2005.

[†]The authors are with the Dept. of Industrial & Management Systems Engineering, School of Science and Engineering, Waseda University, Tokyo, 169-8555 Japan.
a) E-mail: kobayashi@matsu.mgmt.waseda.ac.jp

DOI: 10.1093/ietfec/e89–a.5.1299

Definition 1 (Stopping Sets): A stopping set is a set of variable nodes with the property that every check node connected to a variable node in the stopping set is connected to at least two such nodes.

Stopping sets play the crucial role in the process of a message-passing decoding algorithm of LDPC codes over the BEC [5].

Lemma 1: Assume that we use a LDPC code to transmit over the BEC and that we decode the received word in a message-passing decoding algorithm until either the codeword has been recovered or until the decoder fails to progress further. Let denote χ the subset of the set of variable nodes which is erased by the channel. Then the set of erased symbols which remain when the decoder stops is equal to the unique maximal stopping set of χ.

For the parity-check matrix (1), $\{x_1, x_2, x_3, x_4\}$ is a stopping set; the message-passing decoding algorithm cannot correct all erased symbols if a received word Y is $\{e, e, e, *, *\}$, where $*$ implies an arbitrary symbol.

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}. \tag{1}$$

2.3 Generalized Parity-Check Matrix

If a parity-check matrix for an (N, K) code has more than N columns, it is called a generalized parity-check (GPC) matrix [7]. In general, an M' by N' GPC matrix for an (N, K) code has $N' > N$ columns, and $M' \geq N' - K$ rows, where M' rows span an $N' - K$ dimensional sub-space. As with [7], we follow the convention of always listing the N columns of the GPC matrix that correspond to the N bits of a codeword first. We refer to these bits as *symbol bits*. About the additional $N' - N$ columns, we refer to them as *state bits*. State bits help define the code, however they would not be transmitted. Let s be a set of all indices of state bits, that is $s = \{N + 1, N + 2, \ldots, N'\}$. For a decoder, a received bit which correspond to state bit is treated as always an erased symbol, that is $Y_n = e$, where $n \in s$. A GPC matrix A satisfies $xA = 0$, where x is an N'-tuple of bits such that the first N bits form a codeword v, and the last $N' - N$ state bits are uniquely determined given v and A^T.

3. Decoding Algorithms over the BEC

Decoding of linear codes over the BEC is equivalent to solving simultaneous equations [8]. Thus we can utilize Gaussian elimination for decoding over the BEC. Although this method is MLD, the complexity is $O(N^3)$. In this section, we explain two types of the decoding algorithm over the BEC. One is a message-passing decoding algorithm using a FG which represents a code; this algorithm is described in [5]. The complexity of this algorithm is $O(N)$. The other is a decoding algorithm by searching for rows that have only one symbol "1." We will use the latter algorithm for the proof of Theorem 1. Let $\hat{X}_n^{(t)}$ be an estimated codeword on tth iteration for each algorithms. We consider using a GPC matrix for decoding; the range of n is $1 \leq n \leq N'$ and the range of m is $1 \leq m \leq M'$.

3.1 Algorithm I: Message-Passing Algorithm

Let q_{nm} and r_{mn} be messages between the check node m and the variable node n in a FG. We initialize $\hat{X}_n^{(0)} = Y_n$ for all n and setting $t = 1$. Algorithm I computes each pair of (m, n) satisfying $H_{mn} = 1$ in Step 1 and Step 2:

Step 1.

$$q_{nm} = \hat{X}_n^{(t-1)}. \tag{2}$$

Step 2.

$$r_{mn} = \begin{cases} \sum_{n'} q_{n'm} \bmod 2, & q_{n'm} \neq e \text{ for all } n'; \\ e, & \text{otherwise,} \end{cases} \tag{3}$$

where $n' = \{N(m) \setminus n\}$.

Step 3. Compute for all n:
Let m' be any $m \in M(n)$ such that $r_{mn} \neq e$.

$$\hat{X}_n^{(t)} = \begin{cases} e, & r_{mn} = e \text{ for all } m \in M(n); \\ r_{m'n}, & \text{otherwise.} \end{cases} \tag{4}$$

Step 4.
If $\hat{X}_n^{(t)} = \hat{X}_n^{(t-1)}$ for all n, then the algorithm halts. Setting $t = t + 1$ and continue Step 1 until $\hat{X}_n$ is not e for all n.

3.2 Algorithm II

Let $E^{(t)}$ be a set which has indices of symbol bits and state bits such that $\hat{X}_n^{(t)} = e$ in an iteration t. Let $\eta^{(t)}$ be a matrix which has the same number of columns and rows as H. We initialize $\hat{X}_n^{(0)} = Y_n$ for all n and setting $t = 0$.

Step 1. For all n, setting columns of η as follows:

$$\eta^{(t)}[n] = \begin{cases} H[n], & n \in E^{(t)}; \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

If $|E^{(t)}| = 0$ or all rows of $\eta^{(t)}$ have two or more symbol "1" then this algorithm halts.

Step 2.
For all m such that mth row of $\eta^{(t)}$ has only one symbol "1" in nth column,

$$\hat{X}_n^{(t+1)} = \sum_{n' \in N(m) \setminus n} \hat{X}_{n'}^{(t)} \bmod 2. \tag{6}$$

(We call this operation "$\hat{X}_n$ is determined by mth row").

For all $\hat{X}_{n'}^{(t+1)}$ which is not updated in this iteration, we substitute the value of $\hat{X}_{n'}^{(t)}$ in $\hat{X}_{n'}^{(t+1)}$.

Step 3.

Setting $t = t + 1$ and continue Step 1.

Note that the Algorithm II is equivalent to message-passing algorithm. In Step 2, we need to find the row that has only one symbol "1"; such the row is corresponding to a mth row which the message r_{mn} such that $\hat{X}_n = e$ is not e in (3) of the message-passing algorithm. An XOR operation in Step 2, that is (6), also corresponds to (4) in the message-passing algorithm. Hence, this algorithm and the message-passing algorithm are completely equivalent algorithms; the output of algorithms $\hat{X}$ are same. In addition, all rows of $\eta^{(t)}$ have two or more symbol "1" in Step 1 implies that $E^{(t)}$ is a stopping set.

For example, we consider decoding for the code with the parity-check matrix (1). If $\hat{X}^{(t)} = \{e, e, 0, 0, 0, 0\}$, we have

$$\eta^{(t)} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{7}$$

The second and third rows of $\eta^{(t)}$ have only one symbol "1." Hence we can determine $\hat{X}_1^{(t+1)}$ and $\hat{X}_2^{(t+1)}$ as following:

$$\hat{X}_1^{(t+1)} = (\hat{X}_3^{(t)} + \hat{X}_5^{(t)}) \bmod 2, \tag{8}$$

$$\hat{X}_2^{(t+1)} = (\hat{X}_3^{(t)} + \hat{X}_6^{(t)}) \bmod 2. \tag{9}$$

Eq. (8) is derived from the second row of H, and Eq. (9) is derived from the third row of H.

If $\hat{X}^{(t)} = \{e, e, e, e, 0, 0\}$, we have

$$\eta^{(t)} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}. \tag{10}$$

There is no row which has only one symbol "1." Hence we cannot determine symbols of codeword any more; this algorithm halts.

4. Transformation of a Parity-Check Matrix

In this section, we propose transformation of a parity-check matrix. A transformed matrix is a GPC matrix of the original parity-check matrix. We also show that using a transformed parity-check matrix for decoding, the message-passing algorithm can correct erased symbols which the algorithm cannot correct using the original parity-check matrix.

4.1 Transformation of a Parity-Check Matrix

We consider an arbitrary submatrix P of H that satisfies the following conditions:

- P is a submatrix of H whose row and column length are k and l, respectively.
- Any $k-1$ rows of P are linear independent, and k rows of P are linear dependent.

This transformation is based on P. To apply arbitrary row permutations and column permutations, we can deal with a combination of any columns and any rows of H as submatrix P. We thus define P^c to the ordered set of indices of columns of H corresponding to each column of P; we define P^r to the ordered set of indices of rows of H corresponding to each row of P. They are ordered by the order of the index of columns or rows of P.

Let O be an M by k matrix in which all elements are zero. Let I be a k by k identify matrix. Let Q be a k by N matrix that is defined as follows:

$$Q[i] = \begin{cases} P[j], & i \in P^c, P_j^c = i; \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

Let C be an 1 by $N + k$ matrix in which the all elements at first N columns are zero, and the others are one.

We construct an $(M + k + 1)$ by $(N + k)$ matrix A' as follows:

$$A' := \left[\begin{array}{c|c} H & O \\ \hline Q & I \\ \hline \multicolumn{2}{c}{C} \end{array}\right]. \tag{12}$$

A' is a GPC matrix of H, which the first to Nth columns correspond to symbol bits; the $(N+1)$th to $(N+k)$th column correspond to state bits. This is because the following reasons. Note that no new linear constraint is applied among all symbol bits of the code with an original matrix. The first row to Mth row of A' (that is H and O) is equivalent to H. The $(M + 1)$th row to $(M + k)$th row of A' (that is Q and I) imply that state bits are uniquely determined given symbol bits. The last row of A' (that is C) imply that all rows of P are linear dependent.

We construct A that is a modified matrix of A' with row operations. The row operations are to add $(M + i)$th row to ith row of for all $i (0 \le i \le k - 1)$. It is obvious that A is also GPC matrix of H. In addition, if H is a sparse matrix, Q is also a sparse matrix. Thus, it is obvious that A is a sparse matrix.

For example, we consider the parity check matrix (1) and

$$P = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}, \tag{13}$$

that corresponds to the first 3 rows and 3 columns of H, that is $P^c = \{1, 2, 3\}$ and $P^r = \{1, 2, 3\}$. Thus, we have

$$A' = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}, \quad (14)$$

and

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}. \quad (15)$$

The numbers of symbols "1" in C, I and O are k, k and 0, respectively. When a matrix is transformed from A' to A, the number of symbols "1" in H decreases the number of symbols "1" in Q and increases the number of symbols "1" in I. Thus the number of symbols "1" in a parity-check matrix increases $3k$ by this transformation.

4.2 The Message-Passing Algorithm with a Transformed Matrix

Theorem 1: If a GPC matrix A, which is transformed based on P, satisfies the following conditions, the message-passing algorithm with A can correct some erased bits that cannot be corrected by them with the original matrix H:

- All symbol bits i such that $i \in P^c$ are contained in a stopping set S.
- For all $m \in P^r$ and $n \in S$, there is only one element that is satisfied $A_{mn} = 1$; others are 0.

For example, the submatrix P in (13) and the GPC matrix A in (15) satisfy the above conditions because for all $m \in \{1, 2, 3\}$ and $n \in \{1, 2, 3, 4\}$, there is only one element that is satisfied $A_{mn} = 1$, that is A_{14}.

Proof: We show that (A) for any received word, there is no symbol bit such that the message-passing algorithm with H can be determined but the message-passing algorithms with A cannot be determined; (B) for a received word, there are symbol bits such that the message-passing algorithm with H cannot be determined but the message-passing algorithms with A can be determined. We define n' and m' as the column index and the row index that are satisfied $A_{m'n'} = 1$ on the second condition.

(A) We consider hth row such that $h \in P^r$. For the Algorithm II in the $(t+1)$th iteration, when the hth row of $\eta^{(t)}$ has only one symbol "1" in the nth column, $\hat{X}_n$ is determined by the constraint of the hth row of H. Let E' be $E^{(t)}$ for the occasion. We define such symbol bit n as n^*. E' is equivalent to the positions of erase symbols in the received word that the message-passing algorithm with H can determine $\hat{X}_{n^*}$. It is only necessary for the proof of (A) that we show the Algorithm II with A can determine $\hat{X}_{n^*}$ on the received word that positions of erase symbols are defined by $\{E' \cup s\}$[†].

First, we consider the case of $n^* \in P^c$. By the transformation from A' to A, $A_{hi} = 0$ for all $i \in P^c$; therefore the hth row of $\eta^{(t)}$ has only one symbol "1" at a column corresponding to a state bit. The $(h + M)$th row of $\eta^{(t)}$ has two symbols of "1" at a column corresponding the state bit and n^*. Hence, the Algorithm II can determine the state bit by the hth row of A in the $(t + 1)$th iteration ; the Algorithm II can determine the symbol bit n^* by the $(h + M)$th row of A in the $(t + 2)$th iteration.

Next, we consider the case of $n^* \notin P^c$. By (11) and (12), $A_{(h+M)j} = 0$ for all $j \notin P^c$; therefore the $(h + M)$th row of $\eta^{(t)}$ has only one symbol "1" at a column corresponding to a state bit. The hth row of $\eta^{(t)}$ has two symbols of "1" at a column corresponding the state bit and n^*. Hence, the Algorithm II can determine the state bit by the $(h+M)$th row of A in the $(t+1)$th iteration; the Algorithm II can determine the symbol bit n^* by hth row of A in the $(t + 2)$th iteration.

(B) We assume that

$$\hat{X}_n^{(t)} = \begin{cases} e, & n \in S; \\ 0 \text{ or } 1, & \text{otherwise.} \end{cases} \quad (16)$$

In this case, all rows of $\eta^{(t)}$ of the Algorithm II with H have two or more symbols of "1" as like Eq. (10). However, for the Algorithm II with A, the mth rows of $\eta^{(t)}$ of has only one symbol "1" for all $m \in \{P_r \setminus m'\}$; they are in the columns of $\eta^{(t)}$ corresponding to state bits. We thus can determine X_n for all state bit $(N + 1) \le n \le (N + k)$ except the state bit $(N + j)$ such that $P_{rj} = m'$.

In the $(t + 1)$th iteration of the algorithm, the last row of $\eta^{(t+1)}$ has only one symbol "1" (since in the last row of A, there are symbols of "1" in only the columns corresponding to state bits). Hence, we can determine X_{N+j} such that $P_{rj} = m'$. In the $(t + 2)$th iteration of the algorithm, the m' row of $\eta^{(t+1)}$ has only one symbol "1" in the n'th column. We can determine $X_{n'}$, and n' is in the stopping set. For example, the behavior of the Algorithm II with the matrix (15) is in Appendix. As the example, if all subsets of the set $\{S \setminus X_{n'}\}$ are not stopping sets, we can determine $\hat{X}_i$ for all $i \in S$.

Note: If the conditions in Theorem 1 are satisfied, $\hat{X}_{n'}$ is specified from all linear constraints of the mth rows of H, where $m \in P^r$. For example, we assume a received word $Y = \{e, e, e, e, *, *\}$ for the decoding with parity-check matrix (1). To sum the first row, the second and third row of H, we have $\hat{X}_4 = \hat{X}_5$. We can say that our proposed transformation allowed the message-passing algorithm to obtain such solution. By our proposed transformation, we can correct erased symbols in any stopping set that satisfies the conditions in Theorem 1. If a post-decoding erasure rate depends

[†]As above description, a decoder treat state bits as erase symbols.

on a stopping set (i.e. at low erasure probability) that satisfies the conditions in Theorem 1 for a code, we can improve the erasure rate by our proposed transformation against the stopping set.

5. Relationship to Loop Elimination Methods

Pearl showed that any loops in a graphical model could be erased by the clustering method [9]. This method can be also applied to a FG [4] and can erase small loops in a FG [10]. The FG that represented by submatrix P certainly includes loops. For example, the FG that represented by (13) is length-6 loops. The sum-product algorithm calculates local joint probabilities in a clustered node. Over the BEC, that is equivalent to solving of a simultaneous equation. Our proposed transformation is equivalent to loop eliminations by the clustering method over the BEC[†]. A similarly transformation is proposed about length-4 loop eliminations in [11]. Hence we will use a result of this study to investigate the clustering method over another channels.

6. Numerical Experiments

6.1 Experimental Conditions

We investigate average performances of our proposed transformation by simulations. We examine four code ensembles of which details are shown in Table 1 (denoted by "Ensemble 1-4"). w_c and w_r imply a column weight and a row weight of a parity-check matrix, respectively. #1 implies the number of symbols of "1" in a parity-check matrix. The Ensemble 2 satisfies that the FG that represents a parity-check matrix does not contain length-4 loops. The Ensemble 1, Ensemble 3 and Ensemble 4 satisfy that any two columns and any two rows of a parity-check matrix don't have greater than two "1" symbols in common. This condition means that it is possible to exist length-4 loops in the FG that represents a parity-check matrix. We generate five parity-check matrices from each ensemble. We first find stopping sets of each code as many as possible; we also find submatrices that satisfy conditions of P (in Sect. 4.1) in the parity-check matrix for each stopping set. For each submatrix, if it satisfies the conditions in Theorem 1 then we transform the parity-check matrix based on the submatrix. We apply transformations to the parity-check matrix as many as possible in order from short stopping sets. In Table 2, we show the average data of five transformed parity-check matrixes for each ensemble. N' and M' are column length and row length of a transformed matrix, respectively. R_i implies weight distribution of each rows, that is a rate of the number of rows that have i "1" symbols to M'. We compare decoding performance of these ensembles over the BEC using a message-passing algorithm with a transformed matrix to that with the original parity-check matrix. For each matrix and an erasure probability, 2×10^7 codewords are transmitted.

Table 1 Original matrices.

	N	$Rate$	(w_c, w_r)	#1
Ensemble 1	128	0.5	(4,8)	512
Ensemble 2	128	0.5	(4,8)	512
Ensemble 3	500	0.6	(4,10)	2000
Ensemble 4	500	0.8	(3,15)	1500

6.2 Results and Discussions

Figure 1–Fig. 4 show post-decoding bit erasure rates (BER) and post-decoding word erasure rates (WER) for each ensemble. For the result of the Ensemble 4, we show BER and WER by a MLD. For the Ensemble 3, BER and WER by a MLD are zero, where an erasure probability is between 0.20 and 0.29.

For the result of the Ensemble 1, both BER and WER with transformed matrices are better than that with the original matrices. On the other hand, BER and WER with transformed matrices are only a little better than that with the original matrices for the result of the Ensemble 2. We consider that if submatrix P corresponds to a length-4 loop (i.e. P is a 2 by 2 matrix which elements are all one) for our proposed transformation, not only a one stopping set but also a lot of stopping sets satisfy Theorem 1 for the P. In another case about P, only one or a few stopping sets satisfy Theorem 1 for the P.

For the result of the Ensemble 4, BER and WER by a message-passing algorithm are identical with them by a MLD at low erasure probabilities. It implies that erased symbol bits in small stopping sets in the Ensemble 4 cannot be determined by a MLD, that is, the solution of simultaneous equations to solve for the erased symbol bits is not obvious. Our proposed transformation cannot be applied to such the stopping sets because there is not a subset P that satisfies Theorem 1 for the stopping sets (as we described at Note in Sect. 4). On the other hand, BER and WER by a MLD are zero for the Ensemble 3 at low erasure probabilities. Hence it is possible that there is a subset P that satisfies Theorem 1 for small stopping sets. As shown in the result, we can improve BER and WER especially at low era-

[†]Q in (12) contains in similar loops in P, so we cannot say that the proposed transformation is loop eliminations method.

Table 2 Transformed matrices.

	N'	M'	R_i : average weight distribution of rows	#1
Ensemble 1	305.8	280.2	R_3 : 0.7607 R_4 : 0.1214 R_5 : 0.1071 R_6 : 0.0107	943.8
Ensemble 2	300.2	290.2	R_3 : 0.7448 R_4 : 0.0517 R_5 : 0.1206 R_6 : 0.0793 R_7 : 0.0034	1026.2
Ensemble 3	1141.4	993.4	R_3 : 0.7452 R_4 : 0.0624 R_5 : 0.0363 R_6 : 0.0866 R_7 : 0.0574 R_8 : 0.0121	3660.4
Ensemble 4	909.0	592.4	R_3 : 0.7889 R_4 : 0.0422 R_7 : 0.0051 R_8 : 0.0253 R_9 : 0.0608 R_{10} : 0.0439 R_{11} : 0.0186 R_{12} : 0.0118 R_{13} : 0.0034	2457.4

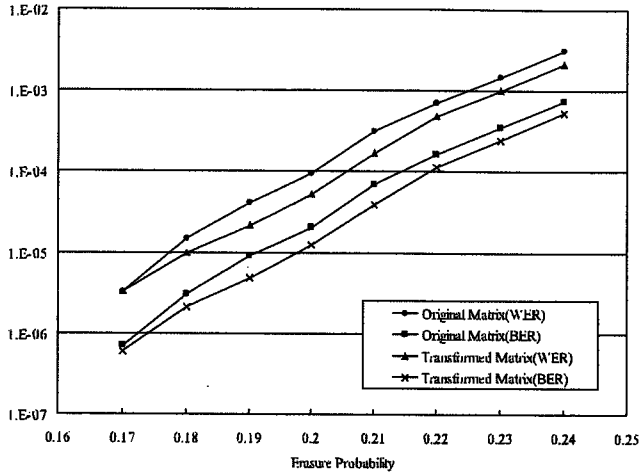


Fig. 1 The result of ensemble 1.

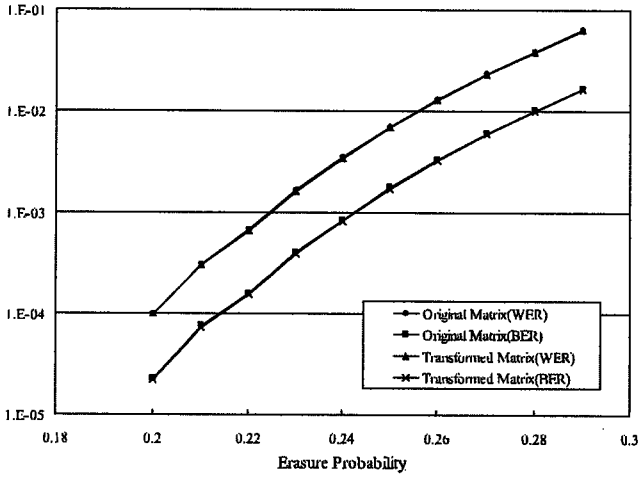


Fig. 2 The result of ensemble 2.

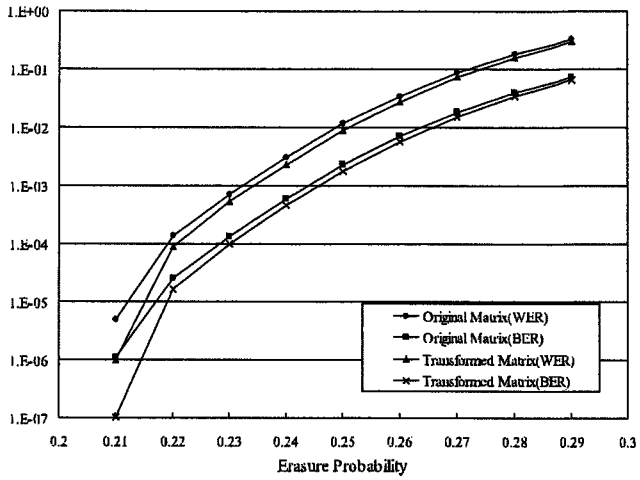


Fig. 3 The result of ensemble 3.

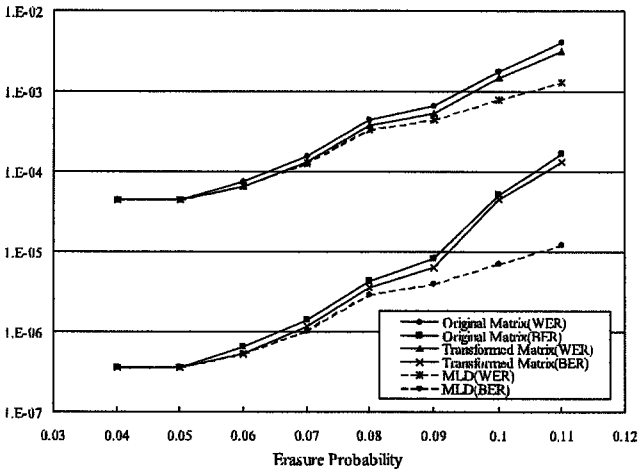


Fig. 4 The result of ensemble 4.

sure probabilities with our proposed transformation against small stopping sets. We consider that our proposed transformation is effective for an ensemble with properties similar to the Ensemble 3.

We consider the tradeoff between the increase of complexity and the improvements of performance by our proposed transformation. Assuming that the computational complexity of a message-passing algorithm is proportional to the number of symbols of "1" in a parity-check matrix, the computational complexity of a message-passing algorithm with a transformed matrix is about 1.5–2 times for that with the original matrix in these simulations as shown in the Table 1 and Table 2. Note that in these simulations, our proposed transformation applies to as many as possible of stopping sets for existing code ensembles. As results of these simulations and descriptions of Note in Sect. 4, if a code satisfies the following conditions, we can improve post-decoding erasure rates with a little increase of complexity by transformation against only the stopping sets:

- The numbers of the smallest stopping sets and stopping sets whose size is near the size of the smallest stopping sets are low.
- There are submatrices P satisfying Theorem 1 for the stopping sets.

7. Concluding Remarks

In this paper, we propose a transformation of a parity-check matrix of any LDPC codes. Over the BEC, the iterative decoding algorithms with this transformation can correct erased symbols of some symbol bits where the algorithm cannot correct using the original parity-check matrix. For the future, we should investigate a good class of codes that is suited to the transformation; we will use the result of this study to investigate encoding and decoding methods over another channels.

Acknowledgment

The authors would like to acknowledge all member of Hira-

sawa Lab. and Matsushima Lab. for their helpful suggestions to this work. This research is partially supported by Category No.15560338 of Grant-in-Aid for Scientific Research, Japan Society for the Promotion of Science.

References

[1] R.G. Gallager, "Low-density parity-check codes," IRE Trans. Inf. Theory, vol.8, no.1, pp.21–28, Jan. 1962.
[2] D.J.C. Mackay, "Good error-correcting codes based on very sparse matrices," IEEE Trans. Inf. Theory, vol.45, no.2, pp.399–431, March 1999.
[3] D.J.C. Mackay and R.M. Nealm, "Near Shannon limit performance of low-density parity-check codes," Electron. Lett., vol.32, pp.1645–1646, Aug. 1996.
[4] F.R. Kschischang, B.J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," IEEE Trans. Inf. Theory, vol.47, no.2, pp.498–519, Feb. 2001.
[5] M. Luby, M. Mitzenmacher, M. Shokrollahi, and D. Spielman, "Efficient erasure correcting codes," IEEE Trans. Inf. Theory, vol.47, no.2, pp.569–584, Feb. 2001.
[6] C. Di, D. Proietti, I.E. Telatar, T.J. Richardson, and R.L. Urbanke, "Finite-length analysis of low-density parity-check codes on the binary erasure channel," IEEE Trans. Inf. Theory, vol.48, no.6, pp.1570–1579, June 2002.
[7] J.S. Yedidia, J. Chen, and M. Fossorier, "Generating code representations suitable for belief propagation decoding," Proc. 40th Allerton Conference Commun., Control, and Computing, CD-ROM, Monticello, IL, Oct. 2002.
[8] P. Elias, "Coding for two noisy channels," Proc. 3rd London Symp., Information Theory, pp.61–76, 1955.
[9] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inferene, Morgan Kaufmann, 1988.
[10] N. Kobayashi, T. Matsushima, and S. Hirasawa, "A note on a decoding algorithm of codes on graphs with small loops," Proc. IEEE Information Theory Workshop, pp.108–112, 2005.
[11] K. Kasai, T. Shibuya, and K. Sakaniwa, "A code-equivalent transformation removing cycles of length four on Tanner graphs," IEICE Technical Report, IT2004-42, Sept. 2004.

Appendix: An Example of Erasure Corrections by the Algorithm II

We show an example of the behavior of the Algorithm II with Eq. (15). We assume that $E^{(t)} = \{1, 2, 3, 4, 7, 8, 9\}$. We have

$$\eta^{(t)} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}. \quad (A\cdot 1)$$

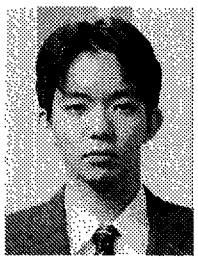
We determine X_8 and X_9 by the second and third rows of $\eta^{(t)}$. We have $E^{(t+1)} = \{1, 2, 3, 4, 7\}$ and

$$\eta^{(t+1)} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}. \quad (A\cdot 2)$$

We determine X_7 by the last row of $\eta^{(t+1)}$. We have $E^{(t+2)} = \{1, 2, 3, 4\}$ and

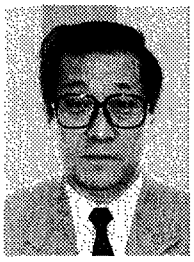
$$\eta^{(t+2)} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \quad (A\cdot 3)$$

If we will continue iterations, all symbol bits are determined.



Naoto Kobayashi was born in Chiba, Japan, on Oct. 1st, 1978. He received the B.E. degree and M.E. degree in Industrial and Management Systems Engineering from Waseda University, Tokyo, Japan, in 2001 and 2003, respectively. He is currently a doctoral student in Industrial and Management Systems Engineering at the Graduate School of Waseda University. His research interests are coding theory and artificial intelligence.

Toshiyasu Matsushima was born in Tokyo, Japan, on Nov. 26, 1955. He received the B.E. degree, M.E. degree and Dr.E. degree in Industrial and Management Systems Engineering from Waseda University, Tokyo, Japan, in 1978, 1980, and 1991, respectively. From 1980 to 1986, he was with Nippon Electric Corporation, Kanagawa, Japan. From 1986 to 1992, he was a lecturer at Department of Management Information, Yokohama College of Commerce. From 1993, he was an associate professor and since 1996 has been a professor of School of Science and Engineering, Waseda University, Tokyo, Japan. From 2001 to 2002, he was a Visiting Professor of the University of Hawaii at Manoa, U.S. From 2005, he has been the Chairperson of the Technical Group on Information Theory of IEICE. His research interests are information theory and its application, statistics, and artificial intelligence. He is a member of the IEEE, the Society of Information Theory and its Applications, the Japan Society for Quality Control, the Japan Industrial Management Association, and the Japan Society of Artificial Intelligence.



Shigeichi Hirasawa was born in Kobe, Japan, on Oct. 2, 1938. He received the B.S. degree in mathematics and the B.E. degree in electrical communication engineering from Waseda University, Tokyo, Japan, in 1961 and 1963, respectively, and the Dr.E. degree in electrical communication engineering from Osaka University, Osaka, Japan, in 1975. From 1963 to 1981, he was with the Mitsubishi Electric Corporation, Hyogo, Japan. Since 1981, he has been a professor of School of Science and Engineering, Waseda University, Tokyo, Japan. In 1979, he was a Visiting Scholar in the Computer Science Department at the University of California, Los Angeles (CSD, UCLA), CA. He was a Visiting Researcher at the Hungarian Academy of Science, Hungary, in 1985, and at the University of Trieste, Italy, in 1986. In 2002, he was also a Visiting Faculty at CSD, UCLA. From 1987 to 1989, he was the Chairperson of Technical Group on Information Theory of IEICE. He received the 1993 Achievement Award, and the 1993 Kobayashi-Memorial Achievement Award from IEICE. In 1996, he was the President of the Society of Information Theory and Its Applications (Soc. of ITA). His research interests are information theory and its applications, and information processing systems. He is an IEEE Fellow, a member of Soc. of ITA, the Operations Research Society of Japan, the Information Processing Society of Japan, the Japan Industrial Management Association, and Institute for Operations Research and the Management Sciences.

PAPER Special Section on Discrete Mathematics and Its Applications

A Note on Construction of Orthogonal Arrays with Unequal Strength from Error-Correcting Codes

Tomohiko SAITO[†a)], *Student Member*, Toshiyasu MATSUSHIMA[†], *Member, and* Shigeichi HIRASAWA[†], *Fellow*

SUMMARY Orthogonal Arrays (OAs) have been playing important roles in the field of experimental design. It has been known that OAs are closely related to error-correcting codes. Therefore, many OAs can be constructed from error-correcting codes. But these OAs are suitable for only cases that equal interaction effects can be assumed, for example, all two-factor interaction effects. Since these cases are rare in experimental design, we cannot say that OAs from error-correcting codes are practical. In this paper, we define OAs with unequal strength. In terms of our terminology, OAs from error-correcting codes are OAs with equal strength. We show that OAs with unequal strength are closer to practical OAs than OAs with equal strength. And we clarify the relation between OAs with unequal strength and unequal error-correcting codes. Finally, we propose some construction methods of OAs with unequal strength from unequal error-correcting codes.

key words: *experimental design, orthogonal arrays, unequal error-correcting codes, nonlinear codes*

1. Introduction

Experimental design is a technique to reduce the number of experiments in quality management [11]. Good experimental design is achieved so that we can estimate the effects of factors and their interactions where the number of experiments is as few as possible. In order to reduce the number of experiments, it is important to construct Orthogonal Arrays (OAs).

Many algorithms to construct OAs were proposed in researches of experimental design [3], [10], [11], [13]. Most of these algorithms are based on properties of projective geometry. These algorithms are so useful that many experimenters use these algorithms to design experiments. But it is still hard problem to construct an optimal OA that have the minimum number of rows for given number of columns and for given interaction effect.

On the other hand, it is known that OAs are closely related to error-correcting codes. The relations are divided into two types: the first is the relation between linear OAs and linear codes, the second is the one between nonlinear OAs and nonlinear codes. And many good OAs were constructed from both linear and nonlinear error-correcting codes [4]. But these OAs are suitable for only cases that equal interaction effects can be assumed, for example, all two-factor interaction effects. Therefore, these OAs are called *OAs with equal strength* in this paper. In experimental design, it is usually assumed that there are partial interaction effects, for example, there is one two-factor interaction effect, but there are not other two-factor interaction effects. So we cannot say that OAs from error-correcting codes are practical.

In this paper, we define *OAs with unequal strength.* We show types of interaction effects that OAs with unequal strength can estimate, and that OAs with unequal strength are closer to practical OAs than OAs with equal strength.

Next, we show the relation between linear OAs with unequal strength and linear unequal error-correcting codes [6]. And we present two construction methods of linear OAs with unequal strength from linear unequal error-correcting codes that is in [2], [6].

Next, we show the relation between nonlinear OAs with unequal strength and codes. And we extend one of the construction methods of linear OAs with unequal strength to construct nonlinear OAs with unequal strength.

Lastly, we show some examples of OAs with unequal strength by our proposed construction methods. And we show that these OAs are more suitable than OAs with equal strength for cases of partial interaction effects.

This paper is organized as follows. In Sect. 2, we describe experimental design as preliminary. In Sect. 3, we describe some relations between OAs and error-correcting codes shown in [4]. In Sect. 4, we define OAs with unequal strength, and show types of interaction effects that OAs with unequal strength can estimate. Furthermore, we show the relation between OAs with unequal strength and unequal error-correcting codes, and propose some construction methods of OAs with unequal strength. In Sect. 5, we show some examples of OAs with unequal strength by our construction methods. The last Sect. 6 concludes our research.

2. Experimental Design

2.1 Factorial Experiments

In this section, we provide a short introduction of factorial experiments. For the detailed explanation, refer to [11].

Example 1: In a certain factory, materials(F_1), machines (F_2), temperatures(F_3) are factors that may affect a ratio of defective products. Each factor has two levels:

Manuscript received August 22, 2005.
Manuscript revised December 6, 2005.
Final manuscript received January 10, 2005.

[†]The authors are with the Department of Industrial and Management Systems Engineering, School of Science and Engineering, Waseda University, Tokyo, 169-8555 Japan.
a) E-mail: tomohiko@matsu.mgmt.waseda.ac.jp

DOI: 10.1093/ietfec/e89–a.5.1307

$F_1 : F_0^1$ (made in A company), F_1^1 (B company)
$F_2 : F_0^2$ (a new machine), F_1^2 (an old machine)
$F_3 : F_0^3$ (100°C), F_1^3 (200°C)
Suppose we want to analyze how the level of factors affects the ratio of defective products.

Let y_{ν_1,ν_2,ν_3} be the ratio of defective products which is given when we experiment with level combination $F_{\nu_1}^1 F_{\nu_2}^2 F_{\nu_3}^3$. We assume that,

$$y_{\nu_1,\nu_2,\nu_3} = \mu + \alpha_{\nu_1}^1 + \alpha_{\nu_2}^2 + \alpha_{\nu_3}^3 + \alpha_{\nu_1,\nu_2}^{1,2} + \alpha_{\nu_1,\nu_3}^{1,3} + \alpha_{\nu_2,\nu_3}^{2,3} + e_{\nu_1,\nu_2,\nu_3},$$

where μ is a constant that has no relation with levels, that is called the *central effect*, $\alpha_{\nu_i}^i$ is the effect that appears when F_i is set for ν_i, that is called the *main effect* of F_i, $\alpha_{\nu_{i_1},\nu_{i_2}}^{i_1,i_2}$ is the effect that appears by combining $F_{\nu_{i_1}}^{i_1}$ with $F_{\nu_{i_2}}^{i_2}$, that is called the *interaction effect* of $F_{i_1}F_{i_2}$, and e_{ν_1,ν_2,ν_3} is a *random error*. Further, we assume that for any $i \in \{1,2,3\}$,

$$\sum_{\nu_i \in \{0,1\}} \alpha_{\nu_i}^i = 0, \tag{1}$$

and, for any $i_1, i_2 \in \{1,2,3\}$, $i_1 \neq i_2$,

$$\sum_{\nu_{i_2} \in \{0,1\}} \alpha_{\nu_{i_1},\nu_{i_2}}^{i_1,i_2} = 0 \quad for \ \forall \nu_{i_1} \in \{0,1\}, \tag{2}$$

$$\sum_{\nu_{i_1} \in \{0,1\}} \alpha_{\nu_{i_1},\nu_{i_2}}^{i_1,i_2} = 0 \quad for \ \forall \nu_{i_2} \in \{0,1\}. \tag{3}$$

Then we can estimate all main effects and interaction effects of two factors if we experiment with all level combinations;

$(F_0^1F_0^2F_0^3), (F_0^1F_0^2F_1^3), (F_0^1F_1^2F_0^3), (F_0^1F_1^2F_1^3),$
$(F_1^1F_0^2F_0^3), (F_1^1F_0^2F_1^3), (F_1^1F_1^2F_0^3), (F_1^1F_1^2F_1^3).$

Let $\hat{\mu}$, $\hat{\alpha}_\phi^i$, $\hat{\alpha}_{\phi,\psi}^{i_1,i_2}$ be a estimator of μ, α_ϕ^i, $\alpha_{\phi,\psi}^{i_1,i_2}$. When we experiment with all level combinations, we calculate,

$$\hat{\mu} = \frac{1}{8} \sum_{(\nu_1,\nu_2,\nu_3)\in\{0,1\}^3} y_{\nu_1,\nu_2,\nu_3},$$

$$\hat{\alpha}_\phi^i = \frac{1}{4} \sum_{(\nu_1,\nu_2,\nu_3)\in\{0,1\}^3, \nu_i=\phi} y_{\nu_1,\nu_2,\nu_3} - \hat{\mu},$$

$$\hat{\alpha}_{\phi,\psi}^{i_1,i_2} = \frac{1}{2} \sum_{\substack{(\nu_1,\nu_2,\nu_3)\in\{0,1\}^3, \\ \nu_{i_1}=\phi,\nu_{i_2}=\psi}} y_{\nu_1,\nu_2,\nu_3} - \hat{\mu} - \hat{\alpha}_\phi^{i_1} - \hat{\alpha}_\psi^{i_2}.$$

Then, for example, $\hat{\alpha}_0^1 = \frac{1}{4}(y_{0,0,0} + y_{0,0,1} + y_{0,1,0} + y_{0,1,1}) - \hat{\mu}$ is as follows:

$$\begin{aligned}
y_{0,0,0} &= \mu + \alpha_0^1 + \alpha_0^2 + \alpha_0^3 + \alpha_{0,0}^{1,2} + \alpha_{0,0}^{1,3} + \alpha_{0,0}^{2,3} + e_{0,0,0},\\
y_{0,0,1} &= \mu + \alpha_0^1 + \alpha_0^2 + \alpha_1^3 + \alpha_{0,0}^{1,2} + \alpha_{0,1}^{1,3} + \alpha_{0,1}^{2,3} + e_{0,0,1},\\
y_{0,1,0} &= \mu + \alpha_0^1 + \alpha_1^2 + \alpha_0^3 + \alpha_{0,1}^{1,2} + \alpha_{0,0}^{1,3} + \alpha_{1,0}^{2,3} + e_{0,1,0},\\
y_{0,1,1} &= \mu + \alpha_0^1 + \alpha_1^2 + \alpha_1^3 + \alpha_{0,1}^{1,2} + \alpha_{0,1}^{1,3} + \alpha_{1,1}^{2,3} + e_{0,1,1},\\
\hline
\hat{\alpha}_0^1 &= (\mu - \hat{\mu}) + \alpha_0^1 + \bar{e}_0^1,
\end{aligned}$$

where $\bar{e}_0^1 = \frac{1}{4}(e_{0,0,0} + e_{0,0,1} + e_{0,1,0} + e_{0,1,1})$. This is because we assumed Eq. (1), (2), (3).

Table 1 An OA with 2 levels and strength 2:$OA(4,3,2,2)$.

0	0	0
0	1	1
1	0	1
1	1	0

2.2 Orthogonal Arrays

Definition 1: [4, Definition 1.1] An $M \times n$ array A with entries from $GF(s)$ is said to be an *Orthogonal Array with s levels and strength t* if every $M \times t$ subarray of A contains each t-tuple based on $GF(s)$ exactly same times as row. We will denote such an array by $OA(M,n,s,t)$.

Example 2: The array in Table 1 is an OA with strength 2, denoted by $OA(4,3,2,2)$.

Let $F_1, F_2, \ldots, F_n$ denote the n factors to be included in the experiment. We assume that each factor has s levels, so we can describe the set of levels as $GF(s)$, where let s be a prime power. If we can assume that there is no interaction effect, i.e.

$$y_{\nu_1,\nu_2,\ldots,\nu_n} = \mu + \alpha_{\nu_1}^1 + \ldots + \alpha_{\nu_n}^n + e_{\nu_1,\nu_2,\ldots,\nu_n},$$

we can reduce the number of experiments by using an OA with strength 2, $OA(M,n,s,2)$. When we use an OA to experimental design, each column corresponds to the factor in the experiment, and each row corresponds to the level combination of the factors with which we experiment.

Example 3: (Continued from Example 1) If we can assume that there is no interaction effect, i.e.

$$y_{\nu_1,\nu_2,\nu_3} = \mu + \alpha_{\nu 1}^1 + \alpha_{\nu_2}^2 + \alpha_{\nu_3}^3 + e_{\nu_1,\nu_2,\nu_3},$$

we can reduce the number of experiments by using the $OA(4,3,2,2)$ in Table 1. We experiment with level combinations as follows:

$$(F_0^1F_0^2F_0^3), (F_0^1F_1^2F_1^3), (F_1^1F_0^2F_1^3), (F_1^1F_1^2F_0^3).$$

Then, we calculate

$$\hat{\mu} = \frac{1}{|\bar{A}|} \sum_{(\nu_1,\nu_2,\nu_3)\in\bar{A}} y_{\nu_1,\nu_2,\nu_3},$$

$$\hat{\alpha}_\phi^i = \frac{1}{|\bar{A}_\phi^i|} \sum_{(\nu_1,\nu_2,\nu_3)\in\bar{A}_\phi^i} y_{\nu_1,\nu_2,\nu_3} - \hat{\mu},$$

where $\bar{A}$ is the set of the rows of $OA(4,3,2,2)$ in Table 1 and $\bar{A}_\phi^i = \{(\nu_1,\nu_2,\nu_3) | (\nu_1,\nu_2,\nu_3) \in \bar{A}, \nu_i = \phi\}$. Then, for example, $\hat{\alpha}_0^1 = \frac{1}{2}(y_{000} + y_{011}) - \hat{\mu}$ is as follows:

$$\begin{aligned}
y_{0,0,0} &= \mu + \alpha_0^1 + \alpha_0^2 + \alpha_0^3 + e_{0,0,0},\\
y_{0,1,1} &= \mu + \alpha_0^1 + \alpha_1^2 + \alpha_1^3 + e_{0,1,1},\\
\hline
\hat{\alpha}_0^1 &= (\mu - \hat{\mu}) + \alpha_0^1 + \bar{e}_0^1,
\end{aligned}$$

where, $\bar{e}_0^1 = \frac{1}{2}(e_{0,0,0} + e_{0,1,1})$. This is because we assumed Eq. (1).

We consider some interaction effects of two factors. Let $I \subset GF(s)^2$ be the set whose element is a pair of indices of two factors in which there may be interaction effect. When we can assume that

$$y_{\nu_1,\nu_2,\ldots,\nu_n} = \mu + \alpha^1_{\nu_1} + \alpha^2_{\nu_2} + \ldots + \alpha^n_{\nu_n} + \sum_{(i_1,i_2)\in I} \alpha^{i_1,i_2}_{\nu_1,\nu_2} + \ldots + e_{\nu_1,\nu_2,\ldots,\nu_n},$$

we need an $M \times n$ array A which satisfies the following three conditions;

- A has strength 2.
- A partially has strength 3, that is, for any i_1, i_2 ($(i_1, i_2) \in I$), every $M \times 3$ subarray, which contains two columns that correspond to F_{i_1} and F_{i_2}, contains each 3-tuple based on $GF(s)$ exactly same times as row.
- A partially has strength 4, that is, for any i_1, i_2, i_3, i_4 ($(i_1, i_2), (i_3, i_4) \in I$), $M \times 4$ subarray, which contains four columns that correspond to F_{i_1}, F_{i_2}, F_{i_3}, and F_{i_4}, contains each 4-tuple based on $GF(s)$ exactly same times as row.

In the special case, if there are all interaction effects of two factors, we need an OA with strength 4. Generally, if there are all interaction effects of k factors, we need an OA with strength $2k$.

3. Orthogonal Arrays and Error-Correcting Codes

3.1 Properties of Orthogonal Arrays

In the following, unless mentioned explicitly, we will consider the case that $s = 2$ for simplicity. An $OA(M, n, 2, t)$ is said to be *linear* if the rows of $OA(M, n, 2, t)$ form a linear vector space. If an $OA(M, n, 2, t)$ is linear, $OA(M, n, 2, t)$ has a basis for the linear vector space. This basis is given in the form of $(\log_2 M) \times n$ matrix called *generator matrix*.

Lemma 1: [4, Theorem 3.27 and 3.29] Let A be an $M \times n$ linear array with 0,1 entries, and G be a generator matrix of A. A is an $OA(M, n, 2, t)$ if and only if any t columns of G are linearly independent over $\{0, 1\}$.

Lemma 2: [4, Theorem 3.30] An $M \times n$ array A with 0, 1 entries is an $OA(M, n, 2, t)$ if and only if

$$\sum_{v=row\,of\,A} (-1)^{u \cdot v} = 0,$$

for all 0, 1 vectors u containing w 1's, for all w in the range $1 \le w \le t$, where the sum is over all rows v of A.

3.2 Linear Orthogonal Arrays and Linear Codes

Let $w(u)$ be the Hamming weight of a vector $u = (u_1, u_2, \ldots, u_n)$. An *error-correcting code* or simply *code* is any collection C of vectors in $GF(s)^n$. The vectors in C are called *codewords*. Let $dist(u, v)$ be the Hamming distance between u and v. We define the *minimal distance* d of a code C to be the minimal distance between distinct codewords:

$$d = \min_{u,v\in C, u\neq v} dist(u, v).$$

If C contains M codewords, then we say that it is a code of length n, size M and minimal distance d over $GF(s)$ or simply $(n, M, d)_s$ code. We consider the case of $s = 2$ as well as OAs.

C is said to be linear if C is a linear vector subspace. If C is linear, C has the dual code $C^\perp$. Let $d^\perp$ be the minimal distance of $C^\perp$. Then $d^\perp$ is said to be the *dual distance* of C.

Theorem 1: [4, Theorem 4.6] If C is a $(n, M, d)_2$ linear code over $\{0, 1\}$ with dual distance $d^\perp$, then the codewords of C form the rows of an $OA(M, n, 2, d^\perp - 1)$ with entries from $\{0, 1\}$. Conversely, the rows of a linear $OA(M, n, 2, t)$ over $\{0, 1\}$ form a $(n, M, d)_2$ linear code over $\{0, 1\}$ with dual distance $d^\perp \ge t + 1$. If the OA has strength t but not $t + 1$, $d^\perp = t + 1$.

Example 4: Let $C = \{000, 011, 101, 110\}$. This is a $(3, 4, 2)_2$ code. Then $C^\perp = \{000, 111\}$, so the dual distance of C is 3. Therefore, the OA corresponding to the code C, that is in Table 1, is an $OA(4, 3, 2, 2)$.

3.3 Nonlinear Orthogonal Arrays and Nonlinear Codes

We will describe binary vectors of length n by polynomials in $z_1, z_2, \ldots, z_n$. For example, $100\ldots0$ will be represented by z_1, $1010\ldots0$ by $z_1 z_3$ and so on. In general $v = v_1 v_2 \ldots v_n$ is represented by $z_1^{v_1} z_2^{v_2} \ldots z_n^{v_n}$, which we abbreviate z^v. We make the convention that $z_i^2 = 1$ for all i. This makes the set of all z^v into a multiplicative group denoted by G. Thus $\{0, 1\}^n$ and G are isomorphic groups, with addition in $\{0, 1\}^n$

$$\begin{aligned} v + w &= (v_1, v_2, \ldots, v_n) + (w_1, w_2, \ldots, w_n) \\ &= (v_1 + w_1, v_2 + w_2, \ldots, v_n + w_n), \end{aligned}$$

corresponding to multiplication in G:

$$z^v z^w = z_1^{v_1} z_2^{v_2} \cdots z_n^{v_n} \cdot z_1^{w_1} z_2^{w_2} \cdots z_n^{w_n} = z^{v+w}.$$

Definition 2: [5, p.133] The *group algebra* QG of G over the rational numbers Q consists of all formal sums

$$\sum_{v\in\{0,1\}^n} a_v z^v, a_v \in Q, z^v \in G.$$

Addition and multiplication of elements of QG are defined in the natural way by

$$\sum_{v\in\{0,1\}^n} a_v z^v + \sum_{v\in\{0,1\}^n} b_v z^v = \sum_{v\in\{0,1\}^n} (a_v + b_v) z^v,$$

$$r \sum_{v\in\{0,1\}^n} a_v z^v = \sum_{v\in\{0,1\}^n} r a_v z^v, r \in Q,$$

and

$$\sum_{v\in\{0,1\}^n} a_v z^v \cdot \sum_{v\in\{0,1\}^n} b_w z^v = \sum_{v,w\in\{0,1\}^n} a_v b_w z^{v+w}.$$

To each $u \in \{0,1\}^n$, we associate the mapping χ_u from G to the rational numbers given by

$$\chi_u(z^v) = (-1)^{u \cdot v}.$$

χ_u is called a *character* of G. χ_u is extended to act on QG by linearity.

$$\chi_u\left(\sum_{v\in\{0,1\}^n} a_v z^v\right) = \sum_{v\in\{0,1\}^n} a_v \chi_u(z^v) = \sum_{v\in\{0,1\}^n} (-1)^{u\cdot v} a_v.$$

Now, let $\gamma = \sum_{v\in\{0,1\}^n} c_v z^v$ be an arbitrary element of the group algebra QG with the property that

$$M = \sum_{v\in\{0,1\}^n} c_v \neq 0.$$

We call the $(n+1)$-tuple $\{A_0, A_1, \ldots, A_n\}$, where

$$A_i = \sum_{w(v)=i} c_v,$$

the *weight distribution* of γ. Then, the transform of $\gamma^\perp$ is defined as follows.

Definition 3: [5, p.136] The *transform* of γ is the element $\gamma^\perp$ of QG given by

$$\gamma^\perp = \frac{1}{M} \sum_{u\in\{0,1\}^n} \chi_u(\gamma) z^u.$$

Now let C be a linear or nonlinear $(n, M, d)_2$ code. C is described by the element $\gamma = \sum_{v\in C} z^v$ of QG. Let $\delta = \frac{1}{M}\gamma^2$, and $\delta^\perp$ is the transform of δ. The weight distribution of $\delta^\perp$ is $\{B_0^\perp, B_1^\perp, \ldots, B_n^\perp\}$, where

$$B_i^\perp = \frac{1}{M} \sum_{w(u)=i} \chi_u(\delta).$$

Then, the dual distance $d^\perp$ of C is defined as follows.

Definition 4: [5, p.139] The *dual distance* $d^\perp$ of a code C is defined by $B_i^\perp = 0$ *for* $1 \le i \le d^\perp - 1$, $B_{d^\perp}^\perp \neq 0$. If C is linear, $d^\perp$ is the minimum distance of $C^\perp$.

Theorem 2: [4, Theorem 4.9][5, p.139] If C is a $(n, M, d)_2$ code over $\{0,1\}$ with dual distance $d^\perp$, then the codewords of C form the rows of an $OA(M, n, 2, d^\perp - 1)$ with entries from $\{0,1\}$. Conversely, the code corresponding to an $OA(M, n, 2, t)$ is a $(n, M, d)_2$ code with dual distance $d^\perp \ge t + 1$. If the OA has strength t but not $t+1$, $d^\perp$ is precisely $t+1$.

4. Orthogonal Arrays with Unequal Strength and Unequal Error-Correcting Codes

4.1 Orthogonal Arrays with Unequal Strength and Its Properties

Definition 5: An $M \times n$ array A with entries from $\{0,1\}$ is said to be an OA with 2 levels and strength $t = (t_1, t_2, \ldots, t_n)$ if every $M \times t_i$ subarray of A, which contains i-th column of A, contains each t_i-tuple based on $\{0,1\}$ exactly same times as row. We will denote such an array by $OA(M, n, 2, t)$. Then we will call an $OA(M, n, 2, t)$ *OA with unequal strength* if the components of t are not mutually equal.

Generally, when an $OA(M, n, 2, (t_1, t_2, \ldots, t_n))$ is applied to experimental designs, we can estimate the following interaction effects; if we can assume that for any $i(\in \{1, 2, \ldots, n\})$, there exists integer s_i $(i = 1, 2, \ldots, n)$, $i_1, i_2, \ldots, i_{t_i - s_i - 2} (\in \{1, 2, \ldots, n\} \setminus \{i\})$, there are no interaction effects of at least $(s_i + 1)$ factors that don't contain i-th factor and i'-th factor such that $t_{i'} - s_{i'} \ge t_i - s_i$, then $OA(M, n, 2, (t_1, t_2, \ldots, t_n))$ can estimate all $(t_i - s_i)$-factor interaction effects that contain $i, i_1, \ldots, i_{t_i - s_i - 2}$, and all $(t_i - s_i - j)$-factor interaction effects that contain $i, i_1, \ldots, i_{t_i - s_i - 2 - j}$ $(j = 1, 2, \ldots, t_i - s_i - 2)$. For example, let F_1, F_2, F_3, F_4, and F_5 be the factors to be included in the experiment, and we use an $OA(M, 5, 2, (t_1, t_2, \ldots, t_5))$ to the experiment, where F_i corresponds to i-th factor of the OA. We consider the case that we use an $OA(M, 5, 2, (4, 2, 2, 2, 2))$. If $s_i = 1$ $(i = 1, 2, \ldots, 5)$, that is, we can assume that there are no $(s_i + 1)$-factor interaction effects that don't contain i-th factor and i'-th factor such that $t_{i'} - s_{i'} \ge t_i - s_i$, then we can estimate interaction effects $F_1F_2F_3$, $F_1F_2F_4$, $F_1F_2F_5$, F_1F_2, F_1F_3, F_1F_4, and F_1F_5, where $i_1 = 2$ for 1st column. Of course, interaction effects that an $OA(M, 5, 2, (4, 2, 2, 2, 2))$ can estimate don't include $(s_i + 1)$-factor interaction effect that don't contain F_i, $F_{i'}$ such that $t_{i'} - s_{i'} \ge t_i - s_i$, for example when $i = 1$, there are no two-factor interaction effects that don't contain F_1. Besides, we consider an $OA(M, 5, 2, (3, 2, 2, 2, 2))$. If $s_i = 1$ $(i = 1, 2, \ldots, 5)$, then we can estimate F_1F_2, F_1F_3, F_1F_4, and F_1F_5. Furthermore, $OA(M, n, 2, (t_1, t_2, \ldots, t_n))$ can estimate all $[\frac{t_i}{2}]$-factor interaction effects that contain i-th factor, where this is the case that $s_i = \frac{t_i}{2}$.

Next, we consider the following three classes of OAs;

- OAs
- OAs with equal strength
- OAs with unequal strength

"OAs" is the class that includes all OAs. So this class includes OAs that can estimate partial interaction effect. And "OAs with equal strength" consists of OAs defined by Definition 1, and "OAs with unequal strength" consists of OAs defined by Definition 5. We compare these three classes. For example, we also take five factors F_1, F_2, F_3, F_4, and F_5, and there exists one two-factor interaction effect F_1F_2. Then if we use an OA that is in "OAs with equal strength" to the experiment, we should use an $OA(M, 5, 2, 3)$. And if we use an OA that is in "OAs with unequal strength" to the experiment, we should use an $OA(M, 5, 2, (3, 2, 2, 2, 2))$. Then an $OA(M, 5, 2, (3, 2, 2, 2, 2))$ may be able to reduce the number of experiments than an $OA(M, 5, 2, 3)$, because an $OA(M, 5, 2, (3, 2, 2, 2, 2))$ is more appropriate to the given

interaction effect. But, in "OAs," there may be an OA that is more appropriate to the given interaction effect, because $OA(M, 5, 2, (3, 2, 2, 2, 2))$ can estimate other interaction effects F_1F_3, F_1F_4 and F_1F_5 that we need not estimate. From the above description, we can say that "OAs with unequal strength" are closer to "OAs" than "OAs with equal strength," so "OAs with unequal strength" is a more practical class. But we cannot say that "OAs with unequal strength" are practical enough, when we compare "OAs with unequal strength" with "OAs."

Next, we show two properties of OAs with unequal strength. The first property (Lemma 3) is extended from Lemma 1, the second (Lemma 4) is extended from Lemma 2. Each lemma is proved in the same way as Lemma 1 and 2.

Lemma 3: Let A be an $M \times n$ linear array with 0, 1 entries, and G be a generator matrix of A. A is an $OA(M, n, 2, (t_1, t_2, \ldots, t_n))$ if and only if any t_i columns of G, which contain the i-th column of G, are linearly independent over $\{0, 1\}$.

Lemma 4: An $M \times n$ array A with 0, 1 entries is an $OA(M, n, 2, (t_1, t_2, \ldots, t_n))$ if and only if

$$\sum_{v=row\,of\,A} (-1)^{u \cdot v} = 0,$$

for all 0, 1 vectors $u = (u_1, u_2, \ldots, u_n)$ such that $u_i \neq 0$ and $w(u) = w$ for all w in the range $1 \leq w \leq t_i$, where the sum is over all rows v of A.

4.2 Linear Orthogonal Arrays with Unequal Strength and Linear Unequal Codes

The separation $(d_1, d_2, \ldots, d_n)$ of linear code C is defined by

$$d_i = \min\{dist(u, v) \mid u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2, \ldots, v_n), u, v \in C, u_i \neq v_i\}, \quad for \quad i = 1, 2, \ldots, n.$$

If a linear code C has the separation whose components are not mutually equal, the code C is called an *unequal error-correcting code*. Let $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$ be the separation of $C^\perp$ which is the dual code of C. Then we will call $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$ the dual separation of C.

The next Theorem 3 shows the relation between linear OAs with unequal strength and linear unequal codes.

Theorem 3: If C is a $(n, M, d)_2$ linear code over $\{0, 1\}$ with dual separation $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$, then the codewords of C form the row of an $OA(M, n, 2, (d_1^\perp - 1, d_2^\perp - 1, \ldots, d_n^\perp - 1))$ with entries from $\{0,1\}$. Conversely, the rows of a linear $OA(M, n, 2, (t_1, t_2, \ldots, t_n))$ over $\{0, 1\}$ form a $(n, M, d)_2$ linear code over $\{0, 1\}$ with dual separation $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$, where $d_i^\perp \geq t + 1, i = 1, 2, \ldots, n$. If the OA has strength t_i but not $t_i + 1$, $d_i^\perp = t_i + 1$ $(i = 1, 2, \ldots, n)$.

Theorem 3 is extended from Theorem 1. Theorem 3 is proved by Lemma 3 and the definition of linear unequal codes. Detailed proof is omitted here.

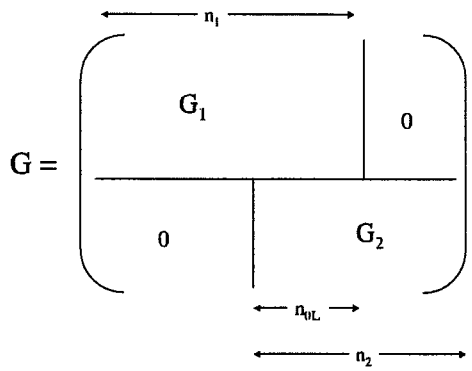


Fig. 1 The construction method of linear OA.

Besides, an unequal error-correcting code with separation $(d_1, d_2, \ldots, d_n)$ can correct all $[\frac{d_i - 1}{2}]$ errors that contain an error in the i-th digit. This corresponds to the fact that an $OA(M, n, 2, (t_1, t_2, \ldots, t_n))$ can estimate all $[\frac{t_i}{2}]$-factor interaction effects that contain the i-th factor.

We show two construction methods of linear OAs with unequal strength. The first method (Construction Method 1) is derived from unequal error-correcting code [6], the second (Construction Method 2) is derived from [2].

Construction Method 1: Let there be two generator matrices of OAs; G_1 is the generator matrix for a linear $OA(M_1, n_1, 2, r_1)$, and G_2 is the one for a linear $OA(M_2, n_2, 2, r_2)$, where $r_2 \leq r_1$. Let G_1 and G_2 be joined as submatrices of G where G_1 and G_2 overlap, as shown in Fig. 1. The OA with generator matrix G is a $(M_1M_2) \times (n_1 + n_2 - n_{0L})$ array. Let $n_{0L} \leq r_2/2$.

Theorem 4: An OA by Construction Method 1 is an $OA(M_1M_2, n_1 + n_2 - n_{0L}, 2, (t_1, t_2, \ldots, t_n))$, where

$$t_i \geq r_1^\dagger \quad (i = 1, 2, \ldots, n_1 - n_{0L}),$$
$$t_i \geq r_2 \quad (i = n_1 + 1, n_1 + 2, \ldots, n_1 + n_2 + n_{0L}),$$
$$t_i \geq r_1 + r_2 - n_{0L} \quad (i = n_1 - n_{0L} + 1, \ldots, n_1).$$

Construction Method 2: Let α denote a primitive element of the field $GF(2^{2m})$. Then $\beta = \alpha^{2^m+1}$ is a primitive element of the field $GF(2^m)$ which is a subfield of the field $GF(2^{2m})$. Consider an OA with 2 levels which have the generator matrix

$$G = \begin{bmatrix} 1 & \alpha & \cdots & \alpha^{2^m} & \alpha^{2^m+1} & \alpha^{2^m+2} & \cdots & \alpha^{2^{2m}-2} \\ 1 & 0 & \cdots & 0 & \beta^3 & 0 & \cdots & 0 \end{bmatrix}. \tag{4}$$

The OA with generator matrix G is a $2^{3m} \times (2^{2m} - 1)$ array, and its strength is $t \geq 2$.

†Because any OA with strength r_1' $(r_1' > r_1)$ is also an OA with strength r_1. So, strength of OA from G_1 is greater than or equal to r_1 in Construction Method 1. Hence, t_i is greater than or equal to r_1.

Theorem 5: Let m be an odd integer. Then the OA with the generator matrix in (4) is an $OA(2^{3m}, 2^{2m} - 1, 2, (t_1, t_2, \ldots, t_n))$, where

$$t_i = 4 \quad (i = 1 + j(2^m + 1), j = 0, 1, \ldots, 2^m - 2),$$
$$t_i \geq 2 \quad (otherwise).$$

The statement of Theorem 5 allows for some modifications and generalizations (cf. [2, Theorem 2]).

4.3 Nonlinear Orthogonal Arrays with Unequal Strength and Codes

Let $\gamma = \sum_{v \in \{0,1\}^n} c_v z^v$ be an arbitrary element of the group algebra QG. Then we will call the $(n+1)$-tuple $\{A_{i,0}, A_{i,1}, \ldots, A_{i,n}\}$, where

$$A_{i,j} = \sum_{v_i \neq 0, w(v)=j} c_v$$

the weight distribution of γ with respect to i.

Now let C be a linear or nonlinear $(n, M, d)_2$ code. C is described by the element $\gamma = \sum_{v \in C} z^v$, of QG. Let $\delta = \frac{1}{M}\gamma^2$ and $\delta^\perp$ be the transform of δ. For $i = 1, 2, \ldots, n$, the weight distribution of $\delta^\perp$ with respect to i is $\{B^\perp_{i,0}, B^\perp_{i,1}, \ldots, B^\perp_{i,n}\}$, where

$$B^\perp_{i,j} = \frac{1}{M} \sum_{u_i \neq 0, w(u)=j} \chi_u(\delta).$$

Then, the dual separation of C is defined as follows.

Definition 6: The *dual separation* $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$ of a code C is defined by, for $i = 1, 2, \ldots, n$,

$$B^\perp_{i,j} = 0, \ for\ 1 \leq j \leq d_i^\perp - 1,$$
$$B^\perp_{i,d_i^\perp} \neq 0.$$

If C is linear, $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$ is the separation of $C^\perp$.

The next Theorem 6 shows the relation between nonlinear OAs with unequal strength and codes. This theorem is extended from Theorem 2.

Theorem 6: If C is a $(n, M, d)_2$ code over $\{0, 1\}$ with dual separation $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$, then the codewords of C form the rows of an $OA(M, n, 2, (d_1^\perp - 1, d_2^\perp - 1, \ldots, d_n^\perp - 1))$ with entries from $\{0, 1\}$. Conversely, the code corresponding to an $OA(M, n, 2, (t_1, t_2, \ldots, t_n))$ is a $(n, M, d)_2$ code with dual distance $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$, where $d_i^\perp \geq t_i + 1$, $i = 1, 2, \ldots, n$. If the OA has strength t_i but not $t_i + 1$, $d_i^\perp$ is precisely $t_i + 1$ $(i = 1, 2, \ldots, n)$.

Proof: Let $\gamma = \sum_{v \in C} z^v$ and $\delta = \frac{1}{M}\gamma^2$. Let $\{B^\perp_{i,0}, B^\perp_{i,1}, \ldots, B^\perp_{i,n}\}$, $i = 1, 2, \ldots, n$, be the weight distribution of $\delta^\perp$ with respect to i. Since the dual separation of C is $(d_1^\perp, d_2^\perp, \ldots d_n^\perp)$,

$$B^\perp_{i,j} = \frac{1}{M} \sum_{u_i \neq 0, w(u)=j} \chi_u(\delta) = 0, \tag{5}$$

for $1 \leq i \leq n$, $1 \leq j \leq d_i^\perp - 1$. Also,

$$\chi_u(\delta) = \chi_u\left(\frac{1}{M}\gamma^2\right) = \frac{1}{M}\chi_u(\gamma)^2 \geq 0. \tag{6}$$

By Eq. (5), (6),

$$\chi_u(\delta) = 0,$$

for any $u = (u_1, u_2, \ldots, u_n)$ such that $u_i = 0$, $w(u) = j$, for $1 \leq i \leq n$, $1 \leq j \leq d_i^\perp - 1$. Therefore, from the definition of character and Lemma 4, the OA corresponding to the code C is $OA(M, n, 2, (d_1^\perp, d_2^\perp, \ldots, d_n^\perp))$.

Conversely, if the OA has strength $t = (t_1, t_2, \ldots, t_n)$, from Lemma 4 we have

$$B_{i,1} = B_{i,2} = \ldots = B_{i,t_i} = 0,$$

for $1 \leq i \leq n$, and so $d_i^\perp \geq t_i + 1$. □

Now we show Construction Method 3 of nonlinear OAs with unequal strength which is extended from the Construction Method 1.

Construction Method 3: Let there be two OAs; $[C_1]$ is $OA(M_1, n_1, 2, r_1)$ and $[C_2]$ is $OA(M_2, n_2, 2, r_2)$, where $r_2 \leq r_1$. Let C_1 be the set of the rows of $[C_1]$ and C_2 be the set of the rows of $[C_2]$. Let

$$C = \{(c_{1,1}, \ldots, c_{1,n_1-n_{0L}}, c_{1,n_1-n_{0L}+1} + c_{2,1}, \ldots, c_{1,n_1} + c_{2,n_{0L}}, c_{2,n_{0L}+1}, \ldots, c_{2,n_2}) \mid for\ \forall(c_{1,1}, c_{1,2}, \ldots, c_{1,n_1}) \in C_1, \forall(c_{2,1}, c_{2,2}, \ldots, c_{2,n_2}) \in C_2\}.$$

The OA whose rows are formed by the vectors in C is $(M_1 M_2) \times (n_1 + n_2 - n_{0L})$ array. Let $n_{0L} \leq r_2/2$.

Theorem 7: The OA by Construction Method 3 is $OA(M_1 M_2, n_1 + n_2 - n_{0L}, 2, (t_1, t_2, \ldots, t_n))$, where

$$t_i \geq r_1 \quad (i = 1, 2, \ldots, n_1 - n_{0L}), \tag{7}$$
$$t_i \geq r_2 \quad (i = n_1 + 1, n_1 + 2, \ldots, n_1 + n_2 - n_{0L}), \tag{8}$$
$$t_i \geq r_1 + r_2 - n_{0L} \quad (i = n_1 - n_{0L} + 1, \ldots, n_1). \tag{9}$$

Proof: Let $M = M_1 M_2$, $n = n_1 + n_2 - n_{0L}$,

$$\gamma = \sum_{v \in C} z^v$$

and,

$$\delta = \frac{1}{M}\gamma^2.$$

Then, the transform of δ is

$$\delta^\perp = \frac{1}{M} \sum_{u \in \{0,1\}^n} \chi_u(\delta) z^u.$$

And, let

$$C_1' = \{(a_{1,1}, a_{1,2}, \ldots, a_{1,n_1}, 0, \ldots, 0) \in \{0, 1\}^n \mid \forall(a_{1,1}, a_{1,2}, \ldots, a_{1,n_1}) \in C_1\},$$

$$C_2' = \{(0,\ldots,0,a_{2,1},a_{2,2},\ldots,a_{2,n_2}) \in \{0,1\}^n \mid \forall(a_{2,1},a_{2,2},\ldots,a_{2,n_2}) \in C_2\},$$

$$\gamma_1' = \sum_{v\in C_1'} z^v, \gamma_2' = \sum_{v\in C_2'} z^v,$$

and,

$$\delta_1' = \frac{1}{M_1}\gamma_1'^2, \delta_2' = \frac{1}{M_2}\gamma_2'^2.$$

Then, we can describe

$$\gamma = \gamma_1' \times \gamma_2',$$

$$\delta = \frac{1}{M}\gamma^2 = \frac{1}{M_1M_2}(\gamma_1' \times \gamma_2')^2 = \delta_1' \times \delta_2'.$$

Moreover, the weight distribution of $\delta^\perp$ with respect to i, $\{B_{i,1}^\perp, B_{i,2}^\perp, \ldots, B_{i,n}^\perp\}$ $(i = 1,2,\ldots,n_1+n_2-n_{0L})$ is as follows;

$$\begin{aligned} B_{i,j}^\perp &= \frac{1}{M}\sum_{u_i\neq 0, w(\boldsymbol{u})=j} \chi_{\boldsymbol{u}}(\delta) \\ &= \frac{1}{M_1M_2}\sum_{u_i\neq 0, w(\boldsymbol{u})=j} \chi_{\boldsymbol{u}}(\delta_1' \times \delta_2') \\ &= \frac{1}{M_1M_2}\sum_{u_i\neq 0, w(\boldsymbol{u})=j} \chi_{\boldsymbol{u}}(\delta_1')\chi_{\boldsymbol{u}}(\delta_2') \\ &= \sum_{u_i\neq 0, w(\boldsymbol{u})=j} \left(\frac{1}{M_1}\chi_{\boldsymbol{u}}(\delta_1')\right)\left(\frac{1}{M_2}\chi_{\boldsymbol{u}}(\delta_2')\right). \end{aligned} \quad (10)$$

1. For the case of $1 \le i \le n_1 - n_{0L}$
   Because $[C_1]$ has strength t_1, from Theorem 2,

$$\frac{1}{M_1}\sum_{\boldsymbol{u}\in\{0,1\}^{n_1}, w(\boldsymbol{u})=j} \chi_{\boldsymbol{u}}(\delta_1) = 0, \quad (11)$$

   for $1 \le j \le t_1$, where $\gamma_1 = \sum_{v\in C_1} z^v, \delta_1 = \frac{1}{M_1}\gamma_1^2$. Also,

$$\chi_{\boldsymbol{u}}(\delta) = \chi_{\boldsymbol{u}}\left(\frac{1}{M}\gamma^2\right) = \frac{1}{M}\chi_{\boldsymbol{u}}(\gamma)^2 \ge 0. \quad (12)$$

   By Eq. (11), (12), for any $\boldsymbol{u}$ such that $\boldsymbol{u} \in \{0,1\}^{n_1}$ and $1 \le w(\boldsymbol{u}) \le r_1$.

$$\chi_{\boldsymbol{u}}(\delta_1) = 0.$$

   Therefore, for any $\boldsymbol{u}'$ such that $\boldsymbol{u}' = (u_1', u_2', \ldots, u_n') \in \{0,1\}^n$, $u_i' \neq 0$ and $1 \le w(\boldsymbol{u}') \le r_1$,

$$\chi_{\boldsymbol{u}'}(\delta_1') = 0. \quad (13)$$

   By Eq. (10), (13), for $1 \le i \le n_1 + n_2 - n_{0L}$ and $1 \le j \le r_1$,

$$B_{i,j}^\perp = 0.$$

2. For the case of $n_1 + 1 \le i \le n_1 + n_2 - n_{0L}$
   In the same way as 1. , for any $\boldsymbol{u}'$ such that $\boldsymbol{u}' = (u_1', u_2', \ldots, u_n') \in \{0,1\}^n$, $u_i' \neq 0$ and $1 \le w(\boldsymbol{u}') \le r_2$,

$$\chi_{\boldsymbol{u}'}(\delta_2') = 0. \quad (14)$$

   By Eq. (10), (14), for $n_1 + 1 \le i \le n_1 + n_2 - n_{0L}$ and $1 \le j \le r_2$,

$$B_{i,j}^\perp = 0.$$

3. For the case of $n_1 - n_{0L} + 1 \le i \le n_1$
   If $u_i' \neq 0$ and $1 \le w(\boldsymbol{u}') \le r_1 + r_2 - n_{0L}$,

$$\chi_{\boldsymbol{u}'}(\delta_1') = 0 \quad or \quad \chi_{\boldsymbol{u}'}(\delta_2') = 0. \quad (15)$$

   By Eq. (10), (15), for $n_1 - n_{0L} + 1 \le i \le n_1$ and $1 \le j \le r_1 + r_2 - n_{0L}$,

$$B_{i,j}^\perp = 0.$$

Hence by Theorem 6, Eq. (7),(8),(9) hold. □

5. Examples of Orthogonal Arrays with Unequal Strength by the Proposed Construction Methods

5.1 Orthogonal Arrays with Unequal Strength by Construction Method 1 and 3

In this section, we show some examples of OAs with unequal strength by Construction Method 1 and 3. And we compare them with optimal OAs with equal strength [4, Table 12.1].

Firstly, we compare the following OAs;

- An $OA(2^{14}, 16, 2, 8)$. This is an optimal $M\times 16$ OA with 2 levels and equal strength 8, that is in [4, Table 12.1].
- An $2^{13} \times 16$ OA with 2 levels and partially strength 8 by Construction Method 1: G_1 in Construction Method 1 is a generator matrix for an $OA(2^7, 9, 2, 5)$. This is an optimal $M \times 9$ OA with 2 levels and equal strength 5, that is in [4, Table 12.1]. G_2 is a generator matrix for an $OA(2^6, 8, 2, 4)$. This is also an optimal $M \times 8$ OA with 2 levels and equal strength 4. And $n_{0L} = 1$.

Then, the number of rows of the OA with unequal strength by Construction Method 1 is fewer than that of the OA with equal strength. Therefore, the OA with unequal strength can reduce more numbers of experiments than the OA with equal strength under partial interaction effects.

Next, we compare the following OAs to show differences between linear and nonlinear OAs with unequal strength.

- (Equal) optimal $M \times n$ OAs with 2 levels and equal strength 4 that is in [4, Table 12.1] $(n = 11, 12, \ldots, 32)$.
- (Method 1) $M \times n$ OAs with 2 levels and partially strength 4 by Construction Method 1 $(n = 11, 12, \ldots, 32)$: G_1 in Construction Method 1 is a generator matrix for an optimal $M_1 \times n_1$ linear OA with 2 levels and equal strength 3 that is in [4, Table 12.1] $(n_1 = 9, 10, \ldots, 30)$, G_2 is a generator matrix for a linear $OA(4, 3, 2, 2)$, and $n_{0L} = 1$.
- (Method 3) $M \times n$ OAs with 2 levels and partially strength 4 by Construction Method 3 $(n = 11, 12, \ldots, 32)$: $[C_1]$ in Construction Method 3 is an optimal $M_1 \times n_1$ linear and nonlinear OA with 2 levels and equal strength 3 that is in [4, Table 12.1] $(n_1 = 9, 10, \ldots, 30)$, $[C_2]$ is an $OA(4, 3, 2, 2)$, and $n_{0L} = 1$.

Table 2 The number of rows of OAs.

n	Equal	Method 1	Method 3
11	128	128	96
12	128	128	96
13	128	128	96
14	128	128	96
15	128	128	128
16	256	128	128
17	256	128	128
18	256	128	128
19	256	256	160
20	512	256	160
21	512	256	160
22	512	256	160
23	512	256	192
24	1024	256	192
25	1024	256	192
26	1024	256	192
27	1024	256	224
28	1024	256	224
29	1024	256	224
30	1024	256	224
31	1024	256	256
32	1024	256	256

The number of rows of each OA is shown in Table 2.

We compare OAs with unequal strength by Construction Method 1 with OAs with equal strength. The number of rows of linear OAs with unequal strength is fewer than that of OAs with equal strength at many n's. Therefore, this linear OAs with unequal strength can reduce more number of experiments than OAs with equal strength under partial interaction effects.

We compare linear and nonlinear OAs with unequal strength by Construction Method 3 with linear OAs with unequal strength by Construction Method 1. The number of rows of OAs by Construction Method 3 is fewer than that of OAs by Construction Method 1. This is because Construction Method 3 is extended from Construction Method 1, so OAs by Construction Method 3 include OAs by Construction Method 1.

5.2 Orthogonal Arrays with Unequal Strength by Construction Method 2

In this section, we will show an example of an OA with unequal strength by Construction Method 2. And we will compare it with an OA with equal strength by using BCH codes [4], [5].

We will compare the following OAs;

- The OA with equal strength that has generator matrix

$$G = \begin{bmatrix} 1 & \alpha & \cdots & \alpha^{2^m+1} & \cdots & \alpha^{2^{2m}-2} \\ 1 & \alpha^2 & \cdots & \alpha^{2^{m+1}+2} & \cdots & \alpha^{2^{2m+1}-4} \end{bmatrix}.$$

  This is an $OA(4096, 63, 2, 4)$.
- The OA with unequal strength by Construction Method 2, where let $m = 3$ in Construction Method 2. This is an $OA(512, 63, 2, (t_1, t_2, \ldots, t_{63}))$, where $t_i = 4$ $(i = 1 + 9j, j = 0, 1, \ldots, 6)$, $t_i \geq 2$ (otherwise).

Then, the number of rows of the OA by Construction Method 2 is fewer than that of the OA with equal strength. Therefore, the OA with unequal strength can reduce more number of experiments than the equal OA under partial interaction effects.

6. Conclusion

In this paper, we defined OAs with unequal strength. We showed types of interaction effects that OAs with unequal strength can estimate, and that OAs with unequal strength are closer to practical OAs than OAs with equal strength. And we showed the relations between OAs with unequal strength and unequal error-correcting codes. And we proposed some construction methods of linear and nonlinear OAs with unequal strength from unequal error-correcting codes. Lastly, we showed some examples of OAs with unequal strength constructed by proposed methods.

Acknowledgments

The authors would like to acknowledge all members of Matsushima Lab. and Hirasawa Lab. in Waseda Univ. for their helpful suggestions to this work. This research is partially supported by Category (C) No.15560338 of Grant-in-Aid for Scientific Research, Japan Society for the Promotion of Science.

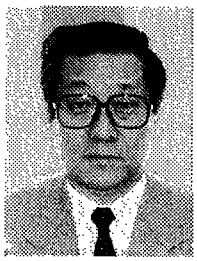
References

[1] G.E.P. Box, W.G. Hunter, and J.S. Hunter, Statistics for Experimenters: An Introduction to Design, Data Analysis, and Model Building, John Wiley & Sons, 1978.
[2] I.M. Boyarinov and G.L. Katsman, "Linear unequal error protection codes," IEEE Trans. Inf. Theory, vol.IT-27, no.2, pp.168–175, March 1981.
[3] R. Fuji-Hara, "On automatical construction for orthogonal designs of experiments," Rep. Stat. Appl. Res., JUES, vol.25, no.1, pp.13–25, 1978.
[4] A.S. Hedayat, N.J.A. Sloane, and J. Stufken, Orthogonal Arrays: Theory and Applications, Springer, New York, 1999.
[5] F.J. MacWilliams and N.J.A. Sloane, The theory of error-correcting codes, North-Holland Publishing, Amsterdam, 1977.
[6] B. Masnick and J. Wolf, "On linear unequal error protection codes," IEEE Trans. Inf. Theory, vol.IT-3, no.4, pp.600–607, Oct. 1967.
[7] T. Saito, T. Yoshida, and T. Matsushima, "A note on the construction of orthogonal designs by using the construction of error correcting codes," Proc. SITA2002, pp.663–666, 2002.
[8] T. Saito, T. Matsushima, and S. Hirasawa, "A note on the construction of orthogonal designs using error correcting codes," Proc. SITA2004, pp.463–466, 2004.
[9] T. Saito, T. Matsushima, and S. Hirasawa, "A note on the construction of nonlinear unequal OAs from error-correcting codes," IEICE Technical Report, IT2005-15, 2005.
[10] K. Suda and H. Miyazaki, "An algorithm which corresponds the multi-factors to orthogonal arrays in the design of orthogonal experiments," J. Japan Industrial Management Association, vol.37, no.6, pp.345–352, 1987.
[11] I. Takahasi, Combinatorial Theory and its Application, Iwanami Syoten, 1979.
[12] Y. Ukita, T. Matsushima, and S. Hirasawa, "A note on learning

Boolean functions by using orthogonal designs," IEICE Trans. Fundamentals, vol.J83-A, no.4, pp.482–490, April 2003.
[13] Y. Washio, Design and Analysis of Experiments, Iwanami Syoten, 1988.

Tomohiko Saito was born in Tokyo, Japan, on Jan. 11, 1978. He received the B.E. degree and M.E. degree in Industrial and Management Systems Engineering from Waseda University, Tokyo, Japan, in 2001 and 2003, respectively. He is currently a doctoral student in Industrial and Management Systems Engineering at the Graduate School of Waseda University. His research interest is coding theory and experimental designs.

Toshiyasu Matsushima was born in Tokyo, Japan, on Nov. 26, 1955. He received the B.E. degree, M.E. degree, and Dr.E. degree in Industrial and Management Systems Engineering from Waseda University, Tokyo, Japan, in 1978, 1980, and 1991, respectively. From 1980 to 1986, he was with Nippon Electric Corporation, Kanagawa, Japan. From 1986 to 1992, he was a lecturer in the Department of Management Information, Yokohama College of Commerce. From 1993, he was an associate professor, and since 1996 he has been a professor in the School of Science and Engineering, Waseda University, Tokyo, Japan. His research interests are information theory and its application, statistics, and artificial intelligence. He is a member of the Society of Information Theory and Its Applications, the Japan Society for Quality Control, the Japan Industrial Management Association, the Japan Society for Artificial Intelligence, and IEEE.



Shigeichi Hirasawa was born in Kobe, Japan, on Oct. 2, 1938. He received the B.S. degree in mathematics and the B.E. degree in electrical communication engineering from Waseda University, Tokyo, Japan, in 1961 and 1963, respectively, and the Dr.E. degree in electrical communication engineering from Osaka University, Osaka, Japan, in 1975. From 1963 to 1981, he was with the Mitsubishi Electric Corporation, Hyogo, Japan. Since 1981, he has been a professor in the School of Science and Engineering, Waseda University, Tokyo, Japan. In 1979, he was a Visiting Scholar in the Computer Science Department at the University of California, Los Angeles (CSD, UCLA), CA. He was a Visiting Researcher at the Hungarian Academy of Science, Hungary, in 1985, and at the University of Trieste, Italy, in 1986. In 2002, he was also a Visiting Faculty member at CSD, UCLA. From 1987 to 1989, he was the Chairman of the Technical Group on Information Theory of IEICE. He received the 1993 Achievement Award and the 1993 Kobayashi-Memorial Achievement Award from IEICE. In 1996, he was the President of the Society of Information Theory and Its Applications (Soc. of ITA). His research interests are information theory and its applications, and information processing systems. He is an IEEE Fellow and a member of Soc. of ITA, the Operations Research Society of Japan, the Information Processing Society of Japan, the Japan Industrial Management Association, and Informs.

論　文

ストリーム暗号への攻撃法の改良に関する一考察
——多次元の相関を利用した攻撃——

細渕　智史[†a)]　　齋藤　友彦[†]　　松嶋　敏泰[†]

A Note on the Improvement of a Fast Correlation Attack on Stream Ciphers

Satoshi HOSOBUCHI[†a)], Tomohiko SAITO[†], and Toshiyasu MATSUSHIMA[†]

あらまし　共通鍵暗号の一種であるストリーム暗号は，鍵を擬似乱数生成器に与えて鍵系列と呼ばれる擬似乱数系列を生成し，これと平文系列との排他的論理和をとることで暗号文系列を生成する方式である．ストリーム暗号に使われる擬似乱数生成器の一種に非線形コンバイナ型乱数生成器があり，これは複数の LFSR と一つの非線形関数で構成される．また，相関攻撃はこのような擬似乱数生成器への攻撃法の一つであり，LFSR の出力系列と鍵系列の相関を用いて LFSR の初期状態を推定する攻撃法である．だが，従来の攻撃法は単体の LFSR を攻撃するものであり，本来複数の LFSR 系列と鍵系列で多次元の相関があるうちの一部分しか推定に使用していない．よって，本論文は Mihaljević らの BP を用いた攻撃法を改良し，多次元の相関を利用した複数の LFSR を同時に攻撃するアルゴリズムを提案する．推定に使用する情報を増やすことより，解読成功確率の向上を見込むことができる．また，このとき推定にかかる計算量は増加してしまうが，BP の並列アルゴリズムによる近似計算を行うことで，計算量の増加を抑えることができる．

キーワード　ストリーム暗号，相関攻撃，非線形コンバイナ型乱数生成器，Belief Propagation

1. まえがき

共通鍵暗号の一種であるストリーム暗号は，鍵を擬似乱数生成器に入力として与えて鍵系列と呼ばれる擬似乱数系列を生成し，これと平文系列との排他的論理和をとることで暗号文系列を生成する暗号方式である．

鍵系列を生成する擬似乱数生成器には様々な種類があり，非線形コンバイナ型乱数生成器もその一つである．これは図 1 のように，鍵を複数の LFSR（線形フィードバックシフトレジスタ）の初期状態として与えて LFSR 系列を生成し，それらを多入力 1 出力の非線形関数で結合して出力するというものである．

ストリーム暗号を攻撃することは，擬似乱数生成器を攻撃することに帰着できる．そして，非線形コンバイナ型乱数生成器への攻撃法の一つに相関攻撃がある [1]．これは，観測された鍵系列と一つの LFSR 系列との相関を用いて LFSR の初期状態（鍵の一部）を推定する攻撃法である．このプロセスを繰り返して LFSR を一つずつ順に攻撃していくことにより，鍵全体を求めることができる．

相関攻撃には数多くの改良法が考案されており，その一つに暗号のモデルを誤り訂正符号のモデルに対応させ，誤り訂正符号における復号アルゴリズムを攻撃に利用したものがある．Mihaljević らは，BP（Belief Propagation）など，線形ブロック符号の繰返し復号法を利用した攻撃法を提案した [2]〜[4]．この攻撃法は，従来の手法よりも計算量を低減させ，解読成功確率を向上させることに成功している．だが，これらの

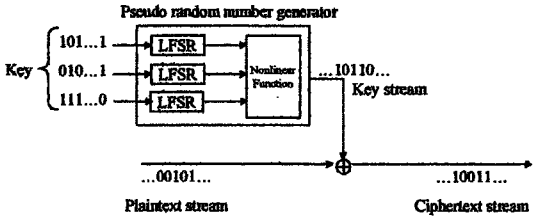


図 1　非線形コンバイナ型乱数生成器の構成
Fig. 1　Nonlinear combiner generator.

† 早稲田大学理工学部経営システム工学科，東京都
Department of Industrial and Management Systems Engineering, Waseda University, 3-4-1 Okubo, Shinjuku-ku, Tokyo, 169-8555 Japan
a) E-mail: hsb@suou.waseda.jp

攻撃法は単体のLFSRを攻撃するものであり，本来複数あるLFSRの初期状態と鍵系列で多次元の相関があるうちの一部分しか推定に使用していない．

そこで，本論文はMihaljević らのアルゴリズムを拡張し，多次元の相関を利用した複数のLFSRを同時に攻撃するアルゴリズムを提案する．このとき推定にかかる計算量は増加してしまうが，BPの並列アルゴリズムによる近似計算を行うことで，計算量の増加を抑えることができる．

更に，シミュレーションを行い解読成功確率などの性能評価を行う．ただし，本論文の提案法に対する耐性を表す指標として，従来用いられていたLimit Noiseをそのまま使うことはできない．Limit Noiseは，非線形関数をBSC（二元対称通信路）とみなした場合，誤り率 p がLimit Noise以下ならば攻撃が必ず成功するという指標であり，シミュレーションによって求められる．だが，本提案法では複数のLFSR系列と鍵系列の多次元の相関を利用するため，BSCで非線形関数を表すことはできない．そこで，提案法に対する耐性を表す指標として，鍵系列とLFSR系列集合との相互情報量を用いた．これは，推定に用いることのできる情報量を表すパラメータとして機能している．

以下に，本論文で使用する変数をまとめる．ただし，対象のLFSRを区別しない場合には上付き添字を省略する．

p：BSCの誤り率
N：鍵系列の観測長
L：LFSRの長さ
$x_1^{(d)}, x_2^{(d)}, ..., x_N^{(d)}$：LFSR$^{(d)}$ からの出力系列
$z_1, z_2, ..., z_N$：鍵系列
S_i：i 時点でのLFSR内部状態
A：LFSR内部状態の遷移行列
Ω_n：ビット n に関係するパリティ検査式集合
w：パリティ検査式の探索パラメータ
B：攻撃で全数探索にあてるビット数
m：パリティ検査式のインデックス
$\omega_n(m)$：ビット n に関係するパリティ検査式 m
α', α'', α''', α''''：正規化定数
d：更新対象のLFSRのインデックス
d'：更新対象外のLFSRのインデックス
$s_n^{(d)}(u), t_n^{(d)}(u), q_{mn}^{(d)}(u), r_{mn}^{(d)}(u)$：ファクタグラフ上で伝搬するメッセージ
$Q_n^{(d)}(u)$：$x_n^{(d)} = u$ となる推定確率
$\hat{X}^{(d)} = [\hat{x}_n^{(d)}]$：LFSR出力の推定系列
$p_{ij,k}$：確率モデルの遷移確率

2. 相関攻撃

相関攻撃は，観測された鍵系列とLFSR系列との相関を用いてLFSRの初期状態を推定する攻撃法である．この攻撃法は既知平文攻撃に分類され，平文，暗号文系列が攻撃者にとって既知である．特にストリーム暗号では，平文と暗号文の排他的論理和が鍵系列になるため，これも既知といえる．更に，暗号の構成は基本的に公開されるものであため，LFSR，非線形関数の構成も既知である．これらの条件のもとで未知の鍵を求めるのが，攻撃の目的である．

3. Mihaljević の攻撃法

3.1 Mihaljević の攻撃法の概要

LFSRの初期状態を求めるには，鍵系列からLFSR系列を求める，LFSR系列からLFSR初期状態の求めるという2段階のプロセスを踏むことになる．このうち後者は，LFSRの初期状態は十分な長さのLFSR系列からBerlekamp-Masseyアルゴリズムにより確定的に求まるため，容易である．よって，非線形関数の出力である鍵系列から入力であるLFSR系列を推定するのが攻撃の目的となる．

しかし，LFSR系列を求めるには多大な計算量がかかるため，非線形関数に確率モデルによる近似を行い，事後確率計算によってLFSR系列を推定する．更に，推定のための事後確率計算にも近似計算を用い，非線形関数の近似，事後確率計算の近似という二重の近似を行う．

まず，複数あるLFSR系列のうち一つの系列に着目し，非線形関数を一つのLFSR系列を入力，鍵系列を出力とする確率モデルに近似する．Mihaljević はこの確率モデルを誤り率 p のBSCとみなすことで暗号のモデルと誤り訂正符号のモデルを対応させた．非線形関数はBSC，LFSR初期状態は情報ベクトル，LFSRの構成は生成行列，LFSR出力 $x_1, x_2, ..., x_N$ は (N, L) パンクチャド符号の符号語，鍵系列 $z_1, z_2, ..., z_N$ は受信語に対応する．ここで，N は鍵系列の観測系列長，L はLFSRの長さを示す．

その上で，Mihaljević は誤り訂正符号の復号アルゴリズムを利用した．このとき，復号性能のより良い復号法を利用すれば暗号への攻撃性能も向上する．

Mihaljević の攻撃法は用いられた復号法によってOSDA（One Step Decoding Algorithm）とIDA（It-

erative Decoding Algorithm) に分類される．OSDA はしきい値復号法を利用したアルゴリズムであり，IDA は繰返し復号法を利用したアルゴリズムである．ここで繰返し復号法として用いられたのは，BF (Bit Flipping)，BP-BF (Belief Propagation based Bit Flipping)，APP (A Priori Propability Decoding)，BP (Belief Propagation) である．そして，この中でも最も高い解読成功確率を示しているのが，BP を利用した攻撃法である．

3.2 BP を用いた攻撃法 [3]

3.2.1 BP (Belief Propagation) [6]

BP は事後確率を近似計算するアルゴリズムである．ここでは，x_n のビット間の制約であるパリティ検査式を用いて，与えられた受信語 $z_n = k$ に対して符号語 $x_n = i$ を得る確率 $Pr(x_n = i \mid z_n = k)$ を計算し，これが最大となる i を推定値 $\hat{x}_n$ として出力する．

図 2 のようなグラフをファクタグラフと呼ぶ．これは，符号語のビット間の制約をグラフ表現したものである．丸いノードが変数を表す変数ノードであり，四角いノードが変数間の関係を表す関数ノードである．この図では，下段の丸いノードが符号語の各ビットを表すビットノードであり，上段の丸いノードがそれに対応した受信語の各ビットを表すビットノードである．下段の四角いノードがパリティ検査式を表すチェックノードであり，上段の四角いノードが非線形関数を表すノードである．それぞれのパリティ検査式からその検査式に含まれる変数のノードに対して枝が伸びており，x_n 間のパリティ制約が表現されている．

BP は，ファクタグラフ上でそれぞれの変数ノードと関数ノード間でメッセージを交換しながら計算を進めていくメッセージパッシングアルゴリズムである．なお，ファクタグラフのノード間でやり取りされる情報をメッセージと呼ぶ．

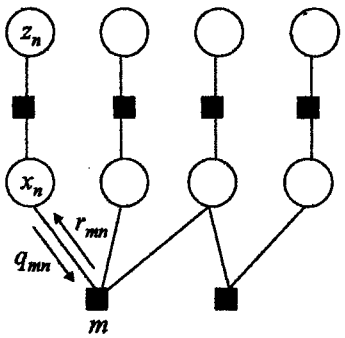


図 2 従来法のファクタグラフ
Fig. 2 Factor graph.

BP はファクタグラフが木構造ならば正しい事後確率が計算できるが，ループが存在する場合でも近似計算となり，実験的に良い結果が得られている．また，近似計算にかかる計算量にはグラフの枝の本数が大きく影響する．

3.2.2 パリティ検査式集合の構成

復号アルゴリズムの前処理として，パリティ検査式集合を構成する．

LFSR 系列には，LFSR の特性によるビット間の制約がある．LFSR のフィードバック多項式を $f(u) = 1 + \sum_{i=1}^{L} b_i u_i$ とすると，LFSR 系列は

$$x_{n+L+1} = b_1 x_{n+1} + b_2 x_{n+2} + \ldots + b_L x_{n+L}. \quad (1)$$

で生成されるため，次の制約式が成立する．

$$x_{n+L+1} + b_1 x_{n+1} + \ldots + b_L x_{n+L} = 0. \quad (2)$$

このような制約式を効率的に取得するために，制約式を符号におけるパリティ検査式に対応させる．パリティ検査式は生成行列から導出できるが，そのための準備として，まずは生成行列に対応する LFSR の働きを式表現する．S_i を i 時点での LFSR 内部状態とし，下式で表す．

$$S_i = \begin{bmatrix} x_{i+L} \\ \vdots \\ x_{i+1} \end{bmatrix}. \quad (3)$$

LFSR 内部状態の遷移行列を A とすると，下式が成り立つ．

$$S_i = AS_{i-1}, \quad i = 1, 2, \ldots. \quad (4)$$

$$S_i = A^i S_0, \quad i = 1, 2, \ldots. \quad (5)$$

A は下式のように記述できる．

$$A = \begin{bmatrix} b_1 & b_2 & b_3 & \cdots & b_{L-1} & b_L \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & \vdots \\ \vdots & \vdots & \vdots & \cdots & 0 & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix}. \quad (6)$$

次に，LFSR の構成からパリティ検査行列 H を導出する．A_1^i を A の i 乗の第 1 行，I_L を L 行 L 列の単位行列とすると，$[x_1 x_2 \ldots x_N] = [x_1 x_2 \ldots x_L]G$ を満たす

生成行列 G から，パリティ検査行列 $H=[P^T I_{N-L}]$ が得られる．ただし，P^T は下式．

$$P^T=\begin{bmatrix} A_1^1 \\ A_1^2 \\ \vdots \\ A_1^{N-L} \end{bmatrix}. \tag{7}$$

このパリティ検査行列 H を疎にしたパリティ検査式集合を［定義 1］より得る．この操作によりファクタグラフの枝の本数を減らすことができ，近似計算にかかる計算量を削減することができる．

［定義 1］ $n=L+1,...,N$ と $w, 1\leq w\leq W$ に対して，以下のプロセスでビット n に関係するパリティ検査式集合 Ω_n を構成する．

- パリティ検査行列の $n-L$ 行目と他の w 行の和を計算する．
- これらの和の中から，ビット $i=B+1, B+2,...,L$ がすべて 0 となるものを Ω_n として記録する．

W, B は攻撃者が自由に設定できるパラメータである．ただし，$B<L$ とする．

3.2.3 復号処理

一般的に LFSR の初期状態 L ビット全体を復号によって求めることは復号性能からいって困難であるため，全数探索を併用する．L ビットのうちはじめの B ビットを全数探索で求め，残りの $L-B$ ビットを復号によって求めることとし，B ビットに適当な値を仮定する．$m\in\Omega_n$ をビット n に関係するパリティ検査式につけたインデックスとし，$q_{mn}(u)$ を m 以外のパリティ検査式により得られた，ビット n の値が u となる確率とする．

$f_n(0)=1-p, f_n(1)=p$ とし，$q_{mn}(u)=f_n(u)$ と初期化して，以下の処理を行う．

［ステップ 1］ $\delta q_{mn}=q_{mn}(0)-q_{mn}(1)$ として以下の計算をする．ただし，$\omega_n(m)$ はビット n に関係するパリティ検査式 m，n' はその検査式に含まれるビットである．

$$\delta r_{mn}=\prod_{n'\in\omega_n(m)}\delta q_{mn'}. \tag{8}$$

$$r_{mn}(u)=(1/2)(1+(-1)^u\delta r_{mn}). \tag{9}$$

［ステップ 2］ 以下の更新を行う．ただし，m' は Ω_n に含まれる m 以外のパリティ検査式．α は $q_{mn}(0)+q_{mn}(1)=1$ となるように与えられる正規化定数であり，同様に α' は $Q_n(0)+Q_n(1)=1$ となるように与えられる正規化定数である．

$$q_{mn}(u)=\alpha f_n(u)\prod_{m'\in\Omega_n\backslash m} r_{m'n}(u). \tag{10}$$

$$Q_n(u)=\alpha' f_n(u)\prod_{m\in\Omega_n} r_{mn}(u). \tag{11}$$

［ステップ 3］ $Q_n(1)>0.5$ ならば $\hat{x}_n=1$，$Q_n(1)\leq 0.5$ ならば $\hat{x}_n=0$ として推定系列 $\hat{X}=[\hat{x}_n]$ を生成する．これがすべてのパリティ検査式を満たせば，$\hat{X}$ を復号結果とし，そうでなければステップ 1 へ戻る．

復号結果を得られないまま定められた反復回数を終えたら，全数探索にあてた B ビットにまた別の値を仮定して，ステップ 1〜3 の処理を行う．これを繰り返して，B ビットにとり得るすべての値を仮定して結果が得られなければ，攻撃失敗．結果を得られたならば，復号結果 $\hat{X}$ が正しいかどうかの最終チェックを行う．$\hat{x}_{L+1}, \hat{x}_{L+2},..., \hat{x}_N$ を用いて $\hat{x}_1, \hat{x}_2,..., \hat{x}_L$ を構成し，これから得られる符号語 $\tilde{x}_1, \tilde{x}_2,..., \tilde{x}_N$ で下式を計算する．

$$S=\sum_{n=1}^{N}\tilde{x}_n\oplus z_n. \tag{12}$$

T をしきい値として $S\leq T$ ならば，$\hat{x}_1, \hat{x}_2,..., \hat{x}_L$ を LFSR 初期状態の正しい推定値として出力する．

4. 多次元の相関を利用した攻撃

従来法は単体の LFSR へ攻撃する手法であった．本章ではこれを応用し，複数の LFSR を同時に攻撃する手法を提案する．ここでは，$\mathrm{LFSR}^{(1)}$ と $\mathrm{LFSR}^{(2)}$，二つの LFSR を同時に攻撃する場合についてのみ記述する．

4.1 概　　要

$\mathrm{LFSR}^{(1)}$ 出力の $x_n^{(1)}$ と $\mathrm{LFSR}^{(2)}$ 出力の $x_n^{(2)}$ のペア $(x_n^{(1)}, x_n^{(2)})$ と z_n の多次元の相関を用いて，$\mathrm{LFSR}^{(1)}$ と $\mathrm{LFSR}^{(2)}$ を同時に攻撃する．

そのために，複数の LFSR を含む暗号のモデルを多端子の通信路モデルに対応させる．2 個の符号語ビット $x_n^{(1)}$，$x_n^{(2)}$ が通信路の入力となり，受信語 z_n が出力となる．このとき，$x_n^{(1)}$ の推定に $x_n^{(2)}$ の確率情報を利用して，$x_n^{(2)}$ の推定にも同様に $x_n^{(1)}$ の確率情報を利用することができる．

より詳しくいえば，多次元の相関確率 $Pr(z_n=k \mid x_n^{(1)}=i, x_n^{(2)}=j)$ が既知であるため，他

の LFSR の確率情報のもとでの $x_n^{(1)}$ の確率情報 $Pr(x_n^{(1)}=i \mid x_n^{(2)}=j, z_n=k)$ を $x_n^{(1)}$ の推定に用いることができるということである．

ここで，相互情報量を推定に用いることのできる情報量を表すパラメータとして用いる．$x_n^{(1)}$ の推定には，従来法では $I(x_n^{(1)};z_n)$ の情報しか利用できていなかったが，提案法では $I(x_n^{(1)};x_n^{(2)},z_n) = I(x_n^{(1)};z_n)+I(x_n^{(1)};x_n^{(2)} \mid z_n)$ の情報量を利用することができ，明らかに情報量が増加している．このようにして推定に使う情報量を増やすことにより解読成功確率を向上させることが，提案法のねらいである．このとき，推定にかかる計算量は増加するため，BP の並列アルゴリズムを用いて計算量の増加を抑える．

複数の LFSR を同時に攻撃することで推定に使う情報量を増加させることと，並列アルゴリズムで計算量を抑えることの 2 点が提案法の特徴である．

4.2 BP の並列アルゴリズムを利用した攻撃

4.2.1 多次元での BP

提案法の多端子の符号モデルをグラフ化すると，図 3 のようなファクタグラフになる．上段と下段の四角いノードがそれぞれ LFSR$^{(1)}$，LFSR$^{(2)}$ 出力のパリティ検査式を表すチェックノード，中段の四角いノードが非線形関数を表すノードである．$x_n^{(1)}$ と $x_n^{(2)}$ が非線形関数の入力，z_n が出力であり，非線形関数の確率モデル $Pr(z_n \mid x_n^{(1)},x_n^{(2)})$ が与えられている．よって，このノードを介して $x_n^{(1)}$ と $x_n^{(2)}$ のノードを結ぶことができる．提案法では，従来法で伝搬するメッセージに加えて，非線形関数ノードから変数ノードへ送られるメッセージ s_n と，変数ノードから非線形関数ノードへ送られるメッセージ t_n の処理を行う．

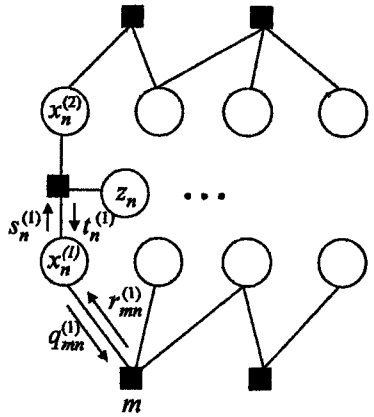


図 3 提案法のファクタグラフ
Fig. 3 Factor graph of proposal method.

4.2.2 アルゴリズム

パリティ検査式の構成や全数探索の仮定は従来法と同様である．$d=1$ ならば $d'=2$，$d=2$ ならば $d'=1$ として，各 d に対して以下のように初期値を与える．

$$q_{mn}^{(d)}(u) = Pr(x_n^{(d)}=u \mid z_n) = \sum_{i\in\{0,1\}} Pr(z_n \mid x_n^{(d)}=u, x_n^{(d')}=i) \, Pr(x_n^{(d)}=u, x_n^{(d')}=i)/Pr(z_n). \quad (13)$$

$$t_n^{(d)}(u) = Pr(x_n^{(d)}=u \mid z_n). \quad (14)$$

[ステップ 1] 以下の更新を $LFSR^{(1)}$，LFSR$^{(2)}$ について順に行う．

他の LFSR の出力系列 $x_n^{(d')}$ の確率情報のもとで $x_n^{(d)}$ の値が u となる確率 $s_n^{(d)}(u)$ を更新する．ただし，α は $s_n(0)+s_n(1)=1$ となるように与えられる正規化定数である．

$$s_n^{(d)}(u) = \alpha \sum_{i\in\{0,1\}} t_n^{(d')}(i) \, Pr(z_n=k_n \mid x_n^{(d)}=u, x_n^{(d')}=i). \quad (15)$$

従来法に $s_n^{(d)}(u)$ を加味して，以下のように更新する．ただし，$\alpha',\alpha'',\alpha'''$ はそれぞれ $r_{mn}(0)+r_{mn}(1)=1$，$q_{mn}(0)+q_{mn}(1)=1$，$Q_n(0)+Q_n(1)=1$ となるように与えられる正規化定数である．

$$r_{mn}^{(d)}(u) = \alpha' \sum_{n''\in\omega_n(m)\backslash n, \sum i_{n''}=0} \prod_{n'\in\omega_n(m)\backslash n} q_{mn'}^{(d)}(i_{n'}^{(d)}). \quad (16)$$

$$q_{mn}^{(d)}(u) = \alpha'' s_n^{(d)}(u) \prod_{m'\in\Omega_n\backslash m} r_{m'n}^{(d)}(u). \quad (17)$$

$$Q_n^{(d)}(u) = \alpha''' s_n^{(d)}(u) \prod_{m\in\Omega_n} r_{mn}^{(d)}(u). \quad (18)$$

パリティ制約下での $x_n^{(d)}$ の値が u となる確率 $t_n^{(d)}(u)$ を更新する．ただし，α'''' は $t_n(0)+t_n(1)=1$ となるように与えられる正規化定数である．

$$t_n^{(d)}(u) = \alpha'''' \prod_{m'\in\Omega_n} r_{m'n}^{(d)}(u). \quad (19)$$

[ステップ 2] $Q_n^{(d)}(1) > 0.5$ ならば $\hat{x}_n^{(d)}=1$，$Q_n^{(d)}(1) \le 0.5$ ならば $\hat{x}_n^{(d)}=0$ として推定系列

$\hat{X}^{(d)} = [\hat{x}_n^{(d)}]$ を生成する．$\hat{X}^{(1)}, \hat{X}^{(2)}$ の両者がすべてのパリティ検査式を満たせばそれを復号結果として出力．そうでなければ，ステップ1へ戻る．B ビットにとり得るすべての値を仮定しても結果が得られなければ，攻撃失敗．

最終チェックのプロセスは従来法と同様である．

ここでは，LFSR2個を同時に攻撃する際のアルゴリズムを例として挙げたが，同様のアルゴリズムで更に多くのLFSRを同時に攻撃することが可能である．

5. シミュレーションによる評価

5.1 解読成功確率と計算量の比較

5.1.1 実験内容

非線形関数に確率モデルを仮定して攻撃実験を行い，従来法と提案法の解読成功確率と計算量を比較する．

$p_{ij,k} = Pr(z_n = k \mid x_n^{(1)} = i, x_n^{(2)} = j)$ と確率モデルの遷移確率を表記する．提案法で仮定する確率モデルは，$p_{aa,a} = 0.7$，$p_{ab,a} = 0.5$，$p_{ba,a} = 0.5$，$p_{bb,a} = 0.3$，$b = a+1$．これを $x_n^{(1)}, x_n^{(2)}$ それぞれで周辺化して，従来法で仮定する確率モデルを得る．ここでは $x_n^{(1)}$, $x_n^{(2)}$ どちらにおいても誤り率 $p = 0.4$ のBSCが得られる．

解読成功確率は実験回数分の攻撃を試み，そのうち攻撃が成功した割合を指す．従来法ではすべてのLFSRで同時に成功した場合に攻撃成功とする．計算量はBPの演算（加算・乗算）回数 × 攻撃成功時の平均反復回数とする．ただし，提案法ではすべてのLFSRに対してメッセージ伝搬を行ったら反復1回とする．

シミュレーション条件は，$N = 1024$，$L = 40$，$B = 26$，$W = 2$，BPの反復回数 $I = 30$，実験回数 $J = 200$．LFSRのフィードバック多項式は，$f(u) = 1 + u^1 + u^3 + u^5 + u^9 + u^{11} + u^{12} + u^{17} + u^{19} + u^{21} + u^{25} + u^{27} + u^{29} + u^{32} + u^{33} + u^{38} + u^{40}$．これらの条件は他の実験でも同様である．

5.1.2 実験結果

実験を行い，表1の結果を得た．

表1 解読成功確率と計算量の比較
Table 1 Comparison of performances.

	解読成功確率	反復回数	計算量
従来法	0.39	15.6	1.01×10^7
提案法	0.46	8.2	1.15×10^7

5.2 従来法の適用できないモデルへの攻撃

5.2.1 実験内容

前節と同様の実験を，従来法では攻撃が困難な場合について行う．

提案法で仮定する確率モデルは3種類．

Model 1は，$p_{aa,a} = 0.7$，$p_{ab,a} = 0.7$，$p_{ba,a} = 0.4$，$p_{bb,a} = 0.2$，$b = a+1$．従来法では，$x_n^{(1)}$ の確率モデルが $p = 0.3$ のBSC，$x_n^{(2)}$ の確率モデルが $p = 0.45$ のBSCに対応する．後者のモデルは p が大きすぎるため従来法での攻撃は困難である．

表2 Model 1の結果
Table 2 Result of Model 1.

	解読成功確率	反復回数	計算量
従来法	0.00	-	-
提案法	0.18	13.2	1.86×10^7

Model 2は，$p_{aa,a} = 0.9$，$p_{ab,a} = 0.9$，$p_{ba,a} = 0.1$，$p_{bb,a} = 0.1$，$b = a+1$．従来法では，$x_n^{(1)}$ の確率モデルが $p = 0.1$ のBSC，$x_n^{(2)}$ の確率モデルが $p = 0.5$ のBSCに対応する．後者のモデルは $p = 0.5$ であるため従来法での攻撃は不可能である．

表3 Model 2の結果
Table 3 Result of Model 2.

	解読成功確率	反復回数	計算量
従来法	0.00	-	-
提案法	0.00	-	-

Model 3は，$p_{aa,a} = 0.9$，$p_{ab,a} = 0.7$，$p_{ba,a} = 0.1$，$p_{bb,a} = 0.3$，$b = a+1$．従来法では，$x_n^{(1)}$ の確率モデルが $p = 0.2$ のBSC，$x_n^{(2)}$ の確率モデルが $p = 0.5$ のBSCに対応する．これも，後者のモデルが $p = 0.5$ であるため従来法での攻撃は不可能である．

表4 Model 3の結果
Table 4 Result of Model 3.

	解読成功確率	反復回数	計算量
従来法	0.00	-	-
提案法	1.00	5.4	7.61×10^6

5.2.2 実験結果

実験を行い，Model 1では表2，Model 2では表3，Model 3では表4の結果を得た．

5.3 同時に攻撃するLFSRの数に関する実験

5.3.1 実験内容

ここまでの実験でLFSR2個を攻撃する場合の性能を評価したが，提案法は更に多くのLFSRを同時に攻撃することもできる．ここでは，同時に攻撃するLFSRの数を更に増やした場合について評価する．

LFSR4個の確率モデルをLFSR1個ずつ，2個ず

つ，若しくはLFSR4個同時に攻撃する．1個ずつ，2個ずつ攻撃する際の確率モデルは，LFSR4個の確率モデルを周辺化して得られる．

これまでの実験では，攻撃対象のLFSRすべてを正しく推定することで攻撃成功となり，この確率を攻撃成功確率としていたが，これでは攻撃対象とするLFSRの個数が異なる場合を比較することができない．よって，本実験ではr個のLFSRを同時に攻撃した場合に，解読成功確率の$1/r$乗をLFSR1個当りの解読成功確率とし，攻撃性能の評価指標として用いる．

また，本実験では相互情報量を解読成功確率を見積もる指標として用いる．従来法ではBSCの誤り率に対して解読成功確率がどの程度であるかを求め，その値によって攻撃法を評価していた．だが，提案法では複数のLFSRを攻撃するため，従来法と同様の評価はできない．そのため，攻撃対象のLFSRの数が違う場合を統一的に測るために相互情報量を用いる．本実験では相互情報量に対してLFSR1個当りの解読成功確率がどの程度であるかを求め，その値によって攻撃法を評価する．

攻撃するのはLFSR4個の確率モデルが2種類．

Model 4は，$p_{aaaa,a}=0.9, p_{aaab,a}, \ldots, p_{baaa,a}=0.7$, $p_{aabb,a}, \ldots, p_{bbaa,a}=0.5$, $p_{abbb,a}, \ldots, p_{bbba,a}=0.3$, $p_{bbbb,a}=0.1$, とする．

Model 5は，$p_{aaaa,a}=0.7, p_{aaab,a}, \ldots, p_{baaa,a}=0.8$, $p_{aabb,a}, \ldots, p_{bbaa,a}=0.5$, $p_{abbb,a}, \ldots, p_{bbba,a}=0.2$, $p_{bbbb,a}=0.3$, とする．

5.3.2 実験結果

Model 4では表5の結果を，Model 5では表6の結果を得た．

5.4 具体的な非線形関数への攻撃

5.4.1 実験内容

ここまでは非線形関数を確率モデルに置き換えたものを攻撃していたが，ここでは実際の擬似乱数生成器に使われている非線形関数を攻撃する．

攻撃対象とするのは，Geffe型乱数生成器と3DRG，4DRGの非線形関数である[5]．Geffe型乱数生成器は3個のLFSRで構成され，非線形関数は$z_n = x_n^{(1)}x_n^{(2)} + x_n^{(2)}x_n^{(3)} + x_n^{(3)}$である．3DRG，4DRGはそれぞれ3個，4個のLFSRをもつDRG（ダイナミック型乱数生成器）である．本実験では，これらの非線形関数に対してすべてのLFSRを同時に攻撃する．

5.4.2 実験結果

Geffe型乱数生成器，3DRG，4DRGのいずれに対しても実験回数200回の攻撃にすべて成功した．

表5 Model 4の結果
Table 5 Result of Model 4.

	相互情報量	LFSR1個当りの解読成功確率
LFSR1個	2.905×10^{-2}	0.55
LFSR2個	3.031×10^{-2}	0.68
LFSR4個	3.443×10^{-2}	0.62

表6 Model 5の結果
Table 6 Result of Model 5.

	相互情報量	LFSR1個当りの解読成功確率
LFSR1個	2.905×10^{-2}	0.55
LFSR2個	3.031×10^{-2}	0.68
LFSR4個	5.725×10^{-2}	0.73

6. 考　　察

表1より，提案法は従来法から計算量をさほど増加させずに解読成功率を向上させることができたことが分かる．他のLFSRの出力$x_n^{(d')}$の確率情報を推定に用いて情報量を増やすことは，攻撃に有効だといえる．

また，表2，表4より，従来法では攻撃が困難なモデルに対して，提案法による攻撃が成功する場合があることも示すことができた．x_nとz_nの一次元の相関が小さく従来法で攻撃できなかった場合でも，$(x_n^{(1)}, x_n^{(2)}, \ldots)$と$z_n$の多次元の相関が大きければ提案法による攻撃が可能だといえる．しかし，一次元の相関を見ればModel 3よりModel 2の方が相関が大きいにもかかわらず，Model 2では攻撃に失敗している．これは，$x_n^{(1)}$から$x_n^{(2)}$へ漏れる情報量が小さいからであると考えられる．この情報量を相互情報量の形で表すと，$I(x_n^{(2)}; x_n^{(1)}) = 0$となる．提案法での攻撃を行う際，攻撃の成否には，確率モデルを周辺化したBSCの誤り率pよりも，相互情報量で表される，推定に使える情報量が影響するといえる．

また，表5，表6より，確率モデルによっては，同時に攻撃するLFSRの個数を増やしたからといって，必ずしもLFSR1個当りの解読成功確率が上がるとは限らないことが分かる．同時に攻撃するLFSRの数を増やすメリットは，相互情報量の増加である．ただし，BPの近似精度が落ちるというデメリットもある．Model 4では相互情報量の増加が顕著であるため，解読成功確率は向上した．Model 5では相互情報量の増加はわずかであるため，近似精度悪化の方が大きく影響し，LFSR1個当りの解読成功確率は減少している．

よって，同時に攻撃するLFSRの数を増やした場合に相互情報量が大きく増加するような非線形関数のクラスに対して提案法は有効であるといえる．

また，提案法では，攻撃対象の非線形関数によって攻撃するLFSRの個数と組合せを適切に選択する必要がある．この選択にLFSRの個数の選択，組合せの選択の2ステップを踏むこととして考察する．

個数選択のステップでは，BPの近似精度の影響があるため，厳密な選択をすることは困難である．相互情報量を目安として実験を繰り返すしかない．

組合せ選択のステップでは，攻撃対象のLFSRの数が同じならばBPの近似精度もほぼ変わらないと考え，最も相互情報量が大きくなる組合せを選択するという方法が考えられる．

更に，具体的な非線形関数に対する攻撃も成功し，提案法が実用的な攻撃法であることが示された．

Geffe型乱数生成器の非線形関数を各LFSRに対してモデル化すると，それぞれ誤り率 $p = 0.25, 0.5, 0.25$ のBSCになる．2番目の p が大きく，$I(x_n^{(2)}; z_n) = 0$ となるため，従来法では2番目のLFSRの初期状態を推定できない．しかし，提案法では攻撃の容易な1番目，3番目のLFSR出力の情報を2番目のLFSR出力の推定に利用できる．相互情報量は $I(x_n^{(2)}; x_n^{(1)}, x_n^{(3)}, z_n) = 0.5$ と大きくなるため，攻撃が可能となる．

4DRGの非線形関数を各LFSRに対してモデル化すると，誤り率 $p = 0.4375, 0.4375, 0.4375, 0.6875$ のBSCになる．1，2，3番目の p が大きすぎるため，従来法でLFSR初期状態を推定するのは困難である．しかし，提案法では4番目のLFSRの情報を利用して，残りのLFSR出力系列も推定できる．3DRGに対しても同様の考察ができる．

7. むすび

本論文では繰返し復号法を利用した相関攻撃を応用し，多次元の相関を利用して複数のLFSRを同時に攻撃する手法を考案した．この手法を用いることで，推定に使う情報量を増やし，解読成功確率の向上を見込むことができる．

文　献

[1] W. Meier and O. Staffelbach, "Fast correlation attacks on certain stream ciphers," J. Cryptol., vol.1, no.3, pp.159–176, 1989.

[2] M.J. Mihaljević, M.P.C. Fossorier, and H. Imai, "A low-complexity and high-performance algorithm for the fast correlation attacks," FSE2000, LNCS 1978, pp.196–212, Springer-Verlag, April 2000.

[3] M.J. Mihaljević, M.P.C. Fossorier, and H. Imai, "On decoding techniques for cryptanalysis of certain encryption algorithms," IEICE Trans. Fundamentals, vol.E84-A, no.4, pp.919–930, April 2001.

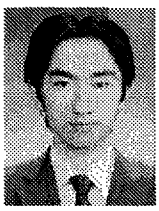
[4] M.J. Mihaljević, M.P.C. Fossorier, and H. Imai, "An algorithm for cryptanalysis of certain keystream generators suitable for high-speed software and hardware implementations," IEICE Trans. Fundamentals, vol.E84-A, no.1, pp.311–318, Jan. 2001.

[5] 白石善明，森井昌克，植松友彦，坂庭好一，"非線形コンバイナ型乱数生成器の特性—線形複雑度，相互情報量，無相関性について，" 信学論（A），vol.J83-A, no.10, pp.1169–1179, Oct. 2000.

[6] 和田山正，低密度パリティ検査符号とその復号法，トリケップス，2002.

[7] 細渕智史，斎藤友彦，松嶋敏泰，"Fast Correlation Attack の改良法に関する一考察，" 第27回情報理論とその応用シンポジウム予稿集，pp.37–40, 2004.

（平成17年4月8日受付，9月6日再受付，10月13日最終原稿受付）



細渕　智史

平15早大・理工・経営システム卒．平17同大大学院修士課程了．同年，日本電気（株）入社．在学中，暗号に関する研究に従事．

齋藤　友彦

平13早大・理工・経営システム卒．平15同大大学院修士課程了．同大大学院博士課程入学．符号理論及び実験計画法に関する研究に従事．

松嶋　敏泰　（正員）

昭53早大・理工・工業経営卒．昭55同大大学院修士課程了．同年，日本電気（株）入社．昭61早大・理工学研究科・博士後期過程入学．平元横浜商科大学講師．平3同大助教授．平4早大・理工学部・工業経営学科（現在経営システム工学科）助教授．平9同大教授．現在に至る．知識情報処理及び情報理論とその応用に関する研究に従事．工博．平13ハワイ大学客員研究員．IEEE，情報理論とその応用学会，人工知能学会，情報処理学会，OR学会，日本経営工学会等各会員．

A Parallel Iterative Decoding Algorithm for Zero-tail and Tail-biting Convolutional Codes

Toshiyasu Matsushima
Waseda University
Shinjuku, Tokyo, JAPAN
e-mail: toshi@matsu.mgmt.waseda.ac.jp

Tomoko K. Matsushima
Polytechnic University,
Sagamihara, JAPAN

Shigeichi Hirasawa
Waseda University,
Shinjuku, Tokyo, JAPAN

I. Introduction

We apply a parallel propagation algorithm[2][3] to the decoding of convolutional codes. The parallel algorithm surpasses the BCJR algorithm in parallel computational complexity and performance when it is applied to tail-biting codes. We analyze the performance of the algorithm by a numerical method similar to the density evaluation[4].

II. A parallel propagation algorithm and the application to convolutional codes

Efficient parallel propagation algorithms were proposed for calculating generalized posterior distribution on extended junction graphs(EJGs) [1][2][3]. If an EJG has no loops, those algorithms halt and calculate the exact marginal generalized posterior distributions on the EJG.

We apply one of the algorithms to the EJGs representing convolutional codes. BNs and factor graphs use state nodes for representing convolutional codes, in order to avoid loops on the graphs. However, EJGs do not need such auxiliary nodes.

Generally, the complexity of iterative decoding algorithms on graphs depends on the dimension of variables in the largest clique and the number of iterations. Let symbols be binary. The dimension of variables in the largest clique on the EJG of a non-recursive convolutional code is $O(2^v)$, where v is the constraint length of the code, while that on the graph using state nodes is $O(2^{2v})$. So the complexity of the EJGs is generally lower than that of the graph using state nodes such as BNs or factor graphs.

III. Numerical performance analysis

Since the EJGs of convolutional codes have no loop, the value calculated by the algorithm converges on the correct posterior probability, and the number of iterations for converging to the correct probability is the same as the code length. The BCJR algorithm also requires the iteration whose number is the same as the code length before it halts. However, in the case where the parallel algorithm calculates approximate values with some suitable precision, the number of the iterations needed for the calculation is the order of the constraint length of the code. Since the strength of the correlation between two cliques is inverse proportion to the distance, the algorithm can calculate the posterior probability of each clique with sufficient precision by using the cliques in its neighborhood.

We analyze the relationship between the error probability and the number of iterations in the parallel algorithm by a numerical method that is similar to the density evaluation method[4]. Since the graphs representing convolutional codes are constructed by the iteration of a simple and symmetric structure, the distribution of the posterior probability can be easily calculated. We can calculate the distribution of output messages by the following formulas.

$$P(w) = \bigotimes_{N_k \in S_l^N(N_i)} P(\log \mu_t^{k \to i}).$$

$$P(\mu_{t+1}^{i \to l}) = \bigotimes_{x \notin D_m} q(N_i) P(\exp w). \quad (2)$$

The error probabilities calculated by the analysis are monotone decreasing with respect to the number of iterations. The analysis shows that the required number of iteration for attaining almost the optimal error probability given by the MAP decoding is $2v$ or $3v$, while the BCJR algorithm cannot halt before the number of iteration reaches the code length. Our simulation shows similar results to the numerical analysis.

IV. The application to tail-biting codes

We apply the algorithms to tail-biting convolutional codes. Since there is no tail in a tail-biting code, the BCJR algorithm cannot be directly applied to the code. So the BCJR algorithm is applied iteratively on the trellis of a tail-biting code until the calculation values converge to constants in the previous research[1]. We call this algorithm TB-BCJR. However, the parallel algorithm can be directly applied to tail-biting convolutional codes, because the algorithm can work not only the EJGs without loops but also the EJGs with loops. Both the parallel algorithm and the TB-BCJR algorithm calculate approximate posterior probability. The bit error probability of the parallel algorithm is lower than that of the TB-BCJR algorithm in our experimental results.

Since the approximate probability of each clique is calculated by using the cliques in its neighborhood in the parallel algorithm, the information of zero-tail is not so important. On the other hand, since the information of zero-tail is very important in the BCJR algorithm, the performance of the TB-BCJR algorithm is lower than that of the BCJR algorithm for zero-tail convolutional codes.

The computational complexity of the TB-BCJR algorithm is much higher than that of the BCJR algorithm for zero-tail convolutional codes. The computational complexity of the parallel algorithm for tail-biting codes, however, is almost the same as that for zero-tail convolutional codes.

[1] Although an intersection node in junction graphs is connected to two clique nodes, an intersection node in EJGs may be connected to more than two clique nodes. The joint distribution represented by EJGs is $P(x_1, \ldots, x_n) = \alpha q(N_1) q(N_2) \cdots q(N_{n_N})$, where $N_l \subset \{X_1, \ldots, X_n\}$ is a clique node. Let $S^N(D_m)$ be the neighboring clique node set of an intersection node D_m. If the information of the random variables in an intersection node is given, the intersection node is called restricted intersection node(RIN). The algorithm propagates the message from a clique node N_i to a clique node N_l connected with it by way of an intersection node D_m as follows:

$$\mu_{t+1}^{i \to l} = \begin{cases} \frac{P^*(N_i)}{\mu_t^{l \to i}} & \text{if } D_m \text{ is a RIN} \\ \sum_{x \notin D_m} q(N_i) \prod_{N_k \in S_l^N(N_i)} \mu_t^{k \to i} & \text{otherwise,} \end{cases} \quad (1)$$

where $S_l^N(N_i) = \{N_k \in S^N(D_h) | D_h \in S^D(N_i), D_h \notin S^D(N_l), k \neq i\}$.

The marginal posterior probability of each clique is calculated by $P_t(N_l) = q(N_l) \prod_{N_k \in S^N(N_l)} \mu_t^{k \to l}$, where $S^N(N_l) = \{N_k \in S^N(D_h) | D_h \in S^D(N_l), k \neq l\}$.

References

[1] J .B .Anderson and S .M. Hladik, *Tailbiting MAP decoders*, Selected Areas in Communications, IEEE Journal, Vol. 16, 1988.

[2] T. Matsushima, T.K. Matsushima and S. Hirasawa *An Alternative Algorithm for Calculating Posterior Probability and Decoding*, Proceedings of IEEE Int. Symp. on Information Theory, 2002.

[3] T. Matsushima, T.K. Matsushima and S. Hirasawa *Calculation of Generalized Posterior Distribution on Junction Graphs* , Proceedings of the 24th Symposium on Information Theory and Its Applications, 2002.

[4] T.J. Richardson, R.L. Urbanke, *The capacity of low-density parity-check codes under message-passing decoding*, IEEE Trans. IT, Vol.47 No. 2, 2001.

0-7803-7728-1/03/$17.00 ©2003 IEEE.

The 26th Symposium on Information Theory
and Its Applications (SITA2003)
Higashiura, Hyogo, Japan, Dec. 15–18, 2003

Parallel Propagation Algorithms for Tailbiting Convolutional Codes

Toshiyasu Matsushima* Tomoko K. Matsushima† Shigeichi Hirasawa*

Abstract— We apply parallel propagation algorithms to decoding of zero-tail and tailbiting convolutional codes. In the application of the parallel algorithm to zero-tail convolutional codes, the iteration number for achieving almost the same bit error probability as that the MAP decoding achieves is 2 or 3 times of the constraint length of codes. So the parallel computational complexity of the parallel propagation algorithm is very low. Our experimental results show that the bit error rate of the parallel decoding algorithm is lower than that of the tailbiting BCJR algorithm. We evaluate the parallel algorithm by numerical methods, which is similar to density evolution method.

Keywords—message propagation algorithm, MAP decoding, tail-biting convolutional codes, the BCJR algorithm, probabilistic reasoning

1 Introduction

The BCJR algorithm[3] is well-known as the MAP decoding procedure, which minimizes symbol error probability, for the convolutional codes whose initial state and final state are known. Such convolutional codes are called zero-tail convolutional code in this paper.

Since there are no tails in a tailbiting convolutional code, the BCJR algorithm can not be directly applied to the code. Several decoding procedure for tailbiting convolutional codes were proposed in previous research. There is a decoding algorithm named tailbiting BCJR algorithm[2], which iterates the forward recursion of the BCJR algorithm on the trellis until the calculated probability of each variable converges to some constant. Unfortunately the tailbiting BCJR algorithm does not guarantee MAP decoding.

The BCJR algorithm is interpreted as the message propagation algorithms that are equal to the Belief Propagation(BP)[13][8] and the sum-product[7] algorithm. The message propagation algorithm can be roughly classified into sequential algorithms and parallel algorithms from the viewpoint of message propagation strategy. The BCJR algorithm is a typical sequential algorithm and the sum-product algorithm for LDPC codes is a typical parallel algorithm. Both converge to the correct marginal posterior probability on graphs without loops. However sequential algorithms cannot work on graphs with loops. So the BCJR algorithm cannot be directly applied to the tailbiting convolutional code.

On the other hand, although parallel algorithms do not guarantee the convergence to the correct posterior probability, they can directly work on graphs with loops. So we apply parallel propagation algorithms to decoding of zero-tail and tailbiting convolutional codes. We proposed parallel propagation algorithms for calculating marginal generalized posterior probability [1] on extended junction graphs (EJGs)[2][9][10][11]. We apply the algorithms to the EJGs representing convolutional codes.

For zero-tail convolutional codes, the parallel algorithms and the BCJR algorithm also need the iteration whose number is the same as the code length before they calculate correct posterior probabilities. However, in the case where the parallel algorithm calculates approximate values with some suitable precision, the number of the iteration for the calculation is the order of the constraint length of the convolutional code.

Since the strength of the correlation between two cliques is inverse proportion to the distance, accurate approximation to the posterior probability of each clique is calculated by using only the cliques in its neighborhood. In the parallel algorithm, almost all nodes except the nodes near the tails do not need the information about the initial and the final state of the tails for decoding. So we expect the parallel algorithms are useful for decoding of tailbiting convolutional codes. The bit error rate of the parallel decoding algorithm is lower than that of the tailbiting BCJR algorithm for some tailbiting convolutional codes.

In Section 2, we review the tailbiting BCJR algorithm of the previous research. In Section 3, we explain parallel propagation algorithms for calculating marginal generalized posterior probability on EJGs. We apply the algorithms to the EJGs representing convolutional codes. In section 4, we compare the parallel algorithm and the BCJR algorithm for decoding zero-tail convolutional codes. We also compare the parallel algorithm and the tailbiting BCJR algorithm for decoding tailbiting convolutional codes. In Section 5, we evaluate the parallel algorithm by numerical methods, which is similar to density evolution method[14].

2 Decoding algorithm for tailbiting convolutional codes

We review the tailbiting BCJR algorithm, which was proposed the previous paper [2] entitled tailbiting MAP decoders. The probabilistic structure of tailbit-

* Waseda University, Tokyo, JAPAN. Email: toshi@matsu.mgmt.waseda.ac.jp
† Polytechnic University, Sagamihara, JAPAN.

[1] In the case that an evidence is given by the distribution of random variables such as $P(X = x) = p_x$, this type of evidence is called distribution-evidence or soft-evidence in the previous research in the AI field. The posterior or conditional distributions given distribution-evidence are formalized as generalized posterior distributions[9]. Although we need not generalized posterior distributions but ordinary posterior distributions in the decoding problem, these algorithms may be applied to decoding.

[2] Although an intersection node in a junction graph(JG)[6] is connected to two clique nodes, an intersection node in an EJG may be connected to more than two clique nodes.

ing convolutional codes is represented by the following formula.

$$P(S^L, y^L) = P(S_1, y_1|S_0) \cdots P(S_0, y_L|S_{L-1}), \quad (1)$$

where S_i is a state and y_i is a received symbol.

The theoretical base of the algorithm depends on the following equation:

$$\begin{aligned} P(S_0|y^L) &= z \sum_{S_{L-1}} P(S_0, y_L|S_{L-1}) \\ &\quad \cdots \sum_{S_0} P(S_1, y_1|S_0) P(S_0|y^L), \quad (2) \end{aligned}$$

where z is normalized constant.

Let α_0 be the row vector whose elements are $P(S_0|y^L)$ and let Γ_i denote the operation $\sum_{S_{i-1}} P(S_i, y_i|S_{i-1})$. We can rewrite Equation (3) as the following equation:

$$\alpha_0 = z\alpha_0\Gamma_0 \cdots \Gamma_L. \quad (3)$$

Thus, α_0 is the normalized left eigenvector of the matrix $\Gamma_0 \cdots \Gamma_L$. Since the operation Γ_i is the forward recursion of the BCJR algorithm, if the forward recursion is iterated enough time, then the resulting output sequence converges to α_0 and the other $P(S_i|y^L)$. This algorithm is called the tailbiting BCJR algorithm in the previous paper.

However, the theoretical base of this algorithm represented by Equation (3) is not correct. From equation (1), the marginal posterior probability $P(S_0|y^L)$ is represented by

$$P(S_0|y^L) = z \sum_{S_1^{L-1}} P(S_1, y_1|S_0) \cdots P(S_0, y_L|S_{L-1}). \quad (4)$$

This equation does not equal to Equation (3). So, unfortunately the tailbiting BCJR algorithm is not the MAP decoder for tailbiting codes.

3 Parallel propagation algorithms for convolutional codes

3.1 Parallel propagation algorithms on EJGs

Parallel propagation algorithms were proposed for calculating marginal generalized posterior probability on extended junction graphs (EJGs) [9][10][11]. Let X_i $i \in I = \{1, \ldots, n\}$ and E_j $j \in I_C \subset I$ be discrete random variables and E_j is called evidence. An EJG is defined by a clique node set $S_N = \{N_1, N_2, \ldots N_{n_N}\}$, an intersection node set $S_D = \{D_1, \ldots, D_{n_D}\}$ and the neighboring node set $S^N(D_m)$ of every intersection node $D_m, m = 1, \ldots, n_D$, where N_l and D_m are subsets of $\{X_1, X_2, \ldots, X_n\}$. Each intersection node is connected to all clique nodes in its neighboring node set with arcs in an EJG.

The typical type of joint distribution represented by EJGs and JGs is shown as follows.

$$P(x_1, \ldots, x_n) = \alpha q(N_1) q(N_2) \cdots q(N_{n_N}). \quad (5)$$

First, restricted intersection nodes (r.i.n.) are defined. If the element of an intersection node is equivalent to a restricted random variable as $D_m = \{X_j\}, j \in I_C$, the intersection node is called a restricted intersection node. If there does not exist an intersection node satisfying $D_m = \{X_j\}, j \in I_C$, the restricted intersection node corresponding to every such restricted random variable X_j is produced and connected to an arbitrary clique node N_l satisfying $X_j \in N_l$.

We explain one algorithm in the proposed parallel propagation algorithms on EJGs. The message from a clique node N_i to a clique node N_l connected with it by way of an intersection node D_m is calculated by

$$\mu_{t+1}^{i \to l} = \begin{cases} \frac{P^*(N_i)}{\mu_t^{l \to i}} & \text{if } D_m \text{ is a r.i.n.} \\ \sum_{x \notin D_m} q(N_i) \prod_{N_k \in S_l^N(N_i)} \mu_t^{k \to i} & \text{otherwise,} \end{cases} \quad (6)$$

where $S_l^N(N_i) = \{N_k \in S^N(D_h) | D_h \in S^D(N_i), D_h \notin S^D(N_l), k \neq i\}$.

The marginal posterior probability of each random variable is calculated by

$$P_t(N_l) = q(N_l) \prod_{N_k \in S^N(N_l)} \mu_t^{k \to l} \quad (7)$$

where $S^N(N_l) = \{N_k \in S^N(D_h) | D_h \in S^D(N_l), k \neq l\}$.

Theorem 1 *If an EJG has no loops, the proposed algorithm halt and calculate the exact marginal generalized posterior distributions on the EJG.*

This algorithm is interpreted as an extended algorithm of the sum-product algorithm or a full parallel type of the generalized distribution low(GDL) algorithm[1 Another algorithm in the proposed parallel propagation algorithms is interpreted as a full parallel type of the HUGIN algorithm[6].

3.2 Application of the parallel propagation algorithms to decoding convolutional codes

We apply the parallel algorithms to the JGs representing convolutional codes. Bayesian Networks(BNs) and factor graphs use state nodes or auxiliary nodes for representing convolutional codes, because of avoiding loops on the graphs. However, JGs do not need such auxiliary nodes for representing non-recursive convolutional codes. The complexity of the JGs is generally lower than that of the graph using state nodes such as BNs or factor graphs.

4 A comparison between the parallel algorithm and the BCJR algorithm

4.1 The parallel algorithm for zero-tail convolutional codes

Since the JGs of convolutional codes have no loop, the value of each symbol calculated by the algorithm converges to the correct posterior probability from Theorem 1. The BCJR algorithm needs the iteration whose number is the same as the code length before it halts.

In the case where the parallel algorithm calculates approximate values with some suitable precision, the number of the iteration for the calculation is the order of the constraint length of the convolutional code.

We evaluate the number of iterations until the parallel algorithm achieves the bit error rate that is the same as that of the BCJR algorithm by some simulation. The evaluated convolutional codes are rate $R = 1/2, 1/3$ and constraint length $v = 2, 3, 4$ and code length $N = 40, 100, 200$. We assume white Gaussian channels. For example, the generator matrix of the rate $R = 1/2$ and constraint length $v = 2$ convolutional code is $G = [1 + D^2, 1 + D + D^2]$. The means of the number of iterations for the rate $R = 1/2$ codes are shown by Table 1.

The simulation and the numerical analysis stated in the next section show that the iteration number for achieving almost the same bit error probability as that the MAP decoding achieves is 2 or 3 times of the constraint length of codes. So the parallel computational complexity of the parallel propagation algorithm is very low, while the total computational complexity is higher than that of the BCJR algorithm.

Table 1: The mean of the number of iterations

R=1/2	N=40	100	200
v=2	4.0	4.7	5.1
3	6.3	6.6	7.0
4	8.4	8.7	9.1

4.2 The parallel algorithm for tailbiting convolutional codes

The parallel algorithm can be directly applied to tail-biting convolutional codes, because the algorithm does not need tails or leaf nodes in the graphs. From the result of the previous section, in the parallel algorithm, almost all nodes except the nodes near the tails do not need the information about the initial and the final state of the tails for decoding. So we expect the parallel algorithms can achieve almost the same bit error probability given by the MAP decoding for tailbiting convolutional codes.

Fig 1 shows the comparison between the parallel decoding algorithm and the tailbiting BCJR algorithm. From our experimental results for several tail-biting convolutional codes, the bit error rate of the parallel decoding algorithm is lower than that of the tailbiting BCJR algorithm.

The parallel computational complexity of the parallel propagation algorithm is lower than that of the tailbiting BCJR algorithm. By combining the parallel algorithm and sequential message propagation strategy, we can reduce the total computational complexity of the parallel propagation algorithm.

5 Numerical analysis for evaluating the performance of the algorithm

We can evaluate the performance of the parallel algorithms for convolutional codes by a numerical analysis that is similar to the density evolution method[14]. We want to analyze the density of out-put messages or the joint posterior probability of symbols in each intersection node, while the density of the posterior probability of individual symbol is analyzed in the density evolution method for LDPC codes. So we introduce a special representation method for the joint posterior probability of symbols in an intersection node.

We explain the representation method by using the following example. Let an intersection node include two symbols X_1, X_2. The joint probability of x_1, x_2 is represented by the following formula:

$$\log P(x_1, x_2) = \log \pi_{00} + x_1 \log \pi_{10} + x_2 \log \pi_{01} + x_1 x_2 \log \pi_{11} \tag{8}$$

where

$$\begin{aligned} \pi_{00} &= P(x_1 = 0, x_2 = 0), \\ \pi_{10} &= \frac{P(x_1 = 1, x_2 = 0)}{P(x_1 = 0, x_2 = 0)}, \\ \pi_{01} &= \frac{P(x_1 = 0, x_2 = 1)}{P(x_1 = 0, x_2 = 0)}, \\ \pi_{11} &= \frac{P(x_1 = 0, x_2 = 0) P(x_1 = 1, x_2 = 1)}{P(x_1 = 0, x_2 = 1) P(x_1 = 1, x_2 = 0)}. \end{aligned} \tag{9}$$

We evaluate the density of π_{10}, π_{01} and π_{11} by following the procedure of the parallel algorithm. The density of the sum of some random variables is given by the convolution of the distribution of the random variables. The convolutions can be efficiently computed using Fourier transform. Since the JGs representing convolutional codes are constructed by the iteration of a simple and symmetric structure, the density of the posterior probability can be calculated by the same iteration.

We can evaluate the symbol error probability of the parallel algorithm for a convolutional code in an arbitrary iteration by using the above density. The error probabilities are monotone decreasing with respect to the number of iteration. The error probability evaluates not only the performance of the parallel algorithm but also the performance of the convolutional code itself. The induced error probabilities are the upper bounds of the error probability of the MAP decoding for the zero-tail and tailbiting convolutional codes.

6 Conclusion

We applied the parallel propagation algorithms to the decoding of zero-tail and tail-biting convolutional codes. We evaluated the performance of the parallel algorithm by numerical analysis and some simulation. Our experimental results show the bit error probability

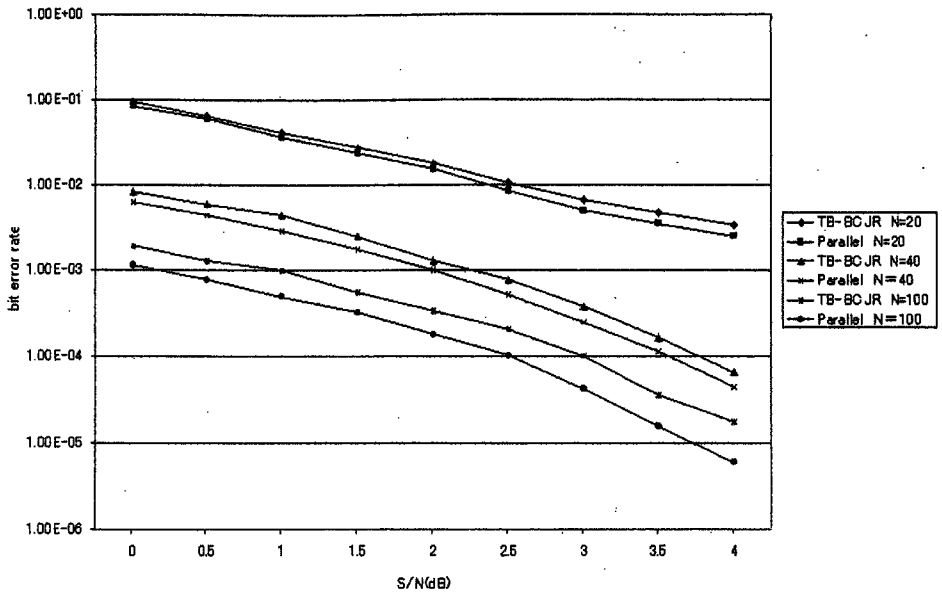


Figure 1: The bit error rate of the parallel algorithm and the tb-BCJR algorithm

of the parallel algorithm is lower than that of the tail-biting BCJR algorithm. The parallel computational complexity of the algorithm is also very low.

Although we could not explain the following extension of the parallel algorithm due to limitation of space, by exchanging sum-operator to max-operator in the parallel algorithm, we proposed a parallel algorithm for minimizing block error probability for convolutional codes. The parallel algorithm is interpreted as a full parallel version of the Viterbi algorithm. We also verified good performance of the parallel algorithm by some simulation.

Acknowledgments

One of the authors, Toshiyasu Matsushima, would like to thank Prof. Marc Fossorier for his precious suggestion. This research is partially supported by JSPS with the Grants-in-Aid for Scientific Research (C) No.15560338.

References

[1] S.M. Aji and R.J. McEliece, *The Generalized Distributive Law*, IEEE Trans. IT, Vol.46 No.2, 2000.

[2] J.B. Anderson and S.M. Hladik, *Tailbiting MAP Decoders*, IEEE Journal on Selected Areas in Communication, Vol.16, Feb. 1998.

[3] L.R. Bahl, J. Cocke, F. Jelinek and J. Raviv, *Optimal decoding of linear codes for minimizing symbol error rate*, IEEE Trans. IT, Vol.20, Mar. 1974.

[4] G.D. Forney, Jr, *Codes on Graphs: Normal Realizations*, IEEE Trans. IT, Vol.47 No.2, 2001.

[5] J. Hagenauer, E. Offer and L. Papke, *Iterative decoding of binary block and convolutional codes*, IEEE Trans. IT, Vol.42, 1996.

[6] F.V. Jensen, K.G. Olesen and S.K. Andersen, *An Algebra of Bayesian Belief Universes for Knowledge-Based System*, Networks, Vol.20, 1990.

[7] F.R. Kschischang, B.J. Fey and H. Loeliger, *Factor Graphs and the Sum-Product Algorithm*, IEEE Trans. IT, Vol.47 No.2, 2001.

[8] R.J. McEliece, D.J.C. MacKay and J. Cheng, *Turbo decoding as an instance of Pearl's "Belief Propagation"*, IEEE J. Sel. Areas Commun., Vol.16 No.2, 1998.

[9] T. Matsushima, T.K. Matsushima and S. Hirasawa, *An Iterative Algorithm for Calculating Posterior Probability and Model Representation* , Proceedings of IEEE Int. Symp. on Information Theory, 2001.

[10] T. Matsushima, T.K. Matsushima and S. Hirasawa, *An Alternative Algorithm for Calculating Posterior Probability and Decoding*, Proceedings of IEEE Int. Symp. on Information Theory, 2002.

[11] T. Matsushima, T.K. Matsushima and S. Hirasawa, *Calculation of Generalized Posterior Distribution on Junction Graphs*, Proceedings of the 25th Symposium on Information Theory and Its Applications, 2002.

[12] M. Osawa, R. Nomura and T. Matshushima, *A study of analysis of alternate decoding algorithm of convoluteional codes* Technical Report IEICE, IT, Jun 2003(Japanese).

[13] J. Pearl, *Probabilistic reasoning in intelligent systems* Morgan Kaufmann, 1988.

[14] T.J. Richardson, R.L. Urbanke, *The capacity of low-density parity-check codes under message-passing decoding*, IEEE Trans. IT, Vol.47 No. 2, 2001.

[15] U. Seino, *High speed decoding by parallel probability propagation*, Master ' s Thesis, Waseda Univ., 2002(Japanese).

The 26th Symposium on Information Theory
and Its Applications (SITA2003)
Higashiura, Hyogo, Japan, Dec. 15–18, 2003

A Method for Reducing Space Complexity of Reliability based Heuristic Search Maximum Likelihood Decoding Algorithms

Hideki YAGI * Toshiyasu MATSUSHIMA * Shigeichi HIRASAWA *

Abstract— In this paper, reliability-based heuristic search methods for maximum likelihood decoding of block codes are considered. Based on the decoding algorithm by Battail and Fang (and its improved technique by Valembois and Fossorier), we deduce a method of reducing the space complexity of the heuristic search maximum likelihood decoding algorithm. The proposed method is applicable to the heuristic search method with a certain class of evaluation functions. Simulation results show the efficiency of the decoding algorithm adopting the proposed method.

Keywords— maximum likelihood decoding, binary block codes, heuristic search, most reliable basis, reliability

1 Introduction

Maximum likelihood decoding (MLD) of block codes minimizes the probability of decoding error when we assume that all codewords have the equal probability to be transmitted. Since the complexity of searching the maximum likelihood (ML) codeword among all codewords is significantly large, many researcher have devoted to develop efficient algorithms of MLD. One of the most efficient MLD algorithms is the reliability based decoding algorithm that uses the column permuted generator matrix in increasing order of reliability.

In this paper, we consider heuristic search MLD algorithms where candidate codewords are generated in increasing value of the evaluation function. G. Battail and J. Fang have proposed a priority-first search method for MLD where the most simple (and primitive) evaluation functions are employed [1] (we will call this method the BF decoding algorithm). Recently, A. Valembois and M. Fossorier have proposed a technique for reducing the space complexity in the BF decoding algorithm with the same class of evaluation functions [4]. Since this kind of evaluation functions employed by both decoding algorithms are the most simple ones, we need to generate more candidate codewords than in the decoding algorithm with more sophisticated ones [2, 3, 5]. We can modify the BM decoding algorithm to perform priority-first MLD even when we use sophisticated evaluation functions [6], however, we cannot adopt the improved technique by Valembois et al. in this case.

In this paper, we propose a method for reducing the space complexity of the priority-first MLD algorithm with effective evaluation functions, which are presented in [3, 5]. Consequently, we show, by computer simulations, that space complexity of the decoding algorithm employing the proposed method is significantly reduced.

2 Reliability based MLD Algorithm

Let $\mathcal{C}$ be a binary linear (n, k, d_{H}) block code of the code length n, the number of information symbols k and the minimum distance d_{H}. We denote a generator matrix of $\mathcal{C}$ by G and the weight profile of $\mathcal{C}$ by $W_{\mathrm{H}}(\mathcal{C})$. We assume any codewords $\boldsymbol{c} = (c_1, c_2, \dots, c_n) \in \{0,1\}^n$ of $\mathcal{C}$ are transmitted over Additive White Gaussian Noise (AWGN) channel. A receiver demodulates a received sequence $\boldsymbol{r} = (r_1, r_2, \dots, r_n) \in \mathcal{R}^n$ into a sequence $\boldsymbol{\theta} = (\theta_1, \theta_2, \dots, \theta_n)$, $\theta_j = \ln \frac{P(r_j|c_j=0)}{P(r_j|c_j=1)}$, where $P(r_j|c_j)$ represents the likelihood of symbol c_j, and inputs it into a soft-decision decoder[1]. Furthermore, a hard-decision sequence $\boldsymbol{z} = (z_1, z_2, \dots, z_n) \in \{0,1\}^n$ is obtained by setting $z_j = 0$ if $\theta_j \geq 0$ and $z_j = 1$ otherwise. The soft-decision decoder estimates the transmitted codeword from $\boldsymbol{\theta}$ and $\boldsymbol{z}$, and output the estimated codeword at the end of decoding.

In reliability-based decoding algorithms, we first find *the most reliable and linearly independent* (MRI) k positions. Then, we permute columns of a generator matrix over MRI positions in increasing value of reliability. The rest of columns are also reordered in increasing value of reliability. We perform the standard row operation with respect to the reordered matrix to make the leftmost k columns the identity matrix. We denote the resultant matrix by $\tilde{G}$.

Let $\tilde{\boldsymbol{\theta}} = (\tilde{\theta}_1, \tilde{\theta}_2, \dots, \tilde{\theta}_n)$ and $\tilde{\boldsymbol{z}} = (\tilde{z}_1, \tilde{z}_2, \dots, \tilde{z}_n)$ be permuted sequences of $\boldsymbol{\theta}$ and $\boldsymbol{z}$, respectively, in the same ordering of columns of $\tilde{G}$. We denote the equivalent code to $\mathcal{C}$ by $\tilde{\mathcal{C}}$, whose codewords are generated by $\tilde{G}$. Let $\boldsymbol{u} = (u_1, u_2, \dots, u_k) \in \{0,1\}^k$ be the leftmost k symbols of $\tilde{\boldsymbol{z}}$, i.e., $u_j = \tilde{z}_j, 1 \leq \forall j \leq k$. The decoder first encodes $\boldsymbol{u}$ by $\tilde{G}$ to obtain the initial codeword $\tilde{\boldsymbol{c}}_{\emptyset} (= \boldsymbol{u}\tilde{G})$. Afterwards, k dimensional vectors, called test error patterns, are iteratively generated and encoded by $\tilde{G}$. For a location set $J \subseteq [1, k]$, the test error pattern $\boldsymbol{t}_J = (t_{J,1}, t_{J,2}, \dots, t_{J,k}) \in \{0,1\}^k$ is determined by setting $t_{J,j} = 1$ if $j \in J$ and $t_{J,j} = 0$ otherwise (J is called the support of $\boldsymbol{t}_J$). Then, $\tilde{\boldsymbol{c}}_J = \tilde{\boldsymbol{c}}_{\emptyset} \oplus \boldsymbol{t}_J\tilde{G}$ is a candidate codeword and this procedure is repeated until a sufficient condition for the ML codeword is satisfied[2].

For a binary vector $\boldsymbol{v} = (v_1, v_2, \dots, v_n) \in \{0,1\}^n$, we define the correlation discrepancy [4] of $\boldsymbol{v}$ as

$$L(\boldsymbol{v}) = \sum_{j|v_j \neq \tilde{z}_j} |\tilde{\theta}_j|. \tag{1}$$

Then $\tilde{\boldsymbol{c}}_{\mathrm{best}}$ is the ML codeword if and only if $L(\tilde{\boldsymbol{c}}_{\mathrm{best}}) = \min_{\tilde{\boldsymbol{c}} \in \tilde{\mathcal{C}}} L(\tilde{\boldsymbol{c}})$.

3 Battail-Fang Decoding Algorithm

Battail et al. have presented a method for generating test error patterns in increasing value of the evaluation function defined as

$$\Delta(\boldsymbol{t}_J) = \sum_{j \in J} |\tilde{\theta}_j|. \tag{2}$$

Let F be arbitrary evaluation function. If F satisfies the following two conditions, the BF decoding algorithm performs priority-first search of test error patterns [4].

(C1) $F(\boldsymbol{t}_J) \leq F(\boldsymbol{t}_{J \cup \{j_m\}})$, for $j_m \notin J$.

(C2) $F(\boldsymbol{t}_J) \leq F(\boldsymbol{t}_{J'}) \Rightarrow F(\boldsymbol{t}_{J \cup \{j_m\}}) \leq F(\boldsymbol{t}_{J' \cup \{j_m\}})$, for $j_m \notin J$ and $j_m \notin J'$.

The function Δ actually satisfies (C1) and (C2).

* Dept. of Industrial and Management Systems Engineering, Waseda University, 3-4-1 Ohkubo Shinjuku-ku, Tokyo, 169-8555 Japan, E-mail: yagi@hirasa.mgmt.waseda.ac.jp

[1] Since the probability of decision error of z_j becomes smaller as the value of $|\theta_j|$ is larger, $|\theta_j|$ is called reliability.

[2] $\oplus$ represents Exclusive OR operation.

Consider k lists of test error patterns $M_1, M_2, \ldots, M_k$. The test error pattern $\boldsymbol{t}_J$ with a support J such that $J \subseteq \{1, 2, \ldots, j_m\}$ and $j_m \in J$ is supposed to be in M_{j_m}[3]. Then for any test error pattern $\boldsymbol{t}_J$ such that $J \neq \emptyset$, the list for storing it is uniquely determined. In a list $M_j, 1 \leq \forall j \leq k$, test error patterns are ordered in increasing value of an evaluation function F.

By the condition (C1), the test error pattern with the minimum value of F in $M_j, 1 \leq \forall j \leq k$, is $\boldsymbol{t}_{\{j\}}$ with the Hamming weight one. For $j = 1, 2, \ldots, k$, we set $M_j = \{\boldsymbol{t}_{\{j\}}\}$ after the initial codeword $\tilde{\boldsymbol{c}}_\emptyset$ is obtained. Then, the decoder searches the test error pattern with the minimum value of F (we will call this pattern the *best pattern*) among the set of ones which have not been found. In the decoding algorithm described below, for any candidate codeword $\tilde{\boldsymbol{c}}_J (= \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}_J \tilde{G})$ given by $\boldsymbol{t}_J$, we assume

$$F(\boldsymbol{t}_J) \leq L(\tilde{\boldsymbol{c}}_J). \tag{3}$$

Furthermore, for a test error pattern $\boldsymbol{t}_J \in M_{j_m}$, we call $\boldsymbol{t}_{J\cup\{j\}}, \forall j > j_m$, the *extended pattern* of $\boldsymbol{t}_J$.

[The BF decoding algorithm]

S1) Set $\tilde{\boldsymbol{c}}_{\rm best} := \tilde{\boldsymbol{c}}_\emptyset$ and $\underline{L} := L(\tilde{\boldsymbol{c}}_\emptyset)$.

S2) Choose the best pattern $\boldsymbol{t}_J \in M_{j_m}$ among the topmost test error patterns in non-empty lists $M_j, 1 \leq j \leq k$. If $F(\boldsymbol{t}_J) \geq \underline{L}$, then output $\tilde{\boldsymbol{c}}_{\rm best}$ and halt the algorithm.

S3) Generate the next candidate codeword by $\tilde{\boldsymbol{c}}_J := \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}_J \tilde{G}$. If $L(\tilde{\boldsymbol{c}}_J) < \underline{L}$, then set $\underline{L} := L(\tilde{\boldsymbol{c}}_J)$ and $\tilde{\boldsymbol{c}}_{\rm best} := \tilde{\boldsymbol{c}}_J$.

S4) At the end of all lists $M_j, \forall j > j_m$, store the extended pattern $\boldsymbol{t}_{J\cup\{j\}}$. Delete $\boldsymbol{t}_J$ from M_{j_m}.

S5) If $M_j = \emptyset$ for all $j = 1, 2, \ldots, k$, then output $\tilde{\boldsymbol{c}}_{\rm best}$ and halt the algorithm. Otherwise, go to S2). □

Valembois et al. have proposed an improved method for choosing the best pattern at S2) where we cost at most $\lceil \log_2 k \rceil$ comparison operations in [4][4]. First, we prepare a binary tree with k leaves where each leaf is allocated to one list M_j. Let the evaluation value of a leaf be that of the topmost test error pattern in M_j. Each node represents one of its successor nodes with the smaller evaluation value. By adopting this relation from leaves to the root node recursively, the root node represents the list whose topmost test error pattern is best. We need $O(k)$ comparison operations to construct this initial tree, however, at most $\lceil \log_2 k \rceil$ comparison operations are only needed in step S2). In the following, the BF decoding algorithm indicates the algorithm with this improved technique.

By (3), the inequality at S2), $F(\boldsymbol{t}_J) \geq \underline{L}$, represents a sufficient condition that $\tilde{\boldsymbol{c}}_{\rm best}$ is the ML codeword. If a tighter sufficient condition for optimality is employed, we only need to generate less candidate codewords to perform MLD. The evaluation function in [3, 5] is more effective than Δ in the sense that it can be a tighter sufficient condition.

For $\forall \tilde{\boldsymbol{c}} \in \tilde{C}$, let $\tilde{\boldsymbol{c}}^{\rm L}$ and $\tilde{\boldsymbol{c}}^{\rm R}$ be the leftmost k symbols and the rightmost $n-k$ symbols of $\tilde{\boldsymbol{c}}$, respectively, i.e., $\tilde{\boldsymbol{c}} = \tilde{\boldsymbol{c}}^{\rm L} \circ \tilde{\boldsymbol{c}}^{\rm R}$[5]. For some $\tilde{\boldsymbol{c}}_{\rm rcf} (= \tilde{\boldsymbol{c}}^{\rm L}_{\rm rcf} \circ \tilde{\boldsymbol{c}}^{\rm R}_{\rm rcf}) \in \tilde{C}$, we define

$$T(\boldsymbol{t}_J, \tilde{\boldsymbol{c}}_{\rm rcf}) = \Big\{ (\boldsymbol{t}_J \oplus \boldsymbol{u}) \circ \boldsymbol{v} \,\Big|\, \boldsymbol{v} \in \{0,1\}^{n-k}, \text{ and } w_{\rm H}(\boldsymbol{t}_J) + d_{\rm H}(\tilde{\boldsymbol{c}}^{\rm R}_{\rm rcf}, \boldsymbol{v}) \in W_{\rm H}(\tilde{C}) \Big\}, \tag{4}$$

where $w_{\rm H}(\cdot), d_{\rm H}(\cdot,\cdot)$ are the Hamming weight and the Hamming distance. Then the evaluation function in [3, 5] is defined as,

$$f(\boldsymbol{t}_J, \tilde{\boldsymbol{c}}_{\rm rcf}) = \min_{\boldsymbol{v} \in T(\boldsymbol{t}_J, \tilde{\boldsymbol{c}}_{\rm rcf})} \Big\{ L(\boldsymbol{v}) \Big\}. \tag{5}$$

Since $L(\boldsymbol{v}) = \Delta(\boldsymbol{t}_J) + \sum_{j=k+1}^{n} (\tilde{z}_j \oplus v_j) |\tilde{\theta}_j| \geq \Delta(\boldsymbol{t}_J)$ for $\forall \boldsymbol{v} \in T(\boldsymbol{t}_J, \tilde{\boldsymbol{c}}_{\rm rcf})$, f can give a tighter sufficient condition for optimality than Δ. Remark that f satisfies (C1) but does not necessarily satisfy (C2) [6]. Therefore, when we store an extended pattern into a list at S4), we need to insert it at the position such that the list remains in increasing value of the evaluation function. In this case, we modify S4) such as

S'4) For all lists M_j such that $j > j_m$, insert the extended patterns $\boldsymbol{t}_{J\cup\{j\}}$ at the position such that the list remains increasing order. Delete $\boldsymbol{t}_J$ from M_{j_m}.

By this modification, the priority-first search of the BF decoding algorithm is maintained [4].

We here describe the complexity of the BF decoding algorithm. In a decoding procedure of a received sequence $\boldsymbol{r}$, the space complexity is $O(k \times M(\boldsymbol{r}))$ where $M(\boldsymbol{r})$ represents the maximum number of test error patterns stored in lists. As for the time complexity, the number of generating test error patterns is dominant as well as the number of encoding them.

4 Proposed Decoding Algorithm

In this section, we propose a method for reducing the space complexity of the BF decoding algorithm with evaluation functions that do not satisfy the condition (C2).

We here define a condition of a evaluation function F.

Definition (C3) For $\forall J \subseteq [1, k]$, if $j_1, j_2 \notin J$ and $1 \leq j_2 < j_1 \leq k$, then a function F satisfies

$$F(\boldsymbol{t}_{J\cup\{j_2\}}) \geq F(\boldsymbol{t}_{J\cup\{j_1\}}). \tag{6}$$

□

For the function f, we show the following lemma.

Lemma 1 The evaluation function f satisfies the condition (C1) and (C3).

(**Proof**) We first show that the function Δ satisfies (C3). By (2), for $\boldsymbol{t}_J$ such that $j_1 \notin J$, $\Delta(\boldsymbol{t}_{J\cup\{j_1\}}) = \sum_{j\in J\cup\{j_1\}} |\tilde{\theta}_j| = \Delta(\boldsymbol{t}_J) + |\tilde{\theta}_{j_1}|$. Since $|\tilde{\theta}_j| \geq |\tilde{\theta}_{j+1}|$ for $1 \leq j \leq k-1$, if $1 \leq j_2 < j_1$ and $j_2 \notin J$, then

$$\begin{aligned}\Delta(\boldsymbol{t}_{J\cup\{j_2\}}) &= \sum_{j\in J\cup\{j_2\}} |\tilde{\theta}_j| + |\tilde{\theta}_{j_1}| - |\tilde{\theta}_{j_1}| \\ &= \Delta(\boldsymbol{t}_{J\cup\{j_1\}}) + |\tilde{\theta}_{j_2}| - |\tilde{\theta}_{j_1}| \geq \Delta(\boldsymbol{t}_{J\cup\{j_1\}}).\end{aligned}$$

Therefore, the function Δ satisfies (C3).

For $\boldsymbol{t}_J$ and $\tilde{\boldsymbol{c}}_{\rm rcf} \in \tilde{C}$, let $\boldsymbol{v}^* = (v_1^*, v_2^*, \ldots, v_n^*)$ be such that $f(\boldsymbol{t}_J, \tilde{\boldsymbol{c}}_{\rm rcf}) = L(\boldsymbol{v}^*)$. Then by (1) and (5),

$$f(\boldsymbol{t}_J, \tilde{\boldsymbol{c}}_{\rm rcf}) = L(\boldsymbol{v}^*) = \Delta(\boldsymbol{t}_J) + \sum_{j=k+1}^{n} (\tilde{z}_j \oplus v_j^*) |\tilde{\theta}_j|. \tag{7}$$

The summation of the right hand side of (7) depends only on Hamming weight of $\boldsymbol{t}_J$. Therefore, if $j_1, j_2 \notin J$ and $1 \leq j_2 < j_1$, then

$$f(\boldsymbol{t}_{J\cup\{j_2\}}) - f(\boldsymbol{t}_{J\cup\{j_1\}}) = \Delta(\boldsymbol{t}_{J\cup\{j_2\}}) - \Delta(\boldsymbol{t}_{J\cup\{j_1\}}), \tag{8}$$

and this implies that the function f satisfies (C3). □

In the following, we consider evaluation functions that satisfy both (C1) and (C3).

[3] i.e., $j_m = \max J$ if $\boldsymbol{t}_J \in M_{j_m}$.

[4] In [4], a method for reducing the space complexity of the BF decoding algorithm with evaluation functions (Δ) satisfying condition (C1) and (C2) has been proposed, however we do not describe it in details here.

[5] $\circ$ represents concatenation of vectors.

The strategy of the proposed method is like *lazy evaluation* where any test error patterns are not generated as long as possible. This approach is similar to an improved method in [4]. We first consider k lists M_j as in the BF decoding algorithm. By the condition (C1), the best pattern among all test error patterns in a list $M_j, 1 \leq \forall j \leq k$, is $\boldsymbol{t}_{\{j\}}$. Furthermore, by the condition (C3), the best pattern among k test error patterns $\boldsymbol{t}_{\{j\}}, 1 \leq \forall j \leq k$, is $\boldsymbol{t}_{\{k\}}$. Therefore, we construct the initial lists as

$$M_j = \begin{cases} \{\boldsymbol{t}_{\{j\}}\}, & \text{if } j = k; \\ \emptyset, & \text{otherwise,} \end{cases} \tag{9}$$

Similar to an improved method of [4], if the proposed method uses a binary tree whose leaves correspond to k lists, no comparison operations are needed to construct the initial tree.

At S2) of the BF decoding algorithm, if $\boldsymbol{t}_J \in M_{j_m}$ is chosen as the best pattern, $k - j_m$ extended pattern of $\boldsymbol{t}_J$ will be stored at S4). However, it is enough to store only its extended pattern $\boldsymbol{t}_{J\cup\{k\}}$ in the list M_k, since the condition (C3) guarantees $F(\boldsymbol{t}_{J\cup\{j\}}) \geq F(\boldsymbol{t}_{J\cup\{k\}})$ for $j < k$.

Following this modification, we need to determine when other extended patterns $\boldsymbol{t}_{J\cup\{j\}}, j < k$, are inserted into lists. Assume that a test error pattern $\boldsymbol{t}_{J\cup\{j_m\}}, j_m \notin J$, is stored in the list M_{j_m} during a decoding procedure. Then extended patterns $\boldsymbol{t}_{J\cup\{j\}}, j < j_m$, cannot be the best pattern at S2), since the condition (C3) guarantees $F(\boldsymbol{t}_{J\cup\{j\}}) \geq F(\boldsymbol{t}_{J\cup\{j_m\}})$. Therefore these extended patterns need to be stored only after $\boldsymbol{t}_{J\cup\{j_m\}}$ is chosen as the best pattern at S2).

Assume that $\boldsymbol{t}_{J\cup\{j_m\}}$ is chosen as the best pattern at S2). By the condition (C3), if $j_m - 1 > \max J$, the extended pattern $\boldsymbol{t}_{J\cup\{j_m-1\}}$ has the smallest value of F next to $\boldsymbol{t}_{J\cup\{j_m\}}$ among extended patterns, i.e.,

$$F(\boldsymbol{t}_{J\cup\{j_m-1\}}) = \min_{\boldsymbol{t}_{J\cup\{j\}}} \left\{ F(\boldsymbol{t}_{J\cup\{j\}}) \,\middle|\, j \notin J, j < j_m \right\}. \tag{10}$$

Therefore, after choosing $\boldsymbol{t}_{J\cup\{j_m\}}$ as the best pattern at S2), $\boldsymbol{t}_{J\cup\{j_m-1\}}$ is inserted into the list M_{j_m-1}. This modification reduces the space complexity significantly. Note that the next candidate pattern $\boldsymbol{t}_{J\cup\{j_m-1\}}$ is easily obtained from the best pattern $\boldsymbol{t}_{J\cup\{j_m\}}$.

We describe a decoding algorithm employing the above method. Before the following algorithm is performed, the decoder constructs the initial binary tree.

[The proposed decoding algorithm]

P1) Set $\tilde{\boldsymbol{c}}_{\text{bcst}} := \tilde{\boldsymbol{c}}_\emptyset$ and $\underline{L} := L(\tilde{\boldsymbol{c}}_\emptyset)$.

P2) Choose the best pattern $\boldsymbol{t}_J \in M_{j_m}$ from non-empty lists. If $F(\boldsymbol{t}_J) \geq \underline{L}$, then output $\tilde{\boldsymbol{c}}_{\text{bcst}}$ and halt the algorithm.

P3) Generate the next candidate codeword by $\tilde{\boldsymbol{c}}_J := \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}_J\tilde{G}$. If $L(\tilde{\boldsymbol{c}}_J) < \underline{L}$, then set $\underline{L} := L(\tilde{\boldsymbol{c}}_J)$ and $\tilde{\boldsymbol{c}}_{\text{bcst}} := \tilde{\boldsymbol{c}}_J$.

P4) a) If $j_m - 1 \notin J$, then insert $\boldsymbol{t}_{J'\cup\{j_m-1\}}$ into the list M_{j_m-1} where $J' = J \setminus \{j_m\}$.

b) If $j_m \neq k$, then insert $\boldsymbol{t}_{J\cup\{k\}}$ into the list M_k. Delete $\boldsymbol{t}_J$ from M_{j_m}.

P5) If $M_j = \emptyset$ for $\forall j = 1, 2, \ldots, k$, then output $\tilde{\boldsymbol{c}}_{\text{bcst}}$ and halt the algorithm. Otherwise, go to S2). □

The step P4) corresponds to the above modification.

We show the validity of the proposed decoding algorithm.

Theorem 1 Assume that an evaluation function F satisfies both (C1) and (C3). During a decoding procedure, if $\boldsymbol{t}_J$ is the best among the set of all test error patterns which has not been chosen as the best pattern, then such $\boldsymbol{t}_J$ has been already generated and stored in the list M_{j_m} such that $j_m = \max J$.

(**Proof**) We first consider the following two cases.

(1) In the case of $\boldsymbol{t}_J \in M_k$.

By the condition (C1), $\boldsymbol{t}_{J'}$ such that $J' = J\setminus\{k\}$ satisfies $F(\boldsymbol{t}_{J'}) \leq F(\boldsymbol{t}_J)$. Therefore, when we assume $F(\boldsymbol{t}_{J'})$ has been chosen as the best pattern at P2), then $\boldsymbol{t}_J$ has been into the list M_k at P4-b).

(2) In the case of $\boldsymbol{t}_J \in M_{j_m}, j_m \neq k$.

From the condition (C3), $\boldsymbol{t}_{J'}$ such that $J' = J\setminus\{j_m\} \cup \{j_m+1\}$ satisfies $F(\boldsymbol{t}_{J'}) \leq F(\boldsymbol{t}_J)$. When we assume $F(\boldsymbol{t}_{J'})$ has been chosen as the best pattern at P2), then $\boldsymbol{t}_J$ has been stored in a list M_{j_m} at P4-a).

Since the first test error pattern $\boldsymbol{t}_{\{k\}}$ has been generated when initial lists has been constructed by (9), the assumptions of (1) and (2) are satisfied by mathematical induction. □

When an extended pattern is inserted in a list at P4), sorting is needed to keep the list in increasing value of F and this complexity may be large[6]. In the following, if a evaluation function F satisfies a certain condition, we show that the time complexity is reduced.

Definition (C4) For $J, J' \subseteq [1, k]$ such that $j_1, j_2 \notin J \cup J'$ and $1 \leq j_2 < j_1 \leq k$, a function F satisfies,

$$F(\boldsymbol{t}_{J\cup\{j_1\}}) \leq F(\boldsymbol{t}_{J'\cup\{j_1\}}),$$
$$\Rightarrow F(\boldsymbol{t}_{J\cup\{j_2\}}) \leq F(\boldsymbol{t}_{J'\cup\{j_2\}}). \tag{11}$$

□

Assume that a function F satisfies (C4). When a test error pattern is inserted into M_k at P4-b), we need sorting to keep a list in increasing value of F. However, during a decoding procedure, the best pattern $\boldsymbol{t}_{J\cup\{j_m\}}$ such that $j_m \notin J$ is chosen from M_{j_m}, it is enough to store $\boldsymbol{t}_{J\cup\{j_m-1\}}$ at the end of M_{j_m-1} from the condition (C4). In case that the proposed decoding algorithm employs a function that satisfies (C4), P4) can be modified as follows:

P'4) a) If $j_m - 1 \notin J$, then store $\boldsymbol{t}_{J'\cup\{j_m-1\}}$ at the end of M_{j_m-1} where $J' = J \setminus \{j_m\}$.

b) If $j_m \neq k$, then insert $\boldsymbol{t}_{J\cup\{k\}}$ into the list M_k. Delete $\boldsymbol{t}_J$ from M_{j_m}.

For the function f, we show the following lemma.

Lemma 2 The evaluation function f satisfies (C4).

(**Proof**) By (8), the following equation holds.

$$f(\boldsymbol{t}_{J\cup\{j_2\}}) - f(\boldsymbol{t}_{J\cup\{j_1\}}) = f(\boldsymbol{t}_{J'\cup\{j_2\}}) - f(\boldsymbol{t}_{J'\cup\{j_1\}}). \tag{12}$$

By transposing (12),

$$f(\boldsymbol{t}_{J\cup\{j_1\}}) - f(\boldsymbol{t}_{J'\cup\{j_1\}}) = f(\boldsymbol{t}_{J\cup\{j_2\}}) - f(\boldsymbol{t}_{J'\cup\{j_2\}}). \tag{13}$$

Then $f(\boldsymbol{t}_{J\cup\{j_2\}}) \leq f(\boldsymbol{t}_{J'\cup\{j_2\}})$ if and only if $f(\boldsymbol{t}_{J\cup\{j_1\}}) \leq f(\boldsymbol{t}_{J'\cup\{j_1\}})$. □

If the function f is employed by the proposed decoding algorithm, P'4) instead of P4) can be used. This saves the time complexity for sorting.

In terms of the time and space complexity of the proposed decoding algorithm, we show the following theorems.

Theorem 2 The proposed decoding algorithm achieves MLD. Then, the maximum list size in the proposed decoding algorithm is less than that of the BF decoding algorithm, if both decoding algorithms employ the same evaluation function satisfying (1) and (3). □

Theorem 3 The number of generating test error patterns in the proposed decoding algorithm is no more than that in the BF decoding algorithm, if both decoding algorithms employ the same evaluation function satisfying (1) and (3). □

[6] Since the total list size is reduced compared to that of the BF decoding algorithm, the complexity for sorting is also reduced.

5 Simulation Results

In this section, we evaluate the effectiveness of the proposed decoding algorithm by computer simulations.

5.1 Conditions for Simulation

For binary (63,30,13) BCH code and binary (104,52,20) quadratic residue (QR) code, we perform MLD by the BF decoding algorithm (we denote with "BF" in tables) and the proposed decoding algorithm (we denote with "Proposed" in tables). At each signal to noise ratio (SNR) E_b/S_0 [dB], both decoding algorithms are carried out 10000 times. In tables, we use the following notations:

$N(\boldsymbol{r})$: the number of generating test error patterns in decoding of $\boldsymbol{r}$

$M(\boldsymbol{r})$: the maximum list size in decoding of $\boldsymbol{r}$

ave : the average value among 10000 decoding

max : the maximum value among 10000 decoding

We use the function f as the evaluation function in both decoding algorithm. Since the function f does not satisfy (C2), S'4) instead of S4) is used in the BF decoding algorithm. We assume that the weight profiles $W_H(C)$ of these two codes are unknown and we use their supersets $W'_H(C) = \{0, d_H, d_H+1, \cdots, n\}$. Furthermore, we set the reference codeword as $\tilde{\boldsymbol{c}}_{ref} = \tilde{\boldsymbol{c}}_{best}$ for the calculation of (5). Each time a new estimated codeword $\tilde{\boldsymbol{c}}_{best}$ is obtained, the reference codeword is updated[7].

5.2 Results and Discussion

By Table 1, the maximum list size $\max M(\boldsymbol{r})$ in the proposed decoding algorithm is less than 1/3 of that in the BF decoding algorithm in each SNR. Furthermore, the average value of the maximum list size $\text{ave}\, M(\boldsymbol{r})$ in the proposed decoding algorithm is less than 1/4 of that in the BF decoding algorithm. These results show that the effectiveness of the proposed decoding algorithm. By Table 2, the values $\max M(\boldsymbol{r})$ and $\text{ave}\, M(\boldsymbol{r})$ in the proposed decoding algorithm are less than 1/4 and 1/5 of ones in the BF decoding algorithm, respectively. Theses results indicate the proposed method also works well for a longer code.

The number of generating test error patterns $N(\boldsymbol{r})$ is one of indices to evaluate time complexity in heuristic search MLD algorithms [2, 4]. By Table 1 and 2, $N(\boldsymbol{r})$ in the proposed decoding algorithm are less than 2/5 of $N(\boldsymbol{r})$ in the BF decoding algorithm even at low SNRs. These results indicate the proposed method reduces the time complexity of the BF decoding algorithm as well as the space complexity.

6 Conclusion and Future Works

In this paper, we propose a method for reducing the space complexity of the Battail-Fang decoding algorithm that is a priority-first heuristic search MLD. The proposed method is applicable to search methods with more effective evaluation functions. The decoding algorithm employing the proposed method are guaranteed to perform MLD since the set of generated candidate codewords there is identical to that in the BF decoding algorithm. The proposed decoding algorithm reduces not only the space complexity but the time one in the BF decoding algorithm. The proposed decoding algorithm can be straightforwardly modified to suboptimal soft-decision decoding by limiting a set of candidate codewords, and the same effectiveness can be anticipated as presented in this paper.

As future works, we need to develop a method for heuristic search MLD algorithm with powerful evaluation functions such as in [2]. The analytical guarantees of the reduction ratio of the complexity in the proposed method to that in the BF decoding algorithm are also needed.

[7] For a justifiable comparison, we assign the same value of the evaluation function to the same test error pattern in both decoding algorithm even when $\tilde{\boldsymbol{c}}_{ref}$ is updated. i.e., the number of generating candidate codewords is the same in both decoding algorithm.

Table 1: The results of decoding for (63,30,13) BCH code

E_b/N_0 [dB]			BF	Proposed
5.0	ave	$N(\boldsymbol{r})$	33.7	2.42
		$M(\boldsymbol{r})$	1.20	0.145
	max	$M(\boldsymbol{r})$	2073	411
4.0	ave	$N(\boldsymbol{r})$	94.6	20.7
		$M(\boldsymbol{r})$	11.6	1.86
	max	$M(\boldsymbol{r})$	9586	2406
3.0	ave	$N(\boldsymbol{r})$	593	205
		$M(\boldsymbol{r})$	99.9	20.7
	max	$M(\boldsymbol{r})$	15191	4435
2.0	ave	$N(\boldsymbol{r})$	2960	1190
		$M(\boldsymbol{r})$	553	130
	max	$M(\boldsymbol{r})$	70519	20552

Table 2: The results of decoding for $(104, 52, 20)$ QR code

E_b/N_0 [dB]			BF	Proposed
6.0	ave	$N(\boldsymbol{r})$	1.53	0.173
		$M(\boldsymbol{r})$	0.248	0.0228
	max	$M(\boldsymbol{r})$	187	18
5.0	ave	$N(\boldsymbol{r})$	33.3	4.91
		$M(\boldsymbol{r})$	3.66	0.358
	max	$M(\boldsymbol{r})$	5215	789
4.0	ave	$N(\boldsymbol{r})$	1210	349
		$M(\boldsymbol{r})$	170	29.5
	max	$M(\boldsymbol{r})$	463354	98746
3.0	ave	$N(\boldsymbol{r})$	41000	14700
		$M(\boldsymbol{r})$	6860	1410
	max	$M(\boldsymbol{r})$	11452892	2673788

Acknowledgement

H. Yagi wishes to express his gratitude to Dr. M. Kobayashi at Shonan Institute of Technology for his valuable comments.

References

[1] G. Battail and J. Fang, "Décodage pondéré optimal descodes linéaires en blocs," Annales des télé-communications, vol.41, nos.11–12, pp.580–604, Nov.–Dec. 1986.

[2] Y.S. Han, C.R.P. Hartmann, and C.C. Chan, "Efficient priority-first search maximum likelihood soft decision decoding of linear block codes," IEEE Trans. Inform. Theory, vol.39, no.5, pp.1514–1523, Sept. 1993.

[3] D. Gazelle and J. Snyders, "Reliability-based code-search algorithm for maximum-likelihood decoding of block codes," IEEE Trans. Inform. Theory, vol.43, no.1, pp.239–249, Jan. 1997.

[4] A. Valembois and M. Fossorier, "An improved method to compute lists of binary vectors that optimize a given weight function with application to soft decision decoding," IEEE Commun. Lett., vol.5, no.3, pp.456–458, Nov. 2001.

[5] M.P.C. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," IEEE Trans. Inform. Theory, vol.41, no.5, pp.1379–1396, Sept. 1995.

[6] A. Valembois and M. Fossorier, "A comparison between "most-reliable-basis reprocessing" strategies", IEICE Trans. fundamentals, vol.E85-A, no.7, pp.1727–1741, July 2002.

The 27th Symposium on Information Theory
and Its Applications (SITA2004)
Gero, Gifu, Japan, Dec. 14–17, 2004

A Heuristic Search Algorithm with the Reduced List of Test Error Patterns for Maximum Likelihood Decoding

Hideki YAGI * Toshiyasu MATSUSHIMA * Shigeichi HIRASAWA *

Abstract— The reliability-based heuristic search methods for maximum likelihood decoding (MLD) generate test error patterns (or, equivalently, candidate codewords) according to their heuristic values. Test error patterns are stored in lists and its space complexity is crucially large for MLD of long block codes. One of the well-known heuristic search methods for MLD is the A* decoding algorithm proposed by Han et al. Based on the decoding algorithms both by Battail and Fang (and its improved technique by Valembois and Fossorier) and by the present authors, we deduce a new method for reducing the space complexity of the A* decoding algorithm. Simulation results show the high efficiency of the proposed method.

Keywords— maximum likelihood decoding, binary block codes, heuristic search, most reliable basis, reliability

1 Introduction

In this paper, we consider the priority-first search-type MLD algorithms where candidate codewords are generated in increasing value of the heuristic function. To the authors' knowledge, G. Battail and J. Fang first proposed a priority-first search method for MLD where a simple evaluation function is employed [1] (we will call this method the BF decoding algorithm). One of the most well-known priority-first search methods for MLD is the A* decoding algorithm proposed by Han et al. [2]. Recently, A. Valembois and M. Fossorier have indicated that a slight modification makes the BF decoding algorithm equivalent to the A* decoding algorithm [6, 7]. Subsequently, Valembois et al. [6] and the present authors [8] have proposed techniques for drastically reducing the space complexity of modified BF decoding algorithms employing some class of heuristic (or evaluation) functions. However, their techniques cannot be straightforwardly used for reducing the space complexity of the A* decoding algorithm in which the search is guided by more sophisticated heuristic functions than that considered in [6, 8].

In this paper, based on the Valembois' indication and the techniques in [8], we propose a method for reducing the space complexity of the A* decoding algorithm. Consequently, we show theoretically that the space complexity for the proposed decoding algorithm is less than that for the A* decoding algorithm, and by computer simulations that the former is significantly reduced.

*Dept. of Industrial and Management Systems Engineering, Waseda University, 3-4-1 Ohkubo Shinjuku-ku, Tokyo, 169-8555 Japan, E-mail: yagi@hirasa.mgmt.waseda.ac.jp

2 Reliability-based MLD Algorithm

Let C be a binary linear (n, k, d) block code of the code length n, the number of information symbols k and the minimum distance d. We denote a generator matrix of C by G and the weight profile of C by $W(C)$. We assume any codewords $\boldsymbol{c} = (c_1, c_2, \ldots, c_n) \in \{0,1\}^n$ of C are transmitted over the Additive White Gaussian Noise (AWGN) channel. The receiver maps a received sequence $\boldsymbol{r} = (r_1, r_2, \ldots, r_n) \in \mathcal{R}^n$ into a sequence $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_n)$, $\theta_j = \ln \frac{P(r_j|c_j=0)}{P(r_j|c_j=1)}$, where $P(r_j|c_j)$ represents the likelihood of the symbol c_j. Furthermore, a hard-decision sequence $\boldsymbol{z} = (z_1, z_2, \ldots, z_n) \in \{0,1\}^n$ is obtained by setting $z_j = 0$ if $\theta_j \geq 0$ and $z_j = 1$ otherwise. The soft-decision decoder estimates the transmitted codeword from $\boldsymbol{\theta}$ and $\boldsymbol{z}$.

In reliability-based decoding algorithms, we use the column-permuted systematic generator matrix $\tilde{G}$ where the leftmost k positions are the *most reliable and linearly independent* (MRI) [2, 5, 6, 7] in non-increasing value of reliability. Let $\tilde{\boldsymbol{\theta}} = (\tilde{\theta}_1, \tilde{\theta}_2, \ldots, \tilde{\theta}_n)$ and $\tilde{\boldsymbol{z}} = (\tilde{z}_1, \tilde{z}_2, \ldots, \tilde{z}_n)$ be permuted sequences of $\boldsymbol{\theta}$ and $\boldsymbol{z}$, respectively, in the same ordering of columns of $\tilde{G}$. Let $\tilde{C}$ be the code whose codewords are generated by $\tilde{G}$. Define $\boldsymbol{u} = (u_1, u_2, \ldots, u_k) \in \{0,1\}^k$ as the leftmost k symbols of $\tilde{\boldsymbol{z}}$, i.e., $u_j = \tilde{z}_j, 1 \leq \forall j \leq k$. The decoder first encodes $\boldsymbol{u}$ by $\tilde{G}$ to obtain the initial codeword $\tilde{\boldsymbol{c}}_\emptyset (= \boldsymbol{u}\tilde{G})$. Afterwards, k dimensional vectors, called *test error patterns* $\boldsymbol{t} \in \{0,1\}^k$, are iteratively generated and encoded by $\tilde{G}$. Then, $\tilde{\boldsymbol{c}} = \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}\tilde{G}$ is a candidate codeword and this procedure is repeated until a sufficient condition for the ML codeword is satisfied[1].

Definition 1 For a location set $J \subseteq [1, k]$, the test error pattern (TEP) $\boldsymbol{t}(J) = (t_1(J), t_2(J), \ldots, t_k(J))$ has element one in J. Such J is called the *support* of $\boldsymbol{t}(J)$. Define that $b(J)$ be the rightmost position in J, i.e., $b(J) = \max J$. For $j > b(J)$, the TEP $\boldsymbol{t}(J \cup \{j\})$ (or simply $\boldsymbol{t}(J \cup j)$) is called an *extended pattern* of $\boldsymbol{t}(J)$. For $j > b(J)$ and $J' = J \setminus b(J)$, the TEP $\boldsymbol{t}(J' \cup j)$ is called an *adjacent pattern* of $\boldsymbol{t}(J)$ in j. □

For a binary vector $\boldsymbol{v} = (v_1, v_2, \ldots, v_n) \in \{0,1\}^n$, we define the *correlation discrepancy* [6, 7] of $\boldsymbol{v}$ as

$$L(\boldsymbol{v}) = \sum_{j|v_j \neq \tilde{z}_j} |\tilde{\theta}_j|. \tag{1}$$

It is well-known that $\tilde{\boldsymbol{c}}_{\text{best}}$ is the ML codeword if and only if $L(\tilde{\boldsymbol{c}}_{\text{best}}) = \min_{\tilde{\boldsymbol{c}} \in \tilde{C}} L(\tilde{\boldsymbol{c}})$.

[1] $\oplus$ represents Exclusive OR operation.

3 Priority-first Search Method of Test Error Patterns

The A* decoding algorithm [2] searches the ML codeword through the trellis or the binary tree of the code. Valembois and Fossorier have indicated that the modified BF algorithm and the A* decoding algorithm are equivalent when both algorithms use the same heuristic function. In this section, we state the A* decoding algorithm from the Valembois' perspective [6].

We first address the heuristic function of the search. For $\tilde{\boldsymbol{c}}_{\rm ref} \in \tilde{C}$ and $\boldsymbol{t} = (t_1, t_2, \ldots, t_k)$, we define

$$T(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref}) = \Big\{ \boldsymbol{v} \,\Big|\, v_j = u_j \oplus t_j \ \text{ for } j \in [1,k], \text{ and } d_H(\boldsymbol{v}, \tilde{\boldsymbol{c}}_{\rm ref}) \in W(\tilde{C}) \Big\}, \quad (2)$$

where $d_H(\cdot,\cdot)$ denotes the Hamming distance. Then the heuristic function considered in [2] is defined as

$$f(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref}) = \min_{\boldsymbol{v} \in T(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref})} \Big\{ L(\boldsymbol{v}) \Big\}. \quad (3)$$

The A* decoding algorithm generates $\boldsymbol{t}$ in increasing value of $f(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref})$. Such $\tilde{\boldsymbol{c}}_{\rm ref}$ is called the *referenced codeword* [3, 6] or the *seed* [2, 5].

The A* decoding algorithm performs the priority-first search with not only the function f but any heuristic functions F satisfying the following condition: for $j \notin J$,

$$\textbf{(C1)} \qquad F\big(\boldsymbol{t}(J)\big) \le F\big(\boldsymbol{t}(J \cup j)\big). \quad (4)$$

It is guaranteed that the A* decoding algorithm finds the most likely codeword if

$$F\big(\boldsymbol{t}(J)\big) \le L(\tilde{\boldsymbol{c}}_J), \quad (5)$$

where $\tilde{\boldsymbol{c}}_J = \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}(J)\tilde{G}$.

Hereafter, we describe how to perform the priority-first search of TEPs using the heuristic function satisfying (C1). Let $M^{(1)}, M^{(2)}, \ldots, M^{(k)}$ be k lists of TEPs. The TEP $\boldsymbol{t}(J)$ is supposed to be in $M^{(b(J))}$ where $b(J) = \max J$. In a list $M^{(j)}, \forall j \in [1,k]$, TEPs are ordered in increasing value of a heuristic function F.

By the condition (C1), the TEPs with the minimum value of F in $M^{(j)}, j \in [1,k]$, is $\boldsymbol{t}(j)$ whose Hamming weight are one. Therefore, we just need to set the initial lists as $M^{(j)} = \{\boldsymbol{t}(j)\}$ for $j \in [1,k]$. Then, the algorithm searches the TEP with the minimum value of F (we will call this pattern the *best pattern*) among the set of those that have not been found.

We here describe the A* decoding algorithm which is equivalent to the BF algorithm with a slight modification.

[The A* decoding algorithm]

S1) Set $\tilde{\boldsymbol{c}}_\emptyset := \boldsymbol{u}\tilde{G}$, $\tilde{\boldsymbol{c}}_{\rm best} := \tilde{\boldsymbol{c}}_\emptyset$ and $\underline{L} := L(\tilde{\boldsymbol{c}}_\emptyset)$. Construct the initial lists of TEPs.

S2) Choose the best pattern $\boldsymbol{t}(J) \in M^{(b(J))}$ among the topmost TEPs in non-empty lists $M^{(j)}$. If $F(\boldsymbol{t}(J)) \ge \underline{L}$, then output $\tilde{\boldsymbol{c}}_{\rm best}$ and halt the algorithm.

S3) Generate the next candidate codeword by $\tilde{\boldsymbol{c}}_J := \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}(J)\tilde{G}$. If $L(\tilde{\boldsymbol{c}}_J) < \underline{L}$, then set $\underline{L} := L(\tilde{\boldsymbol{c}}_J)$ and $\tilde{\boldsymbol{c}}_{\rm best} := \tilde{\boldsymbol{c}}_J$.

S4) For all lists $M^{(j)}$ such that $j > b(J)$, insert the extended patterns $\boldsymbol{t}(J \cup j)$ at the position such that the list remains increasing order. Delete $\boldsymbol{t}(J)$ from $M^{(b(J))}$.

S5) If $M^{(j)} = \emptyset$ for all $j \in [1,k]$, then output $\tilde{\boldsymbol{c}}_{\rm best}$ and halt the algorithm. Otherwise, go to S2). □

In S4), we need to sort the extended pattern so that the list $M^{(j)}$ remains increasing order of the heuristic values. By sorting, the priority-first search of the A* decoding algorithm is maintained [6].

In the original A* decoding algorithm, the only one list of TEPs is used. If we combine the k lists into the united list and order test patterns increasing order of the heuristic values in it, then the above algorithm becomes identical to the original A* decoding algorithm although the behaviors of the two algorithms seem different.

We here describe the complexity of the A* decoding algorithm. In a decoding procedure of a received sequence $\boldsymbol{r}$, the space complexity is $O(k \times M(\boldsymbol{r}))$ where $M(\boldsymbol{r})$ represents the maximum number of TEPs stored in lists. As for the time complexity, that for generating TEPs is dominant as well as that for encoding them.

4 Proposed Decoding Algorithm

In this section, we propose a method for reducing the list size of TEPs in the A* decoding algorithm which dominates the space complexity.

We here define the following condition (C2) for a heuristic function F'.

Definition 2 Let $S^{(0)}$ be a certain subset of $[1,k]$ and $S^{(1)}$ be the complement of $S^{(0)}$. For $J \subseteq [1,k]$, assume $j_1, j_2 \notin J$ and $j_1 < j_2$. If $j_1, j_2 \in S^{(\alpha)}$ with $\alpha \in \{0,1\}$, then a function F' satisfies

$$\textbf{(C2)} \qquad F'\big(\boldsymbol{t}(J \cup j_1)\big) \ge F'\big(\boldsymbol{t}(J \cup j_2)\big). \quad (6)$$

We will call (6) the condition (C2). □

For the function f, we show the following lemma.

Lemma 1 For a referenced codeword $\tilde{\boldsymbol{c}}_{\rm ref} \in \tilde{C}$, let $\boldsymbol{t}_{\rm ref}$ be the TEP of $\tilde{\boldsymbol{c}}_{\rm ref}$. Assuming that $S^{(1)}$ and $S^{(0)}$ be the support of $\boldsymbol{t}_{\rm ref}$ and its complement, respectively. Then for $\boldsymbol{t}(J)$, the heuristic function $f\big(\boldsymbol{t}(J), \tilde{\boldsymbol{c}}_{\rm ref}\big)$ satisfies the condition (C2).

(**Proof**) From (1) and (3), the heuristic function $f(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref})$ of a TEP $\boldsymbol{t} = (t_1, t_2, \ldots, t_k)$ satisfies

$$f(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref}) = \sum_{j=1}^{k} t_j |\tilde{\theta}_j| + \min_{\boldsymbol{v} \in T(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref})} \Big\{ \sum_{j=k+1}^{n} (\tilde{z}_j \oplus v_j) |\tilde{\theta}_j| \Big\}. \quad (7)$$

Denote the second term of the r.h.s. of (7) by $A(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref})$.

Assuming that $j_1 < j_2$, and $j_1, j_2 \in S^{(0)}$, then TEPs $\boldsymbol{t}(J \cup j_1)$ and $\boldsymbol{t}(J \cup j_2)$ have the same Hamming distance from $\boldsymbol{t}_{\rm ref}$. Therefore by (2), (3) and (7),

$$A\big(\boldsymbol{t}(J \cup j_1), \tilde{\boldsymbol{c}}_{\rm ref}\big) = A\big(\boldsymbol{t}(J \cup j_2), \tilde{\boldsymbol{c}}_{\rm ref}\big). \quad (8)$$

Hence we have

$$f(\boldsymbol{t}(J\cup j_1),\tilde{\boldsymbol{c}}_{\rm ref}) - f(\boldsymbol{t}(J\cup j_2),\tilde{\boldsymbol{c}}_{\rm ref}) = \sum_{j\in J\cup j_1}|\tilde{\theta}_j| - \sum_{j\in J\cup j_2}|\tilde{\theta}_j| = |\tilde{\theta}_{j_1}| - |\tilde{\theta}_{j_2}| \geq 0.$$

This inequality shows the function f satisfies (C2) when $j_1, j_2 \in S^{(0)}$. In the case that $j_1, j_2 \in S^{(1)}$, we can prove the lemma similarly. □

In the following, we consider heuristic functions that satisfy both (C1) and (C2).

The strategy of the proposed method is like *lazy evaluation* where any TEPs are not generated as long as possible. This approach is similar to improved techniques in [6, 8] where other heuristic functions are considered. We first consider k lists $M^{(j)}$ as in the A* decoding algorithm of Sect. 3. Hereafter, we assume $S^{(0)} = \{i_1, i_2, \ldots, i_s\}$ and $S^{(1)} = \{i'_1, i'_2, \ldots, i'_p\}$.

By the condition (C1), the best pattern in a list $M^{(j)}, j \in [1,k]$, is $\boldsymbol{t}(j)$ whose Hamming weight is one. Furthermore, the best pattern among s TEPs $\boldsymbol{t}(j), j \in S^{(0)}$, is $\boldsymbol{t}(i_s)$ by the condition (C2). Similarly, the best pattern among p TEPs $\boldsymbol{t}(j), j \in S^{(1)}$, is $\boldsymbol{t}(i'_p)$. Therefore, we may as well construct the initial lists as

$$M^{(j)} = \begin{cases} \{\boldsymbol{t}(j)\}, & \text{if } j \in \{i_s, i'_p\}; \\ \emptyset, & \text{otherwise.} \end{cases} \tag{9}$$

At S2) of the A* decoding algorithm, if $\boldsymbol{t}(J) \in M^{(b(J))}$ is chosen as the best pattern, $k - b(J)$ extended patterns of $\boldsymbol{t}(J)$ will be stored at S4). However, it is enough to store only its extended patterns $\boldsymbol{t}(J\cup i_s)$ and $\boldsymbol{t}(J\cup i'_p)$ in the list $M^{(i_s)}$ and $M^{(i'_p)}$, respectively. This is guaranteed by (C2), since $F(\boldsymbol{t}(J\cup j)) \geq F(\boldsymbol{t}(J\cup i_s))$ for $\forall j \in S^{(0)}$ and $F(\boldsymbol{t}(J\cup j)) \geq F(\boldsymbol{t}(J\cup i'_p))$ for $\forall j \in S^{(1)}$.

Following this modification, we need to determine when to insert other extended patterns $\boldsymbol{t}(J\cup j), j \notin \{i_s, i'_p\}$, into lists. Assume that a TEP $\boldsymbol{t}(J\cup i_q)$ such that $i_q > b(J)$ and $i_q \in S^{(0)}$ has been already stored in the list $M^{(i_q)}$. Since $\boldsymbol{t}(J\cup j)$ such that $j < i_q$ and $j \in S^{(0)}$ cannot be the best pattern, we may as well store these extended patterns only after $\boldsymbol{t}(J\cup i_q)$ is chosen as the best pattern at S2). If $i_{q-1} > b(J)$, $\boldsymbol{t}(J\cup i_{q-1})$ has the smallest heuristic value among all adjacent patterns of $\boldsymbol{t}(J\cup i_q)$ in $S^{(0)}$ from the condition (C2), i.e.,

$$F(\boldsymbol{t}(J\cup i_{q-1})) = \min_{j\in S^{(0)}}\left\{F((\boldsymbol{t}J\cup j)) \mid b(J) < j < i_q\right\}. \tag{10}$$

Therefore, after $\boldsymbol{t}(J\cup i_q)$ is chosen as the best pattern at S2), $\boldsymbol{t}(J\cup i_{q-1})$ is inserted into the list $M^{(i_{q-1})}$. This modification reduces the space complexity significantly. We note that the next generated TEP $\boldsymbol{t}(J\cup i_{q-1})$ and $F(\boldsymbol{t}(J\cup i_{q-1}))$ are easily calculated from the selected pattern $\boldsymbol{t}(J\cup i_q)$ and $F(\boldsymbol{t}(J\cup i_q))$. Similar arguments also hold when $\boldsymbol{t}(J\cup i'_q), i'_q \in S^{(1)}$, has been the best pattern at S2).

We describe a proposed decoding algorithm employing the above method.

[The proposed decoding algorithm]

P1) Set $\tilde{\boldsymbol{c}}_\emptyset := \boldsymbol{u}\tilde{G}$, $\tilde{\boldsymbol{c}}_{\rm best} := \tilde{\boldsymbol{c}}_\emptyset$ and $\underline{L} := L(\tilde{\boldsymbol{c}}_\emptyset)$. Construct the initial lists of TEPs by (9).

P2) Choose the best pattern $\boldsymbol{t}(J) \in M^{(b(J))}$ among non-empty lists. If $F(\boldsymbol{t}(J)) \geq \underline{L}$, then output $\tilde{\boldsymbol{c}}_{\rm best}$ and halt the algorithm.

P3) Generate the next candidate codeword by $\tilde{\boldsymbol{c}}_J := \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}(J)\tilde{G}$. If $L(\tilde{\boldsymbol{c}}_J) < \underline{L}$, then set $\underline{L} := L(\tilde{\boldsymbol{c}}_J)$ and $\tilde{\boldsymbol{c}}_{\rm best} := \tilde{\boldsymbol{c}}_J$.

P4) a) If $b(J) = i_q$ (i.e., $b(J) \in S^{(0)}$) and the adjacent pattern $\boldsymbol{t}(J'\cup i_{q-1})$ exists where $J' = J \setminus b(J)$, then insert it into the list $M^{(i_{q-1})}$.

b) If $b(J) = i'_q$ (i.e., $b(J) \in S^{(1)}$) and $\boldsymbol{t}(J'\cup i'_{q-1})$ exists where $J' = J \setminus b(J)$, then insert it into $M^{(i'_{q-1})}$.

c) If $b(J) < i_s$, then insert $\boldsymbol{t}(J\cup i_s)$ into $M^{(i_s)}$. If $b(J) < i'_p$, then insert $\boldsymbol{t}(J\cup i'_p)$ into $M^{(i'_p)}$. Delete $\boldsymbol{t}(J)$ from $M^{(b(J))}$.

P5) If $M^{(j)} = \emptyset$ for all $j \in [1,k]$, then output $\tilde{\boldsymbol{c}}_{\rm best}$ and halt the algorithm. Otherwise, go to S2). □

The step P4) corresponds to the above modification. Note that we need to store at most three TEPs at P4), while we need to store at most $k - b(J)$ TEPs at S4) of the A* decoding algorithm.

In terms of the time and space complexity of the proposed decoding algorithm, we show the following theorems.

Theorem 1 The proposed decoding algorithm performs MLD. Then, the maximum list size of TEPs in the proposed decoding algorithm is less than that in the A* decoding algorithm, if both decoding algorithms employ the same heuristic function satisfying (C1) and (C2). □

Theorem 2 The number of generated TEPs in the proposed decoding algorithm is no more than that in the A* decoding algorithm, if both decoding algorithms employ the same heuristic function satisfying (C1) and (C2). □

5 Simulation Results

In this section, we evaluate the effectiveness of the proposed decoding algorithm by computer simulations.

5.1 Conditions of Simulations

For the binary (63,30,13) BCH code and the binary (104,52,20) quadratic residue (QR) code, we perform MLD by the A* decoding algorithm (we denote it by "A*" in tables) and the proposed decoding algorithm (we denote it by "Proposed" in tables). At each signal to noise ratio (SNR) $E_{\rm b}/S_0$ [dB], both decoding algorithms are carried out 10,000 times. In tables, we use the following notations:

$N(\boldsymbol{r})$: the number of generated TEPs in decoding of $\boldsymbol{r}$

$M(\boldsymbol{r})$: the maximum list size in decoding of $\boldsymbol{r}$

Ave : the average value among 10,000 decoding

Max : the maximum value among 10,000 decoding

We use the function f as the heuristic function in both decoding algorithms. We assume that the weight

Table 1: The results of decoding for (63,30,13) BCH code

E_b/N_0 [dB]			A*	Proposed
5.0	Ave	$N(\boldsymbol{r})$	33.7	1.78
		$M(\boldsymbol{r})$	1.19	0.176
	Max	$M(\boldsymbol{r})$	2064	437
4.0	Ave	$N(\boldsymbol{r})$	93.6	21.9
		$M(\boldsymbol{r})$	11.4	2.19
	Max	$M(\boldsymbol{r})$	9579	3272
3.0	Ave	$N(\boldsymbol{r})$	583	225
		$M(\boldsymbol{r})$	96.6	24.4
	Max	$M(\boldsymbol{r})$	14321	5103
2.0	Ave	$N(\boldsymbol{r})$	2908	1305
		$M(\boldsymbol{r})$	553	156
	Max	$M(\boldsymbol{r})$	70519	28005

Table 2: The results of decoding for $(104, 52, 20)$ QR code

E_b/N_0 [dB]			A*	Proposed
6.0	Ave	$N(\boldsymbol{r})$	48.5	0.239
		$M(\boldsymbol{r})$	0.248	0.0342
	Max	$M(\boldsymbol{r})$	187	32
5.0	Ave	$N(\boldsymbol{r})$	82.4	5.79
		$M(\boldsymbol{r})$	3.60	0.471
	Max	$M(\boldsymbol{r})$	4932	793
4.0	Ave	$N(\boldsymbol{r})$	1237	357
		$M(\boldsymbol{r})$	166	30.1
	Max	$M(\boldsymbol{r})$	462986	98620
3.0	Ave	$N(\boldsymbol{r})$	33806	13010
		$M(\boldsymbol{r})$	5461	1297
	Max	$M(\boldsymbol{r})$	11452864	2680552

profiles $W(\mathcal{C})$ of these two codes are unknown and we use their supersets $W'(\mathcal{C}) = \{0, d, d+1, \cdots, n\}$.

We set the reference codeword as $\tilde{\boldsymbol{c}}_{\text{ref}} = \tilde{\boldsymbol{c}}_{\text{best}}$ for the calculation of (3). When a temporally best codeword $\tilde{\boldsymbol{c}}_{\text{best}}$ is newly obtained, the reference codeword is updated. In this case, though we need to store TEPs corresponding to old referenced codewords in memory in the proposed decoding algorithm[2], we need not store its discrepancy which is the real number, while storing TEPs needs their heuristic values.

5.2 Results and Discussion

We show the results of the (63,30,13) BCH code and the (104,52,20) QR code in Tables 1 and 2, respectively. By Table 1, the maximum list size $\mathrm{Max}\, M(\boldsymbol{r})$ in the proposed decoding algorithm is less than 2/5 of that in the A* decoding algorithm at each SNR. Furthermore, the average value of the maximum list size $\mathrm{Ave}\, M(\boldsymbol{r})$ in the proposed decoding algorithm is less than 1/3 of that in the A* decoding algorithm. These results show that the effectiveness of the proposed decoding algorithm. By Table 2, the values $\mathrm{Max}\, M(\boldsymbol{r})$ and $\mathrm{Ave}\, M(\boldsymbol{r})$ in the proposed decoding algorithm are less than 1/4 of those in the A* decoding algorithm. Theses results indicate the proposed method also works well for a longer code.

The number of generated TEPs $N(\boldsymbol{r})$ is one of indices to evaluate time complexity in heuristic search MLD algorithms [2, 5, 6]. By Table 1 and 2, $N(\boldsymbol{r})$ in the proposed decoding algorithm are less than 2/5 of $N(\boldsymbol{r})$ in the A* decoding algorithm even at 3.0 [dB]. These results demonstrate the proposed method reduces the time complexity of the A* decoding algorithm as well as the space complexity.

6 Conclusion and Future Works

In this paper, we propose a new priority-first heuristic search method reducing the space complexity of the A* decoding algorithm via the perspective of [6]. The proposed decoding algorithm is guaranteed to perform MLD since the set of generated candidate codewords is identical to that in the original A* decoding algorithm. The proposed decoding algorithm reduces not only the space complexity but the time one in the A* decoding algorithm.

As future works, we need to develop a method for heuristic search MLD algorithm with powerful heuristic functions such as in [5].

[2]Its space complexity may be fairly small since the number of obtained referenced codewords is small. In our simulation, we observed that the maximum number of updating referenced codeword is only 24 among all simulations.

Acknowledgment

H. Yagi wishes to thank Dr. M. Kobayashi at Shonan Institute of Technology and Mr. T. Ishida and Mr. G. Hosoya at Waseda University for their valuable comments.

References

[1] G. Battail and J. Fang, "Décodage pondéré optimal descodes linéaires en blocs," *Annales des télé-communications*, vol.41, nos.11–12, pp.580–604, Nov.–Dec. 1986.

[2] Y.S. Han, C.R.P. Hartmann, and C.C. Chan, "Efficient priority-first search maximum likelihood soft decision decoding of linear block codes," *IEEE Trans. Inform. Theory*, vol.39, no.5, pp.1514–1523, Sept. 1993.

[3] D. Gazelle and J. Snyders, "Reliability-based code-search algorithm for maximum-likelihood decoding of block codes," *IEEE Trans. Inform. Theory*, vol.43, no.1, pp.239–249, Jan. 1997.

[4] M.P.C. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Trans. Inform. Theory*, vol.41, no.5, pp.1379–1396, Sept. 1995.

[5] T. Okada, M. Kobayashi, and S. Hirasawa, "An efficient heuristic search method for maximum likelihood decoding of linear block codes using dual codes," *IEICE Trans. Fundamentals*, vol.E85-A, no.2, pp.485–489, Feb. 2002.

[6] A. Valembois and M. Fossorier, "An improved method to compute lists of binary vectors that optimize a given weight function with application to soft decision decoding," *IEEE Commun. Lett.*, vol.5, no.3, pp.456–458, Nov. 2001.

[7] A. Valembois and M. Fossorier, "A comparison between "most-reliable-basis reprocessing" strategies", *IEICE Trans. fundamentals*, vol.E85-A, no.7, pp.1727–1741, July 2002.

[8] H. Yagi, T. Matsushima, and S. Hirasawa, "A method for reducing space complexity of reliability-based heuristic search maximum likelihood decoding algorithms", *Proc. of the 26th Symposium on Information Theory and its Applications (SITA2003)*, pp.185–188, Hyogo, Japan, Dec. 2003.

International Symposium on Information Theory and its Applications, ISITA2004
Parma, Italy, October 10–13, 2004

Efficient Reliability-based Soft Decision Decoding Algorithm over Markov Modulated Channel

Hideki YAGI†, Toshiyasu MATSUSHIMA†, and Shigeichi HIRASAWA†

† Department of Industrial and Management Systems Engineering,
School of Science and Engineering, Waseda University,
3-4-1 Ohkubo Shinjuku-ku, Tokyo, 169-8555 JAPAN
E-mail: yagi@hirasa.mgmt.waseda.ac.jp

Abstract

We discuss soft-decision decoding which achieves near-maximum likelihood decoding (MLD) of binary block codes over a Markov modulated channel. In this paper, a new soft-decision decoding algorithm using a generalized Expectation Maximization (EM) algorithm is proposed. Each iteration step of the proposed decoding algorithm can be regarded as performing MLD over an additive white Gaussian noise (AWGN) channel, so the proposed decoding algorithm can employ most of conventional efficient methods devised for the AWGN channel. The simulation results show that the proposed decoding algorithm achieves almost the same performance as that of MLD which needs exhaustive search of codewords.

1. Introduction

Soft-decision decoding (SDD) reduces the block error probability of decoding by taking advantage of information of channel noise. On additive white Gaussian noise (AWGN) channels, many researchers have devoted to develop efficient maximum likelihood decoding (MLD) and sub-optimum SDD algorithms for block codes [1, 2, 3, 7].

Recently, additive noise channels with a hidden Markov model have attracted great attentions due to its practicality. On these channels, several symbol-wise maximum a-posteriori probability (MAP) decoding algorithms of block codes have been proposed [6, 8, 9]. However, MLD or sub-optimum SDD algorithms that reduce the block error probability of decoding over these channels have not sufficiently studied. To the authors' knowledge, only sub ML (sequence-wise MAP) decoding algorithm for trellis codes has been devised [5]. To aim at reducing the block error probability of decoding, we need to consider much larger search space of the most likely codeword than that in an AWGN channel, since it is direct product of the spaces of codewords and channel states sequences.

In this paper, we propose a new SDD algorithm of block codes using a generalized Expectation Maximization (EM) algorithm [10] over Markov modulated Gaussian noise (MMGN) channels. In the EM principle, we can reduce the search space at each iteration step. The proposed decoding algorithm is an iterative algorithm and each iteration step can be conducted similar to an MLD algorithm of block codes over an AWGN channel. We then derive (i) reliability measure of binary symbols and (ii) a termination condition of the decoding algorithm. As a result, we show by simulations that the proposed SDD algorithm achieves near MLD with relatively small decoding complexity.

2. Preliminary

2.1. Channel Model

Assume that a channel is modeled as the Markov model with finite discrete states $\mathcal{S} = \{0, 1, \ldots, |\mathcal{S}|-1\}$. At a state $i \in \mathcal{S}$, a Gaussian noise with mean 0 and variance $\sigma^2(i)$ is generated. We denote a state of Markov model at time j with s_j and let $\boldsymbol{s}_j^{(m)} = (s_{j-m+1}, s_{j-m+2}, \ldots, s_j) \in \mathcal{S}^m$ for some integer $m \geq 1$. Then transition probability of the m-th order Markov model can be expressed as[1] $p(s_j = i|\boldsymbol{s}_{j-1}^{(m)})$. Assuming that the Markov model has stationary distribution, let $p(i)$ be the stationary probability at $i \in \mathcal{S}$. We assume that the order of Markov model, transition and stationary probabilities are known to the decoder. This channel is called an MMGN channel.

Let $\mathcal{C}$ be a binary linear (n, k) block code of length n, the dimension of code k with a generator matrix G. A codeword $\boldsymbol{c} = (c_1, c_2, \ldots, c_n) \in \{0,1\}^n$ of $\mathcal{C}$ is mapped into $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, $x_j = (-1)^{c_j} \in \{-1, +1\}$ with equal probability and $\boldsymbol{x}$ is transmitted over an MMGN channel. At the decoder, we estimate

[1] We define that elements of $\boldsymbol{s}_{j-1}^{(m)}$ take no values if the time indices are negative, e.g., $p(s_1|s_{-1}, s_0) = p(s_1|s_0)$ for $m = 2$.

the transmitted codeword $\boldsymbol{c}$ from the received sequence $\boldsymbol{r} = (r_1, r_2, \ldots, r_n) \in \mathcal{R}^n$.

2.2. Reliability-based MLD Algorithm over AWGN channel

We describe reliability-based MLD algorithms with the column-permuted generator matrix [1, 2, 3, 7] that efficiently search the most likely codeword over an AWGN channel. In a decoder, the received sequence $\boldsymbol{r}$ is mapped into a sequence $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_n), \theta_j = \ln \frac{P(r_j|c_j=0)}{P(r_j|c_j=1)}$, where $P(r_j|c_j)$ denotes the likelihood function of c_j. Then, we obtain the hard decision received sequence $\boldsymbol{y} = (y_1, y_2, \ldots, y_n) \in \{0,1\}^n$ by

$$y_j = \begin{cases} 0, & \text{if } \theta_j \geq 0; \\ 1, & \text{otherwise.} \end{cases} \tag{1}$$

For $\forall j \in [1, n]$, $|\theta_j|$ is called j-th reliability (over the AWGN channel)[2]. In this paper, we call any measures which express the confidence value of y_j, j-th reliability.

For $\boldsymbol{c} \in \mathcal{C}$, we define

$$L(\boldsymbol{c}) = \sum_{j|c_j \neq y_j} |\theta_j|. \tag{2}$$

Then, $\arg\max_{\boldsymbol{c}\in\mathcal{C}} \{P(\boldsymbol{r}|\boldsymbol{c})\} = \arg\min_{\boldsymbol{c}\in\mathcal{C}} \{L(\boldsymbol{c})\}$ [7]. An MLD algorithm searches the most likely codeword $\boldsymbol{c}_{\text{best}}$ such that $\boldsymbol{c}_{\text{best}} = \arg\min_{\boldsymbol{c}\in\mathcal{C}} L(\boldsymbol{c})$.

The reliability-based decoder first re-orders the most reliable and linearly independent (MRI) k columns of a generator matrix in non-increasing reliabilities. Then it performs standard row operations to make these k columns the identity matrix. We denote the resultant matrix by $\tilde{G}$. We obtain $\tilde{\boldsymbol{r}}$ and $\tilde{\boldsymbol{y}}$ by the same permutation of $\boldsymbol{r}$ and $\boldsymbol{y}$, respectively. Let $\tilde{\mathcal{C}}$ be the code obtained by the same permutation for $\mathcal{C}$.

Let $\boldsymbol{u} = (\tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_k)$ be the MRI k symbols of $\tilde{\boldsymbol{y}}$ and the initial codeword of search is obtained by $\tilde{\boldsymbol{c}}_0 = \boldsymbol{u}\tilde{G}$. After obtaining $\tilde{\boldsymbol{c}}_0$, we generate k dimensional vectors $\boldsymbol{t} \in \{0,1\}^k$ to obtain candidate codewords by $\tilde{\boldsymbol{c}} = (\boldsymbol{u} \oplus \boldsymbol{t})\tilde{G}$, where we call $\boldsymbol{t} \in \{0,1\}^k$ test error patterns (TEPs)[3]. Then, we iteratively generate candidate codewords and compute their likelihood. The decoder outputs the most likely codeword $\tilde{\boldsymbol{c}}_{\text{best}}$.

The right hand side (r.h.s.) of eq. (2) is a sum of reliabilities (positive real number). Using this structure of eq. (2), (i) acceptance criteria of the most likely codeword and (ii) elimination criteria of unnecessary TEPs are proposed to make the MLD algorithm efficient [1, 2, 3].

[2] $[i, j]$ denotes the set of integers from i to j, for two integers i and j such that $i \leq j$.

[3] $\oplus$ represents Exclusive OR operation.

2.3. Generalized EM Algorithm

The EM algorithm is an iterative procedure for computing (near) maximum likelihood estimation of unknown parameter $\boldsymbol{\psi}$ from observed data $\boldsymbol{z}_{\text{obs}}$. Let $\boldsymbol{z}_{\text{mis}}$ and $\boldsymbol{z} = (\boldsymbol{z}_{\text{obs}}, \boldsymbol{z}_{\text{mis}})$ be "missing data" which cannot be observed directly and "complete data ", respectively. Even when it is practically infeasible to maximize the likelihood function $P(\boldsymbol{z}_{\text{obs}}|\boldsymbol{\psi})$, it is often easy to maximize the joint likelihood function $P(\boldsymbol{z}|\boldsymbol{c})$ that includes missing data [4, 10]. The EM algorithm iteratively maximizes an expectation of $P(\boldsymbol{z}|\boldsymbol{c})$ by the following steps. We here denote the estimated sequence in $(l-1)$-th iteration by $\boldsymbol{\psi}^{(l)}$ where $\boldsymbol{\psi}^{(1)}$ is chosen arbitrarily.

E step: Calculate the following function.

$$Q(\boldsymbol{\psi}|\boldsymbol{\psi}^{(l)}) = E_{\boldsymbol{z}}\Big[\ln P(\boldsymbol{z}|\boldsymbol{\psi})\big|\boldsymbol{z}_{\text{obs}}, \boldsymbol{\psi}^{(l)}\Big]. \tag{3}$$

M step: Search $\boldsymbol{\psi}^{(l+1)}$ such that

$$\boldsymbol{\psi}^{(l+1)} = \arg\max_{\boldsymbol{\psi}} Q(\boldsymbol{\psi}|\boldsymbol{\psi}^{(l)}). \tag{4}$$

Until $\boldsymbol{\psi}^{(l)} = \boldsymbol{\psi}^{(l+1)}$, the above steps are iteratively carried out. □

In M step, a generalized EM (GEM) algorithm sets $\boldsymbol{\psi}$, which satisfies $Q(\boldsymbol{\psi}|\boldsymbol{\psi}^{(l)}) \geq Q(\boldsymbol{\psi}^{(l)}|\boldsymbol{\psi}^{(l)})$ instead of maximizing $Q(\boldsymbol{\psi}|\boldsymbol{\psi}^{(l)})$, as estimated sequence $\boldsymbol{\psi}^{(l+1)}$. Namely, the EM algorithm is a specific instance of GEM algorithms [10].

3. Soft-Decision Decoding Algorithm over MMGN Channel

3.1. Reliability-based SDD algorithm via GEM algorithm

Let $\boldsymbol{s} = (s_0, s_1, \ldots, s_n) \in \mathcal{S}^{n+1}$ be the state sequence of the channel when a transmitted sequence $\boldsymbol{x}$ is input to the channel where $\boldsymbol{x}$ and $\boldsymbol{s}$ are mutually independent. Then, the likelihood of a codeword $\boldsymbol{c}$ is

$$P(\boldsymbol{r}|\boldsymbol{c}) = \sum_{\boldsymbol{s}\in\mathcal{S}^{n+1}} \prod_{j=1}^{n} p(s_j|\boldsymbol{s}_{j-1}^{(m)}) P(r_j|s_j, c_j). \tag{5}$$

To perform MLD, we have to find the codeword which maximizes eq. (5). However, since the r.h.s. of eq. (5) includes marginalization with all $\boldsymbol{s} \in \mathcal{S}^{n+1}$, the conventional efficient algorithm, such as the Viterbi algorithm, cannot be straightforwardly implemented[4]. Then, we will propose a new SDD algorithm via a GEM algorithm.

We regard a sequence $\boldsymbol{s} \in \mathcal{S}^{n+1}$ as missing data in the GEM algorithm. Similar to eq. (3), we define the

[4] The likelihood of a given $\boldsymbol{c}$ can be computed by the well-known Baum-Welch algorithm [4, 8] with the complexity of $O(n)$.

function $Q(\boldsymbol{c}|\boldsymbol{c}^{(l)})$ at l-th step of the GEM algorithm as

$$Q(\boldsymbol{c}|\boldsymbol{c}^{(l)}) = E_{\boldsymbol{S}}\Big[\ln P(\boldsymbol{r},\boldsymbol{s}|\boldsymbol{c})\Big|\boldsymbol{r},\boldsymbol{c}^{(l)}\Big]. \quad (6)$$

For $j \in [1,n]$, we obtain the hard-decision symbol y_j by eq. (1), where $P(r_j|c_j)$ can be computed by $P(r_j|c_j) = \sum_{i\in\mathcal{S}} p(s_j=i)P(r_j|s_j=i,c_j)$.

Theorem 1 For some $\boldsymbol{c}^{(l)} \in \mathcal{C}$, we define

$$M^{(l)}(\boldsymbol{c}) = \sum_{j|\,c_j\neq y_j}\sum_{i\in\mathcal{S}} \frac{p(s_j=i|\boldsymbol{r},\boldsymbol{c}^{(l)})}{\sigma^2(i)}|2r_j|. \quad (7)$$

Then, for $\boldsymbol{c}$ and $\boldsymbol{c}'$,

$$Q(\boldsymbol{c}|\boldsymbol{c}^{(l)}) \geq Q(\boldsymbol{c}'|\boldsymbol{c}^{(l)}) \quad \text{iff} \quad M^{(l)}(\boldsymbol{c}) \leq M^{(l)}(\boldsymbol{c}'). \quad (8)$$

Therefore, for arbitrary subcode $\mathcal{C}'$,

$$\arg\max_{\boldsymbol{c}\in\mathcal{C}'} Q(\boldsymbol{c}|\boldsymbol{c}^{(l)}) = \arg\min_{\boldsymbol{c}\in\mathcal{C}'} M^{(l)}(\boldsymbol{c}). \quad (9)$$

□

Proof: See appendix A.

By Theorem 1, maximizing $Q(\cdot|\boldsymbol{c}^{(l)})$ can be carried out by minimizing $M^{(l)}(\cdot)$. We here define

$$\phi_j^{(l)} = 2r_j \sum_{i\in\mathcal{S}} \frac{p(s_j=i|\boldsymbol{r},\boldsymbol{c}^{(l)})}{\sigma^2(i)}, \quad j\in[1,n]. \quad (10)$$

Then eq. (7) is now $M^{(l)}(\boldsymbol{c}) = \sum_{j|\,c_j\neq y_j} |\phi_j^{(l)}|$ which has similar expression to eq. (2). Therefore, in the l-th M step, we can efficiently search $\boldsymbol{c}^{(l+1)}$ which decreases $M^{(l)}(\cdot)$ by the MLD algorithm described in Sect. 2.2, which minimizes $L(\cdot)$. In this search, $|\phi_j^{(l)}|, j\in[1,n]$, is regarded as j-th reliability.

We describe the proposed SDD algorithm below, where $\boldsymbol{c}^{(1)}$ is some binary sequence.

[The proposed SDD algorithm]

E step: For $j\in[1,n]$ and $i\in\mathcal{S}$, compute $p(s_j = i|\boldsymbol{r},\boldsymbol{c}^{(l)})$ by the Baum-Welch algorithm [4, 8]. For $j\in[1,n]$, we obtain $|\phi_j^{(l)}|$ by eq. (10).

M step: Search $\boldsymbol{c}^{(l+1)}$ satisfying the following equation via an MLD algorithm[5] of Sect. 2.2.

$$M^{(l)}(\boldsymbol{c}^{(l+1)}) \leq M^{(l)}(\boldsymbol{c}^{(l)}). \quad (11)$$

The above steps are iterated until $\boldsymbol{c}^{(l)} = \boldsymbol{c}^{(l+1)}$, then $\boldsymbol{c}^{(l)}$ is output as $\boldsymbol{c}_{\text{best}}$. □

The proposed SDD algorithm is not guaranteed to converge the global maxima because of the EM principal [10].

[5] We will describe the algorithm in detail in the next section.

In the l-th M step, the reliability $|\phi_j^{(l)}|, \forall j\in[1,n]$, satisfies

$$\begin{aligned}\phi_j^{(l)} &= E_{s_j}\Big[\ln\frac{P(r_j|s_j,c_j=0)}{P(r_j|s_j,c_j=1)}\Big|\boldsymbol{r},\boldsymbol{c}^{(l)}\Big] \\ &= \sum_{i\in\mathcal{S}} p(s_j=i|\boldsymbol{r},\boldsymbol{c}^{(l)})\ln\frac{P(r_j|s_j=i,c_j=0)}{P(r_j|s_j=i,c_j=1)}. \quad (12)\end{aligned}$$

i.e., $\phi_j^{(l)}$ is expectation of joint log likelihood ratio of s_j and c_j with $p(s_j=i|\boldsymbol{r},\boldsymbol{c}^{(l)})$. The derivation of eq. (12) will be given in Appendix B.

3.2. The l-th M Step of the Proposed Decoding Algorithm

In this section, we describe the l-th M step of the proposed SDD algorithm.

Let $\tilde{G}^{(l)}$ be a permuted generator matrix whose leftmost k columns are MRI in non-increasing reliabilities $|\phi_j^{(l)}|$. We denote the best codeword obtained so far by $\tilde{\boldsymbol{c}}^*$. The algorithm searches candidate codewords satisfying $M^{(l)}(\tilde{\boldsymbol{c}}) \leq M^{(l)}(\tilde{\boldsymbol{c}}^*)$ and the most likely codeword among them is set to $\tilde{\boldsymbol{c}}^{(l+1)}$.

For a TEP $\boldsymbol{t} = (t_1, t_2, \ldots, t_k)$, we define a evaluation function of $\boldsymbol{t}$ as

$$\Delta^{(l)}(\boldsymbol{t}) = \sum_{j|t_j=1} |\tilde{\phi}_j^{(l)}|. \quad (13)$$

Lemma 1 (Elimination of a TEP) In the l-th M step, assume that a generated TEP $\boldsymbol{t}$ satisfies

$$\Delta^{(l)}(\boldsymbol{t}) \geq M^{(l)}(\tilde{\boldsymbol{c}}^*). \quad (14)$$

Then the candidate codeword $\tilde{\boldsymbol{c}} = (\boldsymbol{u}\oplus\boldsymbol{t})\tilde{G}^{(l)}$ given by $\boldsymbol{t}$ satisfies $M^{(l)}(\tilde{\boldsymbol{c}}) \geq M^{(l)}(\tilde{\boldsymbol{c}}^*)$.

Proof: From the definitions of $M^{(l)}(\boldsymbol{c})$ and $\Delta^{(l)}(\boldsymbol{t})$,

$$\begin{aligned}M^{(l)}(\tilde{\boldsymbol{c}}) &= \sum_{j|t_j=1}|\tilde{\phi}_j^{(l)}| + \sum_{j=n-k+1}^{n}(\tilde{y}_j\oplus\tilde{c}_j)|\tilde{\phi}_j^{(l)}| \\ &= \Delta^{(l)}(\boldsymbol{t}) + \sum_{j=n-k+1}^{n}(\tilde{y}_j\oplus\tilde{c}_j)|\tilde{\phi}_j^{(l)}|. \quad (15)\end{aligned}$$

Therefore, $M^{(l)}(\tilde{\boldsymbol{c}}) \geq \Delta^{(l)}(\boldsymbol{t})$. Eq. (14) implies $M^{(l)}(\tilde{\boldsymbol{c}}) \geq M^{(l)}(\tilde{\boldsymbol{c}}^*)$. □

From Lemma 1, we need not encode a TEP satisfying eq. (14) and the next TEP is generated[6]. In order to judge if the l-th M step can be terminated, we have the following theorem which is readily proven by Lemma 1.

[6] Although more effective evaluation functions than $\Delta^{(l)}(\cdot)$ are devised in [1, 2, 3], we will not describe them for simplicity. However, they can be also applicable to the proposed decoding algorithm in this paper.

Theorem 2 (Condition of Local Termination)
In the l-th M step, let $\mathcal{T}^{(l)}$ be a set of TEPs not generated yet. If

$$\min_{\boldsymbol{t}\in\mathcal{T}^{(l)}} \{\Delta^{(l)}(\boldsymbol{t})\} \geq M^{(l)}(\tilde{\boldsymbol{c}}^*), \tag{16}$$

then the algorithm outputs $\tilde{\boldsymbol{c}}^{(l+1)} = \tilde{\boldsymbol{c}}^*$. □

We can use eqs. (14) and (16) to reduce the number of searched candidate codewords.

We describe the the l-th M Step of the proposed SDD algorithm.

[The l-th M Step of the Proposed Algorithm]

S1) Construct $\tilde{G}^{(l)}$ and generate the initial codeword $\tilde{\boldsymbol{c}}_0 := \boldsymbol{u}\tilde{G}^{(l)}$. If $P(\boldsymbol{r}|\boldsymbol{c}^{(l)}) \geq P(\boldsymbol{r}|\boldsymbol{c}_0)$, then set $\tilde{\boldsymbol{c}}_{\text{best}} := \tilde{\boldsymbol{c}}^{(l)}$, otherwise set $\tilde{\boldsymbol{c}}_{\text{best}} := \tilde{\boldsymbol{c}}_0$. Set $P_{\text{best}} := P(\boldsymbol{r}|\boldsymbol{c}_{\text{best}})$, $M_{\text{best}} := M^{(l)}(\tilde{\boldsymbol{c}}_{\text{best}})$.

S2) Generate a TEP $\boldsymbol{t} \in \mathcal{T}^{(l)}$ and set $\mathcal{T}^{(l)} := \mathcal{T}^{(l)} \setminus \boldsymbol{t}$.

a) If eq. (14) holds for $\boldsymbol{t}$, then go to step S3).

b) Set $\tilde{\boldsymbol{c}} := (\boldsymbol{u} \oplus \boldsymbol{t})\tilde{G}$. If $P_{\text{best}} \geq P(\boldsymbol{r}|\boldsymbol{c})$, then set $P_{\text{best}} := P(\boldsymbol{r}|\boldsymbol{c}), M_{\text{best}} := M^{(l)}(\tilde{\boldsymbol{c}}), \tilde{\boldsymbol{c}}_{\text{best}} := \tilde{\boldsymbol{c}}$.

S3) If eq. (16) is satisfied or $\mathcal{T}^{(l)} = \emptyset$, then output $\tilde{\boldsymbol{c}}^{(l+1)} := \tilde{\boldsymbol{c}}_{\text{best}}$ and terminate the l-th M step. Otherwise go to step S2). □

The proposed SDD algorithm can be terminated by the following conditions: assume that eq. (16) holds and $\tilde{\boldsymbol{c}}^{(l)} = \tilde{\boldsymbol{c}}^*$, then $\tilde{\boldsymbol{c}}^{(l+1)} = \tilde{\boldsymbol{c}}^*$ from Theorem 2. i.e., the SDD algorithm is converged and $\tilde{\boldsymbol{c}}^{(l)}$ is output as the estimated codeword.

4. Evaluation by Simulations

4.1. Conditions

We compare four decoding algorithms: (i) an ideal MLD algorithm given the information of the real state transition sequences (denoted by "Ideal")[7], (ii) the MLD algorithm by exhaustive search (denoted by "MLD"), (iii) the proposed SDD algorithm (denoted by "Proposed"), (iv) the MLD algorithm by regarding noises are AWGN at average SNR [dB] (denoted by "Conventional" or "Conv."). In M step of the proposed SDD algorithm, "Ideal" and "Conventional" algorithms, we generate TEPs according to the method of Gazelle et al. [2]. The initial sequence in the proposed algorithm is set as $\boldsymbol{c}^{(1)} = \boldsymbol{y}$. For each decoding algorithm, at least 10,000 codewords are transmitted until 100 decoding errors occur. We evaluate them by (i) decoding performance (block error rate) and (ii) decoding complexity (the number of searched candidate codewords for each decoding algorithm and the number of iteration for the proposed SDD algorithm[8]).

We assume the first-order Markov Model with $\mathcal{S} = \{0, 1\}$, $p(0) = 0.9$ and $p(1|0) = 0.1$ for the MMGN channel. Furthermore, we assume the variances of the Gaussian distribution of two states satisfy $\sigma^2(1) = \rho\sigma^2(0), \rho \geq 1$.

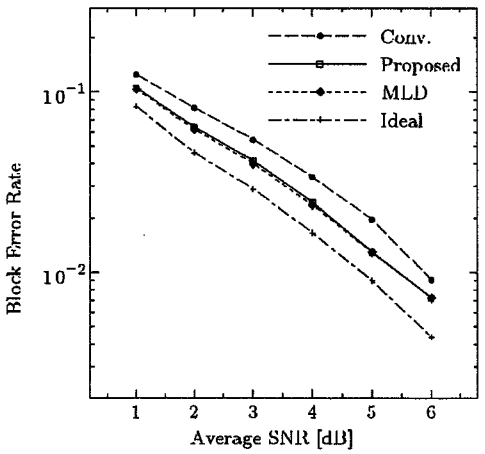


Figure 1: Decoding results for the (24, 12) extended Golay code at $\rho = 5.5$.

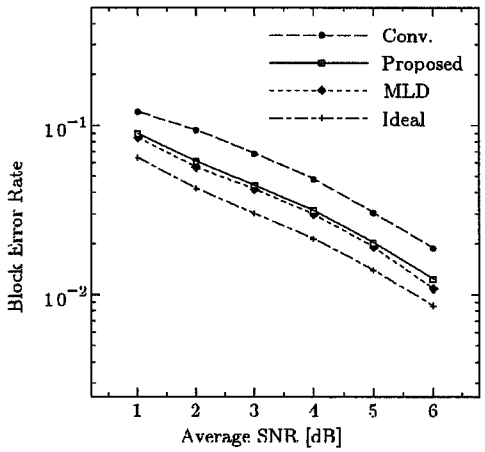


Figure 2: Decoding results for the (24, 12) extended Golay code at $\rho = 8.5$.

4.2. Simulation Results (Decoding Performance)

In Figs. 1, 2 and 3, we show results of decoding performances at $\rho = 5.5, 8.5$ and average SNR 6.0 [dB], respectively, for the (24,12) extended Golay code. In Fig. 4, we show results for the (63,30) BCH code at $\rho = 8.5$

By Fig. 1, block error rates of the MLD and of the proposed SDD algorithms are almost the same at each SNR when $\rho = 5.5$. It can be expected that from small to medium value of ρ, posterior probability $P(s_j|\boldsymbol{r}, \boldsymbol{c}^{(l)})$

[7]It is obvious that results of the "Ideal" algorithm can never be obtained in an actual decoder.

[8]One iteration consists of E and M steps.

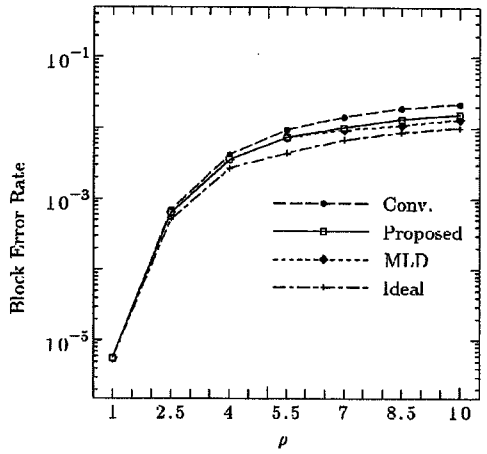


Figure 3: Decoding results for the (24, 12) extended Golay code at average SNR 6.0 [dB].

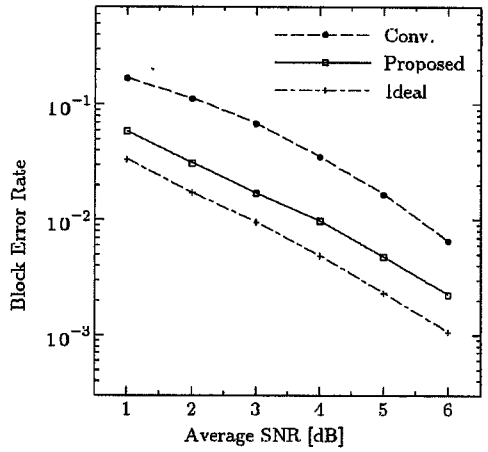


Figure 4: Decoding results for the (63, 30) BCH code at $\rho = 8.5$.

in eqs. (10) and (12) can be estimated highly accurately. By Fig. 2, although we see slight degradation of the performance of the proposed algorithm from MLD at $\rho = 8.5$, it is greatly improved compared with the conventional algorithm. Fig. 3 also shows that the proposed SDD algorithm performs as well as MLD algorithm at each ρ. Note that at $\rho = 1$, the MMGN channel is reduced to the AWGN channel.

By Fig. 4, we see the similar result for the (63,30) BCH code. Remark that the gain of the proposed algorithm from the conventional algorithm is larger than that for the (24,12) extended Golay code. The similar results have obtained at $\rho = 5.5$ and 6.0 [dB] for the (63,30) BCH code.

(Decoding Complexity)

In Tables 1 and 2, we show results of the number of generated candidate codewords for each decoding algorithm at $\rho = 8.5$ for the (24,12) extended Golay code and the (63,30) BCH code, respectively. We also show results of the average (denoted by "ave") and maximum (denoted by "max") number of iterations for the proposed SDD algorithm.

Table 1: The number of candidate codewords for each decoding algorithm and the number of iteration for the (24, 12) extended Golay code at $\rho = 8.5$

SNR [dB]	codewords			iterations	
	MLD	Conv.	Proposed	ave	max
1.0	$3.55 \cdot 10^3$	5.93	9.67	1.74	3
2.0	$3.17 \cdot 10^3$	4.96	7.52	1.55	3
3.0	$2.66 \cdot 10^3$	4.24	5.79	1.30	3
4.0	$2.09 \cdot 10^3$	3.42	4.48	1.02	3
5.0	$1.54 \cdot 10^3$	2.61	3.10	0.754	3
6.0	$1.14 \cdot 10^3$	1.86	1.99	0.558	3

Table 2: The number of candidate codewords for each decoding algorithm and the number of iteration for the (63, 30) BCH code at $\rho = 8.5$

SNR [dB]	codewords		iterations	
	Conv.	Proposed	ave	max
1.0	$4.44 \cdot 10^3$	$1.47 \cdot 10^4$	1.99	3
2.0	$3.86 \cdot 10^3$	$9.55 \cdot 10^3$	1.97	3
3.0	$3.19 \cdot 10^3$	$6.67 \cdot 10^3$	1.88	3
4.0	$2.19 \cdot 10^3$	$3.49 \cdot 10^3$	1.69	3
5.0	$1.33 \cdot 10^3$	$1.26 \cdot 10^3$	1.42	3
6.0	$5.94 \cdot 10^2$	$3.94 \cdot 10^2$	1.14	3

Table 1 shows that the number of generated candidate codewords for the proposed SDD algorithm at 1.0 [dB] is less than 10 which is less than twice that of the conventional algorithm. As for the number of iterations, the average and maximum numbers are no more than two and three, respectively.

Table 2 also shows the effectiveness of the proposed SDD algorithm, whose searched codewords is at most less than four times that for the conventional algorithm, for the (63,30) BCH code at each average SNR. It is noteworthy that, at high average SNRs, the number of searched codewords in the proposed SDD algorithm is less than that for the conventional algorithm. Remark that the exhaustive MLD algorithm cannot be conducted because of its large dimension. As for the number of iterations, the behavior is not different from the case of (24,12) extended Golay code, so we can say that the proposed algorithm is also effective for large codes.

We have the similar results at $\rho = 5.5$ for both codes. These results indicate the effectiveness of Lemma 1 and Theorem 2.

5. Concluding Remarks

In this paper, we propose a new SDD algorithm of block codes over the MMGN channel via a GEM algorithm. The proposed algorithm has an effective termination condition in each iteration step. We show by simulations that the proposed algorithm achieves

near MLD with relatively small complexity.

As for further works, we need to devise a method for eliminating unnecessary iterations of the proposed algorithm. Since the proposed SDD algorithm largely depends on the initial sequence $\boldsymbol{c}^{(1)}$, a measure for selecting a good initial sequence is also needed.

Acknowledgment

The one of the authors, H. Yagi, would like to thank Dr. M. Kobayashi at Shonan Institute of Technology and Mr. T. Ishida and Mr. G. Hosoya at Waseda University for their valuable comments.

Appendix A: Proof of Theorem 1

The function $Q(\boldsymbol{c}|\boldsymbol{c}^{(l)})$ for some $\boldsymbol{c}^{(l)} \in C$ is expanded as:

$$Q(\boldsymbol{c}|\boldsymbol{c}^{(l)}) = \sum_{\boldsymbol{s}\in S^{n+1}} p(\boldsymbol{s}|\boldsymbol{r},\boldsymbol{c}^{(l)}) \ln P(\boldsymbol{r},\boldsymbol{s}|\boldsymbol{c}) = \sum_{\boldsymbol{s}\in S^{n+1}} p(\boldsymbol{s}|\boldsymbol{r},\boldsymbol{c}^{(l)}) \big\{ \ln P(\boldsymbol{r}|\boldsymbol{s},\boldsymbol{c}) + \ln p(\boldsymbol{s}) \big\}, \quad (17)$$

where the last equation is lead by $p(\boldsymbol{s}|\boldsymbol{c}) = p(\boldsymbol{s})$. The first term of eq. (17), which only depends on a choice of $\boldsymbol{c}$, is further expanded as follows:

$$\sum_{\boldsymbol{s}\in S^{n+1}} p(\boldsymbol{s}|\boldsymbol{c}^{(l)},\boldsymbol{r}) \ln P(\boldsymbol{r}|\boldsymbol{s},\boldsymbol{c}) = \sum_{j=1}^{n} \sum_{i\in S} p(s_j = i|\boldsymbol{c}^{(l)},\boldsymbol{r}) \ln P(r_j|s_j = i, c_j). \quad (18)$$

From the assumption of Gaussian distribution at each state $s_j \in S$,

$$\ln P(r_j|s_j = i, c_j) = \Big\{ - \frac{(r_j - (-1)^{c_j})^2}{2\sigma^2(i)} \Big\} + D \quad (19)$$

where D is the independent term of $\boldsymbol{c}$. Substituting the r.h.s. of eq. (19) into eq. (18),

$$\begin{aligned} &\sum_{\boldsymbol{s}\in S^{n+1}} p(\boldsymbol{s}|\boldsymbol{c}^{(l)},\boldsymbol{r}) \ln P(\boldsymbol{r}|\boldsymbol{s},\boldsymbol{c}) \\ &= \sum_{j=1}^{n} \sum_{i\in S} p(s_j = i|\boldsymbol{c}^{(l)},\boldsymbol{r}) \Big\{ - \frac{(r_j - (-1)^{c_j})^2}{2\sigma^2(i)} + D \Big\} \\ &= \sum_{j=1}^{n} \sum_{i\in S} \frac{p(s_j = i|\boldsymbol{c}^{(l)},\boldsymbol{r})}{\sigma^2(i)} r_j (-1)^{c_j} + D', \end{aligned} \quad (20)$$

where D' is the independent term of $\boldsymbol{c}$. Then from eqs. (17) and (20), we have

$$\begin{aligned} Q(\boldsymbol{c}|\boldsymbol{c}^{(l)}) &= \sum_{j=1}^{n} \sum_{i\in S} \frac{p(s_j = i|\boldsymbol{c}^{(l)},\boldsymbol{r})}{\sigma^2(i)} r_j (-1)^{c_j} + D'', \\ &= \sum_{j=1}^{n} \sum_{i\in S} \frac{p(s_j = i|\boldsymbol{c}^{(l)},\boldsymbol{r})}{\sigma^2(i)} |r_j| - M^{(l)}(\boldsymbol{c}) + D'', \end{aligned} \quad (21)$$

where the second term D'' is independent of $\boldsymbol{c}$ and eq. (21) indicates eq. (9). The proof of eq. (8) is straightforward from eq. (9).

Appendix B: Derivation of eq. (12)

From Gaussian distribution of eq. (19), we have

$$\begin{aligned} &\sum_{i\in S} p(s_j = i|\boldsymbol{r},\boldsymbol{c}^{(l)}) \ln \frac{P(r_j|s_j = i, c_j = 0)}{P(r_j|s_j = i, c_j = 1)} \\ &= \sum_{i\in S} p(s_j = i|\boldsymbol{r},\boldsymbol{c}^{(l)}) \\ &\quad \times \Big\{ - \frac{1}{2\sigma^2(i)}(r_j - 1)^2 + \frac{1}{2\sigma^2(i)}(r_j + 1)^2 \Big\} \\ &= \sum_{i\in S} p(s_j = i|\boldsymbol{r},\boldsymbol{c}^{(l)}) \frac{2r_j}{\sigma^2(i)}. \end{aligned} \quad (22)$$

Eq. (22) and the definition of $\phi_j^{(l)}$ prove eq. (12).

References

[1] M. P. C. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Trans. Inform. Theory*, vol.41, no.5, pp.1379 – 1396, Sept. 1995.

[2] D. Gazelle and J. Snyders, "Reliability-based code-search algorithm for maximum-likelihood decoding of block codes," *IEEE Trans. Inform. Theory*, vol.43, pp.239 – 249, Jan. 1997.

[3] Y. S. Han, C. P. R. Hartman, and C. C. Chen, "Efficient priority-first search maximum-likelihood soft decision decoding of linear block codes," *IEEE Trans. Inform. Theory*, vol.39, no.5, pp.1514 – 1523, Sept. 1993.

[4] A. Logothetis and V. Krishnamurthy, "Expectation Maximization algorithms for MAP estimation of jump Markov linear systems," *IEEE Trans. Signal Pro.*, vol.47, no.8, pp.2139 – 2156, Aug. 1999.

[5] W. Turin, "MAP decoding in Channels with Memory," *IEEE Trans. Commn.*, vol.48, no.5, pp.757 – 763, May 2000.

[6] W. Turin, "MAP symbol decoding in Channels with Error Bursts," *IEEE Trans. Inform. Theory*, vol.47, no.5, pp.1832 – 1838, July 2001.

[7] A. Valembois and M. P. C. Fossorier, "An improved method to compute lists of binary vectors that optimize a given weight function with application of soft-decision decoding," *IEEE Commn. Letters*, vol.5, no.11, pp.456 – 458, Nov. 2001.

[8] T. Wadayama, "An iterative decoding algorithm of low density parity check codes for hidden Markov noise channel," *Proc. of Int. Symposium on Information Theory and Its Applications*, Hawaii, U.S.A., Nov. 2000.

[9] T. Wadayama, "An iterative decoding algorithm for channels with Gibbs distributed noise," *Proc. of IEEE Int. Symposium on Information Theory*, Yokohama, Japan, June 2003.

[10] C. F. J. Wu, "On the convergence properties of the EM algorithm," *Ann. Statist.*, vol.11, no.1, pp.95 – 103, Jan. 1983.

The 27th Symposium on Information Theory
and Its Applications (SITA2004)
Gero, Gifu, Japan, Dec. 14-17, 2004

Universal Coding Algorithm for Side Information Context Tree Models

Toshiyasu Matsushima * Shigeich Hirasawa*

Abstract— The problem of universal codes with side information is investigated from Bayes criterion. We propose side information context tree models which are an extension of context tree models to sources with side information. Assuming a special class of the prior distributions for side information context tree models, we propose an efficient algorithm of Bayes code for the models. The asymptotic code length of the Bayes codes with side information is also investigated

Keywords— Source coding with side information, Bayes universal codes, Context tree models

1 Introduction

The problem of source coding with side information in this paper is defined as follows. A side information sequence y^n and a main information sequence x^n are occurred from an information source. A sender sends the side information sequence y^n and the code word $c(x^n)$ that is encoded from x^n and y^n. A receiver decodes the main information sequence x^n from $c(x^n)$ and y^n.

Universal codes with side information have been proposed by Ziv[1], Muramatsu[2], Subarahmanya[3], Yan[4], Uematsu[5] and et al. The asymptotically optimality for some of these codes was proved about i.i.d. and stationary ergodic sources.

Although the asymptotically optimal criterion is used for evaluation of universal codes, universal codes have been evaluated by Bayes, maxmin and minmax redundancy in some previous research. The asymptotic redundancy of the Bayes codes for i.i.d. source was investigated by Clark and Barron[6], and that for context tree models was studied by Gotoh et al.[10]. Efficient Bayes coding algorithms were also investigated for context tree models[7][9]. The CTW(Context Tree Weighting) algorithm[7] was interpreted as a special case of the Bayes coding algorithms.

In this paper, we propose side information context tree models which are an extension of context tree models to sources with side information. Secondly, we deduce universal codes with side information from Bayes criterion. Using a special class of prior distributions for side information context tree models, we propose an efficient algorithm of the Bayes code. Moreover, the asymptotic code length of Bayes codes with side information is also investigated

*Waseda University E-mail: toshi@matsu.mgmt.waseda.ac.jp

2 Preliminary

2.1 Context tree models

The context tree model is a finite state source whose state at t is determined by the postfix of a source sequence x^t. The state set of a context tree model is a complete postfix set. Hence they satisfy the property that no string is a postfix of another. Let m be a context tree model. The state set of m is represented by a $|X|$-ary complete tree T_m^x called a context tree. Each arc in the tree corresponds to a symbol $x \in X$. A path from a leaf to the root in the tree represents a postfix or a context in the model. Let S_m^x be the set of all states in m. S_m^x corresponds to the set of all leaf nodes in the context tree T_m^x. The state $s \in S_m^x$ of a context tree model m at t is determined by the source sequence x^t. This mapping from x^t to a state $s \in S_m^x$ is denoted by $s_m^x(x^t)$.

The conditional probability of x_t of a context tree model is given by the following probability:

$$P(x_t|x^{t-1}) = P(x_t|\theta_{s_m^x(x^{t-1})}, s_m^x(x^{t-1})). \quad (1)$$

A context tree model m is represented by a context set or a leaf set S_m^x and probabilistic parameters $\theta_m^x = \{\theta_s | s \in S_m^x\}$.

2.2 Bayes codes and a special class of the prior probability of context tree models

If the distribution of source sequences $P(x^n)$ is given, there are some coding algorithm such as the arithmetic coding algorithms that encodes the sequences to the code words whose mean code length converges to its entropy $H(X^n)$. Under the condition that such coding algorithm is used for coding, the decision of coding probability $P_C(x^n)$ is the main problem in universal coding. The optimal coding probability of x^n under Bayes criterion is given by

$$P_C(x^n) = \sum_m \int P(x^n|\theta_m, m) P(\theta_m|m) P(m) d\theta_m. \quad (2)$$

Although the time complexity of the Bayes codes for the class of context tree modes whose depth is up to d is $O(2^{|X|^{d-1}})$, if a special class of prior distribution $P(m)$ is assumed, the time complexity is reduced to $O(d)$).

Assumption 1 *We assume that the prior probability $P(m)$ or $P(S_m)$ is represented by the following formula[8]:*

$$P(m) = \prod_{s \in N_m - S_m} (1 - q(s)) \prod_{s_L \in S_m} q(s_L), \quad (3)$$

where N_m denotes the set of all nodes on the context tree model T_m^x and

$$q(s) = \frac{P(S_{m_s})}{\sum_{S \in S^d} P(S_{m_s} \cup S)}, \quad (4)$$

where $m_s \in \{m | s \in S_m\}$, S_d denotes the class of the set of descendant nodes of s that construct a complete tree.

Generally, the right hand side of Formula (4) might have different value at each m_s. However, it has the same value at all m_s under this assumption. Although the assumption restricts the class of prior distributions of context tree models, the complexity for calculating its posterior probability is reduced drastically. The prior probability of m is represented by $\{q(s)|s \in N_m\}$. Moreover, the posterior probability $P(m|x^t)$ is also represented by $\{q(s|x^t)|s \in N_m\}$. This is the mechanism for reducing the computational complexity.

Under the assumption, the conditional coding probability $P_c(x_t|x^{t-1})$ is calculated by the following recursive formulas.

$$P_c(x_t|x^{t-1}) = P_c(x_t|s = s_0) \quad (5)$$

$$P_c(x_t|s) = q(s|x^{t-1})P(x_t|s) + (1 - q(s|x^{t-1})) \prod_{x \in X} P_c(x_t|xs), \quad (6)$$

where xs denotes the context that produced by connecting a symbol x and a context s, and

$$P_c(x_t|xs) = \begin{cases} P_c(x_t|xs) & \text{if } xs \in S(x^t) \\ \arg\max_{\tau \in S_\tau} P_c(x_\tau|xs) & \text{otherwise,} \end{cases} \quad (7)$$

where $S(x^t)$ denotes the set of all $s(x^t)$ ' s and $S_\tau = \{\tau < t | xs \in S(x^\tau)\}$.

The memory structure of the algorithm is represented by a tree whose each node corresponds to a state s that holds $P(x|s)$ and $q(s|x^t)$. The calculation is done through the path from the leaf $s^x(x^{t-1})$ to the root s_0 on the tree. So the time complexity for the calculation is $O(d)$.

The posterior probability $q(s|x^t)$ is updated by the following formula:

$$q(s|x^t) = \frac{P(x_t|s)q(s|x^{t-1})}{P_c(x_t|s)}. \quad (8)$$

The update can be calculated simultaneously with the calculation of the conditional coding probability $P_c(x_t|x^{t-1})$ through the path from the leaf to the root.

The asymptotic code length of the Bayes codes[10] is given by

$$E_{X^n}[-\log P_c(X^n)] = H(X^n|\theta^*_{m^*}, m^*) + \frac{k}{2}\log\frac{n}{2\pi e} + \log\frac{\sqrt{\det I_X(\theta^*_{m^*})}}{p(\theta^*_{m^*})} + o(1), \quad (9)$$

where m^* is a true model, the true parameters $\theta^*_{m^*}$ of the model and $P(\theta_m)$ is the prior probability of θ_m, and k is the number of the parameters.

The CTW algorithm is interpreted as a special case of the Bayes codes whose prior probability $q(s) = 1/2$.

3 Side information context tree models

We extend context tree models for source X^n to models for source X^n with side information Y^n. Let S_m^y be the set of all states for side information Y^n in m. S_m^y corresponds to the set of all leaf nodes in the context tree T_m^y. The state $s_m^y(y^t)$ for side information at t in a side information context tree model m is determined by the postfix of a source sequence y^t.

Let $S_m^{x|s^y}$ be the set of all states for main information sequence X^n given a side information state $s^y \in S_m^y$ in m. $S_m^{x|s^y}$ corresponds to the set of all leaf nodes in the conditional context tree $T_m^{x|s^y}$. The state $s_m^x(x^t|s^y)$ for main information given s^y at t is determined by the postfix of a source sequence x^t and y^t.

The conditional probability of x_{t+1} given (x^t, y^t) of a side information context tree model m is given by

$$P(x_{t+1}|x^t, y^t) = P(x_{t+1}|s_m^x(x^t|s^y), \theta_{s_m^x(x^t|s^y)}, s_m^y(y^t), m). \quad (10)$$

A side information context tree model m is represented by context sets S_m^y, $\{S_m^{x|s^y}|s^y \in S_m^y\}$ and probabilistic parameters $\theta_m^{x|y} = \{\theta_{s^x|s^y}|s^x \in S_m^{x|s^y}, s^y \in S_m^y\}$, $\theta_m^y\{\theta_{s^y}|s^y \in S_m^y\}$.

Example 1 *An example of a context tree T^y and a conditional context tree $T^{x|s^y}$ of a side information context tree model is shown in Fig. 1. Under the condition that $y^{t-1} = \cdots 10$ and $x^{t-1} = \cdots 1$, the context or the state for side information corresponds to the node s_2^y on the context tree T^y and the state for main information corresponds to the node $s_2^{x|s^y}$ on the conditional context tree $T^{x|s^y}$. The conditional probability of x^t is represented by*

$$P(x_t|x^{t-1} = \cdots 1, y^{t-1} = \cdots 10) = P(x_t|S_2^{x|S_2^y}, S_2^y). \quad (11)$$

4 Bayes codes and an efficient algorithm for side information context tree models

4.1 Bayes universal codes with side information

We assume the prior probability of parameters $\theta_m^{x|y}, \theta_m^y$ as $P(\theta_m^{x|y}, \theta_m^y|m) = P(\theta_m^{x|y}|m)P(\theta_m^y|m)$. Under the assumption, the optimal Bayes coding probability of x^n for side information context tree models is given by

$$P_C(x^n|y^n) = \sum_m \int P(x^n|y^n, \theta_m^{x|y}, m)P(\theta_m^{x|y}|m)P(m)d\theta_m^{x|y}. \quad (12)$$

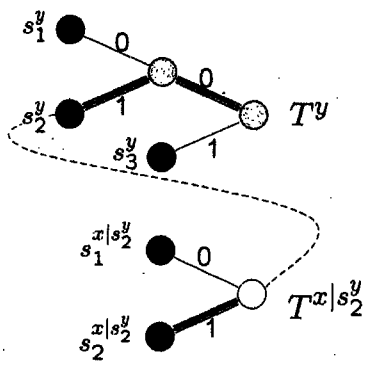


Figure 1: An example of side information context tree models

4.2 An efficient algorithm for side information context tree models

Under general prior probability of side information context tree models, it is very hard to calculate the coding probability $P_C^B(x^n|y^n)$ of the Bayes code. However, assuming a special class of the prior probability $P(m)$, its time complexity can be reduced.

Assumption 2 *We assume that the prior probability $P(m)$ of a side information context tree model is represented by the following formula:*

$$\begin{aligned} & P(m) \\ &= \prod_{s^y \in N_m^y - S_m^y} (1 - q(s^y)) \prod_{s_L \in S_m^y} q(s_L^y) \\ &\quad \prod_{s^x \in N_m^{x|s_L^y} - S_m^{x|s_L^y}} (1 - q(s^x|s_L^y)) \prod_{s_L^x \in S_m^{x|s_L^y}} q(s_L^x|s_L^y). \end{aligned} \tag{13}$$

Under the assumption, the conditional coding probability $P_c(x_t|x^{t-1}, y^{t-1})$ is calculated by the following double recursive formulas.

First, $P_c(x_t|s^y)$ is calculated through the path from the leaf $s^x(x^{t-1}|s^y)$ to the root $s_0^{x|s^y}$ recursively on each conditional context tree $T^{x|s^y}$ given s^y.

$$P_c(x_t|s^y) = P_c(x_t|s^x = s_0^x, s^y) \tag{14}$$

$$\begin{aligned} P_c(x_t|s^x, s^y) &= q(s^x|x^{t-1}, y^{t-1}, s^y)P(x_t|s^x, s^y) + \\ &\quad (1 - q(s^x|x^{t-1}, y^{t-1}, s^y)) \\ &\quad \prod_{x \in X} P_c(x_t|xs^x, s^y). \end{aligned} \tag{15}$$

Secondly, $P_c(x_t|x^{t-1}, y^{t-1})$ is calculated through the path from the leaf $s^y(y^{t-1})$ to the root s_0^y recursively by using $P_c(x_t|s^y)$ on the context tree T^y of side information Y.

$$P_c(x_t|x^{t-1}, y^{t-1}) = P_c(x_t|s^y = s_0^y) \tag{16}$$

$$\begin{aligned} P_c(x_t|s^y) &= q(s^y|x^{t-1}, y^{t-1})P(x_t|s^y) + \\ &\quad (1 - q(s^y|x^{t-1}, y^{t-1})) \\ &\quad \prod_{x \in Y} P_c(x_t|ys^y). \end{aligned} \tag{17}$$

The posterior probability $q(s^x|s^y, x^t, y^t)$ is updated by the following formula:

$$q(s^x|s^y, x^t, y^t) = \frac{q(s^x|x^{t-1}, y^{t-1}, s^y)P(x_t|s^x, s^y)}{P_c(x_t|s^x, s^y)}. \tag{18}$$

The update can be calculated simultaneously with the calculation of $P_c(x_t|s^x, s^y)$ through the path from the leaf $s^x(x^{t-1}|s^y)$ to the root $s_0^{x|s^y}$ on the each conditional context tree $T^{x|s^y}$ given s^y.

The posterior probability $q(s^y|x^t, y^t)$ is updated by the following formula:

$$q(s^y|x^t, y^t) = \frac{q(s^y|x^{t-1}, y^{t-1})P(x_t|s^y)}{P_c(x_t|s^y)}. \tag{19}$$

The update can be calculated simultaneously with the calculation of $P_c(x_t|s^y)$ on the conditional context tree T^y of side information.

Example 2 *We assume that the class of the context tree models T^y whose depth are up to 2 and the class of the conditional context tree models $T^{x|s^y}$ whose depth are up to 2 as shown in Fig. 2. Under the condition that $y^{t-1} = \cdots 10$ and $x^{t-1} = \cdots 11$, the coding probability $P_c(x_t|x^{t-1}, y^{t-1})$ is calculated as the following procedure.*

First, $P_c(x_t|s_4^y)$ is calculated through the path from the leaf $s_5^x|s_4^y$ to the root $s_0^x|s_4^y$ recursively on the conditional context tree $T^{x|s_4^y}$. In the same way, $P_c(x_t|s_1^y)$ and $P_c(x_t|s_0^y)$ are calculated by using $T^{x|s_1^y}$ and $T^{x|s_0^y}$ respectively.

Secondly, $P_c(x_t|x^{t-1}, y^{t-1})$ is calculated through the path from the leaf s_4^y to the root s_0^y recursively by using $P_c(x_t|s^y)$ on the context tree T^y.

Let the depth of the context tree models T^y be d_y and the depth of the conditional context tree models $T^{x|s^y}$ d_x. The time complexity of the proposed algorithm is $O(d_x d_y)$. If Assumption 2 is not assumed, the time complexity of the Bayes code is $O(2^{|Y|^{d_y-1}}2^{|X|^{d_x-1}}$. In the case where parallel computing is done, the parallel time complexity of the proposed algorithm is $O(d_x + d_y)$. The space complexity is $O(2^{d_x}2^{d_y})$.

5 Mean code length of the Bayes codes with side information

We also investigate asymptotic code length of Bayes codes for side information context tree models.

Definition 1 *Let $E_{X^nY^n}$ denote the expectation with respect to $P(X^n, Y^n|\theta_{m^*}^{x|y^*}, \theta_{m^*}^{y*})$, where $\theta_{m^*}^{x|y^*}$ and $\theta_{m^*}^{y*}$ are the true value of $\theta_m^{x|y}$ and θ_m^y respectively. An information matrix $I_{X|Y}(\theta_m^{x|y})$ is defined by*

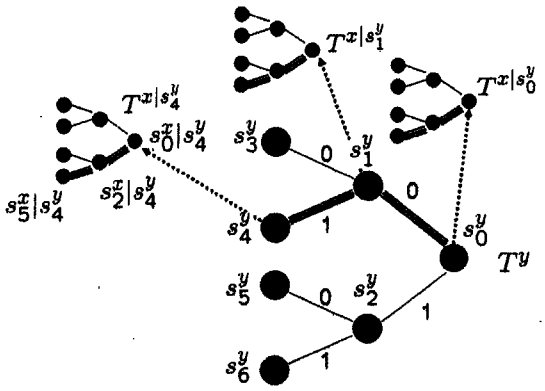


Figure 2: An example of context trees used in the proposed algorithm

$$I_{X|Y}(\theta_m^{x|y}) = -\lim_{n\to\infty}\frac{1}{n}E_{X^nY^n}\left[\frac{\partial^2 \log P(X^n|Y^n,\theta_m^{x|y})}{\partial\theta_m^{x|y}(\partial\theta_m^{x|y})^T}\right] \quad (20)$$

We call $I_{X|Y}$ conditional Fisher information matrix.

Theorem 1 *The average code length of the Bayes codes with side information under some suitable assumption is given by*

$$E_{X^nY^n}[-\log P_c(X^n|Y^n)] = H(X^n|Y^n,\theta_{m^*}^{x|y*},m^*) + \frac{k}{2}\log\frac{n}{2\pi e} + \frac{1}{2}\log\det I_{X|Y}(\theta_{m^*}^{x|y*}) + \log\frac{1}{P(\theta_{m^*}^{x|y*})} + o(1). \quad (21)$$

Although the constant term in the mean code length of an ordinary Bayes code is represented by Fisher information, remark that that of the Bayes codes for side information context is represented by conditional Fisher information.

6 Conclusion

We deduced universal codes with side information from Bayes criterion. Assuming a special prior probability of side information context tree models, we proposed an efficient algorithm of the Bayes code for the models. Moreover, the asymptotic code length of the Bayes codes with side information was investigated.

References

[1] J. Jiv, "Fixed-rate encoding of individual sequences with side information," *IEEE Trans. Inform. Theory*, vol.IT-30, no.2, pp.348-352, March 1984.

[2] J. Muramatsu, "Universal data compression algorithms for stationary ergodic sources based on the complexity of sequences," Ph.D.Thesis, Nagoya University, Nagoya, Japan, 1998.

[3] P. Subrahmanya and T. Berger, "A sliding window Lempel-Ziv algorithm for differential layer encoding in progressive transmission," *Proc. 1995 IEEE Int. Symp. on Inform. Theory*, pp266, Sept. 1995.

[4] E.-H. Yang, A. Kaltchenko, and J.C. Kieffer, "Universal lossless data compression with side information by using a conditional MPM grammer transform," *IEEE Trans. Inform. Theory*, vol.IT-47, pp.2130-2150, Sept. 2001.

[5] T.Uematsu and K.Maeda, "Asymptotic Optimality for Universal Data Compression Algorithm with Side Information Based onIncrementParsing," IEICE A, vol. J85-A, no.1, pp.95-102, Jan. 2002(Japanese).

[6] B.S. Clark and A.R. Brron, "Information-theoretic asymptotics of Bayes methods," *IEEE Trans. Inform. Theory*, vol.IT-36, no.3, pp.453-471, May 1990.

[7] F. M. J. Willems, Y. M. Shtarkov and T. J. Tjalkens, "The Context-Tree Weighting Method:Basic Properties," *IEEE Trans. Inform. Theory*, vol.41, no.3, pp.653-664, May 1995.

[8] T. Matsushima, H. Inazumi and S. Hirasawa, "A Class of Distortionless Codes Designed by Bayes Decision Theory," *IEEE Trans. Inform. Theory*, vol.IT-37, no.5, pp.1288-1293, Sep. 1991.

[9] T. Matsushima and S. Hirasawa, "A Bayes coding algorithm for FSMX sorces," In *Proc. Int. Symp. of Information Theory*, pp.388, 1995.

[10] M. Gotoh, T. Matsushima, S.Hirasawa, "A generalization of B.S.Clarke and A. R. Barron' s asymptotics of Bayes codes for FSMX sources," *IEICE Trans. fundamentals*, vol.E81-A, no.10, pp.2123-2132, Oct. 1998.

[11] T. M. Cover and J. A. Thomas, "Elements of Information Theory," John Wiley & Sons, 1991.

A Note on the Construction of Nonlinear Unequal Orthogonal Arrays from Error-Correcting Codes

Tomohiko Saito Toshiyasu Matsushima Shigeichi Hirasawa

Department of Industrial and Management Systems Engineering
Waseda University
3-4-1 Okubo, Shinjuku-ku, Tokyo, 169-8555, Japan
E-mail: tomohiko@matsu.mgmt.waseda.ac.jp

Abstract Orthogonal arrays have been used in the field of experimental design. Hedayat and Sloane showed the relation between orthogonal arrays and error-correcting codes[1]. And they proposed some construction methods of both linear and nonlinear orthogonal arrays from error-correcting codes. On the other hand, the paper[5] defined unequal orthogonal arrays as new class. It showed that unequal orthogonal arrays are more applicable to experimental design. Furthermore, it showed the relation between unequal orthogonal arrays and unequal error-correcting codes[3], and proposed the construction method of unequal orthogonal arrays from unequal error-correcting codes. But orthogonal arrays from this construction method are all linear. In this paper, we clarify the relation between nonlinear unequal orthogonal arrays and codes. And we propose one of construction methods of nonlinear unequal orthogonal arrays from error-correcting codes.

Key words Experimental Design, Orthogonal Arrays, Unequal Error-correcting Codes, Nonlinear Codes

1 Introduction

Experimental design has been used in many fields, for example in quality management. In experimental design, it is important to design experiments so that we can estimate the effects of factors and their interactions where the number of experiments is as few as possible.

In order to reduce the number of experiments, constructing orthogonal arrays is important[1]. Generally, there are two classes of orthogonal arrays: one is linear orthogonal arrays, the other is nonlinear orthogonal arrays.

Hedayat and Sloane showed the relation between orthogonal arrays and error-correcting codes. And they proposed some construction methods of orthogonal arrays from error-correcting codes[1]. According to their result, its relation can be divided into two types: the first is the relation between linear orthogonal arrays and linear codes, the second is the one between nonlinear orthogonal arrays and nonlinear codes. Furthermore, they proposed some construction methods of orthogonal arrays from linear and nonlinear error-correcting codes.

On the other hand, the paper[5] defined *unequal orthogonal arrays* as new class. From the point of view of the definition, the class with which Hedayat and Slone dealt is *equal orthogonal arrays.* It showed that unequal orthogonal arrays could reduce the number of experiments when we have knowledge about effects of interactions in detail. Furthermore, it showed the relation between unequal arrays and unequal error-correcting codes[3], and proposed construction method of unequal orthogonal arrays from unequal error-correcting codes. But orthogonal arrays from this construction method are all linear.

In this paper, we clarify the relation between nonlinear unequal orthogonal arrays and codes. Furthermore, we extend the construction method of linear unequal orthogonal arrays, and propose one of construction methods of nonlinear unequal orthogonal arrays.

Table 1: $OA(4,3,2,2)$

0	0	0
1	1	0
0	1	1
1	0	1

2 Orthogonal Arrays

2.1 Orthogonal Arrays

Definition2.1:[1] An $M \times n$ array A with entries from $GF(s)$ is said to be an *orthogonal array with s levels and strength t* if every $M \times t$ subarray of A contains each t-tuple based on $GF(s)$ exactly same times as row. We will denote such an array by $OA(M,n,s,t)$.

Example2.1: The array in Table 1 is orthogonal array with strength 2. It is an $OA(4,3,2,2)$.

We consider only the case of $s = 2$.

An orthogonal array $OA(M,n,2,t)$ is said to be linear if the rows of $OA(M,n,2,t)$ form a linear vector

space. If an orthogonal array $OA(M,n,2,t)$ is linear, $OA(M,n,2,t)$ has a basis for the linear vector space. This basis is given in the form of $(\log_2 M) \times n$ matrix called generator matrix.

Next, we show the case that we can reduce the number of experiments when an orthogonal array is applied to an experimental design. Now we consider the case that there is a response variable of interest. And there are three factors F_1, F_2 and F_3 that might affect the response variable. Where each F_i has level $F_{i,0}$ and level $F_{i,1}(i = 1,2,3)$. Then we examine how changes in the levels of the factors affect the response variable. For example, this case is the following case. There is a ratio of defective products as a response variable. And there are three factors that might affect the response variable; the type of catalyst, machine, and material. Where each factor has two levels; catalyst0 and catalyst1, machine0 and machine1 ,and material0 and material1. Then we examine how changes in the levels of the factors affect the response variable.

In the above case, we can estimate all the main effects and all the interaction effects of two factors with the following experimental design:

$$\begin{array}{l}(F_{1,0}F_{2,0}F_{3,0})\\(F_{1,0}F_{2,0}F_{3,1})\\(F_{1,0}F_{2,1}F_{3,0})\\(F_{1,0}F_{2,1}F_{3,1})\\(F_{1,1}F_{2,0}F_{3,0})\\(F_{1,1}F_{2,0}F_{3,1})\\(F_{1,1}F_{2,1}F_{3,0})\\(F_{1,1}F_{2,1}F_{3,1})\end{array} \quad (1)$$

Where the main effect of the factor F_i is measurements which represent how change in the levels of the factor F_i affects the response variable, and the interaction effect of the factors F_i and F_j is measurements which represent how changes in the level combinations of the factors F_i and F_j affect the response variable. Generally, an interaction effect of k factors is measurements which represent how changes in the level combinations of the k factors affect a response variable. And each vector of Eq.(1) corresponds to one time of experiments. For example, $(F_{1,0}, F_{2,0}, F_{3,0})$ correspond to the experiment whose level combination is $F_{1,0}, F_{2,0}, F_{3,0}$. This experimental design contains all level combinations as the vectors. The design such as Eq.(1) is said to be complete design.

Further, we suppose that we know that there is no interaction effect of two factors by experiences. Then we can estimate all the main effect of factors with the following experimental design:

$$\begin{array}{l}(F_{1,0}F_{2,0}F_{3,0})\\(F_{1,0}F_{2,1}F_{3,1})\\(F_{1,1}F_{2,0}F_{3,1})\\(F_{1,1}F_{2,1}F_{3,0})\end{array} \quad (2)$$

This experimental design is made using the orthogonal array $OA(4,3,2,2)$ we show in Table 1. Where, each row of the $OA(4,3,2,2)$ corresponds to the vectors in Eq.(2). For example, 000 which is the first row of the $OA(4,3,2,2)$ correspond to $(F_{1,0}F_{2,0}F_{3,0})$. Then we can reduce the number of experiments using the design that is Eq.(2).

As the above case, when we have knowledge of interaction effects, we can reduce the number of experiments using an orthogonal array. Generally, when an orthogonal array $OA(M,n,2,t)$ is applied to experimental design, the number of experiments is M, and we can estimate interaction effects of at most $[\frac{t}{2}]$ factors.

Next we describe a necessary and sufficient condition for an array to be an orthogonal array.

Theorem 2.1:[1] An $M \times n$ array A with $0,1$ entries is an $OA(M,n,2,t)$ if and only if

$$\sum_{\boldsymbol{u}=row of A} (-1)^{\boldsymbol{u}\cdot\boldsymbol{v}} = 0,$$

for all $0,1$ vectors $\boldsymbol{v}$ containing w 1's, for all w in the range $1 \le w \le t$, where the sum is over all rows u of A.

2.2 Orthogonal Arrays From Codes

2.2.1 Linear Orthogonal Arrays from Linear Codes

Let $w(\boldsymbol{u})$ be the Hamming weight of a vector $\boldsymbol{u} = (u_1, u_2, \ldots, u_n)$. An error-correcting code or simply code is any collection C of vectors in $GF(s)^n$. The vectors in C are called codewords. Let $dist(\boldsymbol{u},\boldsymbol{v})$ be the Hamming distance between two vectors $\boldsymbol{u}$,$\boldsymbol{v}$. We define the minimal distance d of a code C to be the minimal distance between distinct codewords:

$$d = \min_{\boldsymbol{u},\boldsymbol{v}\in C, \boldsymbol{u}\neq\boldsymbol{v}} dist(\boldsymbol{u},\boldsymbol{v}).$$

If C contains M codewords, then we say that it is a code of length n, size M and minimal distance d over $GF(s)$ or simply $(n,M,d)_s$code. We consider the case of $s = 2$ as orthogonal arrays.

C is said to be linear if C is a linear vector subspace. If C is linear, C has the dual code $C^{\perp}$. Let $d^{\perp}$ be the minimal distance of $C^{\perp}$. Then $d^{\perp}$ is said to be a dual distance of C.

Theorem2.2:[1] If C is a $(n,M,d)_2$ linear code over $\{0,1\}$ with dual distance $d^{\perp}$ then the codewords of C form the rows of an $OA(M,n,2,d^{\perp}-1)$ with entries from $\{0,1\}$.

2.2.2 Nonlinear Orthogonal Arrays from Nonlinear Codes

The Group Algebra

we are going to describe binary vectors of length n by polynomials in $z_1, z_2, \ldots, z_n$. For example, $100\ldots0$ will be represented by z_1, $1010\ldots0$ by z_1z_3 and so on. In general $\boldsymbol{v} = v_1v_2\ldots v_n$ is represented by $z_1^{v_1}z_2^{v_2}\ldots z_n^{v_n}$, which we abbreviate $\boldsymbol{z^v}$. We make the convention that $z_i^2 = 1$ for all i. This makes the set of all $\boldsymbol{z^v}$ into a multiplicative group denoted by G. Thus $\{0,1\}^n$ and G are isomorphic groups, with addition in $\{0,1\}^n$

$$\begin{aligned} \boldsymbol{v+w} &= (v_1, v_2, \ldots, v_n) + (w_1, w_2, \ldots, w_n) \\ &= (u_1 + v_1, u_2 + v_2, \ldots, u_n + v_n), \end{aligned}$$

corresponding to multiplication in G:

$$\boldsymbol{z^v z^w} = z_1^{v_1}z_2^{v_2}\cdots z_n^{v_n} \cdot z_1^{w_1}z_2^{w_2}\cdots z_n^{w_n} = \boldsymbol{z^{v+w}}.$$

Definition2.2:[2] The *group algebra* QG of G over the rational numbers Q consists of all formal sums

$$\sum_{\boldsymbol{v}\in\{0,1\}^n} a_{\boldsymbol{v}}\boldsymbol{z^v}, a_{\boldsymbol{v}} \in Q, \boldsymbol{z^v} \in G.$$

Addition and multiplication of elements of QG are defined in the natural way by

$$\sum_{\boldsymbol{v}\in\{0,1\}^n} a_{\boldsymbol{v}}\boldsymbol{z^v} + \sum_{\boldsymbol{v}\in\{0,1\}^n} b_{\boldsymbol{v}}\boldsymbol{z^v} = \sum_{\boldsymbol{v}\in\{0,1\}^n} (a_{\boldsymbol{v}} + b_{\boldsymbol{v}})\boldsymbol{z^v},$$

$$r\sum_{\boldsymbol{v}\in\{0,1\}^n} a_{\boldsymbol{v}}\boldsymbol{z^v} = \sum_{\boldsymbol{v}\in\{0,1\}^n} ra_{\boldsymbol{v}}\boldsymbol{z^v}, r \in Q,$$

and

$$\sum_{\boldsymbol{v}\in\{0,1\}^n} a_{\boldsymbol{v}}\boldsymbol{z^v} \cdot \sum_{\boldsymbol{v}\in\{0,1\}^n} b_{\boldsymbol{w}}\boldsymbol{z^v} = \sum_{\boldsymbol{v},\,\boldsymbol{w}\in\{0,1\}^n} a_{\boldsymbol{v}}b_{\boldsymbol{w}}\boldsymbol{z^v}.$$

To each $\boldsymbol{u} \in \{0,1\}$, we associate the mapping $\chi_{\boldsymbol{u}}$ from G to the rational numbers given by

$$\chi_{\boldsymbol{u}}(\boldsymbol{z^v}) = (-1)^{\boldsymbol{u}\cdot\boldsymbol{v}},$$

where $\boldsymbol{u}\cdot\boldsymbol{v}$ is the scalar product of the vectors $\boldsymbol{u}, \boldsymbol{v}$ over Q. $\chi_{\boldsymbol{u}}$ is called a character of G. $\chi_{\boldsymbol{u}}$ is extended to act on QG by linearity.

$$\begin{aligned} \chi_{\boldsymbol{u}}\Big(\sum_{\boldsymbol{v}\in\{0,1\}^n} a_{\boldsymbol{v}}\boldsymbol{z^v}\Big) &= \sum_{\boldsymbol{v}\in\{0,1\}^n} a_{\boldsymbol{v}}\chi_{\boldsymbol{u}}(\boldsymbol{z^v}) \\ &= \sum_{\boldsymbol{v}\in\{0,1\}^n} (-1)^{\boldsymbol{u}\cdot\boldsymbol{v}} a_{\boldsymbol{v}}. \end{aligned}$$

Let

$$\gamma = \sum_{\boldsymbol{v}\in\{0,1\}^n} c_{\boldsymbol{v}}\boldsymbol{z^v},$$

be an arbitrary element of the group algebra QG with the property that

$$M = \sum_{\boldsymbol{v}\in\{0,1\}^n} c_{\boldsymbol{v}} \neq 0.$$

We call the $(n+1)$-tuple $\{A_0, A_1, \ldots, A_n\}$, where

$$A_i = \sum_{w(\boldsymbol{v})=i} c_{\boldsymbol{v}},$$

the weight distribution of γ.

Definition2.3:[2] The *transform* of γ is the element $\gamma^\perp$ of QG given by

$$\gamma^\perp = \frac{1}{M}\sum_{\boldsymbol{u}\in\{0,1\}^n} \chi_{\boldsymbol{u}}(\gamma)\boldsymbol{z^u}.$$

Nonlinear Orthogonal Arrays from Codes

Now let C be a linear or nonlinear $(n, M, d)_2$code. C is described by the element

$$\gamma = \sum_{\boldsymbol{v}\in C} \boldsymbol{z^v},$$

of QG. Let $\delta = \frac{1}{M}\gamma^2$, and $\delta^\perp$ is the transform of δ. The weight distribution of $\delta^\perp$ is $\{B_0^\perp, B_1^\perp, \ldots, B_n^\perp\}$, where

$$B_i^\perp = \frac{1}{M}\sum_{w(\boldsymbol{u})=i} \chi_{\boldsymbol{u}}(\delta).$$

Definition2.4:[2] The *dual distance* $d^\perp$ of a code C is defined by $B_i^\perp = 0 for 1 \leq i \leq d^\perp - 1$, $B_{d^\perp}^\perp \neq 0$, If C is linear, $d^\perp$ is the minimum distance of $C^\perp$.

Theorem2.3:[1][2] If C is a $(n, M, d)_2$ code over $\{0,1\}$ with dual distance $d^\perp$ then the codewords of C form the rows of an $OA(M, n, 2, d^\perp - 1)$ with entries from $\{0,1\}$.

3 Unequal Orthogonal Arrays

3.1 Unequal Orthogonal Arrays

Definition3.1: An $M \times n$ array A with $0, 1$ entries is said to be an *unequal orthogonal array with* 2 *levels and strength* $\boldsymbol{r} = (r_1, r_2, \ldots, r_n)$ if every $M \times r_i$ subarray of A, which contain ith column of A, contains each r_i-tuple based on $\{0,1\}$ exactly same times as row. We will denote such an array by $OA(M, n, 2, \boldsymbol{r})$.

When $OA(M, n, 2, (r_1, r_2, \ldots r_n))$ is applied to experimental design, we can estimate the effects of interactions of at most$[\frac{r_i}{2}]$ factors which contains ith factor. It was shown that there are cases that unequal array reduce more numbers of experiments than equal orthogonal arrays[5].

For example, this case is the following case. Here, Let $F_i \times F_j$ be the interaction of F_i and F_j. We suppose that there are three factors F_1, F_2 and F_3. And we know that there are $F_1 \times F_2$ and $F_1 \times F_3$. When an equal orthogonal array $OA(M_1, 3, 2, 4)$ is used, we can estimate not only $F_1 \times F_2, F_1 \times F_3$ but $F_2 \times F_3$, although we need not estimate $F_2 \times F_3$. On the other hand, when an unequal orthogonal array $OA(M_2, 3, 2, (4, 2, 2))$ is used, we can not estimate $F_2 \times F_3$. Therefore unequal orthogonal array reduce the number of experiments.

Next we describe a necessary and sufficient condition for an array to be an unequal orthogonal array.

Theorem 3.1: An $M \times n$ array A with $0, 1$ entries is an $OA(M, n, 2, (r_1, r_2, \ldots, r_n))$ if and only if

$$\sum_{\boldsymbol{u}=rowofA} (-1)^{\boldsymbol{u} \cdot \boldsymbol{v}} = 0,$$

for all $0, 1$ vectors $\boldsymbol{v} = (v_1, v_2, \ldots, v_n)$ such that $v_i \neq 0$ and $w(\boldsymbol{v}) = w$ for all w in the range $1 \leq w \leq r_i$, where the sum is over all rows u of A.

3.2 Unequal Orthogonal Arrays from Code

3.2.1 Linear Unequal Orthogonal Arrays from Codes

Linear Unequal Orthogonal Arrays from Codes

The separation $(d_1, d_2, \ldots, d_n)$ of linear code C is defined by

$$d_i = \min\{dist(\boldsymbol{u}, \boldsymbol{v}) \mid \boldsymbol{u} = (u_1, u_2, \ldots, u_n), \boldsymbol{v} = (v_1, v_2, \ldots, v_n), \boldsymbol{u}, \boldsymbol{v} \in C, u_i \neq v_i\}, \quad for \quad i = 1, 2, \ldots, n.$$

Let $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$ be the separation of $C^\perp$ which is the dual code of C. Then $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$ is said to be a dual separation of C.

Theorem3.2: If C is a $(n, M, d)_2$linear code over $\{0, 1\}$ with dual separation $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$, then the codewords of C form the row of an $OA(M, n, 2, (d_1^\perp - 1, d_2^\perp - 1, \ldots, d_n^\perp - 1))$ with entries from {0,1}.

The Construction Method

Now we show the construction method of orthogonal array. This construction method is the method that is applied the construction method of unequal error-correcting code[3] to.

The Construction Method1: Let there be two generator matrix of orthogonal arrays; G_1 is the generator matrix of a linear orthogonal array $OA(M_1, n_1, 2, t_1)$, and G_2 is the one of a linear orthogonal array$OA(M_2, n_2, 2, t_2)$, where $t_2 \leq t_1$. Let G_1 and G_2 be joined as submatrices of G where G_1 and G_2 overlap, as shown in Figure 1. The orthogonal array for which G is generator matrix is $(M_1 M_2) \times (n_1 + n_2 - n_{0L})$ array. Let $n_{0L} \leq t_2/2$.

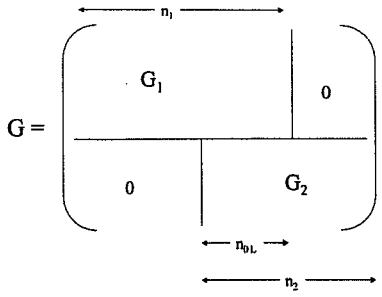


Figure 1: The construction method of linear orthogonal array

Theorem3.3: The orthogonal arrays for which G is generator matrix is $OA(M_1 M_2, n_1 + n_2 - n_{0L}, 2, (r_1, r_2, \ldots r_n))$, where

$$r_i \geq t_1 \quad (i = 1, 2, \ldots, n_1 - n_{0L}),$$
$$r_i \geq t_2 \quad (i = n_1 + 1, n_1 + 2, \ldots, n_1 + n_2 + n_{0L}),$$
$$r_i \geq t_1 + t_2 - n_{0L} \quad (i = n_1 - n_{0L} + 1, \ldots n_1).$$

4 Nonlinear Unequal Orthogonal Arrays

Nonlinear Unequal Orthogonal Arrays from Codes

Let C be a linear or nonlinear $(n, M, d)_2$code and

$$\gamma = \sum_{\boldsymbol{v} \in C} \boldsymbol{z}^{\boldsymbol{v}}.$$

Let $\delta = \frac{1}{M}\gamma^2$ and $\delta^\perp$ is the transform of δ. Now $(B_{i,1}^\perp, B_{i,2}^\perp, \ldots, B_{i,n}^\perp)$, $for\ i = 1, 2, \ldots, n$ is defined by

$$B_{i,j}^\perp = \frac{1}{M} \sum_{u_i \neq 0, w(\boldsymbol{u}) = j} \chi_{\boldsymbol{u}}(\delta).$$

Definition4.1: The *dual separation* $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$ of a code C is defined by

$$B_{i,j}^\perp = 0, \ for\ 1 \leq j \leq d_i^\perp - 1,$$
$$B_{d^\perp}^\perp \neq 0.$$

If C is linear, $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$ is the separation of $C^\perp$.

Theorem4.1: If C is a $(n, M, d)_2$ code over $\{0, 1\}$ with dual separation $(d_1^\perp, d_2^\perp, \ldots d_n^\perp)$ then the codewords of C form the rows of an $OA(M, n, s, (d_1^\perp - 1, d_2^\perp - 1, \ldots, d_n^\perp - 1))$ with entries from $\{0, 1\}$.

Proof: Since the dual separation of C is

$(d_1^\perp, d_2^\perp, \ldots d_n^\perp)$,

$$\begin{aligned} B_{i,j}^\perp &= \frac{1}{M} \sum_{u_i \neq 0, w(\boldsymbol{u})=j} \chi_{\boldsymbol{u}}(\delta) \\ &= 0, \ for\ 1 \leq j \leq d_i^\perp - 1. \end{aligned} \tag{3}$$

Also,

$$\chi_{\boldsymbol{u}}(\delta) = \chi_{\boldsymbol{u}}(\frac{1}{M}\gamma^2) = \frac{1}{M}\chi_{\boldsymbol{u}}(\gamma)^2 \geq 0. \tag{4}$$

Therefore,

$$\chi_{\boldsymbol{u}}(\delta) = 0, \ for\ \boldsymbol{u}\ such\ that\ u_i \neq 0, \ w(\boldsymbol{u}) = j.$$

by Eq.(3), (4). Therefore, for $\boldsymbol{u}$ such that $u_i \neq 0$, $w(\boldsymbol{u}) = j$

$$\begin{aligned} \chi_{\boldsymbol{u}}(\gamma) &= \chi_{\boldsymbol{u}}(\sum_{\boldsymbol{v}\in C} \boldsymbol{z}^{\boldsymbol{v}}) \\ &= \sum_{\boldsymbol{v}\in C} (-1)^{\boldsymbol{u}\cdot\boldsymbol{v}} = 0. \end{aligned}$$

By theorem 2.1, the array of codeword of C forms an orthogonal array of strength $(d_1^\perp, d_2^\perp, \ldots, d_n^\perp)$. □

The Construction Method

The Construction Method2: Let there be two orthogonal arrays; $\overline{C_1}$ is $OA(M_1, n_1, 2, t_1)$ and $\overline{C_2}$ is $OA(M_2, n_2, 2, t_2)$, where $t_2 \leq t_1$. Let C_1 be the set of the rows of $\overline{C_1}$ and C_2 be the set of the rows of $\overline{C_2}$. Let

$$\begin{aligned} C = \{&(c_{1,1}, c_{1,2}, \ldots, c_{1,n_1-n_{0L}+1} + c_{2,1}, \\ &\ldots, c_{1,n_1} + c_{2,n_{0L}}, c_{2,n_{0L}+1}, \ldots, c_{2,n_2}) \mid \\ &for\ \forall(c_{1,1}, c_{1,2}, \ldots, c_{1,n_1}) \in C_1, \\ &\forall(c_{2,1}, c_{2,2}, \ldots, c_{2,n_2}) \in C_2\}. \end{aligned}$$

The orthogonal array whose rows are formed by the vectors in C is $(M_1M_2) \times (n_1 + n_2 - n_{0L})$ array. Let $n_{0L} \leq t_2/2$.

Theorem4.2: The orthogonal arrays whose rows are formed by the vectors in C is $OA(M_1M_2, n_1 + n_2 - n_{0L}, 2, (r_1, r_2, \ldots, r_n))$, where

$$r_i \geq t_1 \quad (i = 1, 2, \ldots, n_1 - n_{0L}), \tag{5}$$
$$r_i \geq t_2 \quad (i = n_1 + 1, n_1 + 2, \ldots, n_1 + n_2 - n_{0L}), \tag{6}$$
$$r_i \geq t_1 + t_2 - n_{0L} \quad (i = n_1 - n_{0L} + 1, \ldots n_1). \tag{7}$$

Proof: Now, let $M = M_1M_2, n = n_1 + n_2 - n_{0L}$,

$$\gamma = \sum_{\boldsymbol{v}\in C} \boldsymbol{Z}^{\boldsymbol{v}} = \sum_{\boldsymbol{v}\in\{0,1\}^n} c_{\boldsymbol{v}} \boldsymbol{Z}^{\boldsymbol{v}},$$

$$\delta = \frac{1}{M}\gamma^2,$$

and

$$\delta^\perp = \frac{1}{M} \sum_{\boldsymbol{u}\in\{0,1\}^n} \chi_{\boldsymbol{u}}(\delta) \boldsymbol{z}^{\boldsymbol{u}}.$$

And, Let

$$\begin{aligned} C_1' = \{&(a_{1,1}, a_{1,2}, \ldots, a_{1,n_1}, 0, \ldots, 0) \in \{0,1\}^n \\ &\mid \forall(a_{1,1}, a_{1,2}, \ldots, a_{1,n_1}) \in C_1\}, \end{aligned}$$

$$\begin{aligned} C_2' = \{&(0, \ldots, 0 a_{2,1}, a_{2,2}, \ldots, a_{2,n_2}) \in \{0,1\}^n \\ &\mid \forall(a_{2,1}, a_{2,2}, \ldots, a_{2,n_2}) \in C_2\}, \end{aligned}$$

$$\gamma_1 = \sum_{\boldsymbol{v}\in C_1'} \boldsymbol{z}^{\boldsymbol{v}}, \gamma_2 = \sum_{\boldsymbol{v}\in C_2'} \boldsymbol{z}^{\boldsymbol{v}},$$

and,

$$\delta_1 = \frac{1}{M_1}\gamma_1^2, \delta_2 = \frac{1}{M_2}\gamma_2^2.$$

Then, we can describe

$$\gamma = \gamma_1 \times \gamma_2,$$

$$\delta = \frac{1}{M}\gamma^2 = \frac{1}{M_1M_2}(\gamma_1 \times \gamma_2)^2 = \delta_1 \times \delta_2.$$

Moreover,

$$\begin{aligned} \delta^\perp &= \frac{1}{M} \sum_{\boldsymbol{u}\in\{0,1\}^n} \chi_{\boldsymbol{u}}(\delta) \boldsymbol{z}^{\boldsymbol{u}} \qquad (8) \\ &= \frac{1}{M_1M_2} \sum_{\boldsymbol{u}\in\{0,1\}^n} \chi_{\boldsymbol{u}}(\delta_1 \times \delta_2) \boldsymbol{z}^{\boldsymbol{u}} \\ &= \frac{1}{M_1M_2} \sum_{\boldsymbol{u}\in\{0,1\}^n} \chi_{\boldsymbol{u}}(\delta_1)\chi_{\boldsymbol{u}}(\delta_2) \boldsymbol{z}^{\boldsymbol{u}} \\ &= \sum_{\boldsymbol{u}\in\{0,1\}^n} (\frac{1}{M_1}\chi_{\boldsymbol{u}}(\delta_1))(\frac{1}{M_2}\chi_{\boldsymbol{u}}(\delta_2)) \boldsymbol{z}^{\boldsymbol{u}}. \qquad (9) \end{aligned}$$

By Eq. (8) and (9),

$$\frac{1}{M}\chi_{\boldsymbol{u}}(\delta) = (\frac{1}{M_1}\chi_{\boldsymbol{u}}(\delta_1))(\frac{1}{M_2}\chi_{\boldsymbol{u}}(\delta_2)).$$

Therefore, for $1 \leq i \leq n, 1 \leq j \leq n$,

$$\begin{aligned} B_{i,j}^\perp &:= \frac{1}{M} \sum_{u_i \neq 0, w(\boldsymbol{u})=j} \chi_{\boldsymbol{u}}(\delta) \\ &= \sum_{u_i \neq 0, w(\boldsymbol{u})=j} (\frac{1}{M_1}\chi_{\boldsymbol{u}}(\delta_1))(\frac{1}{M_2}\chi_{\boldsymbol{u}}(\delta_2)). \end{aligned}$$

For $1 \leq i \leq n_1 - n_{0L}$,

$$B_{i,j}^\perp = 0, \ for\ 1 \leq j \leq t_1,$$

since $\chi_{\boldsymbol{u}}(\delta_1) = 0$, for $\boldsymbol{u}$ such that $u_i \neq 0, w(\boldsymbol{u}) = j$.

For $n_1 \leq i \leq n_1 + n_2 - n_{0L}$,

$$B_{i,j}^\perp = 0, \ for\ 1 \leq j \leq t_2,$$

since $\chi_{\boldsymbol{u}}(\delta_2) = 0$, for $\boldsymbol{u}$ such that $u_i \neq 0, w(\boldsymbol{u}) = j$.

For $n_1 - n_{0L} + 1 \leq i \leq n_1$,

$$B_{i,j}^\perp = 0, \ for\ 1 \leq j \leq t_1 + t_2 - n_{0L},$$

since $\chi_{\boldsymbol{u}}(\delta_1) = 0$, or, $\chi_{\boldsymbol{u}}(\delta_2) = 0$, for $\boldsymbol{u}$ such that $u_i \neq 0, w(\boldsymbol{u}) = j$.

Hence by theorem4.1, Eq.(5),(6),(7) hold. □

5 Discussion

In this section, we will show one of examples of nonlinear unequal orthogonal array constructed by the construction method2. And it will be compared to the linear unequal orthogonal array constructed by the construction method1, and the optimal equal orthogonal array [1, Table 12.1] .

Let A_1 be the orthogonal array which is constructed from $OA(24, 12, 2, 3)$, $OA(4, 3, 2, 2)$[1 Table 12.1] using the construciton method2. $OA(24, 12, 2, 3)$ is a nonlinear orthogonal array. Therefore A_1 is a nonlinear unequal orthogonal array. Then A_1 is the 96×14 array. And the strength of A_1 is $(r_1, r_2, \ldots, r_{14})$, where

$$
\begin{aligned}
r_i &\geq 3 \ \ (i = 1, 2, \ldots, 11), \\
r_i &\geq 2 \ \ (i = 13, 14), \\
r_i &\geq 4 \ \ (i = 12).
\end{aligned}
$$

And let A_2 be the orthogonal array which is constructed from $OA(32, 13, 2, 3)$, $OA(4, 3, 2, 2)$[1 Table 12.1] using the construction method1. Both $OA(32, 13, 2, 3)$ and $OA(4, 3, 2, 2)$ are linear orthogonal arrays. Therefore A_2 is linear unequal orthogonal array. Then A_2 is the 128×15 array. And the strength of A_2 is $(s_1, s_2, \ldots, s_{15})$, where

$$
\begin{aligned}
r_i &\geq 3 \ \ (i = 1, 2, \ldots, 12), \\
r_i &\geq 2 \ \ (i = 14, 15), \\
r_i &\geq 4 \ \ (i = 13).
\end{aligned}
$$

And let A_3 be the orthogonal array $OA(128, 14, 2, 4)$ [1, Table 12.1]. A_3 is optimal equal orthogonal array.

First, we compare A_1 with A_2. The number of row of A_1 is fewer than the one of A_2. Therefore A_1 can reduce more number of experiments than A_2 when the number of factors is 14.

Next, we compare A_1 with A_3 . The number of row of A_1 is fewer than the one of A_3. Therefore A_1 can reduce more number of experiments than A_3 when there are partial intersections.

And although A_2 is unequal and A_3 is equal , the number of experiments of A_2 is same as the one of A_3. But the number of experiments of A_1 is fewer than A_2, A_3.

Hence, it has been shown that there are good orthogonal arrays in orthogonal arrays constructed by the construction method2.

6 Conclusion

In this paper, we clarify the relation between nonlinear unequal orthogonal arrays and error-correcting codes. Furthermore, we extend the construction method of linear unequal orthogonal arrays, and propose one of construction methods of nonlinear unequal orthogonal arrays. And we show that there are good orthogonal arrays in orthogonal array constructed by the proposed construction method.

Acknowledgments

The authors would like to acknowledge all of the member of Hirasawa Lab. and Matsushima Lab. for their helpful suggestions to this work. This research is partially supported by Category (C) No.15560338 of Grant-in-Aid for Scientific Research, Japan Society for the Promotion of Science.

References

[1] A.S.Hedayat, N.J.A.Sloane, and J.Stufken, "Orthogonal Arrays: Theory and Applications," Springer, New York, 1999.

[2] F.J.MacWilliams and N.J.A.Sloane, The theory of error-correcting codes, amsterdam:North-Holland Publishing Co., 1977.

[3] B.Masnick, J.Wolf, "On linear Unequal Error Protection Codes," IEEE Trans. Inform. Theory, Vol.IT-3, No.4, pp.600 - 607, October, 1967.

[4] Yoshifumi Ukita, Toshiyasu Matsushima, Shigeichi Hirasawa, "A Note on Learning Boolean Fonctions by Using Orthogonal Desings," IEICE Trans. Fundamentals, Vol.J83-A, pp482-490, April 2003.

[5] Tomohiko Saito, Takahiro Yoshida, Toshiyasu Matsushima, "A Note on the Construction of Orthogonal Designs by using the Construction of Error Correcting Codes," Proc.of SITA2002, pp663-666.

[6] Tomohiko Saito, Toshiyasu Matsushima, Shigeichi Hirasawa, "A Note on the Construction of Orthogonal Designs Using Error Correcting Codes," Proc.of SITA2004, pp463-466.

A Study of Reliability Based Hybrid ARQ Scheme with Bitwise Posterior Probability Evaluation from Message Passing Algorithm

Daiki KOIZUMI, Naoto KOBAYASHI, Toshiyasu MATSUSHIMA, and Shigeichi HIRASAWA

Waseda University
3-4-1, Okubo, Shinjuku, Tokyo, 169-8555, JAPAN
E-mail: dkoizumi@matsu.mgmt.waseda.ac.jp

Abstract Reliability Based Hybrid ARQ (RBH-ARQ) is one of hybrid ARQ schemes with the modified decision feedback. In RBH-ARQ, the modified feedback is composed of both ACK/NAK signal and unreliable bit index which is evaluated from bitwise posterior probability. In the conventional RB-ARQ, the sender retransmits just unreliable information bits with no coding when unreliable bits are detected on the receiver and retransmission occurs. In the proposed RBH-ARQ, on the other hand, the sender retransmits not information bits but newly encoded parity bits corresponding to the unreliable information bits assuming systematic convolutional coding. Furthermore, the proposed scheme assumes message passing algorithm for Maximum A Posteriori probability (MAP) decoding on the receiver. The receiver puts received bits including retransmitted parity bits all together into our proposing probability model. As a result, better performance can be expected in the proposed RBH-ARQ since our probability model is similar to the celebrated decoding model of multiple Turbo codes. Finally, brief simulation results based on several message passing schedules in our algorithm would be shown.

Key words Reliability Based Hybrid ARQ, Bitwise MAP Decoding, Message Passing Algorithm

1 Introduction

Reliability Based Hybrid ARQ (RBH-ARQ)[4][5] is one of hybrid ARQ schemes with the modified decision feedback. In this scheme, the modified decision feedback is composed of both ACK/NAK signal and unreliable bit index. In order to get unreliable bit index from received sequence, bitwise decoding techniques are required. One of them is Maximum A Posteriori probability (MAP) decoding technique including BCJR algorithm[2] which calculates exact bitwise posterior probability of information bit under received sequence for convolutional codes. Moreover, the celebrated decoding algorithm of Turbo codes[1][3] utilizes parallel BCJR algorithm to calculate bitwise approximate posterior probability. RBH-ARQ normally utilizes such posterior probability to evaluate bitwise reliability of received information sequence. Unreliable bit index is determined by comparing this reliability (log ratio of binary posterior probabilities) and the pre-defined threshold. For example, if the log ratios of certain bits are proved to be less than one, then these bits are regarded as unreliable. After getting this unreliable bit index, the receiver sends back both NAK signal and unreliable bit index to the sender where the feedback channel is assumed to be noiseless as [4][5] do.

When the sender receives the modified decision feedback, the retransmission occurs. The sender then retransmits certain information about unreliable bits depending on the type of RBH-ARQ. In [4], the sender retransmits unreliable information bits with no coding. In another RBH-ARQ scheme[5], nonsystematic convolutional codes are taken and all bits corresponding to unreliable information bits are retransmitted.

Apart from these schemes, in our proposed RBH-ARQ scheme, the sender retransmits newly encoded single parity bit per an unreliable information bit for every retransmission where the half rate (for example) systematic convolutional codes are used in encoding. Moreover, the proposed scheme also assumes message passing algorithm for MAP decoding on the receiver. The receiver puts received bits including retransmitted parity bits all together into our proposing probability model. Since this model is similar to the celebrated decoding model of multiple Turbo codes[3], the better performance can be expected. We prepare several algorithms depending on message passing schedules on our decoding model in simulation and finally brief results are shown.

This paper is organized as follows. The next section 2 defines basic notations and RBH-ARQ model. Section 3 describes the conventional RBH-ARQ schemes in [4][5]. In section 4, we propose the improved RBH-ARQ scheme and show overview of decoding model of multiple Turbo codes. In section 5, we execute some simulations to analyze RBH-ARQ schemes. Last section 6 concludes our research.

2 Basic RBH-ARQ Model

First of all, we describe basic RBH-ARQ model[4][5]. Let $u_i \in \{0,1\}$, $(i = 1,2,\cdots,M)$ be information bit sequence. In Figure 1, the encoder (the sender) takes each u_i and produces output x_i as the codeword. The sequence of x_i is then BPSK modulated and transmitted

under the AWGN channel, assuming that the channel noise parameter is known to the decoder (the receiver).

The decoder beyond the channel takes the sequence of y_i where y_i is the receivedword and calculates bitwise posterior probability $p(u_i|y)$ where $y = y_1 y_2 \cdots y_M$. The error detector calculates the log ratio of (binary) bitwise posterior probabilities, i.e. $\log\{p(u_i = 0|y)/p(u_i = 1|y)\}$, from output of decoder and evaluates the reliability of each information bit. Taking pre-defined threshold λ, if $|\log\{p(u_i = 0|y)/p(u_i = 1|y)\}| < \lambda$ holds, the error detector regards ith bit as unreliable. After the whole sequence of y_i is processed, the error detector sends back both NAK signal and unreliable bit index through noiseless feedback channel. If $|\log\{p(u_i = 0|y)/p(u_i = 1|y)\}| > \lambda$ holds, on the other hand, the error detector sends back ACK signal under feedback channel and performs bitwise Maximum A Posteriori probability (MAP) decoding to estimate $\hat{u}_i$. Since NAK signal contains unreliable bit index, we regard it as not conventional decision feedback but *modified decision feedback.*

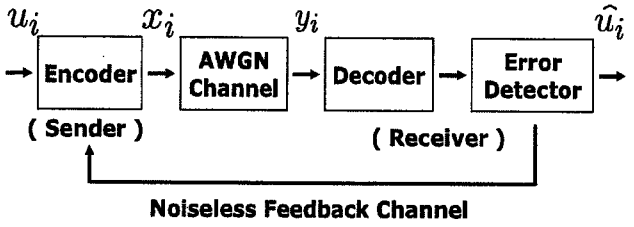


Figure 1: Basic RBH-ARQ Model

For the rest part of this paper, we shall simply call the encoder as the sender, both decoder and error detector as the receiver, respectively. If the modified decision feedback from the receiver contains NAK signal, the retransmission occurs at the sender. In this phase, the sender must retransmit certain information about unreliable bits to the receiver. The retransmitted information depends on the type of RBH-ARQ. The next section explains some types of them in detail.

3 The Conventional RBH-ARQ Schemes [4][5]

Suppose $j = 0, 1, \cdots$ is the number of retransmissions and let $x_i(j)$ be the jth retransmitted codeword, $y_i(j)$ be the corresponding receivedword, respectively, where i is bit index ($i = 1, 2, \cdots, M$) again. Additionally, let $y(j)$ denote the sequence: $y_1(j)y_2(j)\cdots y_M(j)$. In the conventional RBH-ARQ scheme of [4], the sender firstly (when $j = 0$) transmits sequence of Turbo codewords $x_i(0)$. The receiver receives the sequence of $y_i(0)$ through AWGN channel and calculates bitwise posterior probability $p(u_i|y(0))$. For bitwise decoding of Turbo codes, the algorithm described in [1] which is parallel version of BCJR algorithm[2] is well-known. By taking such posterior probability, bitwise reliability evaluation at the receiver can be defined as follows:

Definition 3.1 *(Unreliable Bit Detection)*
When the jth retransmission finishes, the receiver detects unreliable ith bit if the following holds:

$$L_i(j) \triangleq \left|\log \frac{p(u_i = 0|y(0), y(1), \cdots, y(j))}{p(u_i = 1|y(0), y(1), \cdots, y(j))}\right| < \lambda, \quad (1)$$

where λ is pre-defined threshold.

If equation (1) is satisfied, the receiver returns the modified decision feedback which consists of both NAK signal and unreliable bit index. Otherwise the receiver returns ACK signal. These signals are sent through feedback channel to the sender.

If the sender receives NAK signal, the retransmission occurs. Suppose $U(j) \subseteq \{1, 2, \cdots, M\}$ be a set of unreliable bit indices corresponding to the jth retransmission. The sender then retransmits unreliable information bit sequence u_i through AWGN channel where $i \in U(j)$. This means that $x_i(j) = u_i, \forall j > 0$ in [4]. The receiver calculates $L_i(j)$ in (1) and use $\sum_j L_i(j)$ for decoding. The whole retransmission procedure of [4] is shown in Figure 2.

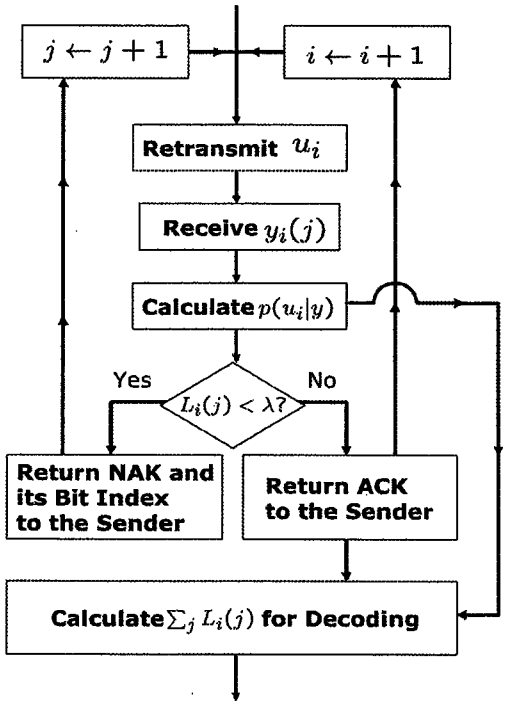


Figure 2: The Conventional RBH-ARQ Retransmission

In another RBH-ARQ scheme[5], the sender firstly transmits the sequence of non-systematic convolutional codeword of $x_i(0)$. As same as [4], the receiver takes the sequence of $y(0)$, performs BCJR algorithm to calculate $p(u_i|y(0))$, and evaluates bitwise reliability in (1). In

this scheme, if the jth retransmission occurs, the sender retransmits not unreliable information bits $u_i, i \in U(j)$ but the whole codeword $x_i(j), i \in U(j)$ corresponding to the unreliable information bits. At the receiver, (the jth) BCJR algorithm is executed and $\sum_j L_i(j)$ is taken for decoding.

4 Proposed RBH-ARQ Scheme

4.1 Brief Procedure Description

In this section, we shall explain the proposed scheme with Figure 3. In our proposed scheme, the sender transmits systematic convolutional codes (half rate for example here) $x_i(0) = (x_{i,0}, x_{i,1})$ where $x_{i,0} = u_i$ and $x_{i,1}$ is parity bit for the first transmission: $j = 0$. The receiver beyond the AWGN channel takes the sequence of $y_i(0) = (y_{i,0}, y_{i,1})$, $(i = 1, 2, \cdots, M)$ and executes BCJR algorithm to calculate posterior probability $p(u_i|y(0))$ where $y(0) = y_1(0)y_2(0)\cdots y_M(0)$.

Nextly, the error detector evaluates bitwise reliability by (1) using $p(u_i|y)$ as [4][5] do. If unreliable bits are detected, the receiver sends back NAK signal as well as unreliable bit index through feedback channel. If the retransmission occurs, the sender newly encodes unreliable bit u_i and retransmits it as $x_i(j)$ to the receiver for the jth retransmission. In this phase, retransmitted bit index should be interleaved since we would make use of decoding algorithm of multiple Turbo codes later. Note that the conventional two schemes retransmit unreliable information bits and the all bits corresponding to the unreliable information bits, respectively.

Lastly, the receiver takes the sequence of $y_i(j)$, $i \in U(j)$ where $U(j)$ is the set of unreliable information bits for jth retransmission again. For every retransmission, the receiver de-interleaves bit index of $y_i(j)$ and executes decoding algorithm of multiple Turbo codes[3] for ith unreliable bit. This decoding strategy preserves suboptimality in terms of calculating posterior probability even if the retransmission frequently occurs, whereas the one in conventional schemes does not.

4.2 Our Decoding Model Similar to that of Multiple Turbo Codes

This subsection explains the decoding model of multiple Turbo codes from which our proposed scheme is derived. In the following explanation, both interleaver and de-interleaver are abbreviated for the simplicity of notation. As same as most of reference books about Turbo codes, we shall basically follow the deriving process of BCJR algorithm[2].

Let $k = 1, 2, \cdots$ be the number of constituent codes where $k = j + 1$, $(j = 0, 1, \cdots)$ holds. By using k, let $x_i^k = (x_{i,0}, x_{i,k})$ be the constituent code and $y_i^k = (y_{i,0}, y_{i,k})$ be the constituent receivedword, respectively. In the decoding of multiple Turbo codes, the posterior probability of $p(u_i|y)$ is approximated by that of constituent code: $p(u_i|y^k)$ where $y^k = y_1^k y_2^k \cdots y_M^k$.

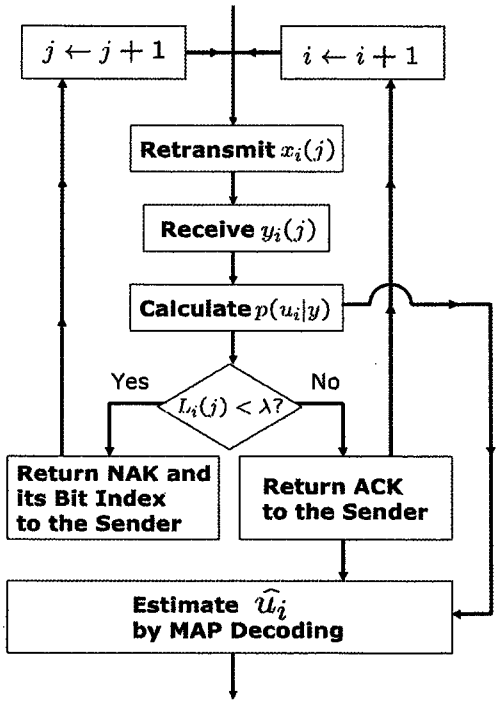


Figure 3: Proposed RBH-ARQ Retransmission

In terms of message passing algorithm on the graphical model such as Factor Graph, Bayesian Network etc. the constituent code can be represented by single subgraph which is subset of the whole graphical model. The output of posterior probability in the subgraph approximates the posterior probability of the multiple Turbo codes. To do this, exchanging extrinsic information between subgraphs is assumed. The above posterior probability then becomes the followings:

$$
\begin{aligned}
& p(u_i = a|y) \\
& \approx \; p(u_i = a|y^k) && (2) \\
& = \; p(u_i = a|y_1^k, y_2^k, \cdots, y_M^k) && (3) \\
& = \; \frac{1}{p(y^k)} \\
& \qquad \sum_{(s_i, s_{i+1}) \in A} p(S_{i,k} = s_i, S_{i+1,k} = s_{i+1}, y^k), && (4)
\end{aligned}
$$

where $a \in \{0, 1\}$, $p(y^k) = p(y_1^k, y_2^k, \cdots, y_M^k)$, both $S_{i,k}$ and $S_{i+1,k}$ are possible states in trellis diagram, and A is a set of states in trellis diagram such that $u_i = a$.

According to [2], the following holds in (4):

$$
\begin{aligned}
& p(S_{i,k} = s_i, S_{i+1,k} = s_{i+1}, y^k) \\
& \quad = \; p(y_{t>i}^k|s_{i+1}) p(s_{i+1}, y_i^k|s_i) p(s_i, y_{t<i}^k), && (5)
\end{aligned}
$$

where t is bit index in trellis diagram.

The first and third terms in RHS of (5) can be recursively calculated using the second term[2]. For the second term, the following transformation is possible:

$$p(s_{i+1}, y_i^k|s_i)$$

$$= p(u_i = a)p(y_{i,0}|u_i)p(y_{i,k+1}|x_{i,k+1}). \quad (6)$$

By substituting (4) by both (5) and (6), we have

$$\begin{aligned} & p(u_i = a|y_1^k, y_2^k, \cdots, y_M^k) \\ &= \frac{1}{p(y^k)} p(u_i = a) \\ &\quad \times \sum_{(s_i, s_{i+1}) \in A} \left[p(y_{t>i}^k | s_{i+1}) \right. \\ &\quad \left. \times \{p(y_{i,0}|u_i)p(y_{i,k+1}|x_{i,k+1})\} p(s_i, y_{t<i}^k) \right]. \quad (7) \end{aligned}$$

In (7), $\sum_{(s_i,s_{i+1})\in A} \left[p(y_{t>i}^k|s_{i+1}) \right.$ $\left. \{p(y_{i,0}|u_i)p(y_{i,k+1}|x_{i,k+1})\}p(s_i, y_{t<i}^k) \right]$ is frequently referred as *extrinsic information*. In the celebrated decoding algorithm of Turbo codes, this extrinsic information is normally exchanged among plural subgraphs. For the case of $k = 2$ (where $j = 1$), there exists two subgraphs and they would be shown in Figure 4 if we take Bayesian Network as a graphical model to express our decoding probability model which is similar to multiple Turbo codes. In Figure 4, note that $x_{i,1}$ as well as $y_{i,1}$ are missing since ith bit is reliable enough and the retransmission does not occur.

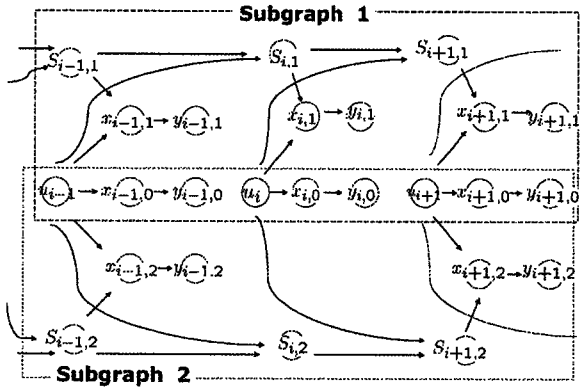


Figure 4: Example of Our Decoding Model which is Similar to that of Multiple Turbo Codes Expressed by Bayesian Network ($k = 2$).

For exchanging extrinsic information among subgraphs, we shall assume message passing algorithm on our probability model. Although there exists several message passing schedules on one given graph, they would be considered in the next section and we assume that full parallel messaging schedule for the simple explanation here. For kth single subgraph, outgoing and incoming messages are defined as the followings:

Definition 4.1 *(Outgoing Message from kth Subgraph) Outgoing message from kth subgraph, $M_{k\rightarrow}$, is equivalent to extrinsic information in equation (7):*

$$\begin{aligned} M_{k\rightarrow} &\triangleq \sum_{(s_i, s_{i+1}) \in A} \left[p(y_{t>i}^k | s_{i+1}) \right. \\ &\quad \left. \{p(y_{i,0}|u_i)p(y_{i,k+1}|x_{i,k+1})\}p(s_i, y_{t<i}^k) \right]. \quad (8) \end{aligned}$$

Definition 4.2 *(Incoming Message to kth Subgraph) Incoming message to kth subgraph, $M_{k\leftarrow}$, is defined as the product of extrinsic information of all subgraphs except for that of kth subgraph:*

$$M_{k\leftarrow} \triangleq \prod_{l=1, l\neq k}^{l=N} M_{l\rightarrow}. \quad (9)$$

On the above definition of incoming message, the following examples can be typical cases:

- For $k = 2$, simply exchanging messages each other:

$$M_{1\leftarrow} = M_{2\rightarrow}, \quad (10)$$
$$M_{2\leftarrow} = M_{1\rightarrow}. \quad (11)$$

- For $k = 3$, the extension of the case of $k = 2$:

$$M_{1\leftarrow} = M_{2\rightarrow} \cdot M_{3\rightarrow}, \quad (12)$$
$$M_{2\leftarrow} = M_{1\rightarrow} \cdot M_{3\rightarrow}, \quad (13)$$
$$M_{3\leftarrow} = M_{1\rightarrow} \cdot M_{2\rightarrow}. \quad (14)$$

Figure 5 shows examples of message passing among subgraphs for $j = 2, 3$. Figure 5 assumes full parallel message passing for simplicity, however, several message passing schedules can be considered:

- Full Parallel Schedule:

  For $k = 2$,
  Subgraph1 $\rightleftharpoons$ *Subgraph2*.

- Semi Parallel Schedule:

  For $k = 3$,
  Subgraph1 $\rightleftharpoons$ *Subgraph2* $\rightleftharpoons$ *Subgraph1* $\rightleftharpoons$ *Subgraph3* $\rightleftharpoons$ *Subgraph1* $\rightleftharpoons$ *Subgraph2*, $\cdots$, and so forth.

- Serial Schedule:

  For $k = 3$,
  Subgraph1 $\rightarrow$ *Subgraph2* $\rightarrow$ *Subgraph3* $\rightarrow$ *Subgraph1* $\rightarrow \cdots$.

These effects for RB-HRQ performance are examined in the next section by simulation.

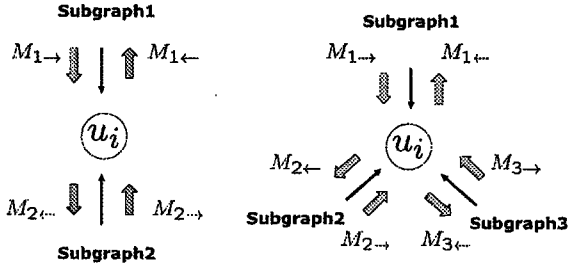


Figure 5: Example of Message Passing for $k = 2, 3$.

5 Simulation

To examine performance of the proposed RBH-ARQ, we took two RBH-ARQ schemes: one is the modified version of [4], the other is proposed one. Brief description of them is the followings:

The Conventional RB-AHQ Scheme:

- For the sequence of $x_i(j)$, the sender transmits half rate of the systematic convolutional codes.
- The sender retransmits *unreliable information bit* for every retransmission, i.e. $x_i(j) = u_i, \forall j > 0$.
- The receiver performs decoding by $\sum_j L_i(j)$.

Proposed RB-AHQ Scheme:

- For the sequence of $x_i(j)$, the sender transmits half rate of the systematic convolutional codes.
- The sender retransmits *single parity bit* corresponding to unreliable information bit for every retransmission, i.e. $x_i(j) = x_{i,k}$, $\forall j > 0$.
- Random interleaver is assumed for every retransmission.
- The receiver performs MAP decoding whose posterior probability is derived from message passing algorithm described in subsection 4.2.

5.1 Simulation1

In this simulation, we compare the performances of conventional and proposed schemes with the following conditions:

- Average Throughput value from 100 ARQ Procedure Executions is taken.
- RBH-ARQ Schemes: Conventional Scheme and Proposed Scheme with Full parallel Messaging Schedule described in the previous section.
- Length of Information Sequence: 1024.
- Constraint Length of Systematic Convolutional Codes: 3.
- Fixed E_s/N_0[dB]: 0.00.
- BER is fixed 7.0×10^{-3}
  by changing threshold λ about the bit reliability.
- Number of Turbo Iterations for Proposed Scheme: 10.

Table 6: Result of Simulation1

Messaging Schedule	Throughput	Retransmission
Conventional	0.468	3.11
Full Parallel	0.451	3.81

5.2 Simulation2

In this simulation, we compared performances of proposed schemes by changing their messaging schedules. The simulation conditions are the followings:

- Average Throughput value from 100 ARQ Procedure Executions is taken.
- Messaging Schedules of Decoding: Full parallel, Semi Parallel, and Serial Schedules described in the previous section.
- Length of Information Sequence: 1024.
- Constraint Length of Systematic Convolutional Codes: 4.
- Fixed E_s/N_0[dB]: -1.00.
- BER is fixed 7.0×10^{-3}
  by changing threshold λ about the bit reliability.
- Number of Turbo Iterations: 10.

With the above conditions, the following Table 7 is obtained.

Table 7: Result of Simulation2

Messaging Schedule	Throughput	Retransmission
Full Parallel	0.383	7.88
Semi Parallel	0.390	7.69
Serial	0.388	7.74

5.3 Discussion

In simulation1, we set relatively advantageous E_s/N_0 for the conventional scheme. From the Table 6, the performances of both schemes are almost same in terms of throughput as well as the number of retransmissions. We analyzed retransmission processes in detail and it turned out that the number of retransmissions reduced rapidly in the conventional scheme. In the conventional scheme, almost all bits are turned over to *reliable* for the second retransmission. Additionally, we should point out that the conventional scheme was extremely faster than the proposed one since its procedure is quite simple. This can be big advantage if the channel condition is relatively good.

In the proposed scheme, on the other hand, the number of unreliable bits did not decrease so rapidly. Only the half of them turned over to *reliable bit* for most of the second retransmission. Under the severe E_b/N_0 conditions, however, BER of the proposd scheme remained around 7.0×10^{-3} without chaning λ drastically. From this result, it is expected that the proposed scheme has better performance under the negative E_s/N_0 as same as the cereblated performance of the multiple Turbo codes

In simulation2, we set relatively low E_s/N_0 to clarify the differences of three messaging schedules. From the Table 7, however, the performances of them have no remarkable differences. According to the data of Turbo

iteration processes, their trends of convergence (of posterior probabilities) were almost same. Hence the convergent values of them were also proved to be almost same.

6 Conclusion

In this paper, we proposed the improved RBH-ARQ scheme with both new retransmission procedure and several messasgeing schedules in decoding. For relatively good channel condition (typically $E_s/N_0 = 0.0$), the convensional and proposed scheme have almost same performance in simulation. But the complexity of calculations in the proposed scheme is extremely larger than the convensional scheme since it has Turbo-like iteration. This cost might be paid under the severe channel conditions such as negative E_s/N_0. For messaging schedules in the proposed scheme, no remarkable difference is obtained.

Acknowledgments

One of authors, D.KOIZUMI appreciates their helpful comments from T.NIINOMI in Kanagawa Institute of Technology and Y.UKITA in Yokohama College of Commerce.

This research is partially supported by Category (C) No.15560338 of Grant-in-Aid for Scientific Research, Japan Society for the Promotion of Science.

References

[1] C.Berrou, A.Glavieux and P.Thitimajshima, "Near Shannon Limit Error-correcting Coding and Decoding: Turbo-codes," In *Proceeding of the IEEE International Conference on Communications*, Vol.2, pp.1064-1070, May. 1993.

[2] L.R.Bahl, J.Cocke, F.Jelinek and J.Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Transactions on Information Theory*, Vol.20, Issue.2, pp.284-287, Mar. 1974.

[3] D.Divsalar and F.Pollara, "Multiple Turbo Codes," In *Proceeding of the IEEE Military Communications Conference*, Vol.1, pp.279-285, Nov. 1995.

[4] H.Kim and J.M.Shea, "New Turbo-ARQ Techniques Based on Estimated Reliabilities," In *Proceeding of the IEEE Wireless Communications and Networking Conference*, Vol.2, pp.843-848, Mar. 2003.

[5] A.Roongta and J.M.Shea, "Reliability-based Hybrid ARQ using Convolutional Codes," In *Proceeding of the IEEE International Conference on Communications*, Vol.4, pp.2889-2893, May. 2003.

In the Proc. of IEEE ISOC ITW2005 on Coding and Complexity; editor M.J. Dinneen; co-chairs U. Speidel and D. Taylor; pages 108-112

A Note on a Decoding Algorithm of Codes on Graphs with Small Loops

Naoto KOBAYASHI
School of Sci. & Eng.
Waseda University
Tokyo, Japan
Email: naoto@ruri.waseda.jp

Toshiyasu MATSUSHIMA
School of Sci. & Eng.
Waseda University
Tokyo, Japan
Email: toshi@matsu.mgmt.waseda.ac.jp

Shigeichi HIRASAWA
School of Sci. & Eng.
Waseda University
Tokyo, Japan
Email: hirasawa@hirasa.mgmt.waseda.ac.jp

***Abstract*— The best-known algorithm for the decoding of low-density parity-check (LDPC) codes is the sum-product algorithm (SPA). The SPA is a message-passing algorithm on a graphical model called a factor graph (FG). The performance of the SPA depends on a structure of loops in a FG. Pearl showed that loops in a graphical model could be erased by the clustering method. This method clusters plural nodes into a single node. In this paper, we show several examples about a decoding on a FG to which the clustering method is applied. And we propose an efficient decoding algorithm for it.**

I. Introduction

In recent years, low density parity check (LDPC) codes [1] have been widely studied. LDPC codes are decoded by the sum-product algorithm (SPA) [2]. The SPA is a message-passing algorithm on a graphical model called a factor graph (FG)[4], and it computes the exact posterior probability if a FG has no loop. The FG that represents any LDPC codes generally has loops. Although the SPA doesn't always compute the exact posterior probability on a FG with loops, it computes a good approximation of a posteriori probability for the decoding of LDPC codes.

The performance of the SPA depends on a structure of loops in a FG. For example, if a FG includes length-4 loops, errors often occur at the bits corresponding to the loops [2]. To avoid this problem, LDPC codes without small loops are used. Pearl showed that any loops in a graphical model could be erased by the clustering method [5][6]. This method collapses plural nodes into a single node. It can be also applied to a FG [4]. For example, clustering two nodes that correspond to a length-4 loop can erase the loop [7]. We call the FG to which the clustering method is applied a "Cluster Factor Graph (CFG)."

Since the structure of the FG that represents any LDPC codes is very complicate, we cannot erase all loops in it by the clustering method. Thus, we consider erasing only some small loops in the FG. We can, however, expect to improve the performance of the SPA on such the CFG, because in the clustering node that contains collapsed plural random variables, the local joint probability of them is calculated. In particular, for a binary erasure channel (BEC), we can prevent the decoding error occurred by stopping sets [12] with the clustering methods. We show the performance of a decoding with the clustering method in this paper.

The computational complexity of the SPA on a CFG is larger than those of the SPA on the original FG, because the number of computation messages is generally increased. For binary party check codes, the computational complexity of the SPA on a CFG can be decreased. In this paper, we explain logic of it and propose an efficient decoding algorithm on a CFG.

This paper is organized as follows: In Section II, we shortly explain a FG and the SPA. In Section III, we explain the clustering method. In Section IV, we interpret the applications of this method for a decoding algorithm and propose an efficient decoding algorithm on a CFG. In Section V, we show several examples for a decoding of LDPC codes by simulations. In Section VI, we discuss the results of the simulations. In Section VII, we summarize the paper.

II. Preliminaries

A. Notation

We assume a codeword $X := \{X_1, X_2, ..., X_N\} \in \{0,1\}^N$ is transmitted over a noisy memory-less channel with transition probabilities $f_n^a (= p(Y_n|X_n = a))$, and let $Y := \{Y_1, Y_2, ..., Y_N\}$ be a received sequence. On receiving Y, we estimate a codeword $\hat{X} := \{\hat{X}_1, \hat{X}_2, ..., \hat{X}_N\} \in \{0,1\}^N$. Let n be an index of nth codeword symbol $(1 \leq n \leq N)$. Let H be a parity-check matrix whose row and column length are M and N, and let H_{mn} be the value of mth rows and nth columns of H $(1 \leq m \leq M)$. An index of each row of the parity-check matrix is denoted by m. We define $N(m) := \{n|H_{mn} = 1\}$ and $M(n) := \{m|H_{mn} = 1\}$ as sets of index to indicate the position of symbol "1" for each rows and columns.

If S is any sets, we define $|S|$ as the number of element of S and define $S_l (1 \leq l \leq |S|)$ as the lth element of S.

B. Factor Graph

A Factor Graph (FG) is a graphical model which has two types of node, a variable node and a check node. The details of a FG are described in [4]. Any liner codes can be represented by a FG, assuming that each symbol and received symbol represent as random variables, and a parity check matrix and a channel assumption represent as constrained conditions. We henceforth describe a variable node corresponding to a codeword symbol n as a "variable node x_n," and a check node

0-7803-9481-X/05/$20.00

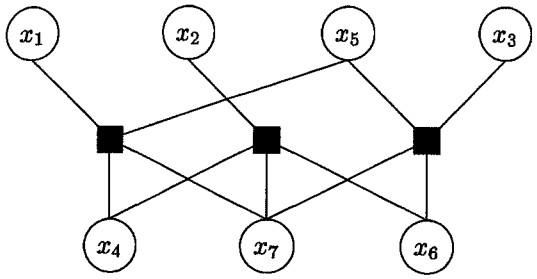


Fig. 1. A factor graph example.

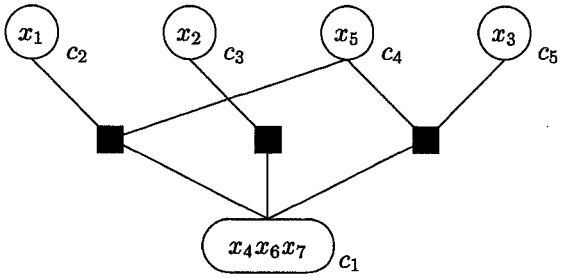


Fig. 2. A cluster factor graph example.

corresponding to a mth row of a parity-check as a "check node h_m." For each x_n, a variable node x_n takes on elements in alphabet A_{x_n}. On binary codes, $A_{x_n} = \{0, 1\}$ for all x_n .

Fig. 1 shows an example of the FG for given the parity check matrix for the $(7, 4)$ Humming code as Eq. (1). We omit variable nodes corresponding to received words and check nodes corresponding to a channel assumption.

$$H = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \tag{1}$$

C. Sum-Product Algorithm

The Sum-Product Algorithm (SPA) is a decoding algorithm for LDPC codes. It is an iterative message-passing algorithm on a FG. The details of the algorithm are described in [1][2]. Although there are several types of descriptions about the SPA, in this paper, we deal with the SPA on a probabilistic domain in [3].

III. Clustering Method

A. Clustering Method

The clustering method [5][6] is to collapse plural nodes into a single node on graphical models. This method can be also applied to a FG [4]. We consider that variable nodes are clustered into a single node[1]. We call a FG to which the clustering method applies a Cluster Factor Graph (CFG), and a new node prepared by clustering is called a "cluster node"[2]. On the other hand, variable nodes which are clustered are called "a variable node is contained in a cluster node." Fig. 2 shows an example of the clustering method. This shows that the variable nodes x_4, x_6 and x_7 are clustered on the FG in Fig. 1.

B. Notation for the Clustering Method

We define new sets for the clustering method. Let $k(1 \leq k \leq N_c)$ be an index of each cluster node, where N_c is the number of cluster nodes in a CFG. We describe a cluster node with an index k as a "cluster node c_k." The variable nodes contained in cluster node c_k are denoted by

$$C_k := \{n | x_n \text{ is contained in } c_k (1 \leq n \leq N)\}. \tag{2}$$

By procedures for the clustering method, we have $C_i \cap C_j = \phi$, where $i \neq j$. A cluster node c_k takes on elements of A_{c_k}, where

$$A_{c_k} := \bigotimes_{n \in C_k} A_{x_n} \tag{3}$$

($\otimes$ implies direct product; it is ordered by C_k). For simply descriptions, we equate a variable node which is not clustered with a cluster node which contains one variable node, that is $|C_k| = 1$. For example in Fig. 2, we define cluster nodes as follows,

$$C_1 = \{4, 6, 7\}, \tag{4}$$

$$C_2 = \{1\}, C_3 = \{2\}, C_4 = \{5\}, C_5 = \{3\}. \tag{5}$$

Let $M'(k)$ be the set of all check nodes which are connected to the cluster node c_k. We have

$$M'(k) = \bigcup_{l=1}^{|C_k|} M(C_{kl}). \tag{6}$$

Let $N'(m)$ be a modified set of $N(m)$ by the clustering method. $N'(m)$ has indexes of cluster nodes. For example in Fig. 2, we have $N'(1) = \{1, 2, 4\}, N'(2) = \{1, 3\}, N'(3) = \{1, 4, 5\}$.

The check node h_1 is connected to the variable node x_4 and x_7 in Fig. 1; however, it is connected to the cluster node $\{x_4, x_6, x_7\}$ in Fig. 2. With the notice of only the CFG in Fig.2, we confuse "the check node h_1 is connected to only the variable node x_4 and x_7" with "the check node h_1 is connected to the variable node x_4, x_6 and x_7." To solve this problem, we define the set D_{km} that is defined between a cluster node c_k and each check node which is connected to c_k, and it is given by

$$D_{km} := \{n | n \in C_k \cap N(m)\}. \tag{7}$$

For example in Fig. 2, we have $D_{11} = \{4, 7\}$ from $N(1) = \{1, 4, 7\}$ and Eq.(4).

[1]Check nodes can be clustered as well. However, we don't deal with them in this paper.

[2]"Cluster Graph [9]" and "Extended Junction Graph [10]" are graphical models that can deal with plural variable nodes as cluster nodes. A CFG is a subclass of them because each variable node is contained in only one cluster node in the model.

C. Characteristics of a CFG and the Clustering Method

Pearl showed that the clustering method could erase independent loops in a graphical model. First, we attempt to erase all loops in the FG that represent any LDPC codes. It is meaningless because such the CFG generally has only one cluster node which contains all variable nodes in the original FG. Thus, we henceforth consider to erase only some small loops in the FG. In addition, the length of a loop is decreased if the collapsed nodes are in the loop. We need to investigate which loops should be clustered.

IV. A Decoding Algorithm on a CFG

A. The Sum-Product Algorithm on a CFG

The SPA calculates local joint probabilities of each elements of A_{c_k} of the cluster node c_k. Thus, we expect to improve the performance of the SPA by the clustering method. If $|D_{km}| \neq |C_k|$, a message from a cluster node c_k to a check node h_m has elements which each alphabet is $\bigotimes_{n \in D_{km}} A_{x_n}$. For calculate this message with an alphabet a, we need to sum the values of the local joint probabilities over the all elements that bits of the alphabet corresponding to each index in D_{km} is a. This operation is called a marginalization.

For the SPA, the number of elements of messages between a cluster node and the neighbor check nodes generally equals the number of elements on which the cluster node takes. Thus, the computational complexity of the SPA on a CFG is larger than those of the SPA on the original FG. However, assuming that using any binary parity check codes, we prepare only two elements of messages "the even-message" and "the odd-message". For the CFG that represent any binary parity check codes, all check nodes in the CFG are constraints of parity check. This constraint implies that the sum of the values of the neighbor variable nodes is zero modulo 2. If some of variable nodes in them are collapsed, we require only the following information to verify the constraint. That is, "the sum of them is zero modulo 2" or "the sum of them is one modulo 2".

For example in Fig. 1 and Fig. 2, the check node h_1 implies

$$X_1 + X_4 + X_5 + X_7 = 0 \bmod 2. \tag{8}$$

According to the above discussions, the minimum requisite information of messages between h_1 and c_1 are not $(X_4 = 0, X_7 = 0)$, $(X_4 = 0, X_7 = 1)$, $(X_4 = 1, X_7 = 0)$ and $(X_4 = 1, X_7 = 1)$ but $(X_4 + X_7 = 0 \bmod 2)$ and $(X_4 + X_7 = 1 \bmod 2)$. For these reasons, the **Horizontal step** of the SPA on a CFG are same computations of the SPA on a FG. On the other hand, we entail the process of calculation each element of cluster nodes and the process of a marginalization on the **Vertical step**. We show these logics as the proposed algorithm to be hereinafter described.

B. For the Binary Erasure Channel

In this subsection, we consider the communication over the binary erasure channel (BEC). If erasures occur on all bits which are contained in any stopping sets [12], the SPA cannot decode by the received sequence. By the clustering method, we can prevent such a decoding error that is occurred by stopping sets. This is due to the equivalent of "a calculation of local joint probabilities" and "solving of a simultaneous equation ".

For example, we deal with the $(7, 4)$ Humming code which parity check matrix is Eq. (1). Assuming that a received sequence is $\{Y_1 = 0, Y_2 = 0, Y_3 = 0, Y_4 = e, Y_5 = 0, Y_6 = e, Y_7 = e\}$, where e implies an erasure. Since $\{x_4, x_6, x_7\}$ are a stopping set, the SPA on the FG cannot decode it. That is, all messages from each check node to the variable nodes x_4, x_6 and x_7 are always an erasure. On the SPA on the CFG, the messages from each check node to the cluster node c_1 are

$$X_4 + X_7 = 0 \bmod 2, \tag{9}$$

$$X_4 + X_6 + X_7 = 0 \bmod 2, \tag{10}$$

$$X_6 + X_7 = 0 \bmod 2. \tag{11}$$

These are a simultaneous equation, and we can solve these and obtain $X_4 = X_6 = X_7 = 0$. If certain conditions are satisfied, we improve the performance of the SPA for the BEC to collapse some of variable nodes in a stopping set by the clustering method. We describe the conditions in appendix.

C. The Proposed Decoding Algorithm

In this subsection, we propose an efficient decoding algorithm on a CFG based on the above descriptions. We descript the algorithm as like the SPA in [3]. For the BEC, we can define the similar efficient algorithm to refer [12]. In this case, we can replace the routines in the **Vertical step** by solution algorithms of a simultaneous equation[3].

Let q^a_{km} and r^a_{mk} be messages between the check node h_m and the cluster node c_k, where a is a type of messages, "even" or "odd". Let $\lambda(A_{c_k})$ be the function that returns all alphabets in A_{c_k}. And let $\lambda_e(A_{c_k}, \{x_i\})$ be the function that returns all alphabets in A_{c_k}, where the bits corresponding to $\{x_i\}$ have even weight ($\{x_i\}$ implies any sets of an index of variable nodes). Let $\lambda_o(A_{c_k}, \{x_i\})$ be the function that returns all elements in A_{c_k}, where the bits corresponding to $\{x_i\}$ have odd weight. Consequently, we have

$$\lambda(A_{c_k}) = \lambda_e(A_{c_k}, \{x_i\}) \cap \lambda_o(A_{c_k}, \{x_i\}). \tag{12}$$

For example in Fig. 2, we have

$$\lambda_e(A_{c_1}, D_{11}) = \{000, 010, 101, 111\}, \tag{13}$$

$$\lambda_o(A_{c_1}, D_{11}) = \{001, 100, 011, 110\}. \tag{14}$$

(Notice that x_4 corresponds to the first bit of A_{c_1}; x_7 corresponds to the third bit of A_{c_1}). We define t_l as the lth bit of a bit sequence t. α is a normalization constant.

On the **Initialization** and the **Horizontal step**, compute through each pair of (m, k), where $k \in N'(m)$ for all $m (1 \leq m \leq M)$:

[3]We can, obviously, compute this by a probabilistic calculation.

Initialization.

$$\begin{cases} q_{km}^{even} = \sum_{t\in\lambda_e(A_{c_k},C_k)} \prod_{l=1}^{|C_k|} f_{C_{kl}}^{t_l}, \\ q_{km}^{odd} = \sum_{t\in\lambda_o(A_{c_k},C_k)} \prod_{l=1}^{|C_k|} f_{C_{kl}}^{t_l}. \end{cases} \tag{15}$$

Horizontal step.

$$\begin{cases} r_{mk}^{even} = ((1+\delta r_{mk})/2), \\ r_{mk}^{odd} = ((1-\delta r_{mk})/2), \end{cases} \tag{16}$$

where

$$\delta r_{mk} = \prod_{k'\in N'(m)\backslash k} (q_{k'm}^{even} - q_{k'm}^{odd}). \tag{17}$$

(*) If $|N'(m)| = 1$, we define $r_{mk}^{even} = 1$ and $r_{mk}^{odd} = 0$.

Vertical step.

Compute for all $t \in \lambda(A_{c_k})$,

$$Q_k^t = \prod_{l=1}^{|C_k|} f_{C_{kl}}^{t_l} \prod_{m\in M'(n)} r_{mk}^{a_{km}(t)}, \tag{18}$$

where

$$a_{km}(t) = \begin{cases} \text{"even"} & t \in \lambda_e(A_{c_k}, D_{km}), \\ \text{"odd"} & other. \end{cases} \tag{19}$$

And compute for each $m \in M'(k)$,

$$\begin{cases} q_{km}^{even} = \alpha(\sum_{t\in\lambda_e(A_{c_k},D_{km})} Q_k^t)/r_{mk}, \\ q_{km}^{odd} = \alpha(\sum_{t\in\lambda_o(A_{c_k},D_{km})} Q_k^t)/r_{mk}. \end{cases} \tag{20}$$

Pseudo-posterior probabilities.

Compute for all n, where $n \in C_k$.

$$\begin{cases} q_n^0 = \alpha \sum_{t\in\lambda_e(A_{c_k},\{n\})} Q_k^t, \\ q_n^1 = \alpha \sum_{t\in\lambda_o(A_{c_k},\{n\})} Q_k^t. \end{cases} \tag{21}$$

V. Numerical Experiments

In this section, we examine the performance of a decoding on the CFG by simulations. We examine MacKay 's regular LDPC code[4] with a code length of 1008, column weight of 3, and row weight of 6. The rate of the code is 0.5, and it has no length-4 loops. We consider collapsing variable nodes that is contained in length-6 loops in the FG that represent this code. The number of length-6 loops of this FG is 165. We decide loops that are clustered to find the length-6 loops that satisfy the conditions in appendix. The CFG that is constructed by this algorithm has no length-4 loops. By this clustering algorithm, we cluster 98 length-6 loops in this FG; however the number of length-6 loops in the made-up CFG is 191. This is caused that length-8 loops which have collapsed nodes transform into length-6 loops. Using these graphs, we compare the SPA with the proposed decoding algorithm (denoted by "CSP"). We evaluate them by a word error (erasure) rate, a bit error (erasure) rate and a computational complexity. For each algorithm, 2×10^4 symbols are transmitted. To examine minutely for the AWGN on the SNR from 2.5 to 3.5, 2.2×10^5 symbols are transmitted over them. The maximum number of iterations is 100 for each algorithm.

[4]http://www.inference.phy.cam.ac.uk/mackay/codes/504.504.3.504

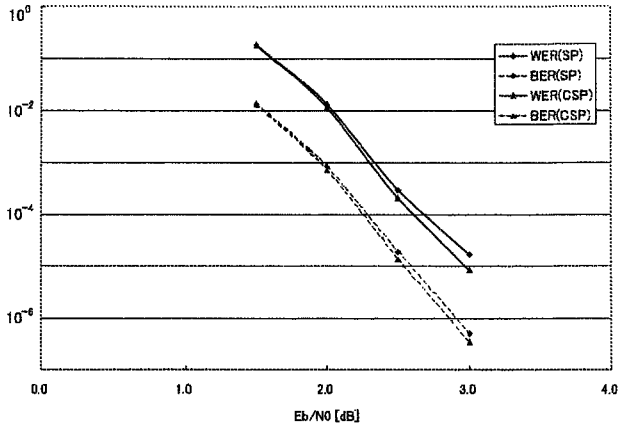


Fig. 3. Simulation result over the AWGN.

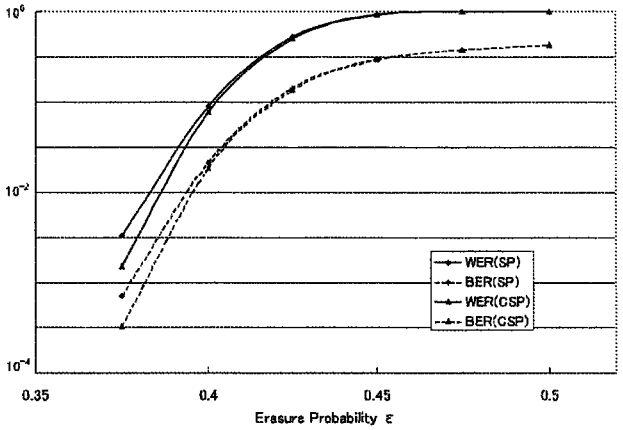


Fig. 4. Simulation result over the BEC.

In Fig. 3, we show the result of the decoding of both algorithms over the AWGN and the result of them over the BEC in Fig. 4. The computational complexity of both algorithms are $O(n)$, if $|N(m)|$, $|M(n)|$ and $|C_k|$ are constant for all m, n and k. Hence, we count the number of the arithmetic operation (addition and multiplication) in one iteration of each algorithm. We show them in Table I. In Table II, we show them in specific figures for the decoding of the above code.

TABLE II

THE NUMBER OF ARITHMETIC OPERATION (SPECIFIC FIGURES)

	SPA	CSP
Horizontal Step.	39312	38166
Vertical Step.	21168	58674
P.P.	5040	4788
total	65520	101628

TABLE I

THE NUMBER OF THE ARITHMETIC OPERATION IN ONE ITERATION

	SPA	CSP
Horizontal Step.	$\sum_{m=1}^{M} \|N(m)\|(2\|N(m)\|+1)$	$\sum_{m=1}^{M} \|N'(m)\|(2\|N'(m)\|+1)$
Vertical Step.	$\sum_{m=1}^{M}(\sum_{n\in N(m)}(2\|M(n)\|+1))$	$\sum_{m=1}^{M}(\sum_{k\in N'(m)}(2^{\|C_k\|}(\|M'(k)\|+\|C_k\|-1)+3))$
P.P.	$(5N)$ $(*)$	$\sum_{k=1}^{N_c}(\|C_k\|(2^{\|C_k\|}+1))$

(*) This formula is derived if the SPA uses the pre-normalization values on the Vertical Step.

VI. Discussion

Fig. 4 illustrates that the decoding performance of the proposed algorithm over the BEC is better than that of the SPA. We think that this is attributed to the decrement of the decoding error that is occurred by stopping sets. On the other hand, the decoding performance of the proposed algorithm over the AWGN is a little better than that of the SPA, as shown in Fig. 3. Notice that the decoding performance with the clustering method is (a little) better than that without the clustering method, although the number of length-6 loops is increased by the clustering method. We may consider that the number of small loops in a FG and the performance of the SPA are not simple proportionality relations.

Table I and Table II illustrate that the number of arithmetic operations of the proposed algorithm is not so enormous as compared with the SPA if $|C_k|$ are small constant numbers for all k. For an implementation, it is more complicate because of a handling of λ_e, λ_o and a_t. Furthermore, the required memory size of the proposed algorithm is more than that of the SPA for the memory of C_k and D_{km}.

VII. Concluding Remarks

In this paper, we show several examples about a decoding on a FG to which the clustering method is applied. We also propose an efficient decoding algorithm for it. In this paper, we use the existing LDPC code, and the clustering algorithm is not so reflected. For the future, we should investigate a good class of codes that is suited to the clustering method. Furthermore, a decoding failure of LDPC codes over the BEC is evaluated exactly for the SPA [12]. We should evaluate it for the proposed algorithm.

Acknowledgment

The authors would like to acknowledge entire member of Hirasawa Lab. and Matsushima Lab. for their helpful suggestions to this work. This research is partially supported by Category No.15560338 of Grant-in-Aid for Scientific Research, Japan Society for the Promotion of Science.

References

[1] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory* Vol.8, pp.21-28, Jan. 1962.

[2] D. J. C. Mackay, "Good Error-Correcting Codes Based on Very Sparse Matrices," *IEEE Trans. Inf. Theory* Vol. 45, pp. 399-431, Mar. 1999.

[3] D. J. C. Mackay, R.M. Nealm "Near Shannon limit performance of low-density parity-check codes," Electron. Lett., Vol. 32, pp. 1645-1646, Aug. 1996.

[4] F. R. Kschischang, B. J. Frey, H-A Loeliger, "Factor Graphs and the Sum-Product Algorithm," *IEEE Trans. Inf. Theory* Vol. 47 ,pp. 498-519, Feb. 2001.

[5] J. Pearl, " Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inferene," Morgan Kaufmann, 1988.

[6] S. J. Russell ,P. Norvig, "Artificial Intelligence," Prentice-Hall, Inc., 1995.

[7] N. Kobayashi, T. Matsushima, S. Hirasawa, "A Note on Decoding of Low Density Parity-Check Codes with Small-Loop (in Japanese)," Symposium on Information Theory and Its Applications, 2004.

[8] S. M. Aji and R. J. McEliece, "The Generalized Distributive Law," *IEEE Trans. Inf. Theory* Vol. 46, pp. 325-343, Mar. 2000.

[9] T.Sato, "Cluster BP and cluster CCCP: Two simple methods for computing Kikuchi approximations (in Japanese)," Tech. Rep. TR03-0001, Dept. of Computer Science, Tokyo Institute of Technology. 2003.

[10] T. Matsushima, S. Hirasawa, "A formalization of generalized probabilistic reasoning and its procedures on Junction trees," submitted to *IEEE Trans. Inf. Theory.*

[11] T. Wadayama, "Low Density Parity-Check Codes and Its Decoding Algorithm (in Japanese)," Triceps, Tokyo, Japan, June. 2002.

[12] M. Luby, M. Mitzenmacher, M. Shokrollahi, and D. Spielman, "Efficient erasure correcting codes" *IEEE Trans. Inf. Theory* Vol. 47, pp. 569-584, Feb. 2001.

Appendix

If a loop in a FG satisfied following conditions then the SPA can determine the variable node x^* that is contained in a stopping set by clustering the loop. Let L be variable nodes in a loop. Let S be variable nodes in a stopping set.

Conditions.

- L is a subset of S.
- All check nodes that are connected to L are only connected to following variable nodes: 1) Variable nodes in L. 2) Only one variable node x^* in S and not in L. 3) Variable nodes not in S and not in L.

The 28th Symposium on Information The
and Its Applications (SITA2005)
Onna, Okinawa, Japan, Nov. 20–23, 2005

A Decoding Algorithm of Low-Density Parity-Check Codes Using Decisions of Erasure Correcting

Gou HOSOYA * Toshiyasu MATSUSHIMA * Shigeichi HIRASAWA *

Abstract— We propose a new iterative decoding algorithm of low-density parity-check codes. The proposed decoding algorithm has used a bipartite graph whose variable nodes are clustered according to a received sequence of the codeword. The clustering procedure has used a decoding algorithm over the binary erasure channel which substitutes check nodes that have two erased variable nodes. We compare decoding results and complexity of the proposed decoding algorithm with the sum-product decoding algorithm.

Keywords— low-density parity-check code, sum-product decoding algorithm

1 Introduction

We propose a new iterative decoding algorithm of low-density parity-check codes. The proposed decoding algorithm has used a bipartite graph whose variable nodes are clustered according to a received sequence of the codeword. The clustering procedure has used a decoding algorithm over the binary erasure channel which substitutes check nodes that have two erased variable nodes. We compare decoding results and complexity of the proposed decoding algorithm with the sum-product decoding algorithm. In recent years, Low-density parity-check (LDPC) codes [6] have been widely studied. LDPC codes with the iterative message-passing decoding algorithm based on belief propagation, which we call the sum-product (SP) decoding algorithm, can be decoded in high performance with low complexity [6]. The SP decoding algorithm is symbol-wise maximum a posterior probability (APP) decoding when a bipartite graph of the code is a tree. When the graph has loops, SP decoding algorithm is not APP decoding. Loops in the bipartite graph are erased by clustering which combines variable nodes together. But bipartite graphs of the LDPC codes have many loops that makes difficult to erase all loops by clustering, and furthermore the decoding complexity on the clustered graph increases exponentially with the maximum number of variable nodes that are contained in the clustered node.

LDPC codes over the binary erausre channels (BEC) have been widely studied [1]. It is well known that the SP decoding over the BEC always fails when the set of erased symbols has a stopping set that is related to loops [4]. In recent studies, some improved version of the SP decoding algorithm have been proposed [2], [5], [7]. Those decoding algorithms can correct erased variable nodes even if they contain the stopping set and have a good trade-off between decoding performance and decoding complexity.

In this paper, we consider to extend the improved SP decoding algorithm over the BEC to noisy channels. We point out that those decoding algorithms generate the clustered graph, so some loops in the bipartite graph are erased by this clustering. We compare decoding results and complexity of the proposed decoding algorithm with the SP decoding algorithm.

* Department of Industrial and Management Systems Engineering, School of Science and Engineering, Waseda University, Okubo 3-4-1, Shinjuku-ku, Tokyo, 169-8555 Japan. E-mail: hosoya@hirasa.mgmt.waseda.ac.jp

2 Preliminaries

2.1 LDPC Codes and bipartite graphs

Let $H = [H_{mn}]$, $m \in [1, M]$, $n \in [1, N]$, be a parity-check matrix whose row and column lengths are M and N, respectively, and $\boldsymbol{c} = (c_1, c_2, \ldots, c_N) \in \{0,1\}^N$ be a codeword of the LDPC code such that $\boldsymbol{c}H^{\mathrm{T}} = \boldsymbol{0}$. Let λ_i and ρ_i denote the fraction of ones of H which are in columns and rows for weight i, respectively, and $\lambda(x) \triangleq \sum_{i=2}^{\infty} \lambda_i x^{i-1}$ and $\rho(x) \triangleq \sum_{i=2}^{\infty} \rho_i x^{i-1}$ be a degree distribution of row and column of ones in H, respectively. LDPC codes are characterized by the code length N and $\lambda(x), \rho(x)$ which are denoted by $C\big(N, \lambda(x), \rho(x)\big)$. Row length M is given by $M = N \frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx}$.

A parity-check matrix is represented by the bipartite graph which consists of two types of nodes called check nodes indexed by position of rows, and variable nodes indexed by position of columns. A check node m and a variable node n are connected with an edge if and only if $H_{mn} = 1$. A loop in the bipartite graph is a closed path that starts from a variable node and returns to the same variable node through edges without passing the same edges more than once. A length of a loop is a number of edges of the closed path.

A stopping set is a subset of variable nodes whose subgraph have check nodes of degree at lease two. The union of stopping sets is also a stopping set, so each graph has an unique maximal stopping set.

2.2 Erasure Decoding for LDPC Codes

2.2.1 SP Decoding Algorithm [1]

The SP decoding algorithm for the BEC can correct an erased variable node whenever its neighboring check node has only one erased variable node. We assume that the check node is labeled as "satisfied" when its all neighboring variable nodes are known, and otherwise it is "unsatisfied". The algorithm is given by the following procedure:

[Sum-Product Decoding Algorithm]

For all unsatisfied check nodes, perform the following:

s1) If the values of all but one of the variable nodes connected to the check nodes are known, set the erased variable nodes to the XOR operation of the other variable nodes and label that check node as "finished". If all the value of variable nodes connected to the check node are known, label the check node as "finished". This procedure is performed sequentially.

s2) Continue s1) until all check nodes are labeled as finished or decoding cannot continue further. □

2.2.2 Improved Decoding Algorithm [2], [7]

The SP decoding algorithm cannot decode whenever the set of erased variable nodes $\mathcal{E}$ has a stopping set. But we can continue the decoding procedure after when the SP decoding algorithm fails. At first, we choose unsatisfied check node that has two erased variable nodes whose positions are denoted by $\epsilon_1, \epsilon_2 \in \mathcal{E}$. Next, we substitute this check node to the other unsatisfied check nodes that have erased variable nodes in position ϵ_2 (or ϵ_1). We assume that the check node, that have two erased variable nodes and is substituted, is labeled as "substituted". This substituted procedure does not increment a number of erased variable nodes of the resulting check node. Moreover, sometimes it can reduce a number of erased variable nodes of the resulting check node. This procedure is continued until all erased variable nodes are corrected or there are no unsatisfied check nodes that have erased variable nodes greater than two and all unsatisfied check nodes that have two erased variable nodes are labeled as "substituted". The improved decoding algorithm is given by the following procedures:

[Improved Decoding Algorithm]

I1) Perform s1) and s2).

I2) If I1) cannot continue further, then label the unsatisfied check node that have two erased variable nodes at positions ϵ_1, ϵ_2 as "substituted", and substitute that check node to other check nodes that have a variable node at position ϵ_2 (or ϵ_1).

I3) If there are unsatisfied check nodes that have one erased symbol, label that check node as "finished", and this variable node is corrected. Next, substitute this corrected variable node to the other "substituted" and "unsatisfied" check nodes that have this corrected variable node. This procedure is performed for all unsatisfied check nodes.

I4) Continue I2) and I3) until all check nodes are labeled as "finished" or all unsatisfied check nodes that have variable nodes greater than two and all unsatisfied check nodes that have two variable nodes are labeled as "substituted". □

3 Proposed Decoding Alogirthm

We consider to extend the improved decoding algortihm to noisy channels, such as the additive white Gaussian noise (AWGN) channels.

3.1 Details of the Algoirthm

From the point of view of message passing algorithm, substituting a check node that has two erased variable nodes regards those two nodes as one clustered node. So we choose erased variable nodes at some ordered manner, and perform improved decoding algorithm, which devoted at section 2.2.2, to make a clustered graph. In this paper, we choose erased variable nodes from the APP values that are calculated by the SP decoding algorithm at the ordinary bipartite graph for a noisy channel. If the log-domain APP value at the position of arbitrary variable node is small, we can treat this node as erasure. The proposed decoding algorithm is given by the followings:

[Proposed Decoding Algorithm]

p1) Perform the SP decoding at the ordinary graph on a noisy channel.

p2) If the estimated sequence is a codeword, stop the algorithm. Otherwise go to p3).

p3) Change suitable number of variable nodes to erasures whose log-domain APP values are small. Perform the erasure decoding discussed in section 2.2.2 to generate a clustered graph. When substituting the check node that has two erased variable nodes, cluster those variable nodes together.

p4) If no clustered nodes are generated, stop the algorithm. Otherwise go to p6).

p5) Perform the SP decoding at the clustered graph on a noisy channel. □

3.2 Decoding Complexity of the Algoirthm

Table 1 shows the number of operations (per one iteration) required for both decoding algorithms [1].

Table 1: Decoding complexity for ordinary and clustered graphs required at one iteraiton

	Ordinary	Clustered
Addition	$3E+N$	$4\tilde{E}-M+\sum \nu_i 2^{\chi_i}$
Multiplication	$8E-M$	$\tilde{E}+(2\tilde{E}+1)\sum 2^{\chi_i}-2\tilde{N}$ $+\sum\sum(\chi_i-\tau_{ij})^2\nu_i 2^{\chi_i}$

E and $\tilde{E}$ denotes the number of edeges in the ordinary graph and clustered graph, respectively. χ_i and ν_i denotes a number of variable nodes in a clustered node i and degree at a cluster node i, respectively. τ_{ij} is a cardinality of a product set between a set variable nodes which are contained at a clustered node i and a check node j, and the clustered node i.

From this table, the complexity of the proposed decoding algorithm grows exponentially with the number of variable nodes in a clustered node.

4 Simulation Results

4.1 Conditions for Simulation

We construct three codes for each $\mathcal{C}_1$ and $\mathcal{C}_2$ which are denoted by $\mathcal{C}_1\big(N_1, \lambda_1(x), \rho_1(x)\big), \mathcal{C}_2\big(N_2, \lambda_2(x), \rho_2(x)\big)$ such that

$$N_1 = 100,\ \lambda_1(x) = x^2,\ \rho_1(x) = x^5, \tag{1}$$

$$N_2 = 300,\ \lambda_2(x) = x^2,\ \rho_2(x) = x^5. \tag{2}$$

Codes that have the same parameter are constructed from different seeds of the random number generator.

We compare the SP decoding algorithm [6] (denoted by "Conv.") with the proposed decoding algorithm (denoted by "Pro." and number of erased variable nodes

[1] Note that the proposed decoding algorithm requires to calculate likelihood values for clustered nodes at the first iteration. We do not count in this table.

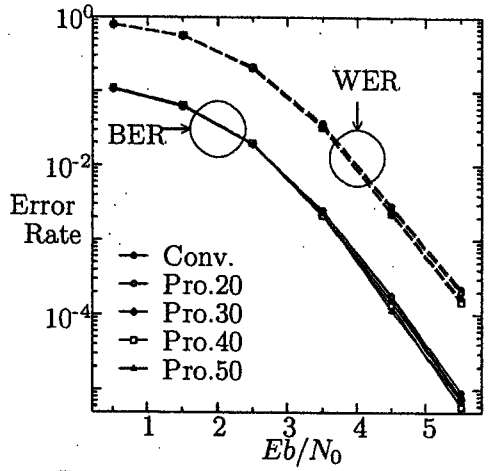


Figure 1: Decoding result for code $\mathcal{C}_1$

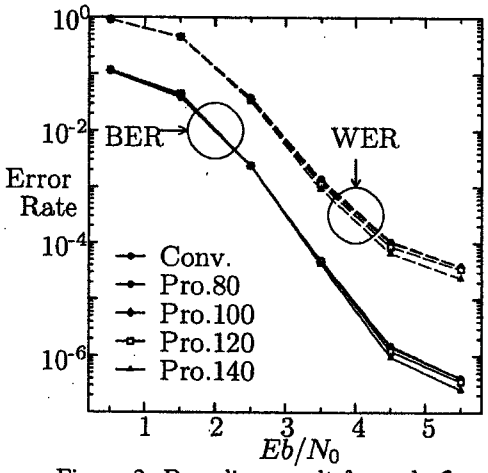


Figure 2: Decoding result for code $\mathcal{C}_2$

which we give). We decode until at least 1×10^6 codewords are transmitted or 50 codewords are failed to decode by the SP decoding algorithm. The maximum number of iterations for both decoding algortihms are set to 40.

We evaluate them by (i) decoding performance, (ii) decoding complexity (the number of real operations required for decoding).

4.2 Simulation Results and Discussions

4.2.1 Decoding Results

Figs. 1 and 2 show bit error rate (BER) and word error rate (WER) of both decoding algorithms for code $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively. From these figures, BER and WER of "Pro." is lower than that of "Conv." for almost all the number of erased variable nodes. The gap between "Conv." and "Pro." becomes larger as Eb/N_0 tents to large value.

4.2.2 Decoding Complexity

Tables 2 and 4 (3 and 5) show the average number (per iteration) of addition and multiplication required for both decoding algorithms of code $\mathcal{C}_1$ ($\mathcal{C}_2$). "Conv." always takes the same number of decoding operations since the bipartite graphs of the codes are unchanged for each received sequences. "Pro." generates the clustered graph according to the received sequence, so the number of decoding operations are changed for each received sequences. From these tables, the decoding complexity of "Pro." is higher than that of "Conv.", since the decoding complexity grows exponentially with the number of variable nodes in the cluster nodes.

Tables 6 and 8 (7 and 9) show the maximum number of addition and multiplication required for both decoding algorithms of code $\mathcal{C}_1$ ($\mathcal{C}_2$). Tables 10 and 11 show maximum number of variable nodes in the clustered nodes for code $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively. From these tables, especially the product operations are much needed and the maximum number of variable nodes in the clustered nodes is relatively large in this case. This is because from table 1, the product operation grows exponentially with the clustered size, it is relative to the maximum number of variable nodes in the clustered nodes.

Table 2: Average number of addition required for decoding of code $\mathcal{C}_1$

Eb/N_0	Conv.	Pro.20	Pro.30	Pro.40	Pro.50
0.5	400	541	460	410	400
1.5	400	467	496	412	410
2.5	400	775	612	490	575
3.5	400	1045	854	823	704
4.5	400	1067	1268	1071	1053
5.5	400	1475	1365	1394	1125

Table 3: Average number of addition required for decoding of code $\mathcal{C}_2$

Eb/N_0	Conv.	Pro.80	Pro.100	Pro.120	Pro.140
0.5	3000	3000	3000	3000	3000
1.5	3000	3047	3093	3000	3063
2.5	3000	3472	3640	3888	3670
3.5	3000	5349	4918	12763	5870
4.5	3000	5304	5893	7310	5813
5.5	3000	3000	7687	6107	6722

5 Conclusion

We have proposed a new iterative decoding algorithm of LDPC codes. The proposed decoding algorithm generates the clustered graph From simulation results, BER and WER of the proposed decoding algorithm are lower than that of the SP decoding algorithm. The decoding complexity of the proposed algorithm is large, since the production operation grows exponentially with respect to a number of variable nodes in the clustered nodes.

Table 4: Average number of multiplication required for decoding of code $\mathcal{C}_1$

Eb/N_0	Conv.	Pro.20	Pro.30	Pro.40	Pro.50
0.5	2350	7204	2536	2383	2350
1.5	2350	3331	3248	2435	2382
2.5	2350	3673	12867	3300	7148
3.5	2350	5050	10962	20681	4053
4.5	2350	4721	19275	12847	12272
5.5	2350	10493	19387	28361	12673

Table 5: Average number of multiplication required for decoding of code C_2

Eb/N_0	Conv.	Pro.80	Pro.100	Pro.120	Pro.140
0.5	7050	7050	7050	7050	7050
1.5	7050	7395	10532	7050	7369
2.5	7050	18376	60762	114058	16604
3.5	7050	152762	126341	5154554	335576
4.5	7050	28795	131966	433669	168856
5.5	7050	7050	50375	161269	392275

Table 6: Maximum number of additon required for decodingof code C_1

Eb/N_0	Pro.20	Pro.30	Pro.40	Pro.50
0.5	2512	1878	1838	400
1.5	2135	2687	2108	1828
2.5	1910	5705	2468	3827
3.5	1979	3247	7983	2138
4.5	1909	3430	3840	3606
5.5	2342	5436	5002	3963

Table 7: Maximum number of additon required for decoding of code C_2

Eb/N_0	Pro.80	Pro.100	Pro.120	Pro.140
0.5	3000	3000	3000	3000
1.5	7682	10447	3000	7876
2.5	9568	17170	47166	11734
3.5	21316	21316	826491	87392
4.5	7977	14374	42822	26195
5.5	3000	7687	12602	31137

Table 8: Maximum number of multiplication required for decoding of code C_1

Eb/N_0	Pro.20	Pro.30	Pro.40	Pro.50
0.5	75166	8156	7106	2350
1.5	27853	65407	14105	6883
2.5	9529	748595	63071	265060
3.5	18704	200165	1081709	34039
4.5	10778	208885	240384	199182
5.5	59396	519767	482558	290931

Table 9: Maximum number of multiplication required for decoding of code C_2

Eb/N_0	Pro.80	Pro.100	Pro.120	Pro.140
0.5	7050	7050	7050	7050
1.5	41173	394524	7050	35387
2.5	273905	2231005	14046881	607043
3.5	3356425	3356425	549706160	30881334
4.5	74876	1368163	9962299	4914654
5.5	7050	50375	888539	6716511

Table 10: Maximum number of variable nodes in the cluster nodes for code C_1

Eb/N_0	Pro.20	Pro.30	Pro.40	Pro.50
0.5	7	9	10	5
1.5	8	10	9	7
2.5	7	8	9	7
3.5	5	9	11	5
4.5	5	8	8	7
5.5	6	9	8	7

Table 11: Maximum number of variable nodes in the cluster nodes for code C_2

Eb/N_0	Pro.80	Pro.100	Pro.120	Pro.140
0.5	8	14	10	7
1.5	14	14	8	6
2.5	15	15	11	8
3.5	10	11	15	12
4.5	7	10	11	13
5.5	0	6	9	13

For further works, to consider other procedures how to choose erased variable nodes is needed. Moreover it should be needed not to make clustered nodes that have a large number of variable nodes, to reduce the decoding complexity.

Acknowledgments

One of the authors, G. Hosoya, would like to thank Dr. M. Kobayashi at Shonan Institute of Technology and Mr. H. Yagi at Waseda University for their valuable comments.

References

[1] M. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Efficient erasure correcting codes," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 569–584, Feb. 2001.

[2] D. Burshtein and G. Miller, "An efficient maximum-likelihood decoding of LDPC codes over the binary erasure channel," *IEEE Trans. Inform. Theory*, vol. 50, no. 11, pp. 2837–2844, Nov. 2004.

[3] T. J. Richardson and R. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 638–656, Feb. 2001.

[4] C. Di, D. Proietti, I. E. Telatar, T. J. Richardson, and R. L. Urbanke, "Finite-length analysis of low-density parity-check codes on the binary erasure channel," *IEEE Trans. Inform. Theory*, vol. 48, no. 6, pp. 1570–1579, June 2002.

[5] H. Pishro-Nik and F. Fekri, "On decoding of low-density parity-check codes over the binary-erasure channel," *IEEE Trans. Inform. Theory*, vol. 50, no. 3, pp. 439–454, March 2004.

[6] R. G. Gallager, "Low density parity check codes," *IRE Trans. Inform. Theory*, vol. 8, pp. 21–28, Jan. 1962.

[7] G. Hosoya, T. Matsushima, and S. Hirasawa "A decoding method of low-density parity-check codes over the binary erasure channel," *Proc. of the 27th Symposium on Information theory and Its Applications*, Gero, Gifu, Japan, pp. 263–267, Dec. 2004.

The 28th Symposium on Information Theory
and Its Applications (SITA2005)
Onna, Okinawa, Japan, Nov. 20-23, 2005

A Note on HTTP Traffic Analysis of the Time Series Model with a Time Varying Density Parameter

Daiki KOIZUMI * Toshiyasu MATSUSHIMA * Shigeichi HIRASAWA *

Abstract— In Internet traffic analysis, there are some approaches assuming the probabilistic distribution on the request arrivals. One of the classical approaches has been to use the time series model with the stationary Poisson distribution. The stationary Poisson distribution, however, can be insufficient to express Internet (heavily burst) traffic.

In this paper, we would deal with a time series model with the time varying (nonstationary) Poisson distribution for HTTP traffic analysis. In our model, the density parameter of the Poisson distribution is time varying. Furthermore, our model has some features in terms of theoretical guarantee of estimated traffic value as well as computational advantage. Finally, we would show some analysis results of the real HTTP traffic data under our model.

Keywords— Internet, HTTP Traffic, Time Series Model, Time Varying Parameter, Poisson Distribution

1 Introduction

In the field of Internet traffic analysis, there are some approaches to assume the probabilistic distribution on the request arrivals. On such research, evaluating the better probabilistic distribution to express the real Internet traffic is important problem[2]. The TELNET and FTP request arrivals, for example, can be expressed by the stationary Poisson distribution[3]. However, the request arrivals on the other protocols, including HTTP, can not always be expressed by the stationary Poisson[2][3] since they are often heavily burst. These results may suggest the insufficiency of the classical stationary Poisson distribution.

By taking the *nonstationary* (=time varying) Poisson distribution, on the other hand, Shino[4] got some interesting results to express real HTTP traffic. However, he assumed the Fourier series [4] for the nonstationary density parameter and used the Genetic Algorithm (GA) to estimate coefficients in the series. His method has huge calculation complexity, moreover, is the approximation of the density parameter.

In this paper, we take a time series model with the time varying (nonstationary) Poisson distribution which is mainly proposed in [1]. This model assumes the probability distribution on the density parameter and has two main natures: the first is that the posterior distribution of the density parameter under the observed series can be easily calculated, the second is that this model guarantees the Bayes optimal prediction on condition that a constant ρ, which is a parameter of time varying, is known. Finally, we would show some analysis results with the real HTTP traffic data and compare its performance against the stationary Poisson model.

* Waseda University, 3-4-1, Okubo, Shinjuku, Tokyo, 169-8555, JAPAN. E-mail: dkoizumi@matsu.mgmt.waseda.ac.jp

2 The Time Series Model with a Time Varying Density Parameter

2.1 The Time Series Model with a Time Varying Density Parameter

In this research, we would like to focus on the number of HTTP request arrivals as the HTTP traffic analysis. Suppose $t = 1, 2, \cdots$ is the discrete time and let x_t be the number of arrivals in a web server at time t. Then we assume the nonstationary Poisson distribution for x_t as follows:

$$p(x_t|\lambda_t) = \frac{e^{-\lambda_t}}{x_t!}(\lambda_t)^{x_t}, \tag{1}$$

where λ_t is a time varying (nonstationary) density parameter.

For the above λ_t, we assume a time series model[1]:

$$\lambda_{t+1} = \frac{u_t}{\rho}\lambda_t\,, \tag{2}$$

where ρ is a constant such that $0 < \rho \leq 1$.

We assume the probability distribution for both λ_t and u_t. For λ_t, suppose the following distribution:

$$p(\lambda_1) \propto \frac{1}{\lambda_1}\,, \tag{3}$$

$$p(\lambda_t) = Ga(\lambda_t|\alpha_t, \beta_t)\,, (t \geq 2). \tag{4}$$

The above $p(\lambda_1)$ actually corresponds to one of the noninformative priors of the density parameter under the Poisson distribution. For $t \geq 2$, we assume the Gamma distribution: $Ga(\lambda_t|\alpha_t, \beta_t)$ where α_t, β_t are the parameters. For u_t, we assume the following Beta distribution:

$$p(u_t) = Be(u_t|\rho\alpha_t, (1-\rho)\alpha_t)\,, \tag{5}$$

where $\rho\alpha_t, (1-\rho)\alpha_t$ are parameters.

2.2 Nonstationarity of the Density Parameter

In (2), a constant ρ expresses the nonstationarity[1]. In the case of $\rho = 1$, λ_t does not vary even if t is increasing. However, if $\rho < 1$, λ_t begins to vary depending on t and in the case of $\rho = 0.5$, the variance of u_t takes the maximum value[1]. Hence ρ expresses the nonstationarity of λ_t.

3 The Nature of the Time Series Model with a Time Varying Density Parameter

In this section, we would point out two natures of the time series model: the one is that the model enables us to calculate the posterior distribution of the density parameter easily, the other is that the model enables us to calculate the Bayes optimal prediction under observed series as well as (known) constant ρ also easily.

3.1 The Posterior Distribution of the Density Parameter

If we assume the time series model described in the previous section, the posterior distribution of $p(\lambda_t|x_1^{t-1})$ where $x_1^{t-1} = x_1 x_2 \cdots x_{t-2} x_{t-1}$ for $t \geq 2$ can be analytically obtained as follows:

- For $t = 1$ (Initial Condition):

$$\begin{aligned} p(\lambda_1|x_1) &\propto p(\lambda_1)p(x_1|\lambda_1) && (6)\\ &\propto \frac{1}{\lambda_1}\frac{e^{-\lambda_1}}{x_1!}(\lambda_1)^{x_1} && (7)\\ &\propto Ga(\lambda_1|x_1, 1)\,, && (8) \end{aligned}$$

- For $t \geq 2$:
  Suppose $\alpha_1 = x_1\,, \beta_1 = 1$ from (8),

$$\begin{aligned} & p(\lambda_t|x_1^{t-1}) \\ &= Ga\left(\lambda_t|\alpha_{t-1}, \beta_{t-1}\right)\,, && (9)\\ & p(\lambda_t|x_1^t) \\ &= \frac{p(x_t|\lambda_t)p(\lambda_t|x_1^{t-1})}{\int_0^\infty p(x_t|\lambda_t)p(\lambda_t|x_1^{t-1})d\lambda_t}\,. && (10)\\ &= Ga\left(\lambda_t|\alpha_{t-1} + x_t, \beta_{t-1} + 1\right)\,, && (11)\\ & p(\lambda_{t+1}|x_1^t) \\ &= Ga\left(\lambda_{t+1}|\rho(\alpha_{t-1} + x_t)\,, \rho(\beta_{t-1} + 1)\right) && (12)\\ &= Ga\left(\lambda_{t+1}|\alpha_t, \beta_t\right)\,. && (13) \end{aligned}$$

On the above simple updating of $p(\lambda_{t+1}|x_1^t)$, (12) is obtained by applying (2) to (11). This calculation is derived from the nature of both the Gamma and Beta distribution.

3.2 The Bayes Optimal Prediction under the Observed Series

Taking the model described in the previous section, suppose a problem to predict x_{t+1} under the observed series: $x_1^t = x_1 x_2 \cdots x_t$.

Let $\hat{x}_{t+1}$ be the estimated value of x_{t+1}, moreover, suppose the following loss function $L(x_{t+1}, \hat{x}_{t+1})$ between x_{t+1} and $\hat{x}_{t+1}$:

$$L(x_{t+1}, \hat{x}_{t+1}) = (x_{t+1} - \hat{x}_{t+1})^2. \qquad (14)$$

Let $\hat{x}_{t+1}^*$ be the Bayes optimal prediction with known constant ρ to minimize the expectation of the loss defined in (14). Then, $\hat{x}_{t+1}^*$ is obtained by the following simple arithmetic calculation:

$$\begin{aligned} & \hat{x}_{t+1}^* \\ &= \sum_{x_{t+1}} x_{t+1} \int_0^\infty p(x_{t+1}|\lambda_{t+1}, x_1^t)p(\lambda_{t+1}|x_1^t)d\lambda_{t+1} \\ &= E[x_{t+1}|x_1^t] \\ &= \frac{\sum_{i=1}^t \rho^{t-i}x_i + \rho^{t-1}\alpha_1}{\sum_{i=1}^t \rho^i + \rho^{t-1}\beta_1}\,. && (15) \end{aligned}$$

4 The Maximum Likelihood Estimation for ρ

The natures described in the previous section hold on condition that a constant ρ in (2) is known, however, if we deal with real data such as the HTTP request data, ρ is unknown and should be estimated. If we take the maximum likelihood estimation of ρ, the objective likelihood function $L(\rho)$ becomes the following:

$$\begin{aligned} & L(\rho) \\ &= \prod_{k=1}^t p(x_{k+1}|x_1^k, \rho)p(x_1|\lambda_1) \\ &= \prod_{k=1}^t \int_0^\infty p(x_{k+1}|x_1^k, \lambda_{k+1})p(\lambda_{k+1}|x_1^k)d\lambda_{k+1} \\ &= \prod_{k=1}^t \frac{(\beta_k)^{\alpha_k}\Gamma(\alpha_k + x_{k+1})}{(\beta_k + 1)^{\alpha_k + x_{k+1}}\Gamma(\alpha_k)x_{k+1}!}\,. && (16) \end{aligned}$$

From (11)(12), the following recursive relationships hold for the two parameters of the Gamma distribution:

$$\begin{aligned} \alpha_k &= \rho^{k-1}\alpha_1 + \sum_{i=1}^k \rho^{k-i}x_i \;\; (k \geq 2)\,, && (17)\\ \beta_k &= \rho^{k-1}\beta_1 + \sum_{i=1}^k \rho^{i-1} \;\; (k \geq 2)\,. && (18) \end{aligned}$$

From (16), (17) and (18), the log-likelihood function of $\log L(\rho)$ becomes the following:

$$\begin{aligned} & \log L(\rho) \\ &= \alpha_1 \log \beta_1 - (\alpha_1 + x_2)\log(\beta_1 + 1) \\ &\quad + \log\left\{\Gamma\left(\alpha_1 + x_2\right)\right\} - \log\left\{\Gamma\left(\alpha_1\right)\right\} \\ &\quad + \sum_{k=2}^t \left\{\left(\rho^{k-1}\alpha_1 + \sum_{i=1}^k \rho^{k-i}x_i\right)\right. \\ &\quad \left. \times \log\left(\rho^{k-1}\beta_1 + \sum_{i=1}^k \rho^i\right)\right\} \end{aligned}$$

$$
\begin{aligned}
&-\sum_{k=2}^{t}\left\{\left(\rho^{k-1}\alpha_1+x_{k+1}+\sum_{i=1}^{k}\rho^{k-i}x_i\right)\right.\\
&\left.\times\log\left(\rho^{k-1}\beta_1+\sum_{i=1}^{k}\rho^{i}+1\right)\right\}\\
&+\sum_{k=2}^{t}\log\left\{\Gamma(\rho^{k-1}\alpha_1+x_{k+1}+\sum_{i=1}^{k}\rho^{k-i}x_i)\right\}\\
&-\sum_{k=2}^{t}\log\left\{\Gamma(\rho^{k-1}\alpha_1+\sum_{i=1}^{k}\rho^{k-i}x_i)\right\}\\
&-\sum_{k=1}^{t}\log(x_{k+1}!)\,. \qquad (19)
\end{aligned}
$$

On the above log-likelihood function, the analytical maximum likelihood estimation for ρ is quite difficult, however, the numerical estimation is doable by simple algorithm such as the gradient method if $\log L(\rho)$ is convex.

5 Simulation Results

5.1 The HTTP Request Data

For the simulation, we used HTTP request data from three different Web servers (A, B and C) in Waseda University. The detail specification is on the Table 1. In Table 1, "Actual Counts" is derived by the following: on the same IP address sending HTTP request, the requests up to *2 seconds* after the initial request is regarded as the exactly *1 request.* We would not insist that such counting is the best method, however, would point out that it is considered in [4]. Additionally, we divided 24 hours into 288 discrete times by taking 5 minutes as a time unit.

Table. 1: The Specification of HTTP Request Data

Server Name	Server A	Server B
Date	Aug.6, 2004	Aug.6, 2004
Time	0:00 - 24:00	0:00- 24:00
Request Counts	16395	64056
Actual Counts	442	37822
Server Name	Server C	
Date	Apr.2nd, 2004	
Time	0:00 - 24:00	
Request Counts	23609	
Actual Counts	21342	

5.2 Preliminary Simulation and its Results

By using the traffic data of Web servers A, B and C (the span is 288 discrete time), we estimated ρ for the preliminary simulation.

Analytically, it is still unknown whether the log-likelihood function $\log L(\rho)$ is convex or not. However, the numerical calculation showed that the three log-likelihood functions are at least all convex. Figure 1 shows the numerical plot of log-likelihood function for Server B. In this case, the maximum likelihood estimation is doable by a numerical method (the simple gradient method).

Table 2 shows the results of $\hat{\rho}$. As previously described in 2.2, the constant ρ expresses the nonstationarity of x_t. Hence according to this result, the degree of nonstationarity becomes the following order: Server A < Server B < Server C.

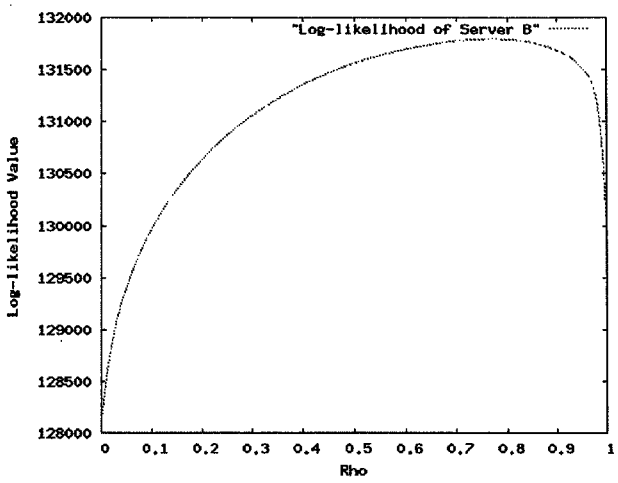


Figure. 1: Log-likelihood Function of Server B

Table. 2: The Maximum Likelihood Estimation Results of ρ

288 Discrete Times	Server A	Server B	Server C
Value of $\hat{\rho}$	0.960	0.769	0.461

5.3 Simulation and its Results

This simulation evaluates the performance of the time series model with a time varying density parameter in terms of accuracy of the prediction. First of all, we estimated the value of $\hat{\rho}$ from the traffic data of 144 discrete times of Server B and C. Then, using these $\hat{\rho}_B, \hat{\rho}_C$ as well as the rest of data (the other 144 discrete times (145-288) of data), we executed the time series prediction and compare the mean square error between the following two models: one is the time series model with time varying density parameter described in 2.1, the other is stationary poisson model.

With the above simulation, we got the estimated $\hat{\rho}$ on Table 3. Then, by using those $\hat{\rho}_B, \hat{\rho}_C$, we got

Table. 3: The Maximum Likelihood Estimation Results of ρ

144 Discrete Times	Server B	Server C
Value of $\hat{\rho}$	0.775	0.397

the mean square error of each server on Table 4. Table 4 shows that the time series model with the no stationary (time varying) density parameter has better predicting performance than the stationary model. Especially for Server C, the nonstationary model performed extremely better prediction. In this case, since $\hat{\rho}_C = 0.397$, the nonstationarity is relatively larger and the model followed observed data well. This tendency can be also observed from Figure 2 and 3. The predicted series from the nonstationary model followed better than that of the exact stationary model. Since $\rho = 1.0$ means to the stationary, we can conclude that this model includes stationary model as a special case and can be more useful. Additionally, this ρ can be useful in terms of the server administration or the decision of technical specification of web servers.

Table. 4: The Mean Square Error of Nonstationary and Stationary Poisson Model

MSE	Server B	Server C
Nonstationary	2.243×10^3	2.989×10^1
Stationary	3.894×10^3	1.787×10^2

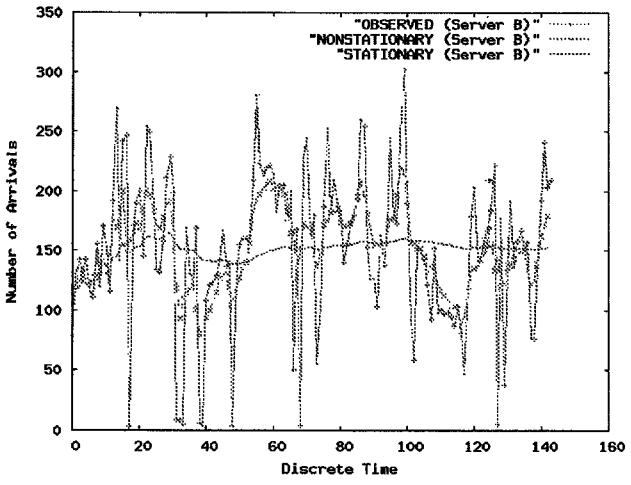


Figure. 2: Observed and Prediction Plot of Server B

6 Concluding Remarks

In this research, we took the time series model with the time varying density parameter and showed some analyzed results of real HTTP request data. Since this

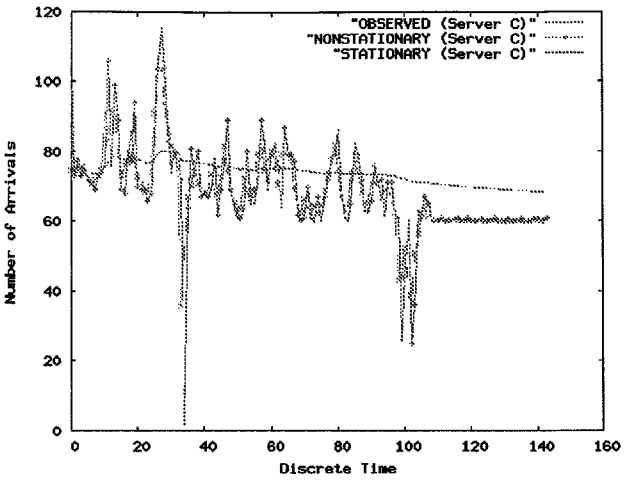


Figure. 3: Observed and Prediction Plot of Server C

model has a constant: $\rho (0 < \rho \leq 1)$, which expresses the nonstationarity, there is more possibility to fit the various types of the real HTTP data than the classical stationary Poisson model.

Acknowledgements

One of the authors, Daiki KOIZUMI, would like to thank Mr. Manabu Kubota and Mr. Yutaka Tajiri in Media Network Center, Waseda University for providing the web traffic data.

This research is partially supported by the Grant-in-Aid for Scientific Research (C) No.15560338 of the Japan Society for the Promotion of Science (JSPS).

References

[1] Kinya IWATA, Takahiro YOSHIDA and Toshiyasu MATSUSHIMA, "A Study of Modeling of Nonstationary Time Series based on Poisson Distribution," *Technical Report of IEICE*, IT-2003-39, pp.93-98, July 2003 (in Japanese).

[2] Shohei SATO and Makiko YOSHIDA, "Next Generation Internet and Traffic Engineering," *IEICE Transactions on Communications*, Vol.J85-B, No.6, pp.875-889, June 2002 (in Japanese).

[3] Vern Paxon and Sally Floyd, "Wide Area Traffic: The Failure of Poisson Modeling, " *IEEE/ACM Transactions on Networking*, Vol.3, No.3, June 1995.

[4] Hideaki SHINO, Keiichi KITAZAWA and Kazuo YANA, "A Method for the Nonstationary Analysis of HTTP Communication Request Occurrences in Internet Access Networks,,' *IEICE Transactions on Communications*, Vol.J84-B, No.8, pp.1494-1504, August 2001 (in Japanese).

Bayes Universal Coding Algorithm for Side Information Context Tree Models

Toshiyasu Matsushima
Waseda University
Tokyo, Japan
E-mail: toshi@matsu.mgmt.waseda.ac.jp

Shigeich Hirasawa
Waseda University
Tokyo, Japan
E-mail: hirasawa@hirasa.mgmt.waseda.ac.jp

Abstract— **The problem of universal codes with side information is investigated from Bayes criterion. We propose side information context tree models which are an extension of context tree models to sources with side information. Assuming a special class of the prior distributions for side information context tree models, we propose an efficient algorithm of Bayes code for the models. The asymptotic code length of the Bayes codes with side information is also investigated.**

I. INTRODUCTION

The problem of source coding with side information in this paper is defined as follows. A side information sequence y^n and a main information sequence x^n are occurred from an information source. A sender sends the side information sequence y^n and the code word $c(x^n)$ that is encoded from x^n and y^n. A receiver decodes the main information sequence x^n from $c(x^n)$ and y^n.

Universal codes with side information have been proposed by Ziv[1], Muramatsu[2], Subarahmanya[3], Yan[4], Uematsu[5] and et al. The asymptotically optimality for some of these codes was proved about i.i.d. and stationary ergodic sources.

Although the asymptotically optimality is a typical criterion of universal codes, they have been evaluated by Bayes, maxmin and minmax redundancy in some previous research. The asymptotic redundancy of the Bayes codes for i.i.d. source was investigated by Clark and Barron[6], and that for context tree models was studied by Gotoh et al.[10]. Efficient Bayes coding algorithms were also investigated for context tree models[7][9]. The CTW(Context Tree Weighting) algorithm[7] is interpreted as the Bayes coding algorithm assuming special prior distributions of context tree models and parameters.

In this paper, we propose side information context tree models which are an extension of context tree models to sources with side information. Secondly, we deduce universal codes with side information from Bayes criterion. Using a special class of prior distributions for side information context tree models, we propose an efficient algorithm of the Bayes code. Moreover, the asymptotic code length of Bayes codes with side information is also investigated

II. PRELIMINARY

A. Context tree models

The context tree model is a finite state source whose state at t is determined by the postfix of a source sequence x^t. The state set of a context tree model is a complete postfix set. Hence they satisfy the property that no string is a postfix of another. Let m be a context tree model. The state set of m is represented by a $|X|$-ary complete tree T_m^x called a context tree. Each arc in the tree corresponds to a symbol $x \in X$. A path from a leaf to the root in the tree represents a postfix or a context in the model. Let S_m^x be the set of all states in m. S_m^x corresponds to the set of all leaf nodes in the context tree T_m^x. The state $s \in S_m^x$ of a context tree model m at t is determined by the source sequence x^t. This mapping from x^t to a state $s \in S_m^x$ is denoted by $s_m^x(x^t)$.

The conditional probability of x_t of a context tree model is given by the following probability:

$$P(x_t|x^{t-1}) = P(x_t|\theta_{s_m^x(x^{t-1})}, s_m^x(x^{t-1})). \quad (1)$$

A context tree model m is represented by a context set or a leaf set S_m^x and probabilistic parameters $\theta_m^x = \{\theta_s | s \in S_m^x\}$.

B. Bayes codes and a special class of the prior distribuiton of context tree models

If a distribution of source sequences $P(x^n)$ is given, there are some coding algorithm such as the arithmetic coding algorithms that encodes the sequences to the code words whose mean code length converges to its entropy $H(X^n)$. Under the condition that such coding algorithm is used for coding, the decision of coding probability $P_C(x^n)$ is the main problem in universal coding.

When a context tree model $m \in M$ and its parameters $\theta_m^x \in \Theta_m^x$ are unknown, the optimal coding probability of x^n under Bayes criterion is given by

$$P_C(x^n) = \sum_{m \in M} \int P(x^n|\theta_m, m) P(\theta_m|m) P(m) d\theta_m. \quad (2)$$

Although the time complexity of the Bayes codes for the class of context tree models M_d whose depth is up to d is $O(2^{|X|^{d-1}})$, if a special class of prior distribution $P(m)$ on the class of context tree models M is assumed, the time complexity is reduced to $O(d)$).

Assumption 1: We assume that the prior probability $P(m)$ or $P(S_m)$ is represented by the following formula[8]:

$$P(m) = \prod_{s \in N_m - S_m} (1 - q(s)) \prod_{s_L \in S_m} q(s_L), \quad (3)$$

where N_m denotes the set of all nodes on the context tree model T_m^x and

$$q(s) = \frac{P(S_{m_s})}{\sum_{S \in S^d} P(S_{m_s} \cup S)}, \quad (4)$$

where $m_s \in \{m | s \in S_m\}$, S_d denotes the class of the set of descendant nodes of s that construct a complete tree.

Generally, the right hand side of Formula (4) might have different value at each m_s. However, it has the same value at all m_s under this assumption. Although the assumption restricts the class of prior distributions of context tree models, the complexity for calculating its posterior probability is reduced drastically. The prior probability of m is represented by $\{q(s) | s \in N_m\}$. Moreover, the posterior probability $P(m|x^t)$ is also represented by $\{q(s|x^t) | s \in N_m\}$. This is the mechanism for reducing the computational complexity.

Let the gathering context tree be $T^x = \bigcup_{m \in M} T_m^x$ and the set of all leaf nodes in T^x be L^x. Under the assumption, the conditional coding probability $P_c(x_t|x^{t-1})$ is calculated by the following recursive formulas.

$$P_c(x_t|x^{t-1}) = P_c(x_t|s = s_0), \quad (5)$$

$$P_c(x_t|s) = \begin{cases} P(x_t|s) & \text{if } s \in L^x \\ (*) & \text{otherwise}, \end{cases} \quad (6)$$

$$(*) = q(s|x^{t-1})P(x_t|s) + (1 - q(s|x^{t-1}))P_c(x_t|s_c), \quad (7)$$

where $s_c = \{xs \in \bigcup_{m \in M} s_m^x(x^t)\}$, s_0 is the root node of T^x and

$$P(x_t|s) = \int P(x_t|s, \theta_s) P(\theta_s|x^{t-1}) d\theta_s. \quad (8)$$

The memory structure of the algorithm is represented by a tree whose each node corresponds to a state s that holds $P(x|s)$ and $q(s|x^t)$. The calculation is done through the path from the leaf $s^x(x^{t-1})$ to the root s_0 on the tree. So the time complexity for the calculation is $O(d)$.

The posterior probability $q(s|x^t)$ is updated by the following formula:

$$q(s|x^t) = \frac{P(x_t|s)q(s|x^{t-1})}{P_c(x_t|s)}. \quad (9)$$

The update can be calculated simultaneously with the calculation of the conditional coding probability $P_c(x_t|x^{t-1})$ through the path from the leaf to the root.

The asymptotic code length of the Bayes codes[10] is given by

$$E_{X^n}[-\log P_c(X^n)] = H(X^n|\theta_{m^*}^*, m^*) + \frac{k}{2} \log \frac{n}{2\pi e} + \log \frac{\sqrt{\det I_X(\theta_{m^*}^*)}}{p(\theta_{m^*}^*)} + o(1), \quad (10)$$

where m^* is a true model, the true parameters $\theta_{m^*}^*$ of the model and $P(\theta_m)$ is the prior probability of θ_m, and k is the number of the parameters.

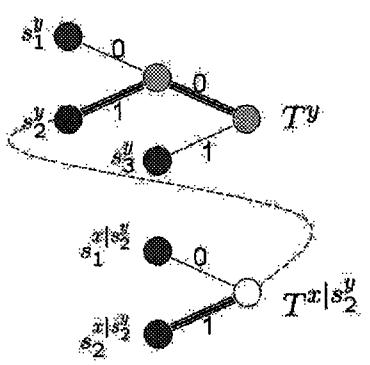


Fig. 1. An example of side information context tree models

The CTW algorithm is interpreted as a special case of the Bayes codes whose prior probability $q(s) = 1/2$.

III. Side Information Context Tree Models

We extend context tree models for source X^n to models for source X^n with side information Y^n. Let S_m^y be the set of all states for side information Y^n in m. S_m^y corresponds to the set of all leaf nodes in the context tree T_m^y. The state $s_m^y(y^t)$ for side information at t in a side information context tree model m is determined by the postfix of a source sequence y^t.

Let $S_m^{x|s^y}$ be the set of all states for main information sequence X^n given a side information state $s^y \in S_m^y$ in m. $S_m^{x|s^y}$ corresponds to the set of all leaf nodes in the conditional context tree $T_m^{x|s^y}$. The state $s_m^x(x^t|s^y)$ for main information given s^y at t is determined by the postfix of a source sequence x^t and y^t.

The conditional probability of x_{t+1} given (x^t, y^t) of a side information context tree model m is given by

$$P(x_{t+1}|x^t, y^t) = P(x_{t+1}|s_m^x(x^t|s^y), \theta_{s_m^x(x^t|s^y)}, s_m^y(y^t), m). \quad (11)$$

A side information context tree model m is represented by context sets S_m^y, $\{S_m^{x|s^y} | s^y \in S_m^y\}$ and probabilistic parameters $\theta_m^{x|y} = \{\theta_{s^x|s^y} | s^x \in S_m^{x|s^y}, s^y \in S_m^y\}$, $\theta_m^y \{\theta_{s^y} | s^y \in S_m^y\}$.

Example 1: An example of a context tree T^y and a conditional context tree $T^{x|s^y}$ of a side information context tree model is shown in Fig. 1. Under the condition that $y^{t-1} = \cdots 10$ and $x^{t-1} = \cdots 1$, the context or the state for side information corresponds to the node s_2^y on the context tree T^y and the state for main information corresponds to the node $s_2^{x|s^y}$ on the conditional context tree $T^{x|s^y}$. The conditional probability of x^t is represented by

$$P(x_t|x^{t-1} = \cdots 1, y^{t-1} = \cdots 10) = P(x_t|S_2^{x|S_2^y}, S_2^y). \quad (12)$$

IV. Bayes codes and an efficient algorithm for side information context tree models

A. Bayes universal codes with side information

We assume the prior probability of parameters $\theta_m^{x|y}, \theta_m^y$ as $P(\theta_m^{x|y}, \theta_m^y|m) = P(\theta_m^{x|y}|m)P(\theta_m^y|m)$. Under the assumption, the optimal Bayes coding probability of x^n for side information context tree models M is given by

$$P_C(x^n|y^n) = \sum_{m\in M} \int P(x^n|y^n, \theta_m^{x|y}, m)P(\theta_m^{x|y}|m)P(m)d\theta_m^{x|y}. \tag{13}$$

B. An efficient algorithm for side information context tree models

Under general prior probability of side information context tree models, it is very hard to calculate the coding probability $P_C^B(x^n|y^n)$ of the Bayes code. However, assuming a special class of the prior probability $P(m)$ on the class of side information context tree models, its time complexity can be reduced.

Assumption 2: We assume that the prior probability $P(m)$ of a side information context tree model is represented by the following formula:

$$P(m) = \prod_{s^y\in N_m^y - S_m^y} (1-q(s^y)) \prod_{s_L\in S_m^y} q(s_L^y) \prod_{s^x\in N_m^{x|s_L^y} - S_m^{x|s_L^y}} (1-q(s^x|s_L^y)) \prod_{s_L^x\in S_m^{x|s_L^y}} q(s_L^x|s_L^y). \tag{14}$$

Under the assumption, the conditional coding probability $P_c(x_t|x^{t-1}, y^{t-1})$ is calculated by the following double recursive formulas.

First, $P_b(x_t|s^y)$ is calculated through the path from the leaf $s^x(x^{t-1}|s^y)$ to the root $s_0^{x|s^y}$ recursively on each conditional gathering context tree $T^{x|s^y}$ given s^y.

$$P_b(x_t|s^y) = P_b(x_t|s^x = s_0^x, s^y) \tag{15}$$

$$P_b(x_t|s^x, s^y) = \begin{cases} P(x_t|s^x, s^y) & \text{if } s^x \in L^{x|s^y} \\ (*) & \text{otherwise}, \end{cases} \tag{16}$$

$$(*) = q(s^x|x^{t-1}, y^{t-1}, s^y)P(x_t|s^x, s^y) + (1-q(s^x|x^{t-1}, y^{t-1}, s^y))P_b(x_t|s_c^x, s^y). \tag{17}$$

Secondly, $P_c(x_t|x^{t-1}, y^{t-1})$ is calculated through the path from the leaf $s^y(y^{t-1})$ to the root s_0^y recursively by using $P_c(x_t|s^y)$ on the gathering context tree T^y of side information Y.

$$P_c(x_t|x^{t-1}, y^{t-1}) = P_c(x_t|s^y = s_0^y), \tag{18}$$

$$P_c(x_t|s^y) = \begin{cases} P_b(x_t|s^y) & \text{if } s^y \in L^y \\ (*) & \text{otherwise}, \end{cases} \tag{19}$$

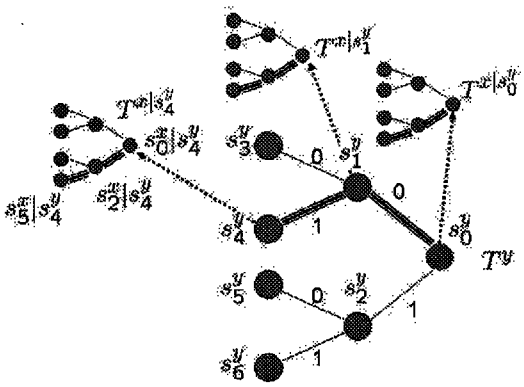


Fig. 2. An example of context trees used in the proposed algorithm

$$(*) = q(s^y|x^{t-1}, y^{t-1})P_b(x_t|s^y) + (1-q(s^y|x^{t-1}, y^{t-1}))P_c(x_t|s_c^y). \tag{20}$$

The posterior probability $q(s^x|s^y, x^t, y^t)$ is updated by the following formula:

$$q(s^x|s^y, x^t, y^t) = \frac{q(s^x|x^{t-1}, y^{t-1}, s^y)P(x_t|s^x, s^y)}{P_c(x_t|s^x, s^y)}. \tag{21}$$

The update can be calculated simultaneously with the calculation of $P_c(x_t|s^x, s^y)$ through the path from the leaf $s^x(x^{t-1}|s^y)$ to the root $s_0^{x|s^y}$ on the each conditional context tree $T^{x|s^y}$ given s^y.

The posterior probability $q(s^y|x^t, y^t)$ is updated by the following formula:

$$q(s^y|x^t, y^t) = \frac{q(s^y|x^{t-1}, y^{t-1})P(x_t|s^y)}{P_c(x_t|s^y)}. \tag{22}$$

The update can be calculated simultaneously with the calculation of $P_c(x_t|s^y)$ on the conditional context tree T^y of side information.

Example 2: We assume that the class of the context tree models T^y whose depth are up to 2 and the class of the conditional context tree models $T^{x|s^y}$ whose depth are up to 2 as shown in Fig. 2. Under the condition that $y^{t-1} = \cdots 10$ and $x^{t-1} = \cdots 11$, the coding probability $P_c(x_t|x^{t-1}, y^{t-1})$ is calculated as the following procedure.

First, $P_c(x_t|s_4^y)$ is calculated through the path from the leaf $s_5^x|s_4^y$ to the root $s_0^x|s_4^y$ recursively on the conditional context tree $T^{x|s_4^y}$. In the same way, $P_c(x_t|s_1^y)$ and $P_c(x_t|s_0^y)$ are calculated by using $T^{x|s_1^y}$ and $T^{x|s_0^y}$ respectively.

Secondly, $P_c(x_t|x^{t-1}, y^{t-1})$ is calculated through the path from the leaf s_4^y to the root s_0^y recursively by using $P_c(x_t|s^y)$ on the context tree T^y.

Let the depth of the context tree models T^y be d_y and the depth of the conditional context tree models $T^{x|s^y}$ d_x. The time complexity of the proposed algorithm is $O(d_x d_y)$. If Assumption 2 is not assumed, the time complexity of

the Bayes code is $O(2^{|Y|^{d_y-1}}2^{|X|^{d_x-1}}$. In the case where parallel computing is done, the parallel time complexity of the proposed algorithm is $O(d_x+d_y)$. The space complexity is $O(2^{d_x}2^{d_y})$.

V. Mean code length of the Bayes codes with side information

We also investigate asymptotic code length of Bayes codes for side information context tree models.

Definition 1: Let $E_{X^nY^n}$ denote the expectation with respect to $P(X^n,Y^n|\theta_{m^*}^{x|y^*},\theta_{m^*}^{y^*})$, where $\theta_{m^*}^{x|y^*}$ and $\theta_{m^*}^{y^*}$ are the true value of $\theta_m^{x|y}$ and θ_m^y respectively. An information matrix $I_{X|Y}(\theta_m^{x|y})$ is defined by

$$I_{X|Y}(\theta_m^{x|y}) = -\lim_{n\to\infty}\frac{1}{n}E_{X^nY^n}\left[\frac{\partial^2\log P(X^n|Y^n,\theta_m^{x|y})}{\partial\theta_m^{x|y}(\partial\theta_m^{x|y})^T}\right]. \quad (23)$$

We call $I_{X|Y}$ conditional Fisher information matrix.

Theorem 1: The average code length of the Bayes codes with side information under some suitable assumption is given by

$$E_{X^nY^n}[-\log P_c(X^n|Y^n)] = H(X^n|Y^n,\theta_{m^*}^{x|y^*},m^*)+\frac{k}{2}\log\frac{n}{2\pi e} +\frac{1}{2}\log\det I_{X|Y}(\theta_{m^*}^{x|y^*})+\log\frac{1}{P(\theta_{m^*}^{x|y^*})}+o(1). \quad (24)$$

Although the constant term in the mean code length of an ordinary Bayes code is represented by Fisher information, remark that that of the Bayes codes for side information context is represented by conditional Fisher information.

VI. Conclusion

We deduced universal codes with side information from Bayes criterion. Assuming a special prior probability of side information context tree models, we proposed an efficient algorithm of the Bayes code for the models. Moreover, the asymptotic code length of the Bayes codes with side information was investigated.

References

[1] J. Jiv, "Fixed-rate encoding of individual sequences with side information," *IEEE Trans. Inform. Theory*, vol.IT-30, no.2, pp.348-352, March 1984.

[2] J. Muramatsu, "Universal data compression algorithms for stationary ergodic sources based on the complexity of sequences," Ph.D.Thesis, Nagoya University, Nagoya, Japan, 1998.

[3] P. Subrahmanya and T. Berger, "A sliding window Lempel-Ziv algorithm for differential layer encoding in progressive transmission," *Proc. 1995 IEEE Int. Symp. on Inform. Theory*, pp266, Sept. 1995.

[4] E.-H. Yang, A. Kaltchenko, and J.C. Kieffer, "Universal lossless data compression with side information by using a conditional MPM grammer transform," *IEEE Trans. Inform. Theory*, vol.IT-47, pp.2130-2150, Sept. 2001.

[5] T.Uematsu and K.Maeda, "Asymptotic Optimality for Universal Data Compression Algorithm with Side Information Based onIncrementParsing," IEICE A, vol. J85-A, no.1, pp.95-102, Jan. 2002(Japanese).

[6] B.S. Clark and A.R. Brron, "Information-theoretic asymptotics of Bayes methods," *IEEE Trans. Inform. Theory*, vol.IT-36, no.3, pp.453-471, May 1990.

[7] F. M. J. Willems, Y. M. Shtarkov and T. J. Tjalkens, "The Context-Tree Weighting Method:Basic Properties," *IEEE Trans. Inform. Theory*, vol.41, no.3, pp.653-664, May 1995.

[8] T. Matsushima, H. Inazumi and S. Hirasawa, "A Class of Distortionless Codes Designed by Bayes Decision Theory," *IEEE Trans. Inform. Theory*, vol.IT-37, no.5, pp.1288-1293, Sep. 1991.

[9] T. Matsushima and S. Hirasawa, "A Bayes coding algorithm for FSMX sorces," In *Proc. Int. Symp. of Information Theory*, pp.388, 1995.

[10] M. Gotoh, T. Matsushima, S.Hirasawa, "A generalization of B.S.Clarke and A. R. Barron' s asymptotics of Bayes codes for FSMX sources," *IEICE Trans. fundamentals*, vol.E81-A, no.10, pp.2123-2132, Oct. 1998.

[11] T. M. Cover and J. A. Thomas, "Elements of Information Theory," John Wiley & Sons, 1991.

社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE

畳み込み符号の並列復号アルゴリズムの性能評価に関する一考察

大澤 雅之　野村 亮　松嶋 敏泰

早稲田大学 理工学部 経営システム工学科

〒169-8655 東京都新宿区大久保 3-4-1

tel.03-5286-3301

fax.03-5286-3301

E-mail: masa0115@toki.waseda.jp

あらまし LDPC 符号，ターボ符号等は近似的シンボル事後確率最大復号法により，高い誤り訂正能力が実験的に確かめられ注目を集めている．近年，それらの符号の復号アルゴリズムの性能評価法として，復号器全体のシミュレーションではなく部分的にモンテカルロ法を用いた方法が提案されている．その代表的な例としてはターボ符号の BCJR 復号，LDPC 符号の Sum Product 復号における EXIT-CHART 法，ターボ符号の BCJR 復号におけるガウス近似法等である．また，一般化事後確率計算アルゴリズムを LDPC 符号，畳み込み符号，ターボ符号等に適用すると，それらの従来の復号アルゴリズムに比べて優れた性能を示すことが実験的に分かっている．本稿では畳み込み符号を一般化事後確率計算アルゴリズムで復号した時の部分的にモンテカルロフィルタ法を用いた性能評価法に関する考察を行う．

キーワード 畳み込み符号，一般化事後確率計算アルゴリズム，モンテカルロフィルタ法

A Study of Analysis of Alternate Decoding Algorithm of Convolutional Codes

Masayuki OSAWA　Ryo NOMURA　and　Toshiyasu MATSUSHIMA

Dep.of Industrial and Management Systems Engineering

Waseda University

3-4-1 Ohkubo,Shinjuku-ku,Tokyo,169-8655 Japan

tel. 03-5286-3301

fax. 03-5286-3301

E-mail: masa0115@toki.waseda.jp

Abstract The maximum a posterior probability decoding of Low Density Parity Check (LDPC) Codes and Turbo Codes is one of the most active subjects in research of error correcting codes. Recently, various methods of analyzing decoding algorithm of these codes using Monte Carlo method partially, not entirely, are proposed. The representative methods of this type are EXIT-CHART for BCJR algorithm on Turbo Codes and for sum product algorithm on LDPC Codes and Gaussian Approximation for BCJR algorithm on Turbo Codes. On the other hand, alternate algorithm for generalized posterior probability on LDPC Codes, Convolutional Codes and Turbo Codes turned out to be superior to sum product algorithm or BCJR algorithm on those codes in many points by simulations. In this paper we consider a method of analyzing alternate algorithm for generalized posterior probability on Convolutional Codes using Monte Carlo Filter method partially.

Keyword Convolutional Codes, An Alternate Algorithm for Generalized Posterior Probability, Monte Carlo Filter Method

1. はじめに

代表的な誤り訂正符号に畳み込み符号，ターボ符号，LDPC 符号がある．畳み込み符号，ターボ符号の代表的な復号アルゴリズムである BCJR アルゴリズム[6]，LDPC 符号の代表的な復号アルゴリズムである Sum P-roduct(SP)アルゴリズム[5]は Belief Propagation(BP)と呼ばれる事後確率計算アルゴリズムと等価である事が分かっている．両者の相違点は，BCJR は直列的に BP を適用した場合であり，SP は並列的に BP を適用した場合であるという点である．BP は Bayesian Network(BN)と呼ばれるグラフ表現において Loop が存在しない場合，正しい事後確率計算が可能である．畳み込み符号の BN 表現は Loop が存在しないため，正しい事後確率計算が保証されている．一方でターボ符号，LDPC 符号の BN 表現は Loop が存在し，正しい事後確率計算は出来ず，近似事後確率計算アルゴリズムとなる．

他の畳み込み符号，ターボ符号，LDPC 符号等の復号アルゴリズムとして一般化事後確率計算アルゴリズム[11]がある．このアルゴリズムには Extended Junction Graph(EJG)と呼ばれる確率モデルを用いる．このアルゴリズムも Loop が存在しない EJG 上で正しい事後確率計算が可能である．畳み込み符号の EJG による表現は Loop が存在しないため，正しい事後確率計算が保証されている．このとき，BCJR アルゴリズムより計算時間を大幅に削減可能であることが実験的に分かっている．ターボ符号，LDPC 符号の EJG 表現は Loop が存在し，正しい事後確率計算は出来ないが，この場合も BP と同様に近似事後確率計算が可能である．LDPC 符号にこのアルゴリズムを適用したとき，BN 表現における Small Loop による性能の劣化を防ぐことができることが実験的に分かっている [7].

これらの復号アルゴリズムの性能評価法としてはシミュレーションによる評価がなされてきた．しかし，シミュレーションによる評価は実験誤差やシミュレーションに時間がかかったりするという問題点がある．そこで，これらの復号アルゴリズムの様々な性能評価法が提案されてきた．特にモンテカルロ法を部分的に用いる方法として，ターボ符号の BCJR 復号，LDPC 符号の Sum Product 復号における EXIT-CHART 法[8]，ターボ符号の BCJR 復号におけるガウス近似法[10]がある．これらはターボ符号，LDPC 符号の各要素符号器についてのみモンテカルロ法を用いればよく符号器全体のシミュレーションは不要である点が特徴である．

本研究では，一般化事後確率計算アルゴリズムを用いた畳み込み符号の復号を対象とし，モンテカルロフィルタ法(以下 M-F 法)を用いて 1 つのクリークのみのメッセージ更新によって，メッセージの密度関数の更新を行う性能評価法を実施する．また，LDPC 符号，ターボ符号の伝播メッセージの密度関数は正規分布で上手く近似されている[9][10]．そこで一般化事後確率計算アルゴリズムによる畳み込み符号の復号の場合の伝播メッセージの正規近似の是非について検討する．

2. 従来研究

2.1. 畳み込み符号と BCJR アルゴリズム

畳み込み符号とは線形シフトレジスタからなる符号器の出力系列の集合である．この符号に対するシンボル事後確率最大復号法としてトレリス線図を用いた BCJR アルゴリズムがある．これは畳み込み符号を BN 表現し，BP を適用したアルゴリズムと等価である．update ルールは符号の最初から最後まで前向きのメッセージを直列に update し，次に逆に最後から最初まで後ろ向きのメッセージを直列に update するというものである．このアルゴリズムの計算時間は明らかに符号長に比例する．畳み込み符号の BN 表現はループを持たないため，このアルゴリズムで正しい事後確率計算が可能である．

2.2. 一般化事後確率計算アルゴリズム[11]

条件付依存関係を持つ確率変数の組をクリークとして表し，各クリーク間の共通部分をセパレータとしてその該当クリークノードを結んだ 2 種類のノードからなるグラフを Extended Junction Graph (EJG) と呼ぶ．EJG 上での確率推論アルゴリズムとして，クリーク間でセパレータに関する分布情報をやりとりする一般化事後確率計算アルゴリズム(以下 DGA：[11]の message propagation(unique type message))がある．このアルゴリズムは一般化事後確率を計算するアルゴリズムであるが，通常の事後確率計算にも適用可能である．DGA は全てのクリーク-セパレータ間でのメッセージのやり取りを同時に行う並列型アルゴリズムである．BP と同様，ループのない EJG についてはこのアルゴリズムで正しい事後確率計算が可能であることが示されている．また，DGA は BP より広い確率モデルのクラスについて正しい事後確率計算が可能である．それは，ループのない BN で表される確率モデルのクラスに比べてループのない EJG で表される確率モデルのクラスのほうが広いことによる．

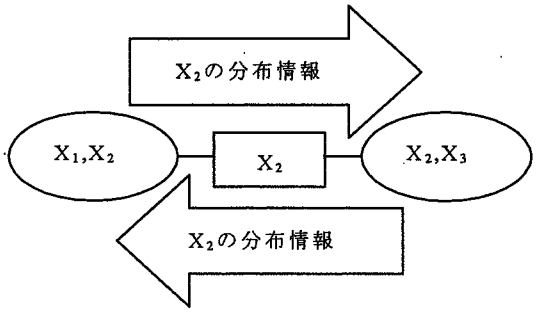


図 1：EJG と DGA のイメージ図
(クリーク={X_1,X_2},{X_2,X_3}／セパレータ={X_2})

DGA の詳細は以下のようになる．

＜ノーテーション＞

$N_1,\ldots,N_{n_N}$ ：クリーク

$D_1,\ldots,D_{n_D}$ ：セパレータ（D_mの分布が与えられている時，これを r.i.n と呼ぶ）

$S^N(D_m)$：D_mの近傍ノード集合

$q(N_i)$：N_iの初期分布

$P^{※}(D_m)$：D_mの与えられた分布

＜アルゴリズムの詳細＞

a)時刻 $t+1$での $N_i \to N_l$のメッセージ（$u_{t+1}^{i\to l}$）

$$u_{t+1}^{i\to l}=\begin{cases}\dfrac{P^{※}(D_m)}{u_t^{l\to i}} & (D_m\text{が}r.i.n)\\ \displaystyle\sum_{x\in D_m} q(N_i)\prod_{N_k\in S_l^N(N_i)} u_t^{k\to i} & (\text{それ以外})\end{cases} \quad (1)$$

※ $S_l^N(N_i)=\{N_k\in S^N(D_h)\mid D_h\in S^D(N_i), D_h\in/S^D(N_l), k\neq i\}$

b)時刻 tでのあるクリーク N_lの事後確率

$$P_t(N_l)=q(N_l)\prod_{N_k\in S^N(N_l)} u_t^{k\to l}. \quad (2)$$

※ $S^N(N_l)=\{N_k\in S^N(D_h)\mid D_h\in S^D(N_l), k\neq i\}$

2.3. 畳み込み符号に適用したときの実験的知見[3]

畳み込み符号を DGA で復号したときの実験的に分かっている特性を以下に述べる．まず，誤り率が収束するまでの反復回数を符号長，拘束長の変化について調べた結果より，収束するまでに必要な反復回数は拘束長に比例し，符号長には比例しないことが分かっている．また，収束したときのビット誤り率は BCJR アルゴリズムのそれとほぼ等しくなることが分かっている．よって，このアルゴリズムは BCJR アルゴリズムとほぼ同じ誤り率特性を計算時間を大幅に減らして達成できるアルゴリズムであるといえる．しかし，このアルゴリズムは並列計算であり，総計算量は大幅に増加する．

3. 畳み込み符号の並列復号の性能評価

本節では，2.3 節で述べた実験的に分かっている畳み込み符号の DGA 復号の良さを確かめるためにその性能評価法を考察する．畳み込み符号の DGA 復号においては，終端の影響を受けないクリーク全てにおいて，メッセージの分布は同じである．そこで，M-F 法を用い，終端の影響を受けない 1 つのクリークのみにおけるメッセージ分布を更新し，誤り率の収束性を見ていく次の性能評価法を実施する．

3.1. モンテカルロフィルタ法による性能評価法

Ⅰ.畳み込み符号の構成から EJG，クリークの分割表を作成する．分割表は符号の構造上ありうる組み合わせを等確率で分配し，それ以外を 0 とする．

Ⅱ.M-F 法を用いて 1 つの終端の影響を受けないクリークにおける(1)式の更新メッセージ，

$\sum_{x\in D_m} q(N_i)\prod_{N_k\in S_l^N(N_i)} u_t^{k\to i}$，の分布を既知の $u_t^{k\to i}$

の分布から求める．ここで求める分布，既知の分布とは共に 5000 個のメッセージの実現値そのものを粒子として持つ分布である．求める更新メッセージの分布は右向き，左向きの 2 種類である．

Ⅲ.Ⅱで求めた粒子による分布を既知の $u_t^{k\to i}$の分布として用いて次の反復で更新されるメッセージの粒子による分布を求める．これを誤り率が収束するまで繰り返す．また，平行して誤り率も求めていく．

3.2. 具体例を用いたモンテカルロフィルタ法による性能評価法の説明

3.1 の性能評価法を具体例を用いて説明する．ここでは，簡単な畳み込み符号{拘束長 1，符号化率 $1/2$，$G=(D,1+D)$}を用いることとする．また，$X_i\in\{0,1\}$とする．この符号の EJG は図 2 になる．

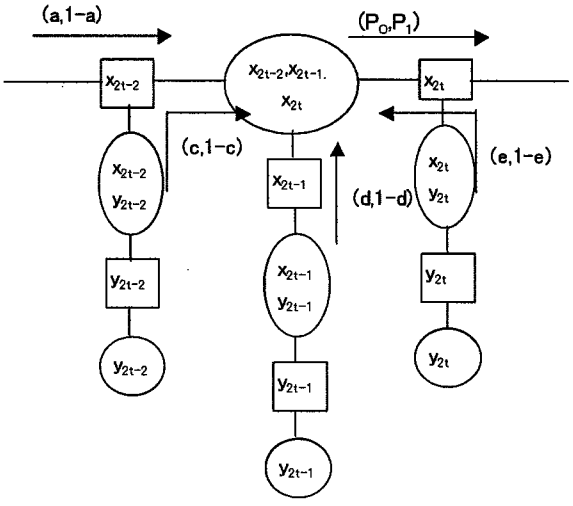


図 2：例の畳み込み符号の EJG

ここで，y_tは時刻 tにおける受信ビットであり，x_tは時刻 tにおける送信ビットである．図 2 ではクリー

ク $(X_{2t-2}, X_{2t-1}, X_{2t})$ から $(X_{2t}, X_{2t+1}, X_{2t+2})$ へのメッセージ (P_0, P_1) を求める場合を考え，それに必要な各メッセージを矢印とアルファベットで表している．

ここで，クリーク $(X_{2t-2}, X_{2t-1}, X_{2t})$ の分割表は図3のようになり，(3)，(4)式が成り立つ．

X_{2t}

X_{2t-2}			0			1	
		0	X_{2t-1}	1	0	X_{2t-1}	1
0	0.25		0	0			0.25
1	0		0.25	0.25			0

図3：クリーク $(X_{2t-2}, X_{2t-1}, X_{2t})$ の分割表

$$P_0 = 0.25 \times (acde + (1-a)(1-c)(1-d)e). \quad (3)$$

$$P_1 = 0.25 \times (ac(1-d)(1-e) + (1-a)(1-c)d(1-e)). \quad (4)$$

通信路を分散 σ^2 の白色ガウス通信路と仮定すると，(x_i, y_i) の関係は，0→1，1→-1 として送信した場合を考えると次のようになる．

$$P(x_i = 0) = \frac{\exp(2y_i/\sigma^2)}{1 + \exp(2y_i/\sigma^2)}. \quad (5)$$

$$P(x_i = 1) = \frac{1}{1 + \exp(2y_i/\sigma^2)}. \quad (6)$$

ここで，送信語を全0とすると，$y_i \sim N(1, \sigma^2)$ である．図2の(a,1-a)の初期値は(0.5，0.5)であり，M-F法でクリーク $(X_{2t-2}, X_{2t-1}, X_{2t})$ からの右方向のメッセージ (P_0, P_1) の粒子による分布を求める．2回目以降，左方向のメッセージも同様に1つのクリークのM-F法によってメッセージの粒子による分布を更新していく．誤り率が収束するまで繰り返せば反復回数ごとの誤り率がクリーク1個のM-F法で計算可能である．

3.3. 正規近似による性能評価

メッセージの正規近似についての検討を次の手順で行う．ここではメッセージの分布はすべて2000個の粒子で表すことにする．また，ここでは拘束長2の畳み込み符号を用いることとする．このときメッセージは4次元となる．$P_{00}, P_{01}, P_{10}, P_{11}$ をその4元メッセージとすると，$\ln\frac{P_{00}}{P_{01}}, \ln\frac{P_{00}}{P_{10}}, \ln\frac{P_{00}}{P_{11}}$ の3つの対数比でメッセージを完全に記述できる．

そこで，まずはこの3つのメッセージがそれぞれ正規分布に従っているかを調べる．具体的には，今回は3回目のメッセージ更新から正規近似を行うので3回目のメッセージ分布について，(7)式の平均の周りの標本r次モーメント m_r' から(8)式の標本歪度，(9)式の標本尖度を求め，それらが近似的に $b_1 \sim N(0, 6/n)$，$b_2 \sim N(0, 24/n)$ となることより有意水準 5%，1%で検定する．

$$m_r' = \sum (X_i - \bar{X})^r / n. \quad (7)$$

$$b_1 = m_3' / (m_2')^{3/2}. \quad (8)$$

$$b_2 = m_4' / (m_2')^2 - 3. \quad (9)$$

また，この3つのメッセージの間には相関があるので，この3つの対数比の平均，分散，共分散を求め，それらのパラメータを持つ3次元正規分布をメッセージの分布と仮定する．そしてその分布からの2000個の乱数から誤り率を計算することで正規近似の良さを見る．

今回，反復回数3回目からのメッセージを正規近似することにする理由は，今回使用した符号器では反復回数2回目のメッセージを正規近似するとその後の反復で誤り率がほとんど変化しないという現象が起こるからである．3回目以降に正規近似を始めると正規近似後5反復の誤り率のM-F法との2乗誤差はほぼ一定になる．収束までの時間が短いことから早く正規近似できなければ意味がないので，反復回数3回目からのメッセージを正規近似することにした．

4. 実験値との比較，正規近似の妥当性

拘束長2の符号器を用い，符号長を 50，100，500，1000 と変えたときの実験値と M-F 法を用いた性能評

価法の誤り率の比較を図4，図5，図6で示す．これらは反復回数3回，6回，10回の時の復号誤り率の比較である．10回のときはすべての誤り率が収束している．図7，図8は符号器の拘束長の違いによる収束性の違いを調べるために，Eb/N0を2に固定したもとで，M-F法によって求めた拘束長1から4までの畳み込み符号の反復回数ごとの誤り率を示している．反復回数2から8回が図7であり，反復回数9から19回が図8である．用いた符号器は以下の4つである．

拘束長1：$G=(D,1+D)$. (10)

拘束長2：$G=(D^2,1+D+D^2)$. (11)

拘束長3：$G=(D^3,1+D+D^2+D^3)$. (12)

拘束長4：$G=(D^4,1+D+D^2+D^3+D^4)$. (13)

図 9,10 は正規近似の妥当性について検討するための，対数尤度比メッセージの正規性の検定結果である．*は 5%有意，**は 1%有意を表す．図 11 は反復回数 4, 6, 8 回のときの正規近似と M-F 法との誤り率の比較図である．ここでは拘束長 2 の符号器を用いた．

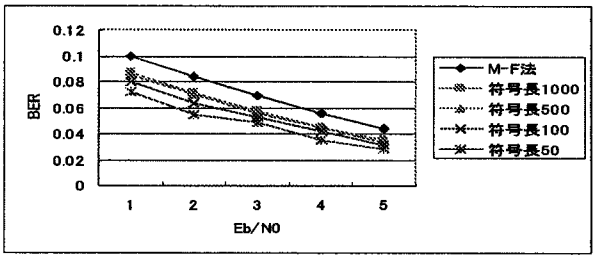


図 4：反復回数 3 回の時の誤り率

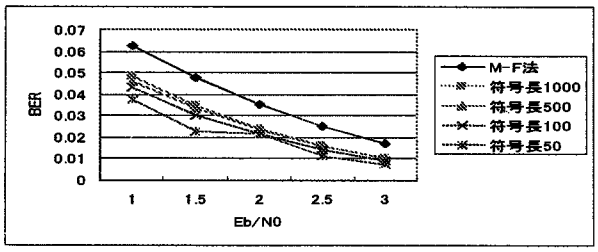


図 5：反復回数 6 回の時の誤り率

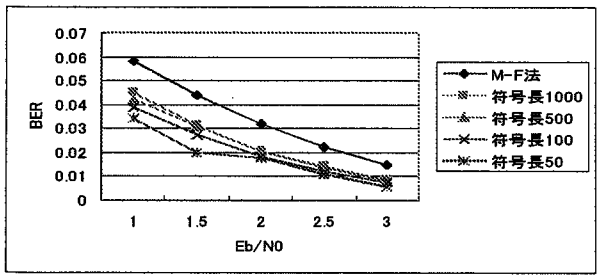


図 6：反復回数 10 回の時の誤り率

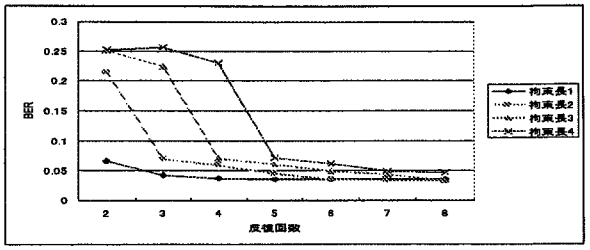


図 7：拘束長の違いによる誤り率の収束の仕方(2-8)

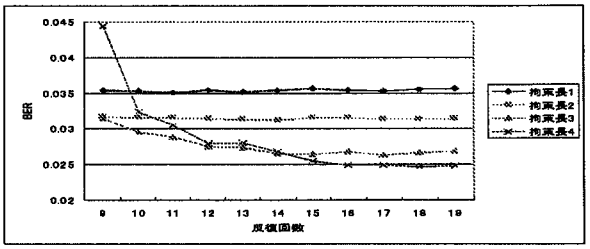


図 8：拘束長の違いによる誤り率の収束の仕方(9-16)

	$\ln(P_{00}/P_{01})$	$\ln(P_{00}/P_{10})$	$\ln(P_{00}/P_{11})$
歪度	0.1208*	0.0228	0.0352
尖度	0.0632	−0.0738	0.0104

図 9：左向きメッセージの正規性の検定

	$\ln(P_{00}/P_{01})$	$\ln(P_{00}/P_{10})$	$\ln(P_{00}/P_{11})$
歪度	0.0593	0.0296	0.1443**
尖度	−0.0349	−0.2155*	−0.121

図 10：右向きメッセージの正規性の検定

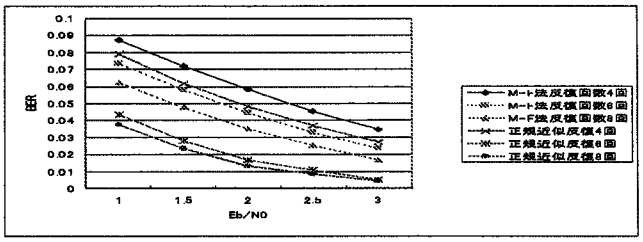


図 11：正規近似による M-F 法との誤り率の差

5. 考察

M-F 法で求めた誤り率は実験値のそれよりも高くなっている．これは終端の影響を考慮していないことから妥当な結果である．また，実験値の誤り率は符号長が長くなるにつれてM-F法で求めた誤り率に近づいていくことが分かる．これは符号長が長くなればその分終端の影響が小さくなることが原因であると思われる．つまりM-F法で求めた誤り率は終端の影響が全くない符号長無限大の場合の誤り率を示しており，これは誤り率の上界であると考えられる．また，図4，図5，図6の3つを見比べてみると誤り率の収束の仕方は M-F 法のそれと各シミュレーションのそれとが非常に似通っていることが分かる．よって M-F 法の結果をこの符

号の収束性の検討に使うことができると考えられる．

M-F法で収束性を調べたのが図7，図8である．M-F法は符号長無限大の場合の誤り率を求めていると考えられることから，収束するまでの反復回数が符号長によらず拘束長に比例するという性質を確認することができた．よってDGAによる畳み込み符号の復号はBCJRアルゴリズムによるそれより計算時間を大幅に削減できる復号アルゴリズムであることが確認できた．符号長が長くなればなるほど計算時間の差が大きくなり，その有用性は増すといえる．

また，拘束長の差による違いについて考察すると，拘束長が短い符号は反復回数が少ない範囲から小さい誤り率を示すが，反復回数を増やしていってもあまり誤り率が改善されない．一方で拘束長が長い符号は反復回数が少ない範囲では誤り率が非常に高いが，反復回数を増やしていくと誤り率が劇的に改善され，最終的には拘束長が短い符号より誤り率が小さくなる．また，図7を見て気づくことに，拘束長3の符号は反復3回まで，拘束長4の符号は反復4回までほとんど誤り率が減少しないということがある．これは各クリークを構成する送信語(X)が拘束長が増えることによって増えていくので，誤り率の改善にはより広範囲の受信情報が必要であるからであると思われる．この傾向はさらに拘束長を増やしたときも成り立つと思われ，DGAで拘束長の大きい畳み込み符号を復号したときの特徴であると思われる．

正規近似に関しては妥当であるとは言い難い．3次元正規分布の仮定が成り立つならば，各対数尤度比メッセージも正規分布に従っていなければならないが，図9，10より正規分布とはいえないものがいくつか現れた．図11の誤り率の比較もかなりかけ離れたものになっている．ターボ符号，LDPC符号においては正規近似の精度が非常に良かったが，今回はメッセージが3次元になったこと，和演算の回数がターボ符号，LDPC符号に比べてかなり少ないため，中心極限定理が成り立つとはいえないことが影響していると考えられる．また，反復2回目から正規近似したときにその後の誤り率が非常に高い値でとまってしまうという問題点があったが，これも上で述べたDGAで拘束長の大きい畳み込み符号を復号したときの特徴が影響していると思われる．

6. まとめ

本研究では畳み込み符号のDGA復号の性能評価法について考察し，シミュレーション値との比較，メッセージの分布を正規分布と仮定した時の妥当性の検討を行った．その結果，この方法は性能評価に有効であること，畳み込み符号のDGA復号の利点を確認できた．また，この性能評価法，DGA復号の利点はTail-biting畳み込み符号においても同様に成り立つと思われる．

今回はM-F法を用いており，LDPC符号のDensity Evolution[1]のような解析的（数値的）性能評価が今後の課題である．

7. 謝辞

本研究を行うにあたり，数多くの御助言，御支援を賜りました早稲田大学松嶋研究室各氏に心より感謝申し上げます．なお，本研究の一部は文部省科学研究費基盤研究(C)(No.15560338)の援助による．

参考文献

[1]T.J.Richardson,R.L.Urbanke,"The capacity of low density parity check codes under message-passing decoding,"IEEE Trans.IT,Vol47,No.2,pp.599-618,2001.

[2]T.Matsusima,T.K.Matsusima,S.Hirasawa,"A parallel iterative decoding algorithm for convolutional codes and tail biting convolutional codes,"ISIT Symp. IT, 2003.

[3]制野宇樹,"確率伝播アルゴリズムの並列処理による高速復号法－事後確率計算アルゴリズムに関する研究,"早稲田大学修士論文,2001.

[4]Y.Weiss,"Correctness of local probability propagation in graphical models with loops,"Neural Computation12, pp.1-41, 2000.

[5]D.J.C.MacKay,"Good error correcting codes based on very sparse matrices,"IEEE Trans.IT,Vol.45,No.2,pp.391-431,1999

[6]L.Bahl,J.Cocke,F.Jelinek,J.Raviv,"Optimal decoding of linear codes for minimizing symbol error rate,"IEEE Trans.IT, Vol20,No6,pp.284-287,1974.

[7]小林直人,野村亮,松嶋敏泰,"低密度パリティチェック符号の復号アルゴリズムに関する一考察," 信学技報, IT2001-81,pp.39-44,2001.

[8]S.T.Brink,"Convergence behavior of iteratively decoded parallel concatenated codes,"IEEE Trans.Comm,Vol49, No.10,pp.1727-1737,2001.

[9] S.Y.Chung,T.J.Richardson,R.L.Urbanke,"Analysis of sum product decoding of low density parity check codes using a Gaussian approximation,"IEEE Trans.IT,vol47, No.2, pp.657-670,2001.

[10]H.E.Gamal,A.R.Hammons Jr,"Analyzing the turbo decoder using the Gaussian Approximation,"IEEE Trans.IT,vol47, No.2,pp.671-686,2001.

[11] T.Matsusima,T.K.Matsusima,S.Hirasawa,"An alternate algorithm for calculating generalized posterior probability and decoding," ISIT Symp. IT, 2002.

[12]北川源四郎,"モンテカルロフィルタおよび平滑化について,"統計数理,44-1,pp31-48.1996.

The 26th Symposium on Information Theory
and Its Applications (SITA2003)
Higashiura, Hyogo, Japan, Dec. 15–18, 2003

区間で一定なパラメータを持つ情報源におけるベイズ符号化法について
Bayes Coding for Sources with Piecewise Constant Parameters

須子 統太* 松嶋 敏泰* 平澤 茂一*
Tota Suko Toshiyasu Matsushima Shigeichi Hirasawa

Abstract— Bayes code is one of the wellknowm universal codes. Bayes code has Bayes optimality in point of minimization of mean redundancy. Sources with piecewise constant parameters are one of nonstationary sources. These sources have abruptly changing parameters. Conventionally, nonpredictive Bayes coding for these sources is shown. In this paper, we propose a predictive Bayes coding algorithm. The computational complexity of the algorithm increases quadratically. Then, we increase the efficiency of the algorithm.

Keywords— source coding, universal coding, Bayes code, nonstationary source.

1 はじめに

情報源の確率構造について完全な情報が与えられていない場合の符号化法，すなわちユニバーサル情報源符号化法については従来から様々な研究が行われている．情報源の分布のクラスのみ仮定し，そのパラメータについては未知である場合を扱うユニバーサル符号のなかで，ベイズ符号は冗長度をベイズ基準のもとで最小にする符号である．[1]

また近年，非定常な情報源における情報源符号化の研究が行われている．その一つに，情報源が未知のパラメータに従い発生し，そのパラメータがある時点で突然変化する情報源（以下，区分定常情報源と呼ぶ）についての研究がある．[3][4][7][8]

この区分定常情報源に対し，Willems[4] が提案した符号化法は非適応的な場合のベイズ符号の一種であると考えられる．区分定常情報源におけるベイズ符号化法は計算量が系列長の指数オーダとなる問題点があり，[4] では効率的な符号化法を提案している．

本研究では区分定常情報源における適応的なベイズ符号化法について扱う．この時，非適応的な場合と同様，計算量が系列長の指数オーダとなる問題点がある．そこで，本研究では非適応的な符号化において効率的なベイズ符号化法を提案する．

* 〒 169-8555 東京都新宿区大久保 3-4-1 早稲田大学理工学部経営システム工学科. Dept. of Industrial & Management Systems Engineering, School of Science and Engineering, Ookubo 3-4-1, Shinjyukuku, Tokyo, 169-8555 Japan. E-mail: suko@matsu.mgmt.waseda.ac.jp

2 区分定常情報源

情報源アルファベットを x_i，長さ N の情報源系列を $x_1^N : x_1, x_2, \cdots, x_N$，また i 番目から j 番目までの部分系列を $x_i^j : x_i, x_{i+1}, \cdots, x_{j-1}, x_j$ とする．区分定常情報源はパラメータの変化パターン m と，その変化パターンのもとでのパラメータの集合 Θ^m によって定まる確率分布で表される．

今，c 回目にパラメータの変化が起きた時点を t_c とする．但し，$t_0 = 1$ である．また，パラメータの変化パターン m は t_c を用いて $m = \{t_1, t_2, \cdots, t_{C(m)}\} \in M$ で表す．さらに，$\Theta^m = \{\theta^m_{t_0}, \theta^m_{t_1}, \cdots, \theta^m_{t_{C(m)}}\}$ とする．但し，θ_{t_c} は定常分布のパラメータとする．このとき，区分定常情報源における系列 x_1^N の発生確率を，

$$P(x_1^N|\Theta^m, m) = \prod_{c=0}^{C(m)} P(x_{t_c}^{t_{c+1}-1}|\theta^m_{t_c}), \qquad (1)$$

と定義する．但し，$t_{C(m)+1} = N+1$ である．

また情報源系列 X_1^N のエントロピーは，

$$H(X_1^N|\Theta^m, m) = -\sum_{x_1^N} P(x_1^N|\Theta^m, m) \log P(x_1^N|\Theta^m, m), \qquad (2)$$

と定義する．

3 ベイズ符号

情報源系列の符号化確率を仮定すれば算術符号を用いることで符号化が可能になる．そのためユニバーサル情報源符号化の問題は系列の符号化確率を決定する問題に帰着される．ベイズ符号ではベイズ基準のもとで冗長度を最小にする符号化確率を決定する．

以降，パラメータの変化パターン m，変化回数 C(m) およびパラメータ Θ^m を未知とし，m，Θ^m の事前分布 $P(m)$，$P(\Theta^m|m)$ は既知であるとする．

区分定常情報源における，非適応的および適応的符号化に対しベイズ符号の符号化確率を以下で示す．

3.1 非適応的ベイズ符号 [1]

系列 x_1^N の符号化確率を $AP(x_1^N)$ とする．今，損失関数を以下で定義する．

$$V(\Theta^m, m, AP, x_1^N) = logP(x_1^N|\Theta^m, m) - logAP(x_1^N). \qquad (3)$$

この時，冗長度，つまり平均符号長とエントロピーの差をリスク関数として以下で定義する．

$$\begin{aligned} & R(\Theta^m, m, AP) \\ &= \sum_{x_1^N} P(x_1^N|\Theta^m, m) log \frac{P(x_1^N|\Theta^m, m)}{AP(x_1^N)} \\ &= L(X_1^N|\Theta^m, m, AP) - H(X_1^N|\Theta^m, m). \quad (4) \end{aligned}$$

本来であればこの冗長度を最小にしたいのだが，全てのパラメータについて最小化することはできない．そこでベイズ基準では，このリスク関数に対し事前分布で期待値をとったベイズリスクを最小化する．ベイズリスクは以下で定義する．

$$\begin{aligned} & BR(P(\Theta^m|m), P(m), AP) \\ &= \sum_m \int_{\Theta^m} R(\Theta^m, m, AP) P(\Theta^m|m) d\Theta^m P(m). \quad (5) \end{aligned}$$

このベイズリスクを最小化する AP，すなわち非適応的ベイズ符号化確率は以下で与えられる．

$$\begin{aligned} & AP^*(x_1^N) \\ &= \sum_m \int_{\Theta^m} P(x_1^N|\Theta^m, m) P(\Theta^m|m) d\Theta^m P(m). \quad (6) \end{aligned}$$

3.2 適応的ベイズ符号 [1]

系列 x_1^{n-1} が与えられたもとでの，次の1シンボル x_n の符号化確率を $AP(x_n|x_1^{n-1})$ とする．非適応的の場合と同様に適応的ベイズ符号化確率は以下で与えられる．

$$\begin{aligned} & AP^*(x_n|x_1^{n-1}) \\ &= \sum_m \int_{\Theta^m} P(x_n|x_1^{n-1}, \Theta^m, m) P(\Theta^m|m, x_1^{n-1}) d\Theta^m \\ &\qquad \times P(m|x_1^{n-1}). \quad (7) \end{aligned}$$

系列 x_1^N に対して非適応的ベイズ符号の符号長と適応的ベイズ符号の符号長は等しくなることが知られている．[1]

$$-\log AP^*(x_1^N) = -\sum_{n=1}^{N} \log AP^*(x_n|x_1^{n-1}). \quad (8)$$

3.3 計算量

今，パラメータの事前分布に自然共役な事前分布を仮定する．この時，(6)，(7) 式における積分計算はデータ数の線形オーダの計算量で計算することができる．他方，m での期待値計算にかかる計算量は $O(|M|)$ となり，考えうるパラメータの変化パターンが多い場合この部分の計算量が符号化確率計算の主要項となる．非適応的な場合 $|M| = 2^{N-1}$，適応的な場合 $|M| = 2^{n-1}$ となり，系列が長くなるにつれて指数的に計算量が多くなることが分る．そのため，系列が長くなればなるほど符号化確率を計算するのが非常に困難になる．

Willems[4] が提案した符号化法は，ある事前分布を仮定したもとでの非適応的なベイズ符号とみなすことができる．[4] では CTW 法を応用することによって，効率的に符号化確率を計算するアルゴリズムを示している．そこで，本研究では適応的な場合について効率的にベイズ符号化確率を計算するアルゴリズムを示す．

4 適応的符号化法の効率化

4.1 m の事前分布

n 時点目での変化パターンの集合を M_n，その要素を $m_n \in M_n$ とする．n 時点目の符号化確率を計算した後，m_n の事後確率は，

$$P(m_n|x_1^n) = \frac{P(m_n|x_1^{n-1}) P^m(x_n|m_n, x_1^{n-1})}{AP^*(x_n|x_1^{n-1})}, \quad (9)$$

によって計算される．但し，

$$\begin{aligned} P^m(x_n|m_n, x_1^{n-1}) = & \int_{\Theta^{m_n}} P(x_n|x_1^{n-1}, \Theta^{m_n}, m_n) \\ & \times P(\Theta^{m_n}|m_n, x_1^{n-1}) d\Theta^{m_n}, \quad (10) \end{aligned}$$

である．

$n+1$ 時点目においてこの変化パターン m_n は，n 時点目までは m_n と同じ変化で，その後 $n+1$ 時点目で変化が起きる m_{n+1}^a と変化が起きない m_{n+1}^b の2つの変化パターンに分けられる．そこで各時点でパラメータの変化が起きる確率を π とする．但し，π は時点によらず一定で，既知であるとする．この時，$P(m_{n+1}^a|x_1^n)$ および $P(m_{n+1}^b|x_1^n)$ は m_n の事後確率を用いて以下で計算することができる．

$$\begin{aligned} P(m_{n+1}^a|x_1^n) &= \pi P(m_n|x_1^n). \quad (11) \\ P(m_{n+1}^b|x_1^n) &= (1-\pi) P(m_n|x_1^n). \quad (12) \end{aligned}$$

π は変化パターン m の事前分布を表現するパラメータとして捉えることができる．もし，変化パターンについて事前情報が全く無い場合，$\pi = 0.5$ とすることで全ての変化パターンに対して等確率な事前分布をふることになる．

4.2 効率的アルゴリズム

変化パターンが与えられたもとでのある区間におけるパラメータ $\theta_{t_c}^m$ を考える．今，$\theta_{t_c}^m$ の事前分布 $P(\theta_{t_c}^m)$ が，変化パターンおよび区間によらず全て等しいと仮定する．次に，二種類の変化パターン m' と m'' を考える．但し，$m' \neq m''$ とする．m'，m'' がそれぞれ，

$$m' = \{\cdots, i, j, \cdots\}, \quad (13)$$

$$m'' = \{\cdots, i, j, \cdots\}, \quad (14)$$

である．つまり，変化が起きていない同一の区間が存在するとする．さらに，変化パターン m が与えられたもとでの符号化確率を，

$$AP(x_n|x_1^{n-1},m) = \int_{\Theta^m} P(x_n|x_1^n,\Theta^m,m)P(\Theta^m|m)d\Theta^m, \quad (15)$$

とする．この時，$i \leq n \leq j-1$ の区間において，

$$AP(x_n|x_1^{n-1},m') = AP(x_n|x_1^{n-1},m''), \quad (16)$$

が成立する．

また，時点1から n までの間で，最後にパラメータが変化した時点を τ_n とする．但し，$\tau_n = 1,2,\cdots,n$ である．この時，τ_n の事前分布 $P(\tau_n)$ を以下で定義する．

$$P(\tau_n) = \sum_{\{m:t_{c(m)}=\tau_n\}} P(m). \quad (17)$$

この時，効率的な適応的ベイズ符号化アルゴリズムを以下で示す．

【効率的アルゴリズム】

step-1.

x_n を読み込む．

step-2.

符号化確率を次式で計算する．

$$AP_{algo}(x_n|x^{n-1}) = \sum_{\tau_n=1}^{n} P^{\tau}(x_n|\tau_n,x_1^{n-1})P(\tau_n|x_1^{n-1}). \quad (18)$$

但し，

$$P^{\tau}(x_n|\tau_n,x_1^{n-1}) = \int_{\theta_{\tau_n}} P(x_n|x_1^{n-1},\theta_{\tau_n})P(\theta_{\tau_n}|x_1^{n-1})d\theta_{\tau_n}. \quad (19)$$

step-3.

$\tau_{n+1} = 1,2,\cdots,n+1$ について $P(\tau_{n+1}|x_1^n)$ を次式で計算する．

$\tau_{n+1} = 1,2,\cdots,n$ の時，

$$P(\tau_{n+1}|x_1^n) = (1-\pi)P(\tau_n|x_1^n) = (1-\pi)\frac{P^{\tau}(x_n|\tau_n,x_1^{n-1})P(\tau_n|x_1^{n-1})}{AP_{algo}(x_n|x_1^{n-1})}. \quad (20)$$

$\tau_{n+1} = n+1$ の時，

$$P(\tau_{n+1}|x_1^n) = \pi. \quad (21)$$

step-4.

step-1 へ戻る．

例 $P(x_{t_c}^{t_{c+1}-1}|\theta_{t_c}^m)$ が i.i.d の場合 (19) 式は，

$$P^{\tau}(x_n|\tau_n,x_1^{n-1}) = \frac{\nu(x_n|x_{\tau_n}^{n-1}) + \beta(x_t|\tau_n)}{\sum_{a=0}^{l-1}(\nu(a|x_{\tau_n}^{n-1}) + \beta(a|\tau_n))}, \quad (22)$$

によって計算される．但し，l は記号の数とし，$\nu(a|x_{\tau_n}^{n-1})$ は時点 τ_n から $n-1$ の区間において記号 a が出現した回数とする．また，$\beta(a|\tau_n)$ はベータ分布のパラメータとする．

効率的アルゴリズムにより計算された符号化確率について以下の定理が成り立つ．

定理 効率的アルゴリズムにより計算された符号化確率はベイズ基準のものとで最適な符号化確率と等しい．つまり，

$$AP_{algo}(x_n|x_1^{n-1}) = AP^*(x_n|x_1^{n-1}), \quad (23)$$

が成立する．

（証明は付録参照）

適応的ベイズ符号化の計算量は $O(2^{n-1})$ であったのに対し，効率的アルゴリズムを用いることで計算量は $O(n)$ となる．定理からも，効率的アルゴリズムがベイズ最適性を保持したまま効率的に符号化確率を計算していることが分る．

5 まとめ

本研究では区分定常情報源におけるベイズ基準のもとで最適な符号化法を示した．これには系列長が長くなるにつれて，指数的に計算量が増えてしまうという問題点があった．そこで適応的な符号化に対して，ベイズ最適性を保持したまま効率的に符号化確率を計算する方法を示した．

ベイズ符号化確率は全ての変化パターンにおける符号化確率を事後確率で重みづけることによって求められるため，全ての変化パターン数の和計算が必要であった．しかしながら，事前分布の設定によっては，符号化する記号が過去のどの時点までと同じパラメータにしたがって発生しているかを考え，その全てのバリエーションを重みづけることでベイズ符号化確率が計算可能になる．そのため，系列長だけの和計算で符号化確率が計算できる．

本研究ではパラメータが変化する確率 π を既知として扱った．しかし，本来この確率は未知であることの方が多い．そのため，π を未知とした場合の符号化法が必要となる．π を未知とした場合のベイズ符号化確率は，

$$AP^*(x_n|x_1^{n-1})$$
$$= \int_\pi \sum_m \int_{\Theta^m} P(x_n|x_1^{n-1}, \Theta^m, m) P(\Theta^m|m, x_1^{n-1}) d\Theta^m$$
$$\times P(m|\pi, x_1^{n-1}) P(\pi|x_1^{n-1}) d\pi, \qquad (24)$$

となる．この時，π の事前分布 $P(\pi)$ をどのように設定すればよいのか，その際に効率的なアルゴリズムが構成可能であるのか，などについては今後の課題としたい．また，区分定常情報源におけるベイズ符号の冗長度に対する理論的な評価についても今後明らかにするべき課題である．

謝辞

本研究を行うにあたり，数多くの御助言，御支援を賜りました，浮田善文氏，並びに松嶋研究室，平澤研究室の各氏に感謝致します．なお，本研究の一部は日本学術振興会科学研究費基盤 (C) 一般 (No.15560338) の援助による．

文献

[1] T. Matsushima, H. Inazumi and S. Hirasawa, "A Class of Distortionless Codes Designed by Bayes Decision Theory" IEEE Trans. Inf. Theory, vol.37, No.5, page 1288, 1991.

[2] T. Matsushima, and S. Hirasawa, "A bayes coding using context tree." In Proc. Int. Symp. on. Inf. Theory, page 386, 1994.

[3] N. Merhav, "On the minimum description length principle for sources with piecewise constant parameters." *IEEE Trans. Inf. Theory*, vol.39, No.6, page 1962, 1993.

[4] Frans M. J. Willems, "Coding for Binary Independent Piecewise-Identically-Distributed Source." IEEE Trans. Inf. Theory, vol.42, No.6, page 2210, 1996.

[5] Frans M. J. Willems, Y. M. Shtarkov and T. J. Tjalkens, "The Context-Tree Weighting Method: Basic Propreties," *IEEE Trans. Inf. Theory*, vol.41, No.3, page 653, 1995.

[6] Frans M. J. Willems, Y. M. Shtarkov and T. J. Tjalkens, "Context Weighting for General Finite-Context Sources." *IEEE Trans. Inf. Theory*, vol.42, No.5, page 1514, 1996.

[7] G. I. Shamir and N. Merhav, "Low-Complexity Sequential Lossless Coding for Piecewise-Stationary Memoryless Sources." *IEEE Trans. Inf. Theory*, vol.45, No.5, page 1498, 1999.

[8] G. I. Shamir and D. J. Costello, Jr., "Asymptotically Oprimal Low-Complexity Sequential Lossless Coding for Piecewise-Stationary Memoryless Sources-Part :The Regular Case." *IEEE Trans. Inf. Theory*, vol.46, No.7, page 2444, 2000.

[9] M. Gotoh and S. Hirasawa, "Statistical model selection based on Bayes decision theory and its application to change detection problem." *Int. J. Production Economics* 60-61, page 629, 2000.

付録

付録では定理の証明を行う．

(証明)

まず，(9)(17)(20) 式より，

$$P(\tau_n|x_1^{n-1}) = \sum_{\{m:t_{c(m)}=\tau_n\}} P(m|x_1^{n-1}), \qquad (25)$$

が成立する．また，(1) 式の定義より，

$$P^m(x_n|x_1^{n-1}, m) = P^\tau(x_n|x_1^{n-1}, \tau_n). \qquad (26)$$

よって，(25)(26) 式より，

$$\begin{aligned} AP^*(x_n|x_1^{n-1}) &= \sum_m P^m(x_n|x_1^{n-1}, m) P(m|x_1^{n-1}) \\ &= \sum_m P^\tau(x_n|x_1^{n-1}, \tau_n) P(m|x_1^{n-1}) \\ &= \sum_{\tau_n=1}^{n} \{P^\tau(x_n|x_1^{n-1}, \tau_n) \sum_{\{m:t_{c(m)}=\tau_n\}} P(m|x_1^{n-1})\} \\ &= \sum_{\tau_n=1}^{n} P^\tau(x_n|x_1^{n-1}, \tau_n) P(\tau_n|x_1^{n-1}) \\ &= AP_{algo}(x_n|x_1^{n-1}). \end{aligned} \qquad (27)$$

また，(11)(12)(17) 式より，(20)(21) 式の計算により $P(m_{n+1}|x_1^n)$ が計算されていることは明らか．

以上より，効率的アルゴリズムによる符号化確率がベイズ符号化確率と等しくなっていることが示せた．

□

The 26th Symposium on Information Theory
and Its Applications (SITA2003)
Higashiura, Hyogo, Japan, Dec. 15–18, 2003

衝突困難ハッシュ関数の安全性について
On the Security of Collision Free Hash Function Family

渋谷知成* 齋藤友彦* 松嶋敏泰*
Tomonari SHIBUYA Tomohiko SAITO Toshiyasu MATSUSHIMA

Abstract— Hash functions play very important role in the field of cryptography and its applications. Especially, hash functions which have collision free property is needed in cryptographic protocols like electronic cash protocols. In this paper, we show the proof of security for combination of hash functions which are elements of collision free hash function family. Moreover, we present a new method for constructing accumulator[2] with the proof of security.

Keywords— Collision free hash function family, combination of hash function, accumulator.

1 はじめに

暗号やその応用分野において，ハッシュ関数は非常に大きな役割を持つ．特に，安全性について示すためには，一方向性や衝突困難性を持ったハッシュ関数が用いられる．まず，一方向性ハッシュ関数を用いることにより，適応的選択暗号文攻撃に対しても安全な暗号および署名方式が構成可能であることが示されている [4]．また，暗号プロトコルでは，その安全性の証明に衝突困難ハッシュ関数の存在が仮定されることが多い．

従来，一方向性ハッシュ関数については，その合成関数も一方向性ハッシュ関数となることが示されている [3]．しかし，合成関数以外の組み合わせについては考えられていない．また，衝突困難ハッシュ関数については，その組み合わせの安全性がまったく考えられていない．

一方，アキュムレータ [2],[5] を用いた認証方式が提案されている．アキュムレータは，衝突困難性を持つ関数により構成される．応用として，電子現金プロトコルも提案されている [7]．アキュムレータは，衝突困難ハッシュ関数の木構造の組み合わせで表すことができるが，その安全性が明示されていないものもある．

そこで，本稿においては，衝突困難ハッシュ関数を木構造に組み合わせた場合の安全性を示す．さらに，安全性の証明が可能であるアキュムレータの一構成法を示す．

2 問題設定

アルゴリズム A について，入力が 1 つの場合は $A(\cdot)$，入力が 2 つの場合は $A(\cdot,\cdot)$，$\cdots$ と記す．A を確率的アルゴリズムとしたとき，任意の ι に対して $A(\iota)$ は，A が入力 ι のときに出力に割り当てる確率空間を示す．ただし，本稿で用いる確率空間は確率的アルゴリズムによって構成されたもののみであり，可算であるとする．ある確率空間 P に対し，$x \leftarrow P$ により，P からランダム選ばれた x であることを示す．入力サイズの多項式時間アルゴリズムの集合を $\mathcal{PA}$ と記し，期待多項式時間である確率的アルゴリズムの集合を $\mathcal{EA}$ と記す．確率的アルゴリズムが期待多項式時間であるとは，アルゴリズムの実行時間の（参照する確率空間上の）期待値が入力サイズの多項式時間であることを示す．$\mathcal{PA} \subset \mathcal{EA}$ であり，$\mathcal{EA}$ に属するアルゴリズムを効率的なアルゴリズムと呼ぶ．ある集合 $\mathbf{S}$ に対し，$x \in_R \mathbf{S}$ により，$\mathbf{S}$ から一様にランダムに選ばれた x であることを示す．また，整数上の多項式の集合を $\mathcal{POLY}$ とする．任意の整数 $n(>0)$ に対し，集合 $\{0,1\}^n$ を群とする演算を $\circ$ で記す．

* 〒 169-8555 新宿区大久保 3-4-1, 早稲田大学理工学部経営システム工学科. Dept. of Industrial and Management Systems Engineering, Waseda University, 3-4-1 Okubo, Shinjuku, Tokyo, 169-8555 Japan. E-mail: shibuya@matsu.mgmt.waseda.ac.jp

3 ハッシュ関数族

定義 1 $\{n_{\mathsf{in},i}\}, \{n_{\mathsf{out},i}\}$ を，$\forall i \ n_{\mathsf{out},i} \le n_{\mathsf{in},i}$ かつ $\exists p \in \mathcal{POLY} \ \ p(n_{\mathsf{out},i}) \ge n_{\mathsf{in},i}$ を満たす増加列とする．このとき，セキュリティパラメータ k に対して以下を満足する関数の集合 $\mathbf{H}_k$ をハッシュ関数族とする．

- $\mathbf{H}_k \triangleq \{\ h \mid h : \{0,1\}^{n_{\mathsf{in},k}} \to \{0,1\}^{n_{\mathsf{out},k}}\}$ (1)
- k の多項式制約の関数 q に対して，$r_k \triangleq q(k)$ としたとき，$\#\mathbf{H}_k \triangleq r_k$ である．
- 入力 1^k に対し，$h \in \mathbf{H}_k$ の記述を一様にランダムに出力するアルゴリズム $G \in \mathcal{EA}$ が存在する．
- 任意の $h \in \mathbf{H}_k$，$x \in \{0,1\}^{n_{\mathsf{in},k}}$ に対して，$h(x)$ を計算するアルゴリズム $C \in \mathcal{PA}$ が存在する．

また，$\mathbf{U} \triangleq \bigcup_k \mathbf{H}_k$ とする．

□

定義 2 任意の $h \in_R \mathbf{U}$ に対し，以下を満足するとき $\mathbf{U}$ を汎用一方向性ハッシュ関数族という．

- 一方向性

$$\forall A \in \mathcal{EA} \, \forall p \in \mathcal{POLY} \, \exists K \, \forall k > K$$
$$\Pr\big[\, h(x) = h(y), x \neq y \mid h \leftarrow G(1^k); x \in_R \{0,1\}^{n_{\mathsf{in},k}}; y \leftarrow A(h, h(x)) \,\big] < \frac{1}{p(k)} \quad (2)$$

また，汎用一方向性ハッシュ関数族に属する関数を汎用一方向性ハッシュ関数という．

□

定義 3 任意の $h \in_R \mathbf{U}$ に対し，以下を満足するとき $\mathbf{U}$ を衝突困難ハッシュ関数族という．

- 衝突困難性

$$\forall A \in \mathcal{EA} \, \forall p \in \mathcal{POLY} \, \exists K \, \forall k > K$$
$$\Pr\big[\, h(x) = h(y), x \neq y \mid h \leftarrow G(1^k); (x,y) \leftarrow A(h) \,\big] < \frac{1}{p(k)} \quad (3)$$

また，衝突困難ハッシュ関数族に属する関数を衝突困難ハッシュ関数という．

□

衝突困難性を持つとき，関数は安全であると定義する．以下，汎用一方向性ハッシュ関数族を $\mathbb{F}_{\mathsf{OW}}$，衝突困難ハッシュ関数族を $\mathbb{F}_{\mathsf{CF}}$ と記す．一方向性置換が存在すれば，汎用一方向性ハッシュ関数族が存在することが示されている [3]．また，Claw Free Permutation Family が存在すれば，衝突困難ハッシュ関数族が存在することが示されている [1]．Claw Free Permutation Family については，素因数分解や離散対数問題の困難さを仮定することにより構成できることが示されている．さらには，衝突困難ハッシュ関数族の存在には，Claw Free Permutation Family のサブクラスである Strong Claw Free Permutation Family を仮定することが適当であることが示唆されている [6]．

定理 1 [3] $h_1, \cdots, h_l \in_R \mathbb{F}_{\mathsf{OW}}$ に対し，汎用一方向性ハッシュ関数の l-composition $h \triangleq h_l \circ \cdots \circ h_1$ は汎用一方向性ハッシュ関数である．

□

汎用一方向性ハッシュ関数の l-composition は以下の図 1 で表される．

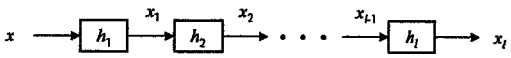


図 1: l-composition

4 衝突困難ハッシュ関数の組合せ

ここでは，衝突困難ハッシュ関数を組み合わせた場合の安全性について考える．汎用一方向性ハッシュ関数は，その合成関数が汎用一方向性ハッシュ関数族に閉じていることが示されている（定理 1）．本稿においては，以下のような組合せを考える．

定義 4 l，d を $\exists\, q \in \mathcal{POLY}$ $(l \le q(k)) \wedge (d \le q(k))$ を満たす定数とする．このとき，$h_{i,j} \in_R \mathbf{H}_{k_{i,j}}$ に対し，以下を満たすような関数 h をハッシュ関数の Tree Combination（以下，TC とする）と呼ぶ．

$$h(x_1, x_2, \cdots, x_l) = x_{d,1} \tag{4}$$

$$\begin{cases} x_{i,j} = h_{i,j}\left(x_{i-1,s_i} \circ \cdots \circ x_{i-1,t_i}\right) \\ x_{1,j} = h_{1,j}(x_j) \quad (j = 1, 2, \cdots, l) \end{cases} \tag{5}$$

ただし，

$$\begin{cases} t_i - s_i \ge 1 \\ s_i = t_{i-1} \\ s_0 = 1\ ,\ t_{\#\{h_{i-1,\cdot}\}} = \#\{h_{i-1,\cdot}\} \end{cases} \tag{6}$$

とする．また，任意の i, j に対し，$h_{i,j}$ が衝突困難ハッシュ関数である TC を衝突困難ハッシュ関数の TC と呼ぶ．

□

TC は，図 2 のように深さ d，葉の数 l の木で表すことができる．また，完全木のみではなく，図 3 のような不均一な木も TC である．

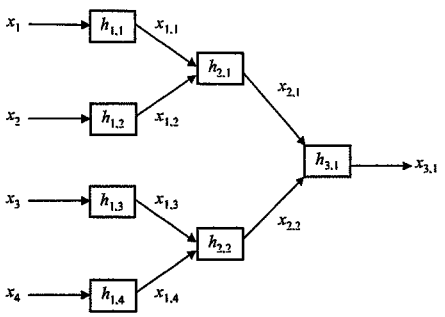


図 2: Tree Combination（完全木）

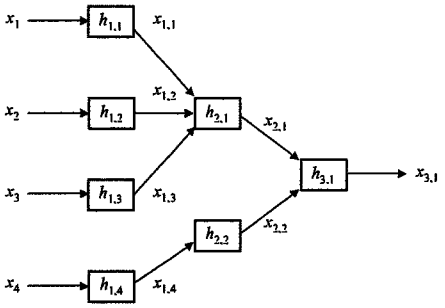


図 3: Tree Combination

定理 2

衝突困難ハッシュ関数の TC h は衝突困難ハッシュ関数である．

□

定理 2 を証明するためには，以下の補題 1 を示せばよい．

定義 5 l を $\exists\, q \in \mathcal{POLY}$ $(l \le q(k))$ を満たす定数とする．このとき，$h_{i,j} \in_R \mathbf{H}_{k_{i,j}}$ に対し，以下を満たす関数 h をハッシュ関数の Partial Tree Combination と呼ぶ．

$$h(x_1, x_2, \cdots, x_l) = x_{2,1} \tag{7}$$

$$\begin{cases} x_{2,1} = h_{2,1}\left(x_{1,1} \circ x_{1,2} \circ \cdots \circ x_{1,l}\right) \\ x_{1,j} = h_{1,j}(x_j) \quad (j = 1, 2, \cdots, l) \end{cases} \tag{8}$$

$$\forall\, j \quad n_{\mathsf{out},k_{1,j}} = n_{\mathsf{in},k_{2,1}} \tag{9}$$

また，任意の i, j に対し，$h_{i,j}$ が衝突困難ハッシュ関数である Partial Tree Combination を衝突困難ハッシュ関数の Partial Tree Combination と呼ぶ．

□

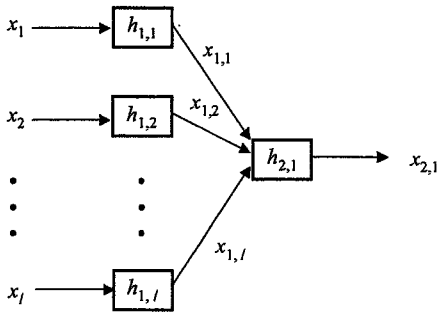


図 4: 衝突困難ハッシュ関数の Partial Tree Combination

補題 1 衝突困難ハッシュ関数の Partial Tree Combination h は衝突困難ハッシュ関数である．

□

[補題 1 の証明]

衝突困難ハッシュ関数の Partial Tree Combination h が衝突困難性を満たさないと仮定する．つまり，

$$\exists A \in \mathcal{EA}\, \exists p \in \mathcal{POLY}\, \forall K\, \exists k > K$$
$$\Pr\left[h(x) = h(y), x \neq y \mid h \leftarrow G\left(1^k\right); (x,y) \leftarrow A(h) \right] \geq \frac{1}{p(k)} \quad (10)$$
$$\left(x = (x_1, \cdots, x_l), y = (y_1, \cdots, y_l) \right)$$

とする．このとき，衝突 (x,y) (s.t. $h(x) = h(y), x \neq y$) が見つかったとすると，

(i) $\forall j \in \{1, \cdots, l\}$ $x_j \neq y_j$ の場合

- $\exists j$ $x_{1,j} = y_{1,j} \Rightarrow h_{1,j}$ で (x_j, y_j) が衝突
- $\forall j$ $x_{1,j} \neq y_{1,j}$
  $\Rightarrow h_{2,1}$ で $(x_{1,1} \circ \cdots \circ x_{1,l}, y_{1,1} \circ \cdots \circ y_{1,l})$ が衝突

(ii) $\exists j \in \{1, \cdots, l\}$ $x_j = y_j$ の場合

$$\mathbf{J} \triangleq \{\, j \mid x_j = y_j \,\}, \overline{\mathbf{J}} \triangleq \{\, 1, \cdots, l \,\} \setminus \mathbf{J} \quad (11)$$

としたとき，

- $\exists j \in \overline{\mathbf{J}}$ $x_{1,j} = y_{1,j} \Rightarrow h_{1,j}$ で (x_j, y_j) が衝突
- $\forall j \in \overline{\mathbf{J}}$ $x_{1,j} \neq y_{1,j}$
  $\Rightarrow h_{2,1}$ で $(x_{1,1} \circ \cdots \circ x_{1,l}, y_{1,1} \circ \cdots \circ y_{1,l})$ が衝突

これは，$h_{i,j}$ が衝突困難ハッシュ関数であることに矛盾する．よって，衝突困難ハッシュ関数の Partial Tree Combination h は衝突困難ハッシュ関数である．

□

[定理 2 の証明]

補題 1 より明らか．

□

5 アキュムレータ

ここでは，アキュムレータの定義およびそれを用いた所属認証を示す [2]．さらに，安全性証明が可能なアキュムレータの一構成法を示す．

5.1 問題設定

証明者 $P_j (j = 1, \cdots, l)$，信頼できる機関 TA，検証者 V を参加者とする．まず，P_j は値 $x_j \in \mathcal{X}$ を持ち，TA に対して x_j を送る．それに対して，TA はリスト $\mathbf{L} = \{x_j | j = 1, \cdots, l\}$ を構成する．その後，P_j は所有する x_j がリスト $\mathbf{L}$ に属することを示したいものとする．そのために，TA は P_j に対し，x_j がリスト $\mathbf{L}$ に所属することの証明書 w_j を送る．

定義 6 $\mathcal{X}$ を集合とする．また，$\mathcal{L} \triangleq \bigcup_l \mathcal{X}^l$ とする．このとき，アキュムレータは衝突困難性ハッシュ関数の組 (f, g, v) により構成される．

$$f : \mathcal{L} \to \mathcal{Z} \quad (12)$$
$$g : \mathcal{L} \times \mathcal{X} \to \mathcal{Z} \quad (13)$$
$$v : \mathcal{X} \times \mathcal{W} \times \mathcal{Z} \to \{True, False\} \quad (14)$$

□

(f, g, v) のセキュリティパラメータ k に対し，

- 完全性

$$\forall \mathbf{L} \in \mathcal{L}\, \forall x \in \mathbf{L} \quad v(x, g(\mathbf{L}, x), f(\mathbf{L})) = True \quad (15)$$

- 健全性

$$\forall A \in \mathcal{EA}\, \forall p \in \mathcal{POLY}\, \exists K\, \forall k > K$$
$$\Pr\left[v(x, w, f(\mathbf{L})) = True, x \notin \mathbf{L}, w \in \mathcal{W}, \mathbf{L} \in \mathcal{L} \mid (x, w, \mathbf{L}) \leftarrow A\left(1^k, f, g, v\right) \right] < \frac{1}{p(k)} \quad (16)$$

を満足するとき，アキュムレータは安全であると定義する．以上より，所属証明が構成される．

1. P_j $(j = 1, \cdots, l)$ は x_j を TA に送る．
2. TA はリスト $\mathbf{L} \triangleq \{\, x_j \mid j = 1, \cdots, l \}$ を構成する．
3. TA は，$z = f(\mathbf{L})$ および $w_j = g(\mathbf{L}, x_j)$ を計算する．
4. TA は，P_j $(j = 1, \cdots, l)$ に，(w_j, z) を送る．また，TA は，V に z を送る．
5. P_j は V に (x_j, w_j, z) を送る．
6. V は $v(x_j, w_j, z) = True$ であるか検証する．

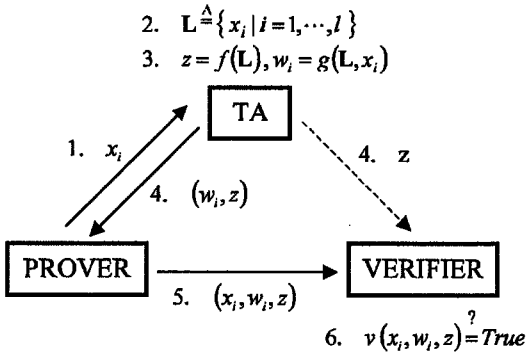


図 5: アキュムレータによる所属証明

5.2 提案法

衝突困難ハッシュ関数を組み合わせることにより，安全性の証明が可能なアキュムレータの構成法を示す．本稿においては，2 つのタイプのアキュムレータを示す．一方は準同型写像を必要としないタイプ [5] であり，もう一方は必要とするタイプ [2] である．前者を Type I，後者を Type II とする．

≪ Tybe I ≫

任意の i, j について $h_{k_{i,j}} \in \mathbf{H}_{k_{i,j}} (\subset \mathbb{F}_{\mathsf{CF}})$ とする．このとき，(f, g, v) を以下のように定める．

$$f(\mathbf{L}) = x_{d,1} \ (\triangleq z) \quad (17)$$
$$\begin{cases} x_{i,j} = h_{i,j}\left(x_{i-1,2j-1} \circ x_{i-1,2j}\right) \\ x_{1,j} = h_{1,j}\left(x_j\right) \end{cases} \quad (18)$$

$$g(\mathbf{L}, x_j) = (c_j, w_j) \tag{19}$$

$$\begin{cases} c_j = (c_{1,j}, \cdots, c_{d-1,j}),\ c_{i,j} \in \{0,1\} \\ w_j = (w_{1,j}, \cdots, w_{d-1,j}),\ w_{i,j} = x_{i,j_i} \end{cases} \tag{20}$$

$$\begin{cases} j_i &= (j_{i-1} + \overline{c_{i-1,j}})/2 \\ j_1 &= j \end{cases} \tag{21}$$

$$\overline{c_{i,j}} = \begin{cases} 0 & (\ c_{i,j} = 1\ ) \\ 1 & (\ c_{i,j} = 0\ ) \end{cases} \tag{22}$$

$$v(x_j, w_j, z) = \begin{cases} True & (z = z_{d,j}) \\ False & (otherwise) \end{cases} \tag{23}$$

$$\begin{cases} z_{i,j} &= h_{i,j_i}(z_{i-1,j} \circ w_{i-1,j}) \\ z_{1,j} &= h_{1,j_1}(x_j) \end{cases} \tag{24}$$

関数 f は図 2 のように完全二分木で表される．また，木の枝には 0,1 のラベルがつけられており，関数 g の出力 c_j は x_j の位置情報を示している（図 6）．関数 v は，x_j,w_j および c_j から，z の値が導ければ $True$ となる．

安全性については，まず，完全性を満たすことは明らかである．f は TC である．また，g の出力 $w_{1,j}$ は衝突困難ハッシュ関数により与えられ．$w_{i,j}(i = 2, \cdots, d-1)$ は，葉の数 2^{i-1}，深さ $\log_2(i-1)+1$ の衝突困難ハッシュ関数の TC により与えられる．つまり，f,g は衝突困難ハッシュ関数である．よって，健全性を満足することは明らかである．ゆえに，Type I は安全である．

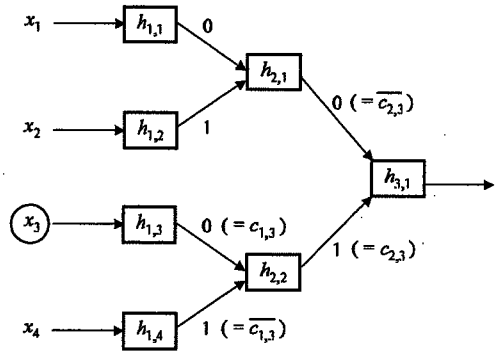


図 6: Type I

≪ Tybe II ≫

任意の i,j について $h_{k_{i,j}} \in \mathbf{H}_{k_{i,j}} (\subset \mathbb{F}_{\mathsf{CF}})$ とする．ただし，任意の $h_{k_{2,1}} \in \mathbf{H}_{k_{2,1}}$ は準同型であり，

$$h_{k_{2,1}}(a \circ b) = h_{k_{2,1}}(a) \bullet h_{k_{2,1}}(b) \tag{25}$$

を満たすものとする．ただし，$\circ$ および $\bullet$ は，関数 $h_{k_{2,1}}$ の定義域 $\{0,1\}^{n_{\mathrm{in},k_{2,1}}}$ および値域 $\{0,1\}^{n_{\mathrm{out},k_{2,1}}}$ 上の演算である．このとき，(f,g,v) を以下のように定める．

$$f(\mathbf{L}) = x_{2,1}\ (\triangleq z) \tag{26}$$

$$\begin{cases} x_{2,1} = h_{2,1}(x_{1,1} \circ \cdots \circ x_{1,l}) \\ x_{1,j} = h_{1,j}(x_j) \end{cases} \tag{27}$$

$$g(\mathbf{L}, x_j) = x_{2,j}\ \left(\triangleq w_j\right) \tag{28}$$

$$\begin{cases} x_{2,j} = h_{2,1}(x_{1,1} \circ \cdots \circ x_{1,j-1} \circ x_{1,j+1} \circ \cdots \circ x_{1,l}) \\ x_{1,j} = h_{1,j}(x_j) \end{cases} \tag{29}$$

$$v(x_j, w_j, z) = \begin{cases} True & (\ h_{2,1}(x_{1,j}) \bullet w_j = z\ ) \\ False & (\ otherwise\ ) \end{cases} \tag{30}$$

関数 f および g は図 4 な木で表される．ただし，f は葉の数が l であり，g は葉の数が $l-1$ である．

安全性については，まず，完全性を満たすことは明らかである．f は葉の数 l，深さ 2 の衝突困難ハッシュ関数の TC である．また，g は葉の数 $l-1$，深さ 2 の衝突困難ハッシュ関数の TC である．つまり，f,g は衝突困難ハッシュ関数である．よって，健全性を満足することは明らかである．ゆえに，Type II は安全である．

Type I のように完全二分木のハッシュ関数を組み合わせで表されるアキュムレータは提案されている [5]．ただし，その安全性については示されておらず，安全性を示すためには，ランダムオラクルモデルを仮定することが必要であると考えられる．Type I では，衝突困難ハッシュ関数族からハッシュ関数を選び，それらを組み合わせることにより，ランダムオラクルモデルを仮定することなくその安全性を示した．Type II のように，準同型写像を必要とするアキュムレータは，離散対数問題に帰着させその安全性が示しているものが提案されている [2]．Type II では，それとは異なる安全性証明を与えた．

6 まとめ

本稿では，衝突困難ハッシュ関数を組み合わせた場合の安全性を示した．さらに，衝突困難ハッシュ関数を組み合わせ，安全性証明が可能なアキュムレータの一構成法を示した．今後の課題としては，安全性のみでなく，効率性などについて考慮することが挙げられる．

本研究の一部は日本学術振興会科学研究費基盤研究 (C) 一般 (No.15560338) の助成による．

参考文献

[1] I.Damgård, "Collision Free Hash Functions and Public Key Signature Scheme," *Eurocrypt'87*, 1988

[2] J.Benaloh, M.de Mare,"One-way Accumulators: A Decentralized Alternative to Digital Signatures," *EUROCRYPT '93*, 1994

[3] M.Naor,M.Yung, "Universal One-Way Hash Functions and their Cryptographic Applications," *STOC89*, 1989

[4] S.Goldwasser, S.Micali, R.Rivest,"A Digital Signature Scheme Against Adaptive Chosen Message Attacks," *SIAM J.COMPUTE. Vol.17, No.2*, pp.281-308,1988

[5] R.Merkle,"Protocols for public key cryptosystems," *IEEE Proceedings of the 1980 Symposium on Security and Privacy*, 1980

[6] W.Ogata, K.Kurosawa,"On claw free families," *IEICE Trans. Vol.E77-A, No.1*, pp.72-80,1994

[7] T.Sander, A.Ta-Shma,"Blind, Auditable Membership Proofs," *Financial Cryptography '00*, 1999

The 26th Symposium on Information Theory
and Its Applications (SITA2003)
Higashiura, Hyogo, Japan, Dec. 15–18, 2003

ベイズ決定理論に基づく予測における近似手法について

A note on one of the approximation methods of the prediction based on Bayes decission theory

江口　公盛*　　須子　統太*　　松嶋　敏泰*
Kimimori Eguchi　　Tota Suko　　Toshiyasu Matsushima

Abstract— In various fields there is a problem which we predict after we got learning data. We mainly discuss one of the methods of the prediction problem.The method is based on Bayes decission theory. However,the theory assures the optimization of the problems,it is too complicated to calculate integral . Now, the approximation techniques are considerated by many persons. In this paper,we discuss one of the techniques and relationship of it to adaboost in the pattern recognition fields.

Keywords—Bayes decission theory , adaboost , Pattren Recognition.

1　はじめに

学習用のデータとしてn個のデータを与えておき，$n+1$個目のデータを予測する手法はさまざまな分野において行われている．その中のひとつとして，ベイズ決定理論に基づき予測する手法がある．この手法は，最適な予測を保証する一方で積分困難な積分計算を要することが知られている．そこで，積分計算を近似するさまざまな手法が研究されている．

そこで，その中で事後分布の分布形が単峰である場合において考案された手法 [2] に着目する．この手法は，ある程度の近似精度を有しながら大幅に計算量を削減している．

本研究では，事後分布が多峰型の分布である場合において積分近似手法を提案考察する．さらに，パターン認識の分野において複数の予測器を組み合わせた予測器を構成するadaboostとの関連性を考察する．adaboostは，識別問題に対して比較的簡単な予測器の構成法と組み合わせ法という利点だけでなく，実験的にその有用性が知られている．しかし，与えられたデータの判別誤差を最小にすることを目標にしたアルゴリズムであり，未知のデータに対する判別誤差を考慮に入れていない．

2　ベイズ決定理論

はじめに問題設定を行い，ベイズ決定理論に基づいて予測を行うことを考える．

* 〒 169-8555 東京都新宿区大久保 3-4-1 早稲田大学理工学部経営システム工学科. Dept. of Industrial & Management Systems Engineering, School of Science and Engineering, Ookubo 3-4-1, Shinjyukuku, Tokyo, 169-8555 Japan. E-mail: eguchi@matsu.mgmt.waseda.ac.jp

2.1　問題設定

入力ベクトルをx，出力をyとする．xおよびyは離散値，連続値どちらの場合でも以下の議論は成り立つが，表記上の簡単のため特に断らない限り以下ではxを連続値，yを離散値としてあつかう．ここで，既に入力x出力yが既知であるデータN個，

$$D=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}, \qquad (1)$$

と入力ベクトルx_{N+1}が与えれられたもとで，未知の出力$\hat{y}_{N+1}$を予測する問題を考える．

2.2　損失関数の定義

まず，学習用のデータDによって求めた関数を$\hat{f}(x)$とする．そして，$\hat{f}(x)$により求まるテスト用の入力x_{N+1}の出力の推定量$\hat{y}_{N+1}$と，真の出力y_{N+1}との損失関数$L(y_{N+1},\hat{y}_{N+1})$を定義する．損失は，以下のようにさまざまな形が考えられる．

0 − 1 損失：

$$L(y_{N+1},\hat{y}_{N+1})=\begin{cases}1 & \hat{y}_{N+1}\neq y_{N+1}\\ 0 & \hat{y}_{N+1}=y_{N+1}\end{cases} \qquad (2)$$

二乗誤差損失：

$$L(y_{N+1},\hat{y}_{N+1})=(y_{N+1},\hat{y}_{N+1})^2 \qquad (3)$$

対数損失：

$$L(y_{N+1},\hat{y}_{N+1})=log\frac{P(y_{N+1}|x_{N+1})}{\hat{P}(y_{N+1}|x_{N+1})} \qquad (4)$$

2.3　リスク関数の定義

損失関数$L(y_{N+1},\hat{y}_{N+1})$に対し，y_{N+1}の条件付分布$p(y_{N+1}|x_{N+1},\theta)$に関して平均をとった期待損失をリスク関数$R(\theta)$と呼び，以下のように定義する．

$$R(\theta)=\sum_{y_{N+1}} L(y_{N+1},\hat{y}_{N+1})p(y_{N+1}|x_{N+1},\theta). \qquad (5)$$

2.4 ベイズリスクの定義

一般に，未知パラメータ θ に関して一様にリスク関数 $R(\theta)$ を最小化する推定量は存在しない．従って，ベイズ決定理論では，θ の事後確率 $p(\theta|\{x_i,y_i\}_{i=1}^N)$ で平均化した平均的なリスクを最小化することを考える [10]．この平均のことをベイズリスク BR と呼び，以下のように定義する．

$$BR=\int R(\theta)p(\theta|\{x_i,y_i\}_{i=1}^N)d\theta. \tag{6}$$

2.5 ベイズ最適な予測

ベイズ最適な予測 $\hat{y}_{N+1}$ を求める式は損失関数の設定により異なる．例えば損失関数を (2) 式としたもとで，ベイズリスク (6) 式を最小化するベイズ最適な予測 $\hat{y}_{N+1}$ は，

$$\hat{y}_{N+1}=\sum_Y y_{N+1}p\left(y_{N+1}|x_{N+1},\{x_i,y_i\}_{i=1}^N\right) \tag{7}$$

となる．ここで $p(y_{N+1}|x_{N+1},\{x_i,y_i\}_{i=1}^N)$ は事後予測分布と呼ばれ，

$$p(y_{N+1}|x_{N+1},\{x_i,y_i\}_{i=1}^N)$$
$$=\int p(y_{N+1}|x_{N+1},\theta)p(\theta|\{x_i,y_i\}_{i=1}^N)d\theta \tag{8}$$

で与えられる．

結果的に (7) 式にあるように予測事後分布を計算してベイズ最適な予測を行っていることがわかる．

また，別の例として損失関数を (3)，(4) 式としたもとでベイズリスク (6) 式を最小化するベイズ最適な予測 $\hat{y}_{N+1}$ もまた，予測事後分布をもとにベイズ最適な予測計算していることになる．つまり，ベイズ決定理論に基づいた予測では，予測事後分布を計算することに帰着される．

しかし，実際には事後予測分布の計算は積分困難な計算になる．そこで次の節で積分困難な場合での事後予測分布の計算法について述べる．

3 事後予測分布の近似手法

積分困難な時な場合の事後予測分布の近似手法は多くの研究がなされている．

以下，パラメータの事後分布が $p(\theta|\{x_i,y_i\}_{i=1}^N)$ が単峰型の分布の場合について述べる．

3.1 近似計算手法 1

MAP 推定法

(8) 式における事後確率分布 $p(\theta|\{x_i,y_i\}_{i=1}^N)$ が単峰型の分布の場合事後確率 $p(\theta_i|\{x_i,y_i\}_{i=1}^N)$ を最大にする θ_i を 1 点求める手法である．この時事後予測分布は以下のように求める．

$$p(y_{N+1}|x_{N+1},\{x_i,y_i\}_{i=1}^N)=p(y_{N+1}|x_{N+1},\theta_i) \tag{9}$$

3.2 近似計算手法 2

野村 [2] による推定法

θ が離散，事後確率分布が単峰型分布のすそが広い場合，パラメータ空間すべてを足し合わせる計算量が膨大になる．

そこで，事後確率を最大とする θ とその近辺を代表点として足し合わせることにより計算量を抑えながらも近似精度の高い手法である．最大とする θ とその近辺のパラメータの集合を Θ_{max} とする．事後予測分布は以下のとおりである．

$$\pi(\theta')=\frac{p\left(\theta'|\{x_i,y_i\}_{i=1}^N\right)}{\sum_{\theta'\in\Theta_{max}}p\left(\theta'|\{x_i,y_i\}_{i=1}^N\right)} \tag{10}$$

$$p\left(y_{n+1}|x_{n+1},\{x_i,y_i\}_{i=1}^N\right)$$
$$\approx\sum_{\theta'\in\Theta_{max}}p\left(y_{n+1}|\theta',\{x_i,y_i\}_{i=1}^N\right)\pi(\theta') \tag{11}$$

以上 2 手法は事後分布が単峰型の場合である．しかし，現実的には，事後分布が多峰型の場合も多く考えられる．

そこで，本研究では，多峰型の場合についての事後予測分布の近似手法について提案し，考察を行う．

4 事後予測分布の導出

事後分布が多峰型に焦点を当て，事後予測分布近似手法を考察する．

一般に事後分布が多峰型の分布であるためには

(事後分布)∝(尤度)×(事前分布)

であることから，尤度と事前分布のどちらかが多峰型の分布でならなければならない．

しかし，事前分布が多峰型である場合与えられたデータ数 N が大きくなると単峰型の分布となるので例として尤度が多峰型であるので事後分布が多峰な分布になるような問題を考える．尤度が ($y\in 0,1$,x:連続値)

$$p\left(\{x_i,y_i\}_{i=1}^N|\theta\right)=\prod_{i=1}^n p(x_i,y_i|\theta) \tag{12}$$

$$=\prod_{i=1}^n\left\{1-\left(\frac{\theta^Tx_i}{a}\right)^2\right\}^{1-y_i}\left\{\left(\frac{\theta^Tx_i}{a}\right)^2\right\}^{y_i} \tag{13}$$

となる時である．(a : 定数) ここで，パラメータ θ の事前分布を一様分布と考えると事後分布 $p(\theta|D)$ は多峰型の分布となり，(8) 式の事後予測分布計算が積分困難となる．

このような問題設定において前節の MAP 推定は，1 つには多峰の中から事後確率が一番最大のものを見つけるのは困難である．また，2 つには予測精度もよいとはいえない．ということがいえると考えられる．

また野村 [2] による推定法も同様である.

そこで，2 点の欠点を補う手法を提案する.

一般に多峰型の関数の最大値を探索する手法はニュートン法を初め，さまざまな手法が考えられるが，多くの手法において局所解に陥る.

ここでは，逆に局所解に陥ることを利用し多くの局所解を求める．それらの局所解を同時に解くためあらかじめ m 個の初期値を用意しておく.

以下提案アルゴリズムを示す.

4.1 提案アルゴリズム

事後分布が多峰な分布で積分計算が困難なときにおける事後分布の導出法を示す.

1) 事後分布の山を局所解も含め算出する.

2) それらの事後確率を重み付けをする.

3) 条件付確率 $p(y|x,\theta)$ に重み付けることで事後予測分布を求める.

提案アルゴリズム

⟨1⟩

パラメータ θ の事後分布 $p\left(\theta|\{x_i,y_i\}_{i=1}^N\right)$ をもとめる.

$$p\left(\theta|\{x_i,y_i\}_{i=1}^N\right)=\frac{p\left(\{x_i,y_i\}_{i=1}^N|\theta\right)p(\theta)}{p\left(\{x_i,y_i\}_{i=1}^N\right)} \tag{14}$$

⟨2⟩

事後確率が高いパラメータ θ を探索アルゴリズムにより求める．また，初期値はパラメータの事前分布に比例する形で m 個の初期値を設定する．ここでは例としてニュートン法により求める.(初期値はパラメータの事前分布に一様分布を考え，等密度に m 個とる.)

$$\dot{I}(\theta) = \frac{\partial p\left(\theta|\{x_i,y_i\}_{i=1}^N\right)}{\partial\theta} \tag{15}$$

$$\ddot{I}(\theta) = \frac{\partial\dot{I}(\theta)}{\partial\theta} \tag{16}$$

$$\theta^{k+1} \leftarrow \theta^k-\frac{\dot{I}(\theta)}{\ddot{I}(\theta)} \tag{17}$$

局所解を $\theta_1,\ldots,\theta_m$ とする.

⟨3⟩

事後確率 $p\left(\theta_i|\{x_i,y_i\}_{i=1}^N\right)$ の重みを計算する

$i=1,\ldots,m$

$$\pi(\theta_i)=\frac{p\left(\theta_i|\{x_i,y_i\}_{i=1}^N\right)}{\sum_{i=1}^m p\left(\theta_i|\{x_i,y_i\}_{i=1}^N\right)} \tag{18}$$

⟨4⟩

事後予測分布を計算する

$$\begin{aligned} & p\left(y_{n+1}|x_{n+1},\{x_i,y_i\}_{i=1}^N\right) \\ = & \int p\left(y_{n+1}|\theta,\{x_i,y_i\}_{i=1}^N\right)p\left(\theta|\{x_i,y_i\}_{i=1}^N\right)d\theta \\ \approx & \sum_{i=1}^m p\left(y_{n+1}|\theta_i,\{x_i,y_i\}_{i=1}^N\right)\pi(\theta_i) \end{aligned} \tag{19}$$

5 考察

4 節の提案手法において $\theta_1,\cdots,\theta_i,\cdots,\theta_i,\cdots,\theta_m, i\neq j$ である時 $\theta_i=\theta_i$ などのように峰のパラメータが同じところに収束するところが多々存在する．この時は，事後分布がすその広い形であるためであると考えられる．したがって，積分計算を考える上で，θ_i の近辺を θ の候補として用意しておくほうがよいと考えられる.

ステップ 2 における解探索手法はさまざまな求め方が考えられる.

次に，パターン認識の分野において複数の予測器を予測器を構成する adaboost について関連性を考察する.

5.1 adaboost

5.1.1 adaboost の概要

adaboost の構成法 (アルゴリズムは付録) は，T 個の予測器を重み付けし，ひとつの予測器 $F_T(x)$ を求めるものである.

また,i 番目のデータが正解ならばデータの重みを小さくし，不正解ならばデータの重みを大きくする.(アルゴリズム中のステップ 3)

すなわち，学習しにくいデータ自身の重みを大きくしておき，常に一つ前の予測器が予測しづらい部分を重点的に予測するアルゴリズムであると考えられる.

しかし，このアルゴリズムは，経験損失を最小にすることを目標にしたアルゴリズムである．したがって，理論的に汎化誤差を小さくするためのアルゴリズムとはいえない.

5.1.2 ベイズ決定理論と adaboost

adaboost のアルゴリムにおける T 個の予測器を逐次的に求めそれらを重み付けすることにより精度のよい予測器を求めた.

その求め方は，初めの一つ目の予測器はデータの重みが一様なので，最尤法により求めていると考えられる．次に，データの重みを変えることで，他の高い山を求めていると考えられる.

一方で，4 節で求めた提案手法は，事後分布が多峰であるときその局所解を一括で m 個求め，それらの事後分布を重み付けすることによりベイズ決定理論に基づいた予測の近似を行った.

つまり，adaboost においては，求める T 個の予測器は逐次パラメータの事後確率が高いパラメータをまたは

その近辺を求めているのではないかと考えられる.

ここで, この見方と関連した研究として向内 [6] がある.

6 まとめ

本研究ではある事後分布が多峰型の分布におけるベイズ決定理論に基づいた予測に関して考察を行った. 取り扱ったモデルでは事後分布が多峰型の分布における積分計算が困難な場合に着目し, 事後確率が高くなるパラメータを局所解を含め求めることで予測事後分布を求める手法を提案した.

さらに adaboost のアルゴリズムの過程で求めることになる T 個の予測器は事後分布が多峰性な分布である場合, 局所解を含めた事後確率が高いパラメータの付近を重点的に求めようとしているという考察をした.

謝辞

本研究を行うにあたり, 数多くの御助言, 御支援を賜りました, 浮田善文氏, 野村 亮氏, 桑田修平氏, 並びに松嶋研究室の各氏に感謝致します. なお, 本研究の一部は日本学術振興会科学研究費基盤 (C) 一般 (No.15560338) の援助による.

文献

[1] 松嶋 敏泰, "帰納・演繹推論と予測 -決定理論による学習モデル-" , 第 1 回 情報論的学習理論ワークショップ IBIS'98,

[2] 野村 亮, 松嶋 敏泰, 平澤茂一 " メモリ量を低減した近似ベイズ符号化アルゴリズム" 電子情報通信学会論文誌 A Vol.J86-A No.1 pp.46-59 2003.

[3] 須子 統太, 野村 亮, 松嶋 敏泰, "拡張された階層モデルにおける予測アルゴリズムについて" The 25thSymposium on Imformation Theory and Its Application(SITA2002)pp755-758

[4] 桑田 修平, 吉田 隆弘, 松嶋 敏泰"ベイズ決定理論に基づくロバストなパターン認識手法に関する一考察" The 25thSymposium on Imformation Theory and Its Application(SITA2002)pp283-286

[5] R.E.Schapire,Y.Freund,P.Bartlett and W.S.Lee, "Boosting the Margin;A New Explanation for the Effectiveness of Voting Methods," The Annals of Statistics,vol.26,no.5 page 1651-1686, 1998.

[6] 向内隆文, 上田修功 "勾配法による混合分布の構成" IBIS,page 37, 2001.

[7] Y.Freund,R.E.Schapire, "Experiments with a new Boosting Algorithm" In Machine Learning ;Proceedings of the 13th Internatonal Conference page 148-156, 1996.

[8] Y.Freund,R.E.Schapire, "A Decision Theoretic Generalization of On-Line Learning and an Application to Boosting" Journal of Computer and System Sciences,vol.55,no.1 page 119-139, 1997

[9] J.Freidman,T.Haste and,R.Tibshirani " Additive Logistic Regression:a Statistical View of Boosting" The Annals of Statistics,vol.28,no.2 page 337-407, 2000.

[10] James O. Berger: *Statistical Decision Theory and Bayesian Analysis*, Springer, 1985.

付録

adaboost のアルゴリズムについて述べる.
adaboost のアルゴリズムを以下に示す.

input

$D = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$

$L(W, D_t)$;weak learn

start

$D_0(i) = \frac{1}{N}, i = 1, \ldots N.$

$F_0(x) = 0$

for $t = 0$ to T do

step

$1. w_t(i) = \frac{D_t(i)}{\sum_{i=1}^{N} D_t(i)}$

$2. f_{t+1} = L(W, D_t)$

$3. e_{t+1} = \sum_{i=1}^{N} w_t(i) |f_t(x_i) - y_i|$

$4. \beta_t = \frac{e_t}{1-e_t}$

$5. w_{t+1}(i) = \frac{w_t(i)}{\beta_t}$

output

$$F_T(x) = \begin{cases} 1 & if \quad \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t}(x)\right) f_t(x) \\ & \quad \geq \frac{1}{2} \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t}\right) \\ 0 & otherwise \end{cases}$$

$L(W, D_t)$;weak learn

t; 反復回数

$D_0(i)$; データの重み

$F_0(x)$; 初期の予測器

$F_T(x)$; 最終的に求める予測器

$f_t(x)$;t 番目の予測器

e_t; 誤り確率

w_t;t 番目の予測器の重み

社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE

Tailbiting 畳込み符号の復号アルゴリズムに関する一考察

岡野 洋平† 小泉 大城† 松嶋 敏泰‡

† 早稲田大学大学院 理工学研究科 〒169-8555 東京都新宿区大久保 3-4-1
‡ 早稲田大学 理工学部 経営システム工学科

E-mail: † {okano, dkoizumi, toshi}@matsu.mgmt.waseda.ac.jp

あらまし 事後確率最大(MAP)復号により誤り訂正を行う場合，符号の確率モデルのグラフ表現に複数のループが存在すると，確率伝播型の復号アルゴリズムの出力は近似事後確率となる．このような場合，アルゴリズムの収束性は保証されず，確率伝播スケジュールと事後確率の近似精度との関係などの復号特性の解析は非常に困難である．しかし，Tailbiting(TB)畳込み符号のように，グラフ表現が 1 ループのみからなる場合は，Lauritzen の三角化を用いることで，現実的な計算量で正しい事後周辺確率を求めることができる．本研究ではこのことを利用して，拘束長に対する系列長が短い場合に焦点を当て，TB 畳込み符号に並列型確率伝播アルゴリズムを適用して復号した際の反復計算ごとの近似事後確率と正しい事後確率との Kullback-Leibler(KL)情報量の挙動を調べることによって，並列型確率伝播アルゴリズムの近似精度による復号特性を考察する．

キーワード Tailbiting(TB)畳込み符号，事後確率最大(MAP)復号，並列型確率伝播アルゴリズム

A Study of Decoding Algorithm of Tailbiting Convolutional Codes

Youhei OKANO† Daiki KOIZUMI† and Toshiyasu MATSUSHIMA‡

† Graduate School of Science and Engineering, WASEDA University

‡ Department of Industrial and Management Systems Engineering, WASEDA University

3-4-1, Ohkubo, Shinjuku-ku, Tokyo, 169-8555, JAPAN
E-mail: † {okano, dkoizumi, toshi}@matsu.mgmt.waseda.ac.jp

Abstract On the error-correction by MAP decoding, if graphical model of coding-decoding system has loops, Belief-Propagation algorithm calculates approximate posterior probability. In this case, it is difficult to analyze decoding performance especially in terms of accuracy of approximate posterior probability as well as message passing schedule. If graphical model of a code has only one loop, however, the exact posterior probability can be calculated by using the method proposed by Weiss. As a result, we can quantify the difference between the approximate and exact posterior probability. By using this property, we assume both Tailbiting (TB) convolutional codes and parallel message passing decoding algorithm. We investigate by Kullback-Leibler (KL) divergence between approximate posterior probability calculated by message passing calculation and exact posterior probability at every iterative count and then consider characteristics of parallel message passing algorithm on TB convolutional codes.

Keyword Tailbiting (TB) convolutional codes, Maximum a posteriori(MAP) decoding, Parallel messege passing algorithm

1. はじめに

近年盛んに研究されている事後確率最大(MAP)復号で優れた性能を示す符号のひとつに Turbo 符号がある．Turbo 符号を復号する場合，その確率モデルのグラフ表現はループを持つため，復号の際に計算される近似事後確率の精度やグラフ上のメッセージパッシングスケジュールなどの解析は困難である．一方，Turbo 符号の要素符号として，重要な誤り訂正符号に畳込み符号がある．畳込み符号には現実的な計算量で正しい事後確率を計算し，復号ビット誤り率最小を達成する直列型メッセージパッシングを行う最適な復号アルゴリズム(BCJR アルゴリズム[1])が存在する．また，畳込み符号の初期状態と終了状態を任意の同じ状態に設定した TB 畳込み符号[2]の復号には近似アルゴリズム(TB-BCJR)が良く知られている．この符号の確率モデルをグラフ表現すると 1 つの大きなループからなり，別の近似アルゴリズムとして並列型確率伝播アルゴリズムが提案されていて，両者のアルゴリズムは同じ近似事後確率に収束する．違いは，メッセージの更新において，一方向にのみ確率分布を逐次的に更新する直列型と，全ての確率変数の確率分布を一度に更新する並列型の違いである．前者は，1 更新ごとに 1 つの確率分布を更新するだけなので，1 回の計算量は比較的少ないがメッセージが収束するまで更新を行う必要があり，後者は，1 回の計算量は多いがどの更新時点においても近似事後確率が得られるという特徴がある．

さらに，TB 畳込み符号のグラフ表現は 1 つの大きなループ からなるので，Weiss の方法[4]を利用して，計算量は膨大ではあるものの実時間内で正しい事後確率を求めることもできる．

本研究では，拘束長 3 の畳込み符号器を用い，拘束長に対する系列長が短いもの(10, 20, 40)に焦点を当て，並列型確率伝播アルゴリズムを用いて復号した場合の各シンボルのメッセージの更新ごとの近似事後確率と Weiss の方法を利用して求めた正しい事後確率との差(KL 情報量)の推移をみることで，並列型確率伝播アルゴリズムの近似精度に基づく復号性能について考察する．

2. 準備

2.1. TB 畳込み符号

畳込み符号とは任意の時点 t に，符号器状態が $S_t=\{0,1,\cdots,2^{\nu-1}\}$（$\nu$ は遅延素子数．今回は $\nu=2$ の場合を用いる．）の元である線形符号である．畳込み符号の符号化には，ｉ）定められた時点に符号器の状態を全零状態にもっていく方法(zero-tail)，ii）初期状態を任意として初期状態と終了状態を同じにする方法等がある．ii）の場合が TB 畳込み符号である．

2.2. 近似事後確率と正しい事後確率との KL 情報量

復号する際に求める事後確率は，全受信系列 Y_1^N が得られたもとで状態遷移のとる事後確率 $P(s_{t-1},s_t|Y_1^N)$ である．ここで、正しい事後確率を $P^{\text{正しい}}(s_{t-1},s_t|Y_1^N)$，復号アルゴリズムによる近似事後確率を $P^{\text{近似}}(s_{t-1},s_t|Y_1^N)$ とおくと，KL 情報量は，

$$P^{\text{正しい}}(s_{t-1},s_t|Y_1^N)\times\log\left(P^{\text{正しい}}(s_{t-1},s_t|Y_1^N)\Big/P^{\text{近似}}(s_{t-1},s_t|Y_1^N)\right). \quad (1)$$

で求めることができる．

実験では，各シンボルにおけるメッセージ更新ごとの KL 情報量の挙動をみる．

3. 復号アルゴリズム

3.1. TB-BCJR アルゴリズム[3]

BCJR アルゴリズムは，全受信系列 Y_1^N から情報記号 u_t を推定するために，(2)式で与えられる事後確率 $P(u_t|Y_1^N)$ を分配則により効率的に求めるアルゴリズムである．

$$\begin{aligned}P(u_t|Y_1^N)&=P(s_{t-1},s_t,Y_1^N)\Big/P(Y_1^N)\\&=qP(s_{t-1},Y_1^{t-1})\cdot P(s_{t-1},s_t,Y_t)\cdot P(Y_{t+1}^N|s_t)\\&=q\alpha_{t-1}(s_{t-1})\cdot\gamma_t(s_{t-1},s_t)\cdot\beta_t(s_t).\end{aligned} \quad (2)$$

TB-BCJR アルゴリズムでは，BCJR アルゴリズムと異なり，初期状態と終了状態の分布を任意に定め，前向き計算・後向き計算ともに終了時点で止めず，引き続き計算を行う．グラフで見ると，1 つのループを 1 方向に対して直列的にメッセージを更新しているものと解釈できる．終了条件は全てメッセージが収束したときとする．以下，アルゴリズムを示す．（s_t は時点における符号器状態である．）

〔アルゴリズム〕

《ステップ 1》 －初期化－

$$\alpha_0(0,\cdots,2^\nu-1)=1/2^\nu,\quad \beta_N(0,\cdots,2^\nu-1)=1/2^\nu. \quad (3)$$

《ステップ 2》 －前向き計算－

$$\alpha_t(s_t)=\sum_{s_{t-1}\in S_{t-1}}\alpha_{t-1}(s_{t-1})\cdot\gamma_t(s_{t-1},s_t). \quad (4)$$

$$\alpha_t^\diamond(s_t)=1/\sum_{s_t\in S_t}\alpha_t(s_t).\ (t=1,\cdots,N,N+1,\cdots) \quad (5)$$

（ただし $\gamma_t(s_{t-1},s_t)=\gamma_n(s_{n-1},s_n)$ （$n=t \bmod N$）で，終了条件は $|\alpha_t^\diamond(s_t)-\alpha_{t-N}^\diamond|\approx 0$）

《ステップ 3》 －後向き計算－

前向き計算と同様な流れで後向きに行う．

《ステップ 4》

《ステップ 2》・《ステップ 3》で求めた $\alpha_t^\diamond(s_t)$，$\beta_t^\diamond(s_t)$ を用いて，各時点の事後確率 $P(u_t|Y_1^N)$ を求める．

3.2. 並列型確率伝播アルゴリズム[5]

条件付依存関係を持つ確率変数の組をクリークとして表し，各クリーク間の共通部分をセパレータとしてその該当クリークノードを結んだ2種類のノードからなるグラフを Extended Junction Graph (EJG) と呼ぶ．EJG 上での確率推論アルゴリズムとして，クリーク間でセパレータに関する分布情報をやり取りする．

並列型確率伝播アルゴリズムは，EJG において，全てのクリークから隣接するセパレータに交互に確率伝播を行うことにより，それぞれのクリーク内の相関のある確率変数の結合確率分割表を更新していく並列型の近似事後確率計算法である．直列型の復号法と違い，1 回の更新で全ての事後確率を更新させることが可能である．以下，アルゴリズムを示す．

《ステップ 1》

既知情報をもつクリーク C_k の確率分割表 $P(C_k)$ は以下のように示すことができる．

更新後の結合確率＝更新前の結合確率
×更新前の周辺確率/更新後の周辺確率

《ステップ 2》

セパレータ S_i は隣接する C_i からメッセージを受け取り，確率分割表 $P(S_i)$ を(6)式により更新させる．全ての S_i において同時に行う．

$$P^{\text{更新後}}(S_i) = P^{\text{更新前}}(S_i) \times \prod_{C_i,C_{i+1}} \left(P(C_{i,i+1} \cap S_i) \Big/ P^{\text{更新前}}(C_{i,i+1} \cap S_i)\right) \quad (6)$$

《ステップ 3》

C_i は隣接する S_i からメッセージを受け取り，確率分割表 $P(C_i)$ を(7)式により更新させる．全ての C_i において同時に行う．

$$P^{\text{更新後}}(C_i) = P^{\text{更新前}}(C_i) \times \prod_{C_i,C_{i+1}} \left(P(S_{i-1,i} \cap C_i) \Big/ P^{\text{更新前}}(S_{i-1,i} \cap C_i)\right) \quad (7)$$

《ステップ 4》

《ステップ 3》で収束したら終了．収束しなければ《ステップ 2》へ．

3.3. Lauritzen の三角化を用いて正しい事後確率を計算する方法

正しい事後確率を求めるために，TB 畳込み符号のグラフ表現が 1 つのループからなることを利用した Weiss の方法を用いる．この方法においては，ある時点の状態遷移のもつ事後確率を求めたい場合，その時点のクリークから始めて隣りのクリークの確率分布の行列を逐次的に更新していき，一周させた行列の対角成分が正しい事後確率となる．以下，例を用いて説明する．

〔例〕事後確率 $P(u_t|Y_1^N)$ を求めたい場合．時点 t から始まるグラフのクリークのもつ関数 $P(s_t,s_{t+1}|Y_{t+1})$ とその隣接するクリークのもつ関数 $P(s_{t+1},s_{t+2}|Y_{t+2})$ との積をとり，取り得る状態 s_{t+1} の総和をとる．つまり，(8)式のように結合確率の周辺化を行う．

$$\sum_{s_1,\cdots,s_{t-2}}^{s_{t+2},\cdots,s_N} P(s_1,\cdots,s_{t-1},s_t,\cdots,s_N|Y_1^N) = \sum_{s_{t-2}} \left(\cdots\left(\sum_{s_{t+2}} \left(\sum_{s_{t+1}} P(s_t,s_{t+1}|Y_{t+1}) P(s_{t+1},s_{t+2}|Y_{t+2})\right) \times P(s_{t+2},s_{t+3}|Y_{t+3})\right)\cdots P(s_{t-1},s_t|Y_t)\right). \quad (8)$$

4. 並列型アルゴリズムの近似精度をみるためのシミュレーション

並列型アルゴリズムによる近似事後確率が実際どのような挙動を示しているかを，正しい事後確率を基準にその差を追うことで見ていく．

4.1. 実験条件

以下の条件のもとでシミュレーションを行う．

1. 以下の拘束長 3 の畳込み符号器を用いる．
$$G = \left(1+D^2, 1+D+D^2\right) \quad (9)$$
2. 白色ガウス通信路を仮定し，主に SN 比={0.5, 1.0, 1.5}についてみる．
3. 拘束長 3 に対して短い系列長={10, 20, 40}に焦点を当ててみる．

4.1.1. 実験①(KL 情報量の挙動を見る)

まず，メッセージ更新ごとの KL 情報量の挙動を調べ，その特徴を見る．

[結果]

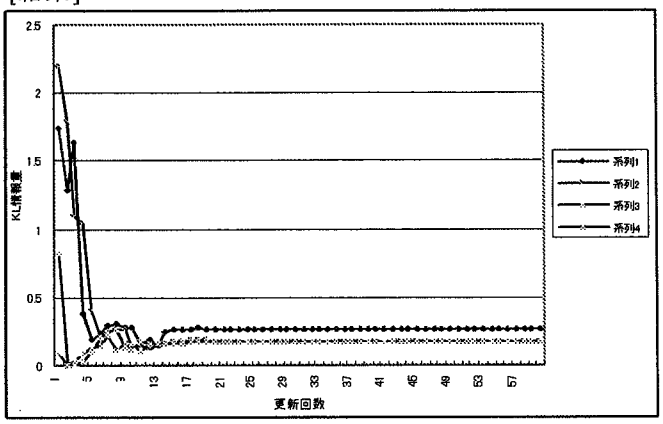


図 1：系列長=10，SN 比=1 のときのある 4 つの系列の KL 情報量の推移

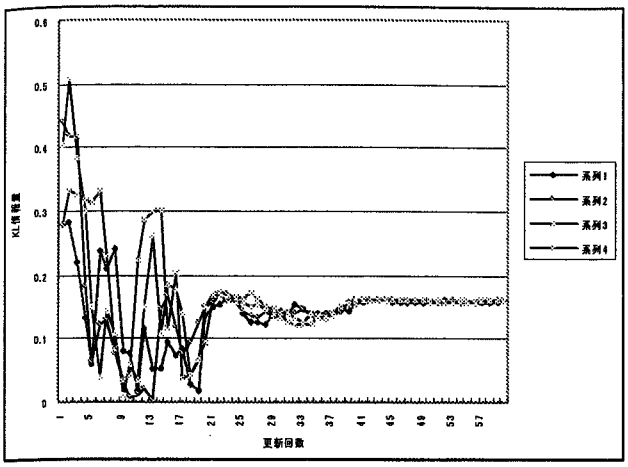


図 2：系列長=20，SN 比=1 のときのある 4 つの系列の KL 情報量の推移

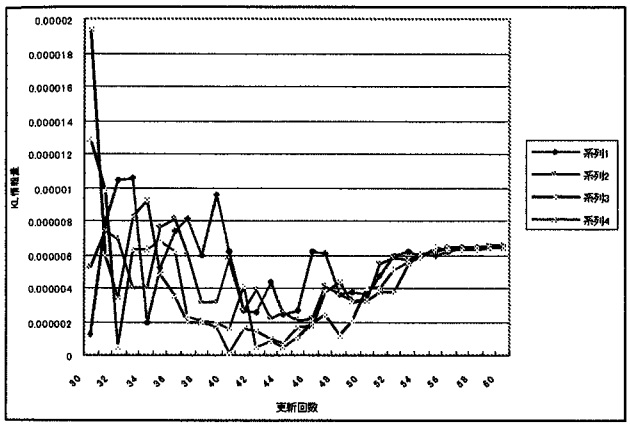


図 3：系列長=40，SN 比=1 のときのある 4 つの系列の KL 情報量の推移

4.1.2. 実験②(KL 情報量がどの更新回数で最小値をとるのかをみる)

4.1.1.の実験により，KL 情報量は収束する前に一度，収束した KL 情報量より小さい値をとっていることが分かる．よって，実際にどの更新回数で KL 情報量が最小値をとるのかを調べ，その割合をみる．以下の結果は全て SN 比=1 で行っている．

[結果]

表 1：系列長 10 のとき

更新回数	3以下	4	5	6	7	8	9	10	11	12	13	14	15以上
割合(%)	6.4	10	16.2	15.6	13.7	9.9	7	5.3	4.4	3	1.8	1.2	5.5

表 2：系列長 20 のとき

更新回数	11以下	12	13	14	15	16	17	18	19	20	21	22	23以上
割合(%)	5.4	3.6	5.9	9	12.3	13.8	12.8	9.8	7.6	5.2	4.4	2.8	7.4

＊ただし，0.2%の割合で単調減少となる系列が現れた．

表 3：系列長 40 のとき

更新回数	31以下	32	33	34	35	36	37	38	39	40	41	42	43以上
割合(%)	5	3.7	5.7	9.4	12.3	13.8	13	9.7	7.6	5.3	4.3	2.7	7.5

＊ただし，3.7%の割合で単調減少となる系列が現れた．

4.1.3. 実験③(正しい事後確率により復号したときの誤りの個数と収束した近似事後確率により復号したときの誤りの個数の割合)

[結果]

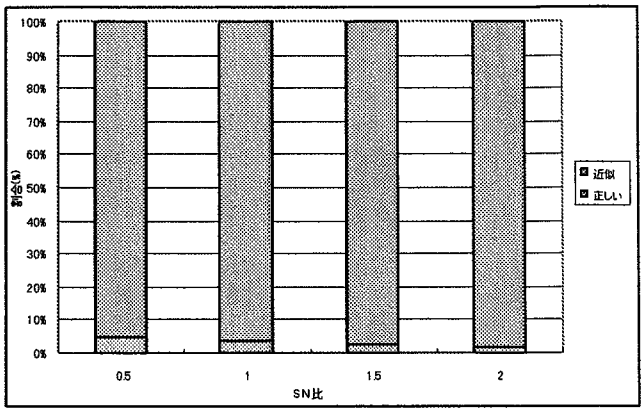


図 4：系列長=10，SN=1 の場合

＊系列長=20, 40 となるにつれて，正しい事後確率による復号誤りの割合は減少する傾向にある．

4.1.4. KL 最小の誤り率と収束後の誤り率の比較

復号する際，誤りが発生する系列のうち，正しい事後確率に基づいて復号しても誤る系列は取り除く．その集合の中で，KL が最小のものの誤り率と KL を収束させるまでメッセージを更新して復号した場合の誤り率をみる．また正しい事後確率で復号したときの誤り率についても比較する．

[結果]

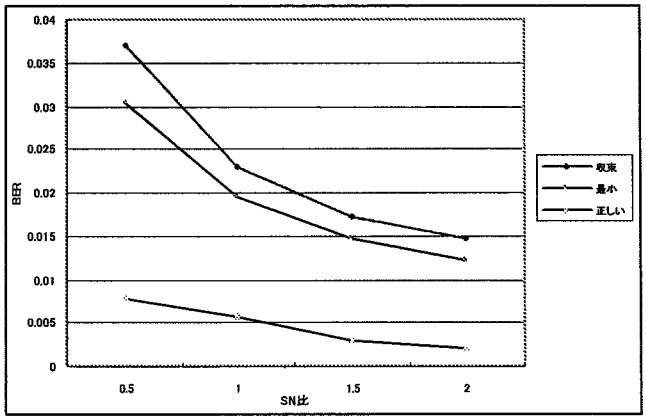


図 3：系列長 10 の場合

5. 考察

①より，系列長が短い場合，KL 情報量は収束する前にほぼ必ず収束値よりも低い値をとることが分かる．つまり，アルゴリズムの正しい事後確率への近似精度はメッセージが収束する前に止めた方が良いことが分

かる．系列長が長くなるにつれて KL 情報量が単調減少していく系列が増えていく．おそらくこれは 1 ループの大きさが大きくなるにつれてメッセージを伝える密度が小さくなり，各シンボルの更新回数ごとの KL 情報量の変化が小さくなっていくからと思われる．

②より，KL 情報量が最小値をとる更新回数の頻度は系列長の長さの直前で止めたときがもっとも多いことがわかる．つまり，系列長が短い場合に限り，1 つのノードから送るメッセージを全ノードに伝える直前で止めた方が近似精度は良いことが分かる．

③より，全体の復号誤りの個数に対する正しい事後確率計算に基づく復号誤りの個数の割合は低いことが分かる．つまり復号アルゴリズムの性能の悪さによる誤りの発生が大部分を占めるので，復号アルゴリズムの改良次第ではさらに復号誤り率を下げることができると考えられる．

④より，並列型アルゴリズムの更新回数は KL 情報量最小で止めた方が性能が良いことが明らかになった．つまり，直列型では KL 情報量を収束させた段階でなければ復号できないのが，並列型では収束させる前に止めたほうが復号誤り率を小さくすることができる．ただし，今回は個々のシンボルのとる KL 情報量が最小となる更新回数で復号したが，本来並列型アルゴリズムを止める場合，その全シンボルの更新回数を同時に止めることになり，今回の結果は並列型の復号アルゴリズムの方が直列型の復号アルゴリズムよりも性能が良くなる可能性があるという一つの指針を示したに過ぎない．

6. まとめ

今回は拘束長に対する系列長が短い場合に，TB 畳込み符号に並列型アルゴリズムを用いた場合の更新回数ごとの近似精度を評価した．また，メッセージを収束させる前にそれ以下の値をとる KL 情報量が多く存在することが分かり，実際に KL 最小の値を取る更新回数で復号した方が復号誤りは減少することも確認した．今後は拘束長をさらに増やした場合の KL 情報量の推移はどのようになっているのか，また収束の早さに対する KL 情報量の挙動をみることによっても新しい性質が明らかになってくるものと思われる．

7. 謝辞

本研究を行うにあたり，数多くの御助言，御支援を賜りました早稲田大学松嶋研究室各氏に心より感謝申し上げます．特に松嶋研究室博士課程 2 年須子統太氏，斉藤友彦氏に深く感謝致します．なお，本研究の一部は文部省科学研究費基盤(C)一般(No.15560338)の援助による．

文　献

[1] L.R.Bahl *et al.*, " Optimal decoding of linear codes for minimizing symbol error rate, " *IEEE Trans. Inform. Theory.*, vol.20, pp.284-287, Mar. 1974.

[2] H.H.Ma and J.K.Wolf," On tail biting convolutional codes," *IEEE Trans. Commun.*, vol. COM-34, pp. 104-111, Feb. 1986.

[3] J.Anderson, " Tailbiting MAP Decorders, " *IEEE Trans. Commun.*, vol.16, No.2, pp.297-302, 1998.

[4] Y.Weiss, " Correctness of Local Propagation in Graphical Models with Loops," *Neural Computation*, vol.12,pp.1-41,2001.

[5] T. Matsushima, T. K. Matsushima, S. Hirasawa, "An alternate algorithm for calculating generalized posterior probability and decoding," In. Proc. Int. Symp. Inf. Theory, 2002.

[6] 制野宇樹，" 確率伝播アルゴリズムの並列処理による高速復号法－事後確率計算アルゴリズムに関する研究－，" 早稲田大学修士論文，2001.

The 27th Symposium on Information Theory
and Its Applications (SITA2004)
Gero, Gifu, Japan, Dec. 14-17, 2004

Small-Loopを含む低密度パリティ検査(LDPC)符号の復号に関する研究
A Note on Decoding of Low Density Parity Check Codes with Small-Loop

小林 直人* 松嶋 敏泰* 平澤 茂一*
Naoto Kobayashi Toshiyasu Matsushima Shigeichi Hirasawa

Abstract— Low Density Parity Check(LDPC) codes decoded by Sum-Product algorithm(SPA) have good performances.If graphical model of LDPC codes include loop structure, especially loop size of 4 structures(Small-Loop), performance of SPA gets worse. Cluster method[?] or algorithm with junction graph[?] can reduce loop structure from graphical model,but these method are more complicated and requires more memory than SPA. We propose a new method of clustering Small-Loop. This method requires same memory size as SPA. And we show the usefulness of this method by computer simulation.

Keywords— LDPC codes , Sum-Product Algorithm , small-loop , clustering

1 はじめに

疎な検査行列を持つ低密度パリティチェック (LDPC) 符号は，復号アルゴリズムとしてSum-Productアルゴリズム (SPA) を用いると非常に良い復号特性を持つため，近年活発に研究がなされている [?].

SPAは確率モデルをグラフィカルモデルで表現し，ノード間でメッセージ伝播を行うことで周辺事後確率を計算するアルゴリズムである．グラフィカルモデルにループ構造が存在しない場合は，SPAは正確な事後確率を計算することができる．ループ構造が存在する場合は，特に理論的保証は無いが，その計算結果が事後確率の近似となることが実験的に確かめられている．LDPC符号は，一般的にグラフィカルモデルにループ構造が存在するため，SPAの計算結果は近似事後確率となるが，特に長さが4のループ (Small-Loopと呼ぶ) はSPAの近似性能を悪くする [?]．その対処法として，小さなLoopを含まないように符号を構成する方法や，Loop構造に含まれる複数の符号語ビットを1つの変数として扱い（クラスタリング）復号する方法が行われている．後者の研究としては，[?] のクラスターBPや [?] のJunction Graphを用いたアルゴリズム等が存在するが，共に計算式がSPAより複雑となり，メッセージ伝播に必要なメモリ量が増加する．

本研究では，2元符号の性質を利用することで，SPAのメッセージ伝播に必要なメモリ量を増加させずに，Small-Loopをクラスタリングした場合と同じ結果を求めるアルゴリズムを提案する．さらに，Small-Loopを含む比較的符号長の短い符号でシミュレーションを行いクラスタリングの有用性を検証した．なお，本稿ではSmall-Loopのクラスタリングについてのみ取り扱っているが，これ以外のループ構造についても同様のアルゴリズムが実現可能である（この場合，クラスタリングする変数の組み合わせにより，同一の符号で複数のグラフ表現が可能となるため，その考慮が必要となる）．

*〒 169-8555 東京都新宿区大久保 3-4-1 早稲田大学理工学部経営システム工学科 Dept. of Industrial & Management Systems Engineering, School of Science and Engineering, Ookubo 3-4-1, Shinjyukuku, Tokyo, 169-8555 Japan. E-mail: kobayashi@matsu.mgmt.waseda.ac.jp

2 準備

2.1 低密度パリティチェック符号

(2元) 低密度パリティチェック (LDPC) 符号 [?] とは，パリティ検査行列 H に含まれるシンボル1の個数が，符号長に対して非常に少ない符号である．シンボル1の配置は基本的にランダムに決定される．

2.2 Sum-Productアルゴリズム

Sum-Productアルゴリズム (SPA) は，確率モデルをグラフィカルモデルで表現し，そのノード間でメッセージ伝播を行うことで事後周辺確率を計算するアルゴリズムである．グラフィカルモデルにループ構造が存在する場合は，正確な事後確率は計算できないが，符号の復号のように，グラフ構造に特殊な条件を持たせたり，また終了条件を定めた下で用いることにより，その計算結果が事後確率の近似となることが知られている．

以下にLDPC符号の復号に用いる際のアルゴリズムを示す．

$\mathcal{N}(m) \equiv \{n : H_{mn} = 1\}$, $\mathcal{M}(n) \equiv \{m : H_{mn} = 1\}$ とする．H_{mn} はパリティ検査行列の m 行 n 列目の値である $(0 \le m < M, 0 \le n < N)$．また，無記憶通信路を仮定し，符号語 $(x_0, x_1, ..., x_{N-1}) \in \{0,1\}^N$ を送信，受信語 $(y_0, y_1, ..., y_{N-1}) \in \mathcal{R}^N$ を受信した元での，各符号語ビット x_i の尤度を $f_i^a = \Pr\{y_i | x_i = a\}$ とする．

$H_{mn} = 1$ を満たす全ての組 (m,n) に対し計算を行う．

Initialization.

$$\begin{cases} q_{mn}^0 = f_n^0 \\ q_{mn}^1 = f_n^1 \end{cases} \tag{1}$$

と初期化する．

Horizontal step.

$$\begin{cases} r_{mn}^0 = ((1 + \delta r_{mn})/2) \\ r_{mn}^1 = ((1 - \delta r_{mn})/2) \end{cases} \tag{2}$$

但し

$$\delta r_{mn} = \prod_{n' \in \mathcal{N}(m) \backslash n} (q_{mn'}^0 - q_{mn'}^1) \tag{3}$$

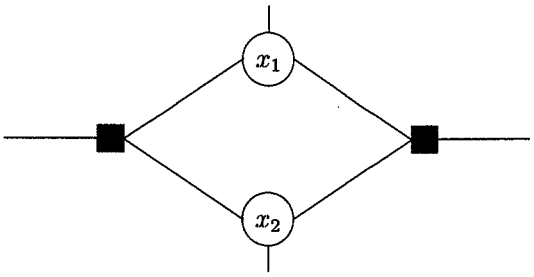


図 1: Factor Graph の例

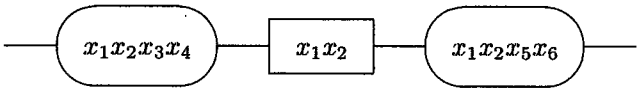


図 2: Junction Graph の例

Vertical step.

$$\begin{cases} q^0_{mn} = \alpha(f^0_n \prod_{m' \in \mathcal{M}(n)\backslash m} r^0_{m'n}) \\ q^1_{mn} = \alpha(f^1_n \prod_{m' \in \mathcal{M}(n)\backslash m} r^1_{m'n}) \end{cases} \tag{4}$$

(α は正規化定数)

Pseudoposterior probabilities.

$$\begin{cases} q^0_n = \alpha(f^0_n \prod_{m' \in \mathcal{M}(n)} r^0_{m'n}) \\ q^1_n = \alpha(f^1_n \prod_{m' \in \mathcal{M}(n)} r^1_{m'n}) \end{cases} \tag{5}$$

Decoding.

$$C_n = \begin{cases} 0 & (q^0_n \geq q^1_n) \\ 1 & (q^0_n < q^1_n) \end{cases} \tag{6}$$

として

$$(C_0, ..., C_{N-1})H^t = 0 \tag{7}$$

を満たしていれば，$(C_0, ..., C_{N-1})$ を推定符号語として終了．そうでなければ，**Horizontal step.** へ戻る (指定回数反復しても，(??) を満たさない場合，その時点の $(C_0, ..., C_{N-1})$ を推定符号語として終了 (もしくは推定語非発見として終了)).

2.3 符号のグラフ表現

線形符号は Tanner Graph[?] や Junction Graph[?] と呼ばれるグラフィカルモデルで表現することができる．Tanner Graph は符号語の各ビットに対応する変数ノードと，検査行列の各行（線形制約）に対応するチェックノードから構成される．Junction Graph は検査行列の各行（線形制約）に対応する Cluster ノードと，符号語の各ビットの集合に対応する Intersection ノードから構成される．

パリティ検査行列 H に，式 (??) のようなシンボル 1 の配置があった場合にタナーグラフで表現すると，長さ 4 のループ構造 (Small-Loop) となる (図??)．Small-Loop が存在すると，SPA の復号性能が悪くなることが実験的に確かめられている [?].

$$H = \begin{pmatrix} & \vdots & & \vdots & \\ \cdots & 1 & \cdots & 1 & \cdots \\ & \vdots & & \vdots & \\ \cdots & 1 & \cdots & 1 & \cdots \\ & \vdots & & \vdots & \end{pmatrix} \tag{8}$$

同様に Junction Graph で表現すると，図??のように Loop 構造にならない．この場合，対象となる符号語 2 ビット (図??における x_1, x_2) が，4 元に拡大されたノードとして扱われる（「クラスタリング」される）．従って，このグラフィカルモデルに SPA を適用すると，このノードに対するメッセージは，$\{q^{00}_{mn}, q^{01}_{mn}, q^{10}_{mn}, q^{11}_{mn}\}$ のように 4 元となり，このメッセージを用いる計算を他の計算と場合分けする必要が生じる．

なお本稿では，式 (??) のようなシンボル 1 が 2 つ重複するような箇所が，3 行又は 3 列以上存在するような符号を除外して考える．

3 提案法

3.1 クラスタリングされたノードの計算

本章では符号を Junction Graph で表現し，クラスタリングされた符号語ビットが存在する場合の SPA による復号方法を考える．簡単のため 2 つの符号語ビットがクラスタリングされた場合のみ説明するが，それ以上の符号語ビットがクラスタリングされた場合でも同様のことが言える．

SPA の **Horizontal step.** は，

$$\begin{cases} r^0_{mn} = \sum_{\{x_{n'}: n' \in \mathcal{N}(m)\backslash n\}} \psi(x_n = 0, \{x_{n'}\}) \\ \qquad \times \prod_{n' \in \mathcal{N}(m)\backslash n} q^{x_{n'}}_{mn'} \\ r^1_{mn} = \sum_{\{x_{n'}: n' \in \mathcal{N}(m)\backslash n\}} \psi(x_n = 1, \{x_{n'}\}) \\ \qquad \times \prod_{n' \in \mathcal{N}(m)\backslash n} q^{x_{n'}}_{mn'} \end{cases} \tag{9}$$

($\psi(\{x_n\})$ は $\{x_n\}$ が偶重みの場合 1，奇重みの場合 0 を返す関数) の式変形となっている．

式 (??) の n がクラスタリングされた符号語ビットの場合，$\{r^{00}_{mn}, r^{01}_{mn}, r^{10}_{mn}, r^{11}_{mn}\}$ という 4 つのメッセージを計算・保持する必要があるが，$\psi(\{x_n\})$ の性質（線形符号の性質）により $r^{00}_{mn} = r^{11}_{mn}$，$r^{01}_{mn} = r^{10}_{mn}$ となるため，それぞれを r^0_{mn}，r^1_{mn} で代表させることができる．

また，式 (??) の n' がクラスタリングされた符号語ビット n^* を含む場合は，式 (??) を

$$\delta r_{mn} = \Big(\prod_{n' \in \mathcal{N}(m)\backslash\{n, n^*\}} (q^0_{mn'} - q^1_{mn'})\Big)(Q^{even}_{mn^*} - Q^{odd}_{mn^*}) \tag{10}$$

(Q^{even}_{mn} は変数ノード n がとり得る偶重みの元に対してのメッセージ q_{mn} の和，Q^{odd}_{mn} は奇重みの元に対してのメッセージ q_{mn} の和を表す) とすることで，同様の計算を行うことができる．従って，**Vertical step.** において，n がクラスタリングされた符号語ビット n^* の場合，

$\{q^{00}_{mn^*}, q^{01}_{mn^*}, q^{10}_{mn^*}, q^{11}_{mn^*}\}$ の4つを保持する必要はなく，$q^0_{mn^*} = Q^{even}_{mn^*}$，$q^1_{mn^*} = Q^{odd}_{mn^*}$ とすれば良い．

以上2つの性質を用いると，ループ構造に含まれる変数ノード，チェックノードを場合分けして計算することで，SPA のメッセージ伝播に必要なメモリ量を増加させずに，クラスタリングした場合の計算と同じ結果を求めることができる．

以上のことを Small-Loop 構造に適用したアルゴリズムを次節に示す．

3.2 提案アルゴリズム

L_A を Small-Loop に含まれる2つの変数ノードのうち任意の1つの変数ノードの集合とし，L_B を Small-Loop に含まれる変数ノードのうち L_A に含まれないノードの集合とする．Small-Loop の定義として，$A = \{(m,n)|n$ と m が同一のSmall-Loop内のノード$, n \in L_A\}$，$B = \{(m,n)|n$ と m が同一のSmall-Loop内のノード$, n \in L_B\}$ というチェックノード m と変数ノード n のペアの集合を用意する．

$\beta(n)$, $\gamma(n)$, $\zeta(n)$ は $n \in L_A$ に対して定義され，$\beta(n) \in L_B$ は n が含まれる Small-Loop の n では無い変数ノードを表し，$\gamma(n)$, $\zeta(n)$ は n が含まれる Small-Loop のチェックノードをそれぞれ表す $(\gamma(n) \neq \zeta(n))$．

$H_{mn} = 1$ を満たす全ての組 (m,n) に対し計算を行う．

Initialization.

$((m,n) \notin A, (m,n) \notin B)$ の場合

式 (??) と同じ計算を行う．

$((m,n) \in A)$ の場合

$$\begin{cases} q^0_{mn} = f^0_n f^0_{\beta(n)} + f^1_n f^1_{\beta(n)} \\ q^1_{mn} = f^0_n f^1_{\beta(n)} + f^1_n f^0_{\beta(n)} \end{cases} \tag{11}$$

と初期化する．$((m,n) \in B)$ については計算しない（利用しない）．

Horizontal step.

$$\begin{cases} r^0_{mn} = ((1 + \delta r_{mn})/2) \\ r^1_{mn} = ((1 - \delta r_{mn})/2) \end{cases} \tag{12}$$

$((m,n) \notin A, (m,n) \notin B)$ の場合

式 (??) と同じ計算を行う．

$((m,n) \in A)$ の場合

$$\delta r_{mn} = \prod_{n' \in \mathcal{N}(m) \backslash \{n, \beta(n)\}} (q^0_{mn'} - q^1_{mn'}) \tag{13}$$

$((m,n) \in B)$ については計算しない（利用しない）．

Vertical step.

$(n \notin L_A, n \notin L_B)$ の場合

式 (??) と同じ計算を行う．

$(n \in L_A \vee n \in L_B)$ の場合

$$\begin{cases} Q_{00} = \prod_{m' \in \mathcal{M}(n) \backslash \{\gamma(n), \zeta(n)\}} r^0_{m'n} \\ \quad \times \prod_{m'' \in \mathcal{M}(\beta(n)) \backslash \{\gamma(n), \zeta(n)\}} r^0_{m''n} \\ \quad \times r^0_{\gamma(n)n} r^0_{\zeta(n)n} f^0_n f^0_{\beta(n)} \\ Q_{01} = \prod_{m' \in \mathcal{M}(n) \backslash \{\gamma(n), \zeta(n)\}} r^0_{m'n} \\ \quad \times \prod_{m'' \in \mathcal{M}(\beta(n)) \backslash \{\gamma(n), \zeta(n)\}} r^1_{m''n} \\ \quad \times r^1_{\gamma(n)n} r^1_{\zeta(n)n} f^0_n f^1_{\beta(n)} \\ Q_{10} = \prod_{m' \in \mathcal{M}(n) \backslash \{\gamma(n), \zeta(n)\}} r^1_{m'n} \\ \quad \times \prod_{m'' \in \mathcal{M}(\beta(n)) \backslash \{\gamma(n), \zeta(n)\}} r^0_{m''n} \\ \quad \times r^1_{\gamma(n)n} r^1_{\zeta(n)n} f^1_n f^0_{\beta(n)} \\ Q_{11} = \prod_{m' \in \mathcal{M}(n) \backslash \{\gamma(n), \zeta(n)\}} r^1_{m'n} \\ \quad \times \prod_{m'' \in \mathcal{M}(\beta(n)) \backslash \{\gamma(n), \zeta(n)\}} r^1_{m''n} \\ \quad \times r^0_{\gamma(n)n} r^0_{\zeta(n)n} f^1_n f^1_{\beta(n)} \end{cases} \tag{14}$$

を計算した後，

$(n \in L_A, (n,m) \notin \{L_A, L_B\})$ ならば

$$\begin{cases} q^0_{mn} = \alpha((Q_{00} + Q_{01})/r^0_{mn}) \\ q^1_{mn} = \alpha((Q_{10} + Q_{11})/r^1_{mn}) \end{cases} \tag{15}$$

$(n \in L_B, (n,m) \notin \{L_A, L_B\})$ ならば

$$\begin{cases} q^0_{mn} = \alpha((Q_{00} + Q_{10})/r^0_{mn}) \\ q^1_{mn} = \alpha((Q_{01} + Q_{11})/r^1_{mn}) \end{cases} \tag{16}$$

$(n \in L_A, (n,m) \in \{L_A, L_B\})$ ならば

$$\begin{cases} q^0_{mn} = \alpha((Q_{00} + Q_{11})/r^0_{mn}) \\ q^1_{mn} = \alpha((Q_{01} + Q_{10})/r^1_{mn}) \end{cases} \tag{17}$$

を計算する．$(n \in L_B, (n,m) \in \{L_A, L_B\})$ の場合は計算しない．

Pseudoposterior probabilities.

$(n \notin L_A, n \notin L_B)$ の場合

式 (??) と同じ計算を行う．

$(n \in L_A)$ の場合

$$\begin{cases} q^0_n = \alpha(Q_{00} + Q_{01}) \\ q^1_n = \alpha(Q_{10} + Q_{11}) \end{cases} \tag{18}$$

$(n \in L_B)$ の場合

$$\begin{cases} q^0_n = \alpha(Q_{00} + Q_{10}) \\ q^1_n = \alpha(Q_{01} + Q_{11}) \end{cases} \tag{19}$$

Decoding. SPA と同様に行う．

このアルゴリズムは Small-Loop 内の変数ノードへのメッセージの伝播を，1つのノード $(n \in L_A)$ に代表させて行っている．また，式 (??) は同一の Small-Loop においてのメッセージに対しては同じ計算となるので，1つの Small-Loop に対して式 (??) は1回計算すればよい．

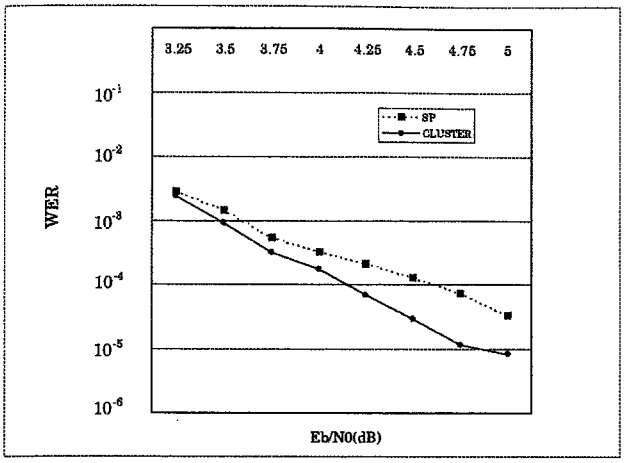


図 3: 実験 1 結果

3.3 計算量評価

SPA と提案アルゴリズムについて演算回数を簡単に比較する．符号長 N，符号化率 0.5，H の列重みを 3，行重みを 6，Small-Loop の個数が L の符号を考える．演算は和，積それぞれ 1 回とカウントし，正規化は考えないこととする．比較を表??に示す．

表 1: 演算回数の比較

SPA		提案アルゴリズム (CLUSTER)	
式 (??)	$12N$	式 (??) +式 (??)	$12N-10L$
式 (??)	$12N$	式 (??)+式 (??)〜(??)	$12N+12L$

4 シミュレーション

4.1 実験条件

符号は符号長 200，符号化率 0.5，H の列重みを 3，行重みを 6 の LDPC 符号のクラス，通信路は白色ガウス通信路を用いる．但し，式 (??) のようなシンボル 1 が 2 つ重複するような箇所が，3 行又は 3 列以上存在するような符号は除外した．SPA，提案アルゴリズム共に最大反復回数は 400 回とする．

4.2 実験 1

上記条件で符号をランダムに 5 つ生成し (Small-Loop の個数は 20 個前後)，それぞれについて SPA，提案アルゴリズムを用いて復号する．図??に SNR3.25〜5.00 のブロック誤り率のグラフを示す (各 SNR, 符号に対しての実験回数は 600000 回)．

4.3 実験 2

上記条件で符号をランダムに 1 つ生成し，ある 1 つの Small-Loop に含まれる 2 つの符号語ビットを意図的に送信誤りとし復号を行う（送信時にビットを反転させる）．その他の符号語ビットは通常通り送信する．表??に，各 SNR において 5000 回実験した内の，アルゴリズムの復号誤りパターンの個数を示した．

4.4 考察

実験 2 より Small-Loop 構造のビット部分に誤りが発生した場合，SPA では復号誤りが発生しやすくなるが，Small-Loop をクラスタリングすると，その誤りの発生を抑えることができることがわかる．このことにより，平均的な復号誤り率も Small-Loop をクラスタリングした方がよくなると考えられるが，実験 1 の結果よりこれを確認できた．

表 2: 実験 2 結果

SPA	CLUSTER	SNR					
		5.00	4.50	4.00	3.50	3.00	2.50
ERROR (UNDETECT)	RIGHT	180 (138)	263 (190)	377 (222)	423 (208)	512 (197)	544 (171)
ERROR	ERROR	67	126	175	299	575	1068
RIGHT	ERROR (UNDETECT)	2 (2)	2 (1)	6 (2)	18 (8)	35 (9)	85 (6)

※ ERROR は復号誤り，(UNDETECT) は未検出誤りを表す
符号語 2 ビットに意図的にノイズを加えているため SNR は厳密な値ではない

5 まとめ

本稿では，SPA でのメッセージ伝播に必要なメモリ量を増加させることなく Small-Loop をクラスタリングするアルゴリズムを提案した．このアルゴリズムは，Small-Loop による SPA の近似性能が悪くなる影響を抑えることができるため，Small-Loop が存在する LDPC のクラスを，実用することのできる符号のクラスとして用いることができるようになった．

今後の課題として，Small-Loop 以外のループ構造をクラスタリングした場合の，その計算量やモデル構築方法の検討などが挙げられる．また，Small-Loop を含む符号のクラスと，Small-Loop を取り除いた符号のクラスの符号性能の差を明確にする必要もある．

謝辞

本研究を行うにあたり，数多くの御助言，御支援を賜りました松嶋研究室，平澤研究室の各氏に感謝致します．なお，本研究の一部は日本学術振興会科学研究費基盤 (C) 一般 (No.15560338) の援助による．

参考文献

[1] D.J.C.Mackay,"Good Error-Correcting Codes Based on Very Sparse Matrices," *IEEE Trans.Inf. Theory* Vol.45,pp.399-431,Mar,1999.

[2] T.Sato,"Cluster BP and cluster CCCP: Two simple methods for computing Kikuchi approximations" ,Tech. Rep. TR03-0001, Dept. of Computer Science, Tokyo Institute of Technology ,2003.

[3] T.Matsushima,S.Hirasawa,"A formalization of generalized probabilistic reasoning and its procedures on Junction trees", submitted to *IEEE Trans.Inf. Theory.*

[4] B.J. Frey, F.R. Kschischang, and H.-A. Loeliger,"Factor graphs and the sum-product algorithm" ,*IEEE Trans.Inf. Theory*,Vol.47,pp. 498-519,Feb,2001

[5] 和田山 正,"低密度パリティ検査符号とその復号法", トリケップス社,2002.

The 27th Symposium on Information Theory
and Its Applications (SITA2004)
Gero, Gifu, Japan, Dec. 14-17, 2004

誤り訂正符号を利用した直交計画の構成法に関する一考察
～逐次実験に適した直交計画について～
A Note on the Construction of Orthogonal Designs Using Error Correcting Codes

斉藤友彦* 松嶋敏泰* 平澤茂一*
Tomohiko Saito Toshiyasu Matsushima Shigeichi Hirasawa

Abstract— In the field of the experimental design in statistics, a orthogonal design was studied to get the many information with the few number of times of an experiment. It was shown that there were many relations between the problem of constructing orthogonal designs and of constructing linear code in error-correcting codes. In this paper, the sequential experimental designs which were considered in experimental designs are investigated from the view points of error-correcting codes.

Keywords— error-correcting codes, orthogonal designs, sequential experimental designs

1 はじめに

統計学における実験計画法の分野では，少ない実験回数でより多くの情報を得るために直交計画が利用されている [1][2]．直交計画では，できるだけ少ない実験回数で，与えられた要因 (因子) 及びそれらの間にいくつかの想定される交互作用を全て分離推定できるように計画を構成することが重要である．

一方，誤り訂正符号の分野において，線形符号は理論，実用両面において重要な符号クラスの一つである [6]．線形符号において，符号構成問題，すなわち符号長，最小距離が与えられた下で，できるだけ情報長が大きい符号を構成する問題，は重要な研究テーマであり，従来多くの研究がされている．

直交計画及び線形符号はどちらも線形部分空間として表現することができる．従来，この2つを構成する問題はほぼ等価であることが示されており，そして線形符号の構成法を利用した，直交計画の構成法が提案されている [1]．

ところで，実験計画法では一般的に，想定すべき交互作用があらかじめ分かっており，それに基づいて直交計画が構成される．しかし，現実問題において，想定すべき交互作用があらかじめ分からない場合があり，このような場合，実験で得られたデータからの推定結果を基に，逐次的に実験を追加することによって対応している．このような実験を逐次実験と呼ぶ [8][4]．しかし，従来，この逐次実験に適した直交計画を構成する問題はほとんど考えられていない．

一方，誤り訂正符号においても，通信路の状態があらかじめ分からない場合があり，このような場合，最初に送信された受信後の推定結果を基に，逐次的に符号の冗長部分を再送させることによって対応している．このような方式を適応型 ARQ 方式と呼ぶ．この適応型 ARQ 方式に適した符号を incremental redundancy codes と呼び，従来多くの incremental redundancy codes の構成法が提案されている [5]．

本稿では誤り訂正符号の視点から，逐次実験に適した直交計画の構成法に関して考察を行う．そして，逐次実験に適した直交計画を構成する問題は incremental redundancy codes を構成する問題とほぼ等価であることを示し，従来提案されている incremental redundancy codes の構成法が，逐次実験に適した直交計画の構成法に利用できることを示す．

*〒 169-8555 東京都新宿区大久保 3-4-1　早稲田大学理工学部経営システム工学科. Dept. of Industrial and Management Systems Engineering, Waseda Univ, Okubo 3-4-1,Shinjuku,Tokyo,169-8555 Japan. E-mail: tomohiko@matsu.mgmt.waseda.ac.jp

2 準備

2.1 直交計画

いま n 個の因子を $F_1, F_2, \cdots, F_n$ とし，各因子に与える条件 (水準) の集合を $\{\Omega_1, \Omega_2, \cdots, \Omega_n\}$ とする．ここで，水準の集合の大きさ，すなわち水準数は $|\Omega_i| = \omega_i, (i = 1, 2, \cdots n)$ とする．また，t 因子間に存在する交互作用を t 次の交互作用と呼び，$F_{i_1} \times \cdots \times F_{i_t}$ とかく．

ここで，

$$\begin{aligned} \mathbf{\Omega} &= \Omega_1 \times \Omega_2 \times \cdots \times \Omega_n \\ &= \{(\nu_1, \nu_2, \cdots, \nu_n); \forall \nu_i \in \Omega_i, i = 1, \cdots, n\}, \end{aligned}$$

とする．因子 F_i を ν_i に対応させ，ベクトル $\boldsymbol{\nu} = (\nu_1, \cdots, \nu_n)$ を $\mathbf{\Omega}$上の点とする．このとき点 $\boldsymbol{\nu}$ の集合 $\mathbf{\Gamma}(\subseteq \mathbf{\Omega})$ を計画と呼ぶ．また，$\mathbf{\Omega}$を完全計画と呼び，$\mathbf{\Omega}$上の全ての点に関して，すなわち全ての水準組合せで，実験を行うことをを完全実験と呼ぶ．

以下，ω_i が一定，すなわち $\omega_i = q, \Omega_i = GF(q), (i = 1, \cdots, n)$ の場合についてのみ考える．ただし，q は素数の累乗である．

定義 1(直交計画)

計画 $\mathbf{\Gamma}$ を考える．但し，$\mathbf{\Gamma} \subseteq GF(q)^n$, $|\mathbf{\Gamma}| = K$ とする．$\mathbf{\Gamma}$ の中で因子 $F_{i_1}, \cdots, F_{i_r}$ の水準が $\varphi_1, \cdots, \varphi_r$ であるような点全体を

$$\mathbf{\Gamma}^{i_1, \cdots, i_r}_{\varphi_1, \cdots, \varphi_r} = \{(\nu_1, \cdots, \nu_n) \in \mathbf{\Gamma}; \nu_{i_1} = \varphi_1, \cdots, \nu_{i_r} = \varphi_r\}, \quad (\varphi_j \in GF(q))$$

とする．このとき，

$$|\mathbf{\Gamma}^{i_1, \cdots, i_r}_{\varphi_1, \cdots, \varphi_t}| = \frac{K}{\omega_{i_1} \cdots \omega_{i_r}}, (\forall \varphi_1 \in GF(q), \cdots, \forall \varphi_t \in GF(q))$$

となるような計画 $\mathbf{\Gamma}$ を因子数 n，実験回数 K，水準数 q，強さ r の直交計画と呼び，$OA(K,n,q,r)$ と書く．また，$\mathbf{\Gamma}$ 上の全ての点に関して実験を行うことを直交実験と呼ぶ．□

直交計画 $\mathbf{\Gamma}$ が線形ベクトル空間であるとき，生成行列 G によって，

$$\mathbf{\Gamma} = \{\boldsymbol{\nu} = \boldsymbol{\theta} G; \boldsymbol{\theta} \in GF(q)^m\}, \tag{1}$$

と表現される．ただし，

$$G = \begin{bmatrix} g_{11} & \cdots & g_{1n} \\ \vdots & & \vdots \\ g_{m1} & \cdots & g_{mn} \end{bmatrix} = \begin{bmatrix} \boldsymbol{g_1} & \boldsymbol{g_2} & \cdots & \boldsymbol{g_n} \end{bmatrix}, \tag{2}$$

である．このとき直交計画 $\mathbf{\Gamma}$ は線形であると呼ぶ．

以下，直交計画が線形の場合についてのみ考える．

定理 1 [1][2]

直交計画 $\mathbf{\Gamma}$ が強さ r であるための必要十分条件は生成行列 G の任意の r 個以下の列ベクトルが $GF(q)$ 上一次独立である．□

因子 F_i を $\boldsymbol{g}_i$ に対応させた場合，(つまり F_i と ν_i が対応している)，この対応をわりつけと呼ぶ．

ここで，全ての t 次の交互作用が想定される場合，強さ $2t$ の直交計画が必要となる．したがって，直交計画を構成する問題は因子数 n，強さ $r(=2t)$ が与えられた下で，実験回数，すなわち (2) 式における m の値が，最も小さくなる (生成) 行列を構成する問題に帰着する．

本来，実験計画法では，より細かい交互作用に関する情報 (例えば交互作用は $F_1 \times F_2$ のみ存在し他の交互作用は存在しない等) が与えられる．しかし，全ての t 次の交互作用を想定する場合は実用上重要であり，本稿ではこの場合のみ考える．

2.2 線形符号

n 次元線形ベクトル空間 $\{0,1,\cdots,q-1\}^n$ の k 次元部分空間を $(n,k)_q$ 線形符号と呼ぶ．ここで，n を符号長，k を情報長と呼ぶ．また，$(n,k)_q$ 線形符号の最小 (ハミング) 距離が d であるとき，$(n,k,d)_q$ 線形符号と呼ぶ．$(n,k,d)_q$ 線形符号 C は $(n-k)\times n$ の検査行列 H によって次のように表現することができる．

$$C = \{\boldsymbol{c} \in GF(q)^n; \boldsymbol{c}H^T = \boldsymbol{0}\}.$$

このとき，検査行列 H と最小距離 d には次のような関係がある．

定理 2 [6]

線形符号 C の最小距離が d であるための必要十分条件は検査行列 H の任意の $d-1$ 個以下の列ベクトルが $GF(q)$ 上一次独立である．□

線形符号を構成する問題は，符号長 n，最小距離 d が与えられた下で，$n-k$ の値が最も小さくなる (検査) 行列 H を構成する問題に帰着する．

3 従来研究

3.1 直交計画と線形符号

直交計画と線形符号の関係について以下のような定理が示されている．

定理 3 [1]

C を $(n,k,d)_q$ 線形符号とする．C の双対符号 $C^\perp$ の各符号語を点とし，その点の集合を $\mathbf{\Gamma}$ とする．このとき，$\mathbf{\Gamma}$ は $OA(n,q^{n-k},q,d-1)$ となる．□

3.2 逐次実験

実験計画法では一般的に，想定すべき交互作用効果があらかじめ分かっている場合について議論がされている．しかし，現実問題において想定すべき交互作用効果があらかじめ分からない場合がある．このような場合，実験によって得られたデータの推定結果から逐次的に実験を追加することによって対応している．このような実験を逐次実験と呼ぶ．以下，例を用いて逐次実験について説明する．

例 1 (逐次実験)

因子 F_1,F_2,F_3,F_4 が存在し，各因子の水準数は 2 とする．このとき 1 次近似もしくは 2 次近似のモデルどちらかが仮定できるとする．すなわち以下の 2 つのモデルのどちらかが仮定できるとする．

$$\begin{aligned} y_{i,j,k,l} &= \theta_0 + \theta_{1,i} + \theta_{2,j} + \theta_{3,k} + \theta_{4,l} + e_{i,j,k,l} \\ y_{i,j,k,l} &= \theta_0 + \theta_{1,i} + \theta_{2,j} + \theta_{3,k} + \theta_{4,l} \\ &+ (\theta_1\theta_2)_{i,j} + \ldots + (\theta_3\theta_4)_{k,l} + e_{i,j,k,l}, \end{aligned}$$

但し，θ_0 を一般平均，$\theta_{i,j}(j \in \{0,1\})$ を因子 $F_i(i = 1,\ldots,4)$ の主効果，$(\theta_i\theta_j)_{k,l}(k,l \in \{0,1\})$ を因子 F_i,F_j $(i,j = 1,\ldots,4)$ の交互作用効果とし，$\sum_j \theta_{i,j} = 0$，$\sum_k (\theta_i\theta_j)_{k,l} = \sum_l (\theta_i\theta_j)_{k,l} = 0$ を満たすものとする．また，$e_{i,j,k,l}$ は偶然誤差を表し，平均 0，分散 σ^2 の正規分布に従うものとする．

このとき以下の 2 つの生成行列 G_1,G_2 で定義される計画 $\mathbf{\Gamma}_1$,$\mathbf{\Gamma}_2$ を用意する．

$$G_1 = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix},$$

$$G_2 = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

但し，$\mathbf{\Gamma}_1$ は $OA(2^3,4,2,3)$ であり，$\mathbf{\Gamma}_2$ は $OA(2^4,5,2,4)$ である．また，$\mathbf{\Gamma}_1$ は $\mathbf{\Gamma}_2$ の部分空間，すなわち $\mathbf{\Gamma}_1 \subset \mathbf{\Gamma}_2$ である．

このとき，以下のように実験及び各効果の推定を行う．まず，1 次近似のモデルだと仮定して，計画 $\mathbf{\Gamma}_1$ を用いて実験を行う．そして，実験で得られたデータから各主効果及び交互作用効果に関して分散分析 [7] を行いその結果，仮定したモデルが正しいと判断された場合，その

まま各主効果を推定し実験を終了する．もし，仮定した1次近似のモデルが誤りであり，2次の交互作用が存在すると判断された場合，計画Γ_2を用いて実験を行い(因子$F_1,\ldots,F_4$をG_2の2列目から5列目にわりつける)，各主効果，2次の交互作用効果を推定し，実験を終了する．但し，Γ_2について実験を行う際，Γ_1はΓ_2の部分空間なので，$\Gamma_2\backslash\Gamma_1$に関して追加実験を行えば良い．

ここで，Γ_1は強さ3の直交計画なので，2次の交互作用効果を求めることはできず，求められた2次の交互作用効果は他の交互作用効果と交絡している．例えば，$F_1\times F_2$の効果を求めた場合，その値は$F_1\times F_2$の効果と$F_3\times F_4$の効果の和になっている．従って，Γ_1に関して実験を行い，分散分析の結果，2次の交互作用が存在すると判定された場合，それは他の2次の交互作用と交絡した結果であり，追加実験を行うことによって，その効果を分離する．□

3.3 incremental redundancy codes

本節では，適応型ARQ方式で利用されるincremental redundancy codesの定義，及びその構成法について述べる．まず，incremental redundancy codesの定義は以下の通りである．

定義2 [5] (incremental redundancy codes)

C_iを$(n_i,k,d_i)_q$線形符号とする$(i=1,2,\ldots,l)$．但し，$n_1=k+1,n_{i+1}=n_i+1$ $(i=1,2,\ldots,l-1)$とする．ここで，C_iの生成行列G_iが，

$$G_i=[I_k|\mathbf{p}_1|\mathbf{p}_2|\ldots|\mathbf{p}_i]\quad(i=1,2,\ldots,l),$$

となるとき，$\{C_1,C_2,\ldots,C_l\}$をincremental redunadancy codesと呼ぶ．但し，I_kは$k\times k$単位行列，$\mathbf{p}_i$ $(i=1,2,\ldots,l)$を$k\times 1$列ベクトルとする．□

次に，[5]で提案されたincremental redundancy codesの構成法について述べる．以下のアルゴリズムにおいて，$q=2$とし，情報長kは入力として与えられるものとする．また，$A(d,i)$を符号長がn_i，ハミング重みがdの符号語の数とする．

[符号構成アルゴリズム]

1. $i=1$とし，$G_0=[I_k]$とする．
2. 長さkの列ベクトル($\mathbf{0}$を除く)から，$A(d_{i-1},i-1)$の値を最も減らすものを$\mathbf{p}_i$とする．このとき，複数の列ベクトルが$\mathbf{p}_i$として選択される場合がある．
3. 2.で選択された全ての$\mathbf{p}_i$から，更に$A(d_{i-1}+1,i-1)$の値を最も減らすものを選択する．
4. $A(n_{i-1},i-1)$まで続け，残った全ての$\mathbf{p}_i$について$G_i=[G_{i-1}|\mathbf{p}_i]$を決定する．
5. 2.-4.までを全てのG_{i-1}について行う．
6. 以下を満たすG_i'が存在する場合，G_iを削除する$(G_i,G_i'$は2.-5.において選択された生成行列$)$．

$$\exists_{1\le r\le n_i}(\forall_{0\le l<r}A(l,i)=A'(l,i))\wedge A(r,i)>A'(r,i).$$

7. 任意の置換σに関して，以下のような$\{G_i,G_i'\}$の組がある場合どちらか一方を削除する．

$$G_i'=\left[I_k|\mathbf{p}_{\sigma(1)}|\ldots|\mathbf{p}_{\sigma(i)}\right],G_i=[I_k|\mathbf{p}_1|\ldots|\mathbf{p}_i].$$

8. 任意の置換σに関して以下のような$\{G_i,G_i'\}$の組がある場合どちらか一方を削除する．

$$G_i'=\left[\begin{array}{c|c}I_k & \begin{array}{c}z_{i,\sigma(1)}\\ \vdots\\ z_{i,\sigma(k)}\end{array}\end{array}\right],G_i=\left[\begin{array}{c|c}I_k & \begin{array}{c}z_{i,1}\\ \vdots\\ z_{i,k}\end{array}\end{array}\right].$$

但し，$z_{i,j}=(p_{j,1},\ldots p_{j,i}),(1\le j\le k)$とする．
9. $i\le l$ならばiを1増やし，2.へ．

4 逐次実験とincremental redundancy codes

4.1 逐次型直交計画

本節では，逐次型直交計画の定義，及びその逐次実験への利用法について述べる．まず，逐次型直交計画を以下のように定義する．

定義3 (逐次型直交計画)

Γ_iを$OA(q^i,n_i,q,r_i)$とする$(i=1,2,\ldots,l)$．但し，$n_i-i=k$(定数)，$n_{i+1}=n_i+1(i=1,2,\ldots,l)$とする．ここで，$\Gamma_i$が$\Gamma_{i+1}$の部分空間，すなわち

$$\Gamma_1\subset\Gamma_2\subset\ldots\subset\Gamma_l,$$

となるとき$\{\Gamma_1,\Gamma_2,\ldots,\Gamma_l\}$を逐次型直交計画と呼ぶ．□

逐次型直交計画は，例えば以下のように逐次実験に利用することができる．まず，Γ_i，Γ_{i+1}を用意する．但し，Γ_i，Γ_{i+1}はそれぞれ$OA(q^i,n_i,q,2t+1)$，$OA(q^{i+1},n_i+1,q,2t+2)$とする．まずΓ_iを利用して直交実験を行う．そして，実験で得られたデータから分散分析を行い，$t+1$次以上の交互作用が想定される場合，Γ_{i+1}を利用して直交実験を行う．その際，$\Gamma_i\subset\Gamma_{i+1}$なので$\Gamma_{i+1}\backslash\Gamma_i$についてのみ追加実験を行う．この他にも，$\{\Gamma_1,\Gamma_2,\ldots,\Gamma_l\}$の中から三つ直交計画を選択し，3段階の逐次実験にするなど，逐次型直交計画の様々な利用法が考えられる．

4.2 逐次型直交計画とincremental redundancy codes

逐次型直交計画とincremental redundancy codesの間には以下のような関係がある．

定理4

C_iを$(n_i,k,d_i)_q$線形符号とし$(i=1,2,\ldots,l)$，そして$\{C_1,C_2,\ldots,C_l\}$はincremental redundancy codeであるとする．このとき，C_iの双対符号$C_i^{\perp}$の各符号語を点とし，その点の集合をΓ_iとする．このとき，Γ_iは$OA(q^{n_i-k},n_i,q,d_i-1)$であり，$\{\Gamma_1,\Gamma_2,\ldots,\Gamma_l\}$は逐次型直交計画となる．

(証明) $\mathbf{v}_1\in\Gamma_i$とする．このとき$\mathbf{v}_1$は$\mathbf{v}_1G_i^T=\mathbf{0}$を満たす．よって$[v_1,0](\in GF(q)^{n_i+1})$

$$[\mathbf{v}_1,0]G_{i+1}^T=[\mathbf{v}_1,0]\left[\begin{array}{c}G_i^T\\ \mathbf{p}_{i+1}^T\end{array}\right]=\mathbf{0},$$

を満たす．よって，任意の$\mathbf{v}_1\in\Gamma_i$に関して，$[\mathbf{v}_1,0]\in\Gamma_{i+1}$を満たす．したがって，$\Gamma_i\subset\Gamma_{i+1}(i=1,2,\ldots l)$となる．□

表 1: $k=4$ の incremental redundancy codes を利用した場合

因子数 n_i	強さ r_i	因子数 n_i	強さ r_i
5	1	11	4
6	1	12	5
7	1	13	5
8	2	14	6
9	3	15	7
10	3	16	7

4.3 符号構成法を利用した逐次型直交計画

3.3 節で述べた構成法により incremental redundancy codes を構成し，その双対符号から，逐次型直交計画を構成することができる．本稿では一例として情報長 $k=4$ の場合における incremental redundancy codes を利用した逐次型直交計画の因子数，強さを表 1 に示す．このとき実験回数はそれぞれ $2^{n_i-k}=2^{n_i-4}$ 回となる．

5 考察

本節では，例を用いて 4.3 節で示した逐次型直交計画の有効性について考察する．

例 2

因子数 9，水準数 2 の場合について考える．また 1 次近似，2 次近似のモデルである確率がそれぞれ $p, 1-p$ であるとする．

初めに，逐次型直交計画を利用した場合について考える．まず表 1 における因子数 9，実験回数 2^5，強さ 3 の直交計画を利用する．そして分散分析の結果，2 次の交互作用効果が存在すると判断された場合，表 1 における因子数 11，実験回数 2^7，強さ 4 の直交計画を利用する．このときの平均実験回数は

$$2^5 \times p + 2^7 \times (1-p) = 128 - 96p, \qquad (3)$$

となる．

次に，上記のモデルに対して逐次型ではない，最適な直交計画，すなわち因子数及び強さが与えられた下で最も実験回数が少ない直交計画，を利用した場合について考える．[1]，pp.319 より，因子数 9，強さ 3，4 における最適な直交計画の実験回数は 24，2^7 となる．

まず，因子数 9，実験回数 24，強さ 3 の直交計画を利用した場合を考える．このとき，モデルが 1 次近似のモデルであれば良いが，もし 2 次近似のモデルの場合，最初に行った 24 回の実験で得られたデータを捨て，改めて，因子数 9，実験回数 2^7，強さ 4 の直交計画を利用して実験を行わなければならない．したがって，この場合の平均実験回数は

$$24 \times p + (24 + 2^7) \times (1-p) = 152 - 128p, \qquad (4)$$

となる．ここで，(3)，(4) 式を比較した場合，$p < \frac{3}{4}$ の場合，最適な直交計画を利用したほうが平均実験回数が少なくなり，$p > \frac{3}{4}$ の場合，逐次型直交計画を利用したほうが平均実験回数が少なくなる．

次に，初めから因子数 9，実験回数 2^7，強さ 4 の直交計画を利用した場合を考える．この場合，1 次近似のモデル，2 次近似のモデルのどちらの場合でも，実験回数は 2^7 となる．したがって平均の実験回数は 2^7 となる．したがって，p がどのような値をとっても必ず逐次型直交計画を利用した方が平均実験回数が少なくなる． □

6 まとめ

本稿では，実験計画法における逐次実験と，誤り訂正符号における適応型 ARQ 方式で利用される incremental redundancy codes との対応関係を示した．そして，従来提案された incremental redundancy codes の符号構成法が逐次実験に適した直交計画の構成に利用できることを示した．今後の課題として，incremental redundancy codes の符号構成法を改良し，より実験計画法の問題に適した直交計画の構成法を提案することが挙げられる．

謝辞

本研究を行うにあたり，ご助言を頂いた，松嶋研究室，平澤研究室の皆様に深く感謝致します．なお，本研究の一部は日本学術振興会研究費基盤研究 (C)(No.15560338) の援助による．

参考文献

[1] A.S.Hedayat,N.J.A.Sloane,and J.Stufken, "Orthogonal Arrays: Theory and Applications," Springer ,New York,1999.

[2] 高橋磐郎, 組合せ理論とその応用, 岩波全書 316, 東京, 1979.

[3] 浮田善文, 松嶋敏泰, 平澤茂一, "直交計画を用いたブール関数の学習に関する一考察," 信学論 (A),vol.J86-A,no.4,pp482-490,Apr.2003.

[4] 浮田善文, 松嶋敏泰, 平澤茂一, "ブール関数の逐次実験計画を用いた学習に関する一考察," 信学技報,COMP2002-52,2002.

[5] D.Cygan and E.Offer, "Short Linear Incremental Redundancy Codes Having Optimal Weight Structure Profile," IEEE Trans. Inform. Theory,vol.IT-37,pp. 192 - 195, January 1991.

[6] 平澤茂一, 西島利尚, 符号理論入門, 培風館, 東京,1999.

[7] 永田靖, 入門実験計画法, 日科技連出版社, 東京,2000.

[8] 奥野忠一, 芳賀敏郎, 実験計画法, 培風館, 東京,1969.

[9] 斉藤友彦, 吉田隆弘, 松嶋敏泰, "誤り訂正符号構成法を利用した直交計画の構成法に関する一考察," 第 25 回情報理論とその応用シンポジウム予稿集,pp.663-666.

The 27th Symposium on Information Theory
and Its Applications (SITA2004)
Gero, Gifu, Japan, Dec. 14–17, 2004

区間で一定なパラメータを持つ非定常情報源における ベイズ符号の冗長度について

Redundancy of Bayes Codes for Nonstationary Sources with Piecewise Constant Parameters

須子 統太* 松嶋 敏泰* 平澤 茂一*
Tota Suko Toshiyasu Matsushima Shigeichi Hirasawa

Abstract— In this paper we treat universal source coding when the parameters of the probabilistic model of source are known. Bayes code is one of the wellknowm universal codes. Bayes code has Bayes optimality in point of minimization of redundancy. Recently, many researches of Bayes code for nonstationary sources are done. Researches for sources with piecewise constant parameters are one of them. But the Asymptotic character of mean redundancy for this source is not known. In this paper, we evaluate asymptotic mean redundancy of this source with the conditions of change number is not kown. And we show that Bayes code has universality with some condtions.

Keywords— source coding, universal coding, Bayes code, nonstationary source

1 はじめに

情報源の分布のクラスのみ仮定し，そのパラメータについては未知である場合のユニバーサル符号化法については従来から数多くの研究がなされている．その中でもベイズ符号は冗長度をベイズ基準のもとで最小にする符号であるとして知られている．[5] また，定常情報源におけるベイズ符号の平均冗長度については Clark と Barron によって厳密な評価が行われている．[4]

近年，非定常な情報源に対するベイズ符号の研究が行われている．その1つに区間で一定なパラメータを持つ非定常情報源を仮定したものがある．[1][3][8][9][12] この情報源は，ある区間では定常なパラメータに従い系列が発生しているが，そのパラメータがある時点で突然変化する．そのため，一定区間では定常な情報源とみなすことができるが，複数回のパラメータの変化が起きることで全体としては非定常な情報源として扱われる．

この情報源に対する具体的なベイズ符号化アルゴリズムとしては，CTW 法を応用した効率的な一括型符号化アルゴリズム [3] や，効率的な逐次型の符号化アルゴリズム [12]，近似計算を用いることで計算量を軽減したアルゴリズムなどが提案されている．[8][9]

*〒 169–8555 東京都新宿区大久保 3-4-1 早稲田大学理工学部経営システム工学科 Dept. of Industrial & Management Systems Engineering, School of Science and Engineering, Ookubo 3-4-1, Shinjyukuku, Tokyo, 169-8555 Japan. E-mail: suko@matsu.mgmt.waseda.ac.jp

また，この非定常情報源に対する理論評価を行っている研究がいくつかある．Merhav は，総変化回数が既知である条件のもとで，任意の語頭符合に対する平均符号長の下界を示している．更に，同条件のもとでベイズ符号の平均符号長の上界も示しているが，平均冗長度の漸近的な性質についての明確な議論はない．[1] また，総変化回数を未知としたもとで，Willems は有限時点における個別系列に対する冗長度の上界を示している．[3] しかし，この Willems の結果もまた，漸近的な冗長度の性質についての明確な議論はなされていない．

本研究ではこの非定常情報源に対し，総変化回数が未知としたもとで，ベイズ符号の漸近的な性能評価を行う．その際，パラメータの変化にモデルを仮定することにより，平均冗長度についての評価を行う．そのもとで，ベイズ符号の弱ユニバーサル性についても示す．

2 情報源モデルとベイズ符号

2.1 区間で一定なパラメータを持つ情報源

情報源アルファベットを $x_i \in \{0,1,\cdots,k\}$，長さ N の情報源系列を $x_1^N : x_1, x_2, \cdots, x_N$，また i 番目から j 番目までの部分系列を $x_i^j : x_i, x_{i+1}, \cdots, x_{j-1}, x_j$ とする．情報源はパラメータの変化パターン m と，その変化パターンのもとでのパラメータの集合 Θ_m によって定まる確率分布 $P(x_1^N|\Theta_m, m)$ で表されるとする．

いま，ある変化パターン m において c 回目にパラメータの変化が起きた時点を t_c^m とし，総変化回数を $C(m)$ とする．(但し，以降ノーテーションの煩雑を避けるため t_c^m は単に t_c と表記する．また，便宜上 $t_0 \triangleq 1, t_{C(m)+1}-1 \triangleq N$ と定義する．) パラメータの変化パターン m は t_c を用いて $m \triangleq \{t_0, t_1, \cdots, t_{C(m)}\} \in M$ と定義する．またパラメータ集合は，$\Theta_m \triangleq \{\theta_{t_0}, \theta_{t_1}, \cdots, \theta_{t_{C(m)}}\}$ とする．但し，θ_{t_c} は定常分布の k 次元パラメータで，各 θ_{t_c} は独立であるとする．このとき，系列 x_1^N の出現確率は，

$$P(x_1^N|\Theta_m, m) = \prod_{c=0}^{C(m)} P(x_{t_c}^{t_{c+1}-1}|\theta_{t_c}), \qquad (1)$$

で表される．真の m, Θ_m をそれぞれ $m^*, \Theta_{m^*}^*$ と表し，未知であるとする．また，真の総変化回数 $C(m^*)$ も未

知とする．以降，真の c 回目の変化時点 $t_c^{m^*}$ は t_c^* と表記することとする．

2.2 ベイズ符号

情報源系列の符号化確率を仮定すれば算術符号を用いることで符号化が可能になる．そのためユニバーサル情報源符号化の問題は系列の符号化確率を決定する問題に帰着される．ベイズ符号では符号長と理想符号長との差をベイズ基準のもとで最小にする符号化確率を決定する．

今，系列 x_1^N に対する任意の符号化確率を $Q(x_1^N)$ とする．この時，個別系列に対する冗長度を損失関数として以下で定義する．

$$V(\Theta_{m^*}^*, m^*, Q, x_1^N) = \log P(x_1^N|\Theta_{m^*}^*, m^*) - \log Q(x_1^N). \quad (2)$$

この時，平均冗長度をリスク関数として以下で定義する．

$$R(\Theta_{m^*}^*, m^*, Q) = \sum_{x_1^N} P(x_1^N|\Theta_{m^*}^*, m^*) \log \frac{P(x_1^N|\Theta_{m^*}^*, m^*)}{Q(x_1^N)}. \quad (3)$$

本来であればこの平均冗長度を最小にしたいのだが，全てのパラメータについて最小化することはできない．そこでベイズ基準では，このリスク関数に対し事前分布で期待値をとったベイズリスクを最小化する．ベイズリスクは以下で定義する．

$$BR(P(\Theta_m|m), P(m), Q) = \sum_m \int_{\Theta_m} R(\Theta_m, m, Q) P(\Theta_m|m) d\Theta_m P(m). \quad (4)$$

今，m, Θ_m の事前分布 $P(m), P(\Theta_m|m)$ が得られたもとで，ベイズリスクを最小にする符号化確率 P_c は以下で与えられる．

$$P_c(x_1^N) = \sum_m \int_{\Theta_m} P(x_1^N|\Theta_m, m) P(\Theta_m|m) d\Theta_m P(m). \quad (5)$$

これが，ベイズ符号の符号化確率となる．また，ベイズ符号の個別系列に対する冗長度，および平均冗長度は，それぞれ $V(\Theta_{m^*}^*, m^*, P_c, x_1^N), R(\Theta_{m^*}^*, m^*, P_c)$ で与えられる．

2.3 パラメータの変化モデルと事前確率

情報源のパラメータ変化に対して以下の仮定を置く．

仮定 2.1 系列長 N の情報源系列に対し各時点 n において，次式で表される確率 α でパラメータの変化が起きるとする．

$$\alpha = \frac{\rho}{\omega^{(1-s)} N^s}. \quad (6)$$

但し，$0 \le s \le 1, 0 \le \omega \le N, 0 < \rho \le \omega$ となる実数パラメータでいずれも既知であるとする．

パラメータの変化モデルを仮定することは，変化パターン m の事前確率を与えていると考えることができる．仮定2.1を用いて，変化パターン m の事前確率 $P(m)$ を以下で定義する．

$$P(m) = \alpha^{C(m)} (1-\alpha)^{(N-C(m)-1)}. \quad (7)$$

また，上記仮定のもとでパラメータの変化モデルの性質として以下の補題が得られる．(大数の強法則)

補題 2.1 *仮定 2.1 のもとで，確率 1 で次式が成り立つ．*

$$\lim_{N\to\infty} \frac{C(m^*)}{N} = \alpha. \quad (8)$$

2.4 弱ユニバーサル性

ユニバーサル符号の満たすべき性質として，弱ユニバーサルを定義する．

定義 2.1 ある符号化確率 Q が，

$$\lim_{N\to\infty} \frac{1}{N} R(\Theta_{m^*}^*, m^*, Q) = 0, \quad (9)$$

を満たす時，Q を弱ユニバーサルという．

3 平均冗長度の評価

ベイズ符号が弱ユニバーサルとなる場合とならない場合についてそれぞれ平均冗長度の評価を行う．

3.1 弱ユニバーサルとなる場合

まず，変化モデルにおいて，$0 < s \le 1$ である場合について考える．

個別系列に対する冗長度を，次式で展開すると，

$$\log \frac{P(x_1^N|\Theta_{m^*}^*, m^*)}{P_c(x_1^N)} = \log \frac{P(x_1^N|\Theta_{m^*}^*, m^*)}{P_m(x_1^N|m^*)} + \log \frac{P_m(x_1^N|m^*)}{P_c(x_1^N)}, \quad (10)$$

となる．但し，

$$P_m(x_1^N|m^*) = \int_{\Theta_{m^*}} P(x_1^N|\Theta_{m^*}, m^*) P(\Theta_{m^*}|m^*) d\Theta_{m^*}, \quad (11)$$

とする．また，任意の m' に対し，$0 \le P(m') \le 1$ より，

$$P_c(x_1^N) = \sum_m P_m(x_1^N|m') P(m') \ge P_m(x_1^N|m') P(m'), \quad (12)$$

が成り立つ．よって，次式が成り立つ．

$$\log \frac{P_m(x_1^N|m^*)}{P_c(x_1^N)} \le \log \frac{1}{P(m^*)}. \quad (13)$$

これを (10) 式に代入し，両辺を $P(x_1^N|\Theta^*_{m^*},m^*)$ で期待値をとると，

$$\begin{aligned} & R(\Theta^*_{m^*},m^*,P_c) \\ \leq & \sum_{x_1^N} P(x_1^N|\Theta^*_{m^*},m^*)\log\frac{P(x_1^N|\Theta^*_{m^*},m^*)}{P_m(x_1^N|m^*)} \\ & +\log\frac{1}{P(m^*)}, \end{aligned} \tag{14}$$

となる．以降 (14) 式右辺を評価していく．

(14) 式右辺第一項目に注目する．(1) 式の情報源の定義より，

$$\begin{aligned} & \sum_{x_1^N} P(x_1^N|\Theta^*_{m^*},m^*)\log\frac{P(x_1^N|\Theta^*_{m^*},m^*)}{P_m(x_1^N|m^*)} \\ & = \sum_{c=0}^{C(m^*)}\left\{\sum_{x_{t_c^*}^{t_{c+1}^*-1}} P(x_{t_c^*}^{t_{c+1}^*-1}|\theta^*_{t_c^*})\log\frac{P(x_{t_c^*}^{t_{c+1}^*-1}|\theta^*_{t_c^*})}{P_m(x_{t_c^*}^{t_{c+1}^*-1})}\right\}, \end{aligned} \tag{15}$$

と展開できる．但しここで，

$$P_m(x_{t_c}^{t_{c+1}-1}) = \int_{\theta_{t_c}} P(x_{t_c}^{t_{c+1}-1}|\theta_{t_c})P(\theta_{t_c}|m)d\theta_{t_c}, \tag{16}$$

とする．(15) 式の右辺の各項は，それぞれが，定常な各区間を一つの定常情報源とみなした時のベイズ符号の平均冗長度を表している．

十分大きな N' より大きな系列長 N に対し，定常情報源におけるベイズ符号の平均冗長度は，系列長 N の関数として次式で表されることが知られている．[4]

$$r(N) = \frac{k}{2}\log N + O(1). \tag{17}$$

また，c 回目の変化がおきてから $c+1$ 回目の変化が起きるまでの定常区間の長さを N_c とする．$N_c \leq N'$ なる区間の平均冗長度を δ_c とし，N の関数 $\epsilon(N)$ が存在するとする．(15) 式は log の凸性より，

$$\begin{aligned} & \sum_{x_1^N} P(x_1^N|\Theta^*_{m^*},m^*)\log\frac{P(x_1^N|\Theta^*_{m^*},m^*)}{P_m(x_1^N|m^*)} \\ = & \sum_{\{c:N_c>N'\}} r(N_c) + \sum_{\{c':N_{c'}\leq N'\}} \delta_{c'} \\ = & \frac{C(m^*)+1}{C(m^*)+1}\sum_{\{c:N_c>N'\}} r(N_c) + \sum_{\{c':N_{c'}\leq N'\}} \delta_{c'} \\ \leq & (C(m^*)+1)r\left(\frac{1}{\alpha}\right) + \epsilon(N), \end{aligned} \tag{18}$$

となる．この時，定常区間の長さがある任意の定数 N'' 以下になる確率は，

$$\begin{aligned} \Pr\{N_c \leq N''\} &= \alpha\sum_{i=1}^{N''}(1-\alpha)^{i-1} \\ &= 1-(1-\alpha)^{N''+1}, \end{aligned} \tag{19}$$

となり，$0 < s \leq 1$ において，$N \to \infty$ とすると 0 へ近づく．そのため，$N_c \leq N'$ となる区間が出現する確率は無視できるほど小さくなり，$\epsilon(N)$ も 0 へ近づく．

以上の結果および，補題 2.1 より，

$$\begin{aligned} & \sum_{x_1^N} P(x_1^N|\Theta^*_{m^*},m^*)\log\frac{P(x_1^N|\Theta^*_{m^*},m^*)}{P_m(x_1^N|m^*)} \\ & \leq (\alpha N+1)r\left(\frac{1}{\alpha}\right), \end{aligned} \tag{20}$$

また，(14) 式右辺第二項目は，補題 2.1 より，

$$\log\frac{1}{P(m^*)} = \log\frac{1}{\alpha^{\alpha N}(1-\alpha)^{(N-\alpha N-1)}}. \tag{21}$$

以上より，次の定理および系を得る．

定理 3.1 $0 < s \leq 1$ の時，次式が成立する．

$$\begin{aligned} & R(\Theta^*_{m^*},m^*,P_c) \\ \leq & \left\{\frac{(\alpha N+1)k}{2}\log\frac{1}{\alpha}\right. \\ & + \left.\log\frac{1}{\alpha^{\alpha N}(1-\alpha)^{(N-\alpha N-1)}} + O(N^{(1-s)})\right\}. \end{aligned} \tag{22}$$

系 3.1 $0 < s \leq 1$ の時，*(5)* 式による符号化は弱ユニバーサルとなる．

(証明)

$$\frac{1}{N}R(\Theta^*_{m^*},m^*,P_c) \leq O\left(\frac{\log N^s}{N^s}\right), \tag{23}$$

となることより明らか． □

3.2 弱ユニバーサルとならない場合

次に，変化モデルにおいて，$s = 0$ である場合について考える．今，

$$\sum_{x_1^N} P(x_1^N|\Theta^*_{m^*},m^*)\log\frac{P_m(x_1^N|m^*)}{P_c(x_1^N)} \geq 0. \tag{24}$$

より，平均冗長度の下界が次式で得られる．

$$\begin{aligned} & R(\Theta^*_{m^*},m^*,P_c) \\ & \geq \sum_{x_1^N} P(x_1^N|\Theta^*_{m^*},m^*)\log\frac{P(x_1^N|\Theta^*_{m^*},m^*)}{P_m(x_1^N|m^*)}. \end{aligned} \tag{25}$$

また，仮定 2.1 より，$s = 1$ において，個々の定常区間の長さは有限の値を持つ．有限の区間における平均冗長度は定数とみなせることと，補題 2.1 より，

$$\begin{aligned} & \sum_{c=0}^{C(m^*)}\left\{\sum_{x_{t_c^*}^{t_{c+1}^*-1}} P(x_{t_c^*}^{t_{c+1}^*-1}|\theta^*_{t_c^*})\log\frac{P(x_{t_c^*}^{t_{c+1}^*-1}|\theta^*_{t_c^*}|m^*)}{P_m(x_{t_c^*}^{t_{c+1}^*-1})}\right\} \\ & = \sum_{c=0}^{C(m^*)} O(1) \\ & = O(N). \end{aligned} \tag{26}$$

以上より，次の定理および系を得る．

定理 3.2 $s=0$ の時，次式が成立する．

$$\frac{1}{N}R(\Theta^*_{m^*}, m^*, P_c) \geq O(1). \quad (27)$$

系 3.2 $s=0$ の時，*(5)* 式による符号化は弱ユニバーサルとならない．

(証明) 定理より明らか． □

4 まとめ

本研究では，区間で一定なパラメータを持つ非定常情報源に対するベイズ符号の平均冗長度の評価を行った．結果，$0<s\leq 1$ の時，ベイズ符号の平均冗長度が 0 となり弱ユニバーサルとなり，$s=0$ の時には平均冗長度が定数オーダとなり弱ユニバーサルとならないことが分った．つまり，パラメータの総変化回数が N よりも遅い速度で増える場合に弱ユニバーサル性が保証され，N と比例した速度で増える場合には弱ユニバーサル性が保証されない．

また，この情報源に対するベイズ符号を実現する効率的な逐次符号化アルゴリズムについては [12] にて議論されているが，効率化した場合においても系列長 N 全てを符号化するのには総計算量が $O(N^2)$ かかってしまう．重みづける変化パターン m の数を減らすことで計算量の削減が可能であると考えられるが，その際の近似の精度を評価するにあたって，今回の結果が応用できると期待される．

謝辞

本研究を行うにあたり，数多くの御助言，御支援を賜りました，浮田善文氏，並びに松嶋研究室，平澤研究室の各氏に感謝致します．なお，本研究の一部は日本学術振興会科学研究費基盤 (C) 一般 (No.15560338) の援助による．

参考文献

[1] N. Merhav, "On the minimum description length principle for sources with piecewise constant parameters." *IEEE Trans. Inf. Theory*, vol.39, No.6, page 1962, 1993.

[2] Frans M. J. Willems, Y. M. Shtarkov and T. J. Tjalkens, "The Context-Tree Weighting Method: Basic Propreties," *IEEE Trans. Inf. Theory*, vol.41, No.3, page 653, 1995.

[3] Frans M. J. Willems, "Coding for Binary Independent Piecewise-Identically-Distributed Source." *IEEE Trans. Inf. Theory*, vol.42, No.6, page 2210, 1996.

[4] B.S. Clark and A.R. Brron, "Information-theoretic asymptotics of Bayes methods," *IEEE Trans. Inf. Theory*, vol.IT-36, no.3, pp.453-471, May 1990.

[5] T. Matsushima, H. Inazumi and S. Hirasawa, "A Class of Distortionless Codes Designed by Bayes Decision Theory" *IEEE Trans. Inf. Theory*, vol.37, No.5, page 1288, 1991.

[6] T. Matsushima, and S. Hirasawa, "A bayes coding using context tree." *In Proc. Int. Symp. on. Inf. Theory*, page 386, 1994.

[7] Toshiyasu Matsushima and Shigeichi Hirasawa, "A Bayes Coding Algorithm for FSMX Sources," *In Proc. Int. Symp. on Inf. Theory*, page.388, 1995.

[8] G. I. Shamir and N. Merhav, "Low-Complexity Sequential Lossless Coding for Piecewise-Stationary Memoryless Sources." *IEEE Trans. Inf. Theory*, vol.45, No.5, page 1498, 1999.

[9] G. I. Shamir and D. J. Costello, Jr., "Asymptotically Oprimal Low-Complexity Sequential Lossless Coding for Piecewise-Stationary Memoryless Sources-Part :The Regular Case." *IEEE Trans. Inf. Theory*, vol.46, No.7, page 2444, 2000.

[10] William Feller, An Introduction to Probability Theory and Its Application, John Willy & Sons, 1957.

[11] 韓太舜, 小林欣吾, 情報と符号化の数理, 岩波講座応用数学, 岩波書店,1994.

[12] 須子統太, 松嶋敏泰, 平澤茂一, "区間で一定なパラメータを持つ情報源におけるベイズ符号化法について," 第 26 回情報理論とその応用シンポジウム予稿集,pp.165-168, 2003.

The 27th Symposium on Information Th
and Its Applications (SITA2004)
Gero, Gifu, Japan, Dec. 14-17, 2004

階層モデルにおけるベイズ予測の漸近評価に関する一考察

A Note on the Asymptotics of Bayes Prediction for Hierarchical Models

宅味 丈夫* 須子 統太* 松嶋 敏泰*
Takeo Takumi Tota Suko Toshiyasu Matsushima

Abstract— In this paper, we consider bayes prediction for hierarchical models including gaussian mixture model, hidden Markov model, and so on. We derive the bayes predictive distribution weighted by posterior probability of models and evaluate the asymptotics of its cumulative loss and risk.

Keywords— bayes prediction, hierarchical model, universal modelling

1 はじめに

階層モデル，すなわち高次のモデルのパラメータ空間が低次のモデルのパラメータ空間を含むモデルは，ARモデルなどに代表されるマルコフ型のモデルのみならず，混合正規分布や隠れマルコフモデルなども含んでおり，音声・画像認識・時系列予測など幅広い分野で用いられている．近年，こうした階層モデルにおけるベイズ予測に関する研究が理論・応用の双方から進められている．

パラメータと分布が1対1対応することを特定可能と定義する．階層モデルのうちで，特定可能なモデルにおけるベイズ予測に対しては漸近正規性に基づいて累積損失・累積リスクの漸近評価がなされている [1][3].

一方，高次のモデルが低次のモデルを含む場合にパラメータと分布との1対1対応関係が崩れることを特定不能と定義する．例えば，混合正規分布においては真の分布を表す最小の混合数を持つモデルに対して冗長な混合数を持つモデルを適用した場合に特定不能となる．特に，パラメータ空間において特定不能な部分が次元を持つ集合となる場合にはフィッシャー情報行列が退化することから漸近正規性が成立しない．従って，このような場合におけるベイズ予測の漸近評価については未解決な部分が多い．

近年，代数幾何及び多変数複素関数論を用いた解析によって，特定不能性を有する階層モデルにおけるベイズ予測の漸近評価が考案されている [6]．具体的にはモデルを単一に固定した場合のベイズ予測に対して累積リスクを漸近評価する方法が示されている．しかし，この方法で正確な評価が行うことができるのは特殊な場合に限られる．また，累積リスクの上界について評価はなされているが [6][8]，どの程度まで小さくできるのかについては明確にされていない．

本研究においては，モデルを未知とし，モデルの事後確率で重みつけを行ったベイズ予測について漸近評価を行う．まず，その累積損失・累積リスクが漸近的にこれらを最小にするモデルを用いてベイズ予測を行った場合と等価であることを示す．また，それぞれの下界についても考察を行うことで，より厳密な漸近評価が実現できることを示す．

*〒169-8555 東京都新宿区大久保3-4-1 早稲田大学理工学部経営システム工学科. Dept. of Industrial and Management Systems Engineering, Waseda Univ., Okubo 3-4-1, Shinjuku, Tokyo, 169-8555 Japan. E-mail: takumi@matsu.mgmt.waseda.ac.jp

2 問題設定

x^n を $\mathbb{R}^L$ 上に値をとる確率変数列からのサンプルであるとする．$K \in \mathcal{K} = \{1, \cdots, K_{\max}\}$ を離散ラベルとするモデルを m_K，確率分布のクラスを

$$\mathcal{H}_{m_K} = \{p(\cdot|\theta_{m_K}, m_K) | \theta_{m_K} \in \Theta_{m_K}\}, \tag{1}$$

とし，

$$\mathcal{H} = \cup_{K \in \mathcal{K}} \mathcal{H}_{m_K}, \quad \mathcal{H}_{m_1} \subset \mathcal{H}_{m_2} \subset \cdots \subset \mathcal{H}_{m_{K_{\max}}}, \tag{2}$$

なる階層構造が成立しているものとする．真の分布が $p^*(x^n) \in \mathcal{H}$ であると仮定し，

$$K_0 \equiv \min\{K | \exists \theta_{m_K}, \forall x^n, p(x^n|\theta_{m_K}, m_K) = p^*(x^n)\}, \tag{3}$$

とする．また，カルバック距離を

$$H(\theta_{m_K}) \equiv \int p^*(x) \log \frac{p^*(x)}{p(x|\theta_{m_K}, m_K)} dx, \tag{4}$$

とすることで，

$$\{\theta^*_{m_K}\} \equiv \arg\min_{\theta_{m_K}} H(\theta_{m_K}), \tag{5}$$

とする．ここで，

$$p^*(x^n) = p(x^n|\theta^*_{m_{K_0}}, m_{K_0}), \tag{6}$$

が成り立つことに注意する．

例（混合正規分布）
混合正規分布の確率分布は，

$$p(x^n|\theta_{m_K}, m_K) = \prod_{i=1}^{n} \left[\sum_{k=1}^{K} a_k f(x_i|b_k) \right], \tag{7}$$
$$\theta_{m_K} = \{\{a_k\}_{1 \le k \le K-1}, \{b_k\}_{1 \le k \le K}\}.$$

ここで，

$$\sum_{k=1}^{K} a_k = 1; \ a_k \ge 0, \ 1 \le k \le K, \tag{8}$$

であり，

$$f(x_i|b_k) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left\{ -\frac{(x_i - \mu_k)^2}{2\sigma_k^2} \right\}, \ 1 \le k \le K, \tag{9}$$

である．これは，K について階層構造 (2) を有する．

3 ベイズ決定理論

ベイズ決定理論の立場から最適な予測分布 $\hat{p}(x_{n+1}|x^n)$ を構成する．

・損失関数

$$\log p(x_{n+1}|x^n,\theta_{m_K},m_K)-\log\hat{p}(x_{n+1}|x^n), \quad (10)$$

・リスク関数　$R_n(m_K,\theta_{m_K},\hat{p})$

$$\int p(x^n|\theta_{m_K},m_K)\Big\{\int p(x_{n+1}|x^n,\theta_{m_K},m_K) \times\log\frac{p(x_{n+1}|x^n,\theta_{m_K},m_K)}{\hat{p}(x_{n+1}|x^n)}dx_{n+1}\Big\}dx^n, \quad (11)$$

・ベイズリスク関数　$BR_n(P(\cdot),w(\cdot),\hat{p})$

$$\sum_K P(m_K)\int w(\theta_{m_K}|m_K)R_n(m_K,\theta_{m_K},\hat{p})d\theta_{m_K}, \quad (12)$$

とする．ベイズリスク関数を最小にする $\hat{p}(x_{n+1}|x^n)$ がベイズ最適な予測分布であり，これは，

$$\hat{p}^*(x_{n+1}|x^n)=\sum_K P(m_K|x^n) \times\int p(x_{n+1}|x^n,\theta_{m_K},m_K)w(\theta_{m_K}|m_K,x^n)d\theta_{m_K}, \quad (13)$$

となる [4]．

4 評価基準

本来，予測分布はリスクを最小にするように求めたいが，全てのパラメータに対し，一様に最小にすることは不可能であるために，事前分布で平均化したベイズリスクを最小にすることを考えている．従って，ベイズ基準によって得られた予測分布がどの程度のリスクを持つのかについて評価する必要がある．

$$CR_n\equiv\int p^*(x^n)\log\frac{p^*(x^n)}{\hat{p}^*(x^n)}dx^n,\ \hat{p}^*(x^n) \equiv\sum_K P(m_K)\int p(x^n|\theta_{m_K},m_K)w(\theta_{m_K}|m_K)d\theta_{m_K}, \quad (14)$$

とおくと，

$$R_n(m_{K_0},\theta^*_{m_{K_0}},\hat{p}^*)=CR_{n+1}-CR_n, \quad (15)$$

なる関係が成立するため，$R_n(m_{K_0},\theta^*_{m_{K_0}},\hat{p}^*)$ を評価するためには累積リスク CR_n を評価すればよい．また，本研究においては概収束の意味でのより強い結果を与えるために，まず累積損失 $CL(X^n)$ を評価し，次にその期待値である累積リスク CR_n を評価する．

5 漸近評価

5.1 特定不能性の問題

以下のような例について考える．

真の分布　：$p(x|1)$
学習モデル　：$p(x|a,b,c)=(1-a)p(x|b)+ap(x|c)$

とすると，

$$\{a=0,b=1\}\cup\{a=1,c=1\}\cup\{b=c=1\}, \quad (16)$$

は真の分布と等価となる．このように，真の分布を表すパラメータが次元を持つ集合となるとき，あるパラメータの近傍で，確率分布が変化しない方向が存在する．つまり，局所で特定不能になることからフィッシャー情報行列のランクが縮退し，カルバック距離 $H(\theta_{m_K})$ を 1 点のパラメータの近傍のみでテイラー展開して 2 次近似する従来の漸近展開は適用できない．従って，別の解析法を考える必要がある．

5.2 条件

各 m_K について以下が成り立つと仮定する．

条件 1（事前分布に関する条件）

1. 事前分布 $w(\theta_{m_K}|m_K)$ は無限回微分可能でそのサポートがコンパクト（このサポートを Θ_{m_K} と表す）．
2. $\theta^*_{m_K}$ の近傍がサポート Θ_{m_K} に含まれる．

条件 2（対数尤度に関する条件）

1. $\psi(x,\theta_{m_K})=\log p^*(x)-\log p(x|\theta_{m_K},m_K)$ は θ_{m_K} について解析関数（高階微分可能でテイラー展開がある定義域で絶対収束する関数）であり，Θ_{m_K} を含む複素空間の開集合 $\Theta^c_{m_K}$ 上の解析関数に解析接続できる．
2. 条件

$$\int\sup_{\theta_{m_K}\in\Theta_{m_K}}|\psi(x,\theta_{m_K})|^2p^*(x)dx<\infty, \quad (17)$$

が成立する．

例えば，混合正規分布において混合比が 0 になるような場合には，条件 2 を満たさないことが知られているが，代替的な条件を仮定することでこの問題を回避できることが示されている [8]．ただし，$p(x|\theta_{m_K},m_K)$ が解析関数でないような場合，例えば，混合正規分布において分散が 0 になるような場合はあらかじめ除外すべきことを示唆している．

条件 3（特定可能性）
$\theta_{m_{K_0}}\neq\theta'_{m_{K_0}}$ に対して $p(\cdot|\theta_{m_{K_0}},m_{K_0})\neq p(\cdot|\theta'_{m_{K_0}},m_{K_0})$ が成り立つ．

例えば，指数型分布族の混合分布を考えた場合，パラメータの大小で順序つけを行うことで条件 3 が満たされる．

5.3 漸近評価の詳細

以下，

$$\hat{p}(x^n|m_K)\equiv\int p(x^n|\theta_{m_K},m_K)w(\theta_{m_K}|m_K)d\theta_{m_K}, \quad (18)$$

とする．

補題 1　条件 1，2 が成り立つとき，$K\geq K_0$ において

$$\log\frac{p^*(x^n)}{\hat{p}^*(x^n|m_K)}=\lambda^{(1)}_{m_K}\log n+o(\log n),\ a.s. \quad (19)$$

ここで，$\lambda^{(1)}_{m_K}$ は

$$J(\lambda_{m_K})=\int H(\theta_{m_K})^{\lambda_{m_K}}w(\theta_{m_K}|m_K)d\theta_{m_K}, \quad (20)$$

の原点に最も近い極 $\left(-\lambda^{(1)}_{m_K}\right)$ に対応している[1]．

[1] $J(\lambda_{m_K})$ の極を原点に近いものから順に並べると，

$$0>-\lambda^{(1)}_{m_K}>-\lambda^{(2)}_{m_K}>\cdots.$$

となる [6]．

(証明)

$$h(x,\theta_{m_K}) \equiv \frac{1}{\sqrt{H(\theta_{m_K})}}\Big\{nH_n(\theta_{m_K}) - H(\theta_{m_K})\Big\}, \quad (21)$$

$$H_n(\theta_{m_K}) \equiv \frac{1}{n}\log\frac{p^*(x^n)}{p(x^n|\theta_{m_K},m_K)}, \quad (22)$$

とし,

$$\zeta_n(\theta_{m_K}) \equiv \frac{1}{\sqrt{n}}\sum_{i=1}^{n} h(x_i,\theta_{m_K}), \quad (23)$$

とすると,

$$\begin{aligned}
&\frac{p^*(x^n)}{\hat{p}(x^n|m_K)} \\
&= \int \exp\big(-nH_n(\theta_{m_K})\big)w(\theta_{m_K}|m_K)d\theta_{m_K} && (24)\\
&= \int \exp\Big(-nH(\theta_{m_K}) + \sqrt{nH(\theta_{m_K})}\zeta_n(\theta_{m_K})\Big) \\
&\qquad \times w(\theta_{m_K}|m_K)d\theta_{m_K} && (25)\\
&= \int_0^\infty dt \int \delta(t - H(\theta_{m_K}))w(\theta_{m_K}|m_K) \\
&\qquad \times \exp\big(-nt + \sqrt{nt}\zeta_n(\theta_{m_K})\big)d\theta_{m_K} && (26)\\
&= \int_0^\infty \frac{dt}{n}\int \delta\Big(\frac{t}{n} - H(\theta_{m_K})\Big)w(\theta_{m_K}|m_K) \\
&\qquad \times \exp\big(-t + \sqrt{t}\zeta_n(\theta_{m_K})\big)d\theta_{m_K}, && (27)
\end{aligned}$$

と展開できる. ここで, 相加・相乗平均を用いて

$$\begin{aligned}
&-\frac{1}{2}\big(t + \zeta_n(\theta_{m_K})^2\big) \\
&\qquad \le \sqrt{t}\zeta_n(\theta_{m_K}) \le \frac{1}{2}\big(t + \zeta_n(\theta_{m_K})^2\big), && (28)
\end{aligned}$$

とすることで, $\log p^*(x^n) - \log\hat{p}(x^n|m_K)$ を上下からバウンドすることを考える. ここで,

$$\begin{aligned}
A(\alpha) &\equiv \int_0^\infty \frac{dt}{n}\int \delta\Big(\frac{t}{n} - H(\theta_{m_K})\Big) \\
&\qquad \times w(\theta_{m_K}|m_K)d\theta_K \cdot \exp(-\alpha t), && (29)\\
B &\equiv \frac{1}{2}\zeta_n(\theta_{m_K})^2, && (30)
\end{aligned}$$

とすると,

$$\begin{aligned}
&\log A\Big(-\frac{3}{2}\Big) - B \\
&\qquad \le \log\frac{p^*(x^n)}{\hat{p}(x^n|m_K)} \le \log A\Big(-\frac{1}{2}\Big) + B, && (31)
\end{aligned}$$

となる. まず, 条件 1, 2 が成り立つとき, $\frac{1}{2} \le \alpha \le \frac{3}{2}$ について

$$A(\alpha) = \lambda^{(1)}_{m_K}\log n + o(\log n), \quad (32)$$

が成り立つ [6]. 一方, 条件 1, 2 が成り立つとき, $h(x,\theta_{m_K})$ について平均 0 で分散が有限であることが示されているので [7], 重複対数の法則が適用ができ,

$$\zeta_n(\theta_{m_K}) = O\Big(\sqrt{2\log\log n}\Big), \ a.s. \quad (33)$$

となる. これより,

$$B = o(\log n), \ a.s. \quad (34)$$

を得る. 従って, (31) とから題意を得る. □

補題 1 に対して以下の結果が成立する.

補題 2[6] d_{m_K} をモデル m_K におけるパラメータ数とする. 条件 1, 2 のもとで $K > K_0$ のとき,

$$\lambda^{(1)}_{m_K} \le \frac{d_{m_K}}{2}. \quad (35)$$

補題 3 条件 1〜3 のもとで,

$$\lambda^{(1)}_{m_{K_0}} = \frac{d_{m_{K_0}}}{2}. \quad (36)$$

(証明) 条件 3 より m_{K_0} では $\theta^*_{m_{K_0}}$ においてフィッシャー情報行列が正則である. 従って, $H(\theta_{m_{K_0}})$ を $\theta^*_{m_{K_0}}$ の近傍でテイラー展開し, 平均値の定理を用いると, $c(\theta_{m_{K_0}}) > 0$ が存在して

$$\begin{aligned}
&H(\theta_{m_{K_0}}) \\
&= c(\theta_{m_{K_0}})(\theta_{m_{K_0}} - \theta^*_{m_{K_0}})'I(\theta^*_{m_{K_0}})(\theta_{m_{K_0}} - \theta^*_{m_{K_0}}), && (37)
\end{aligned}$$

とできる. $u = I^{1/2}(\theta_{m_{K_0}} - \theta^*_{m_{K_0}})$ とおくことで, $H(u) = c(u)\|u\|^2$ の形に変形できる. このとき, $J(\lambda_{m_{K_0}})$ の最大の極 $\lambda^{(1)}_{m_{K_0}}$ は $\frac{d_{m_{K_0}}}{2}$ となる [6]. □

定理 $K \ge K_0$ において最小の $\lambda^{(1)}_{m_K}$ をとる K を K^* とおくと, 条件 1, 2 のもとで

$$CL(X^n) = \lambda^{(1)}_{m_{K^*}}\log n + o(\log n), \ a.s. \quad (38)$$

が成り立つ. さらに, 条件 3 が成り立つ場合, $\lambda^{(1)}_{m_{K^*}} \le \frac{d_{m_{K_0}}}{2}$ となる.

(証明)

$$\begin{aligned}
&-\log\hat{p}(x^n) \\
&\quad = -\log\hat{p}(x^n|m_{K^*})P(m_{K^*}) \\
&\qquad -\log\Big\{1 + \sum_{K\neq K^*}\frac{\hat{p}(x^n|m_K)P(m_K)}{\hat{p}(x^n|m_{K^*})P(m_{K^*})}\Big\}, && (39)
\end{aligned}$$

のように展開する. 第 1 項については

$$\begin{aligned}
-\log\hat{p}(x^n|m_{K^*}) &= -\log p(x^n|\theta^*_{m_{K^*}},m_{K^*}) \\
&\quad + \lambda^{(1)}_{m_{K^*}}\log n + o(\log n), \ a.s. && (40)
\end{aligned}$$

とできるので, 第 2 項が概収束の意味で $o(\log n)$ であることを示せば題意を得る. ここで,

$$\begin{aligned}
&\log\frac{\hat{p}(x^n|m_K)}{\hat{p}(x^n|m_{K^*})} \\
&= \log\frac{p(x^n|\theta^*_{m_K},m_K)}{p(x^n|\theta^*_{m_{K^*}},m_{K^*})} \\
&+ \log\frac{p(x^n|\theta^*_{m_{K^*}},m_{K^*})}{\int p(x^n|\theta_{m_{K^*}},m_{K^*})w(\theta_{m_{K^*}}|m_{K^*})d\theta_{m_{K^*}}} \\
&- \log\frac{p(x^n|\theta^*_{m_K},m_K)}{\int p(x^n|\theta_{m_K},m_K)w(\theta_{m_K}|m_K)d\theta_{m_K}}, && (41)
\end{aligned}$$

として考える. $K < K_0$ の場合, 条件 2-2 より, 右辺第 1 項は $-Cn + o(n)$, $a.s.$, 補題 1 より第 2 項は $\lambda^{(1)}_{m_{K^*}}\log n +$

$o(\log n)$, $a.s.$, 第 3 項は $\lambda^{(1)}_{m_K}\log n + o(\log n)$, $a.s.$ となるので,

$$\frac{\hat{p}(x^n|m_K)}{\hat{p}(x^n|m_{K^*})} = O(e^{-n}),\ a.s. \tag{42}$$

を得る. また, $K > K_0$ の場合, 第 1 項は 0 であり, 第 3 項は補題 1 と定義より $\lambda^{(1)}_{m_K}\log n + o(\log n)$, $\lambda^{(1)}_{m_K} \geq \lambda^{(1)}_{m_{K^*}}$, $a.s.$ であるために,

$$\frac{\hat{p}(x^n|m_K)}{\hat{p}(x^n|m_{K^*})} = o(n),\ a.s. \tag{43}$$

を得る. (42), (43) から (39) の第 2 項が概収束の意味で $o(\log n)$ であることが示せた. また, $\lambda^{(1)}_{m_{K^*}} \leq \frac{d_{m_{K_0}}}{2}$ は補題 3 と $\lambda^{(1)}_{m_{K^*}}$ の定義より明らかである. □

証明の流れは特定可能な階層モデルを対象とした Gotoh[3] のものを踏襲しているが, [3] の証明が最尤推定量に対する重複対数の法則を用いて累積損失を評価しているのに対し, 補題 1 のような最尤推定量によらない方法を用いて累積損失を評価している点で大きく異なっている. (例えば混合モデルにおいて真の分布に対し冗長なモデルを仮定すると, 最尤推定量の一致性・漸近正規性が失われ [3] と同等の方法で解析することは不可能である.)

系　条件 1, 2 のもとで

$$CR_n = \lambda^{(1)}_{m_{K^*}}\log n + o(\log n), \tag{44}$$

が成り立つ. さらに, 条件 3 が成り立つ場合 $\lambda^{(1)}_{m_{K^*}} \leq \frac{d_{m_{K_0}}}{2}$ となる.

6　考察

定理においては真の分布を表す最小のモデルを用いたベイズ予測よりも冗長なモデルを用いたベイズ予測による累積損失及び累積リスクが小さくなる可能性も考慮に入れている. ただし, Merhav の結果 [5] を用いると, このような場合は仮に存在したとしても漸近的に無視できることが示される. 以下, この事実について考察する.

補題 4[5]

$$C_n \equiv \sup_w I_w(\Theta; X^n),$$

$$I(\Theta; X^n) \equiv \int w(\theta)d\theta \cdot \int p(x^n|\theta) \times \log \frac{p(x^n|\theta)}{\int p(x^n|\theta)w(\theta)d\theta} dx^n, \tag{45}$$

する. 確率測度を構成する任意の密度関数 $\hat{p}(x^n)$ に対して,

$$\int p(x^n|\theta)\log\frac{p(x^n|\theta)}{\hat{p}(x^n)}dx^n > (1-\epsilon)C_n,\ \forall\epsilon > 0, \tag{46}$$

が, $B \subset \Theta$ を除いて成立する. ここで B は事前密度 $w(\cdot)$ に基づく確率測度 $W(\cdot)$ において

$$W(B) \leq \frac{1 + C_n - I_w(\Theta; X^n)}{\epsilon C_n}, \tag{47}$$

となる集合である. ここで, $n \to \infty$ において $C_n \to \infty$ となり, $C_n - I_w(\Theta; X^n) \to 0$ なるような事前密度 $w(\cdot)$ が存在すれば (47) の右辺は漸近的に 0 に収束する.

条件 1-3 より Clarke らの結果 [2] が適用でき, ジェフェリーの事前分布が漸近的に C_n を達成することを示すことができる. 従って,

$$C_n = \frac{d_{m_{K_0}}}{2}\log n + O(1) \tag{48}$$

なることが示される. 本研究で扱っている対象はこの問題のサブクラスとなっており, 特定可能性を保証することでほとんど全ての場合 $\lambda^{(1)}_{m_{K^*}}$ が $\frac{d_{m_{K_0}}}{2}$ で下界されることが示される.

7　まとめ

階層モデルにおいてモデルの事後確率で重みをつけたベイズ予測分布の累積損失及び累積リスクが漸近的にはこれらを最小にするモデルを用いてベイズ予測を行った場合と等価になることを示した. さらに, それぞれの下界についても考察を行い, 厳密な漸近評価ができることを示した. 今後の課題としては, $\lambda^{(1)}_{m_K}$ についてより強い意味での下界を与えることで, $o(\log n)$ 項の詳細な解析について検討していきたい.

謝辞

本研究を行うにあたり, 数多くの御助言, 御支援を賜りました松嶋研究室の各氏に感謝致します. なお, 本研究の一部は日本学術振興会科学研究費基盤 (C) 一般 (No.15560338) の援助による.

参考文献

[1] B.S.Clarke and A.R.Barron, "Information-theoretic asymptotics of bayes methods," *IEEE Trans. on Inform. Theory*, vol.36, pp.453-471, 1990

[2] B.S.Clarke and A.R.Barron, "Jeffreys' prior is asymptotically least favorable under entropy risk," *J. Statist. Plan. Inference*, vol.41, pp.37-60, 1994

[3] M.Gotoh, T.Matsushima, and S.Hirasawa, "A generalization of B.S.Clarke and A.R.Barron's asymptotics of bayes codes for FSMX sources," *IEICE Trans. Fundamentals*, vol.E81-A, pp.2123-2132, 1998

[4] T.Matsushima, H.Inazumi, and S.Hirasawa, "A class of distortionless codes designed by bayes theory," *IEEE Trans. on Inform. Theory*, vol.37, pp.1288-1293, 1991

[5] N.Merhav and M.Feder, "A strong version of the redundancy-capacity theorem of universal coding," *IEEE Trans. on Inform. Theory*, vol.41, pp.714-722, 1995

[6] S.Watanabe, "Algebraic analysis for non-identifiable learning machines," *Neural Computation*, vol.13, pp.899-933, 2001

[7] 渡辺澄夫,"代数的な特異点をもつ学習モデルの学習誤差と汎化誤差," 電子情報通信学会論文誌, vol.J84-A, pp.99-108, 2001

[8] 渡辺澄夫, 山崎啓介, 青柳美輝,"混合正規分布の特異点の非解析性について," 信学技法 NC, vol.50, pp.41-46, 2004

The 27th Symposium on Information Theory
and Its Applications (SITA2004)
Gero, Gifu, Japan, Dec. 14–17, 2004

Fast Correlation Attackの改良法に関する一考察
A Note on Improvement of a Fast Correlation Attack

細渕 智史 Satoshi HOSOBUCHI *
斉藤 友彦 Tomohiko SAITO *
松嶋 敏泰 Toshiyasu MATSUSHIMA *

Abstract— A stream cipher is a symmetric encryption algorithm which use a pseudo-random sequence(keystream) generator. Nonlinear combiner generator is one of general keystream generators for certain stream ciphers. This generator is composed of several LFSRs and a nonlinear function. Fast correlation attack is one of the popular attacks for keystream generators.

M. J. Mihaljevic, M. P. C. Fossorier, and H. Imai have proposed the improved fast correlation attack which is based on the iterative decoding algorithm. In this paper, we propose an improved method which can be widely applied. In detail, the method is applicapable to attacking multi LFSRs.

Keywords— stream cipher, fast correlation attack, nonlinear combiner generator, belief propagation

1 はじめに

ストリーム暗号とは，鍵を種として鍵系列と呼ばれる擬似乱数系列を生成し，これと平文系列との排他的論理和をとることで暗号文系列を生成する暗号方式である．この鍵系列を生成する擬似乱数生成器の代表的な構成法に，複数の線形フィードバックシフトレジスタ (LFSR) と多入力1出力の非線形関数を組み合わせたものがある．特に，鍵をLFSRの初期状態として与えることで簡易的な擬似乱数であるLFSR系列を生成し，複数のLFSRで得られたそれら系列を非線形関数で結合して出力系列とするものを非線形コンバイナ型乱数生成器と呼ぶ．

このようなLFSRに基づく擬似乱数生成器に対する攻撃手法の1つにFast Correlation Attackがある [1]．これは観測された鍵系列とLFSRのうち1つの出力系列との相関を用いてLFSRの初期状態 (鍵) を推定する攻撃である．このプロセスを繰り返してLFSRを1つずつ順に攻撃していくため，分割統治攻撃とも呼ばれる．

Fast Correlation Attackには数多くの改良法が考案されており，その一つに暗号のモデルを誤り訂正符号のモデルに対応させ，誤り訂正符号における復号アルゴリズムを攻撃に利用したものがある．さらに，復号アルゴリズムを利用した攻撃は大きく2つに分類することができる．一つは線形ブロック符号の復号法を利用したものであり，もう一つは畳み込み符号の復号法を利用したものである．

線形ブロック符号の復号法を利用したものの一つにMihaljevicにより提案されたアルゴリズムがある [2][3][5]．これは誤り訂正符号における閾値復号法もしくは繰り返し型復号法を利用したものである．

従来，これらのアルゴリズムは単体のLFSRを攻撃した場合の性能評価をしており，複数のLFSRを対象とした場合については特に言及されていない．

そこで，本稿ではMihaljevicらにより提案された繰り返し型復号法を利用した攻撃法を複数のLFSRを攻撃対象とした場合について考察を行う．そしてMihaljevicらのアルゴリズムを拡張し，より複数のLFSRを対象とした場合に適したアルゴリズムを提案する．

*〒 169-8555 東京都新宿区大久保 3-4-1 早稲田大学理工学部経営システム工学科．Dept. of Industrial and Management Systems Engineering, Waseda Univ., Okubo 3-4-1, Shinjuku, Tokyo, 169-8555 Japan. E-mail: hoso@matsu.mgmt.waseda.ac.jp

2 従来研究

本章では繰り返し型復号法を利用したFast Correlation Attackについて述べる．Fast Correlation Attackは観測されたある長さの鍵系列とLFSR系列との相関を用いてLFSRの初期状態 (鍵) を推定する攻撃である．この攻撃法は既知平文攻撃であり，平文，暗号文系列が攻撃者にとって既知のため，鍵系列も既知といえる．さらに，擬似乱数生成器の構成なども既知である．

2.1 暗号のモデルと符号のモデルの対応

近年の改良法では，LFSR初期状態 S_0 を情報ビット，LFSRの構成を生成行列，LFSR出力を符号語，非線形関数をBSC(2元対称通信路)，鍵系列を受信語とみなすことで暗号化のモデルを符号化のモデルに対応させ，LFSR初期状態の推定に符号の復号アルゴリズムを用いている．

観測系列長を N とする．攻撃対象のLFSRは長さ L，特性多項式を $f(u)$ とし，LFSR出力系列は X とすると，$X = x_1, x_2, ..., x_N$ はLFSRの初期状態 $X_0 = x_1, x_2, ..., x_L$ を情報ビットとするバイナリ (N,L) パンクチャド符号 C の符号語とみなせる．さらに，非線形関数出力 (鍵系列)$Z = z_1, z_2, ..., z_N$ は受信語とみなせる．また，非線形関数の構成は既知であるが多入力1出力であるため，出力から入力を知ることは出来ない．LFSR出力からLFSR初期状態は確定的に求まるため，この非線形関数の出力から入力を推定するのが攻撃の目的となり，符号のモデルで言えば受信語から符号語を復号することに当たる．

ここで，復号アルゴリズムを適用するため非線形関数を確率モデルと仮定する．具体的には，$Pr(x_i = z_i) = 1-p$ として，非線形関数は誤り率 p のBSCとする．非線形関数は既知のため，p は既知である．推定に使われる繰り返し型復号法には，BP-BF(Belief Propagation based Bit Flipping)，BP(Belief Propagation) 等がある．ただし，一般的に攻撃対象にしているLFSRでは初期状態の全ビットを復号によって求めることは復号性能からいって困難であるため，全数探索を併用する．LSFRの L ビットのうち初めの B ビットを全数探索で求め，残りの $L-B$ ビットを復号によって求めることとする．

次節では，繰り返し型復号法を用いたFast Correlation Attackのプロセスを詳しく説明する．

2.2 繰り返し型復号法を用いたFast Correlation Attack

2.2.1 LFSRの式表現

S_i を i 時点でのLFSR内部状態とし，下式で表す．

$$S_i = \begin{bmatrix} x_{i+L} \\ \vdots \\ x_{i+1} \end{bmatrix} \quad (1)$$

LFSR内部状態の遷移行列をAとすると，下式が成り立つ．

$$S_i = AS_{i-1}, \quad i = 1, 2, ... \quad (2)$$

$$S_i = A^i S_0, \quad i = 1, 2, ... \quad (3)$$

LFSR の特性多項式を $f(u) = 1 + \sum_{i=1}^{L} b_i u_i$ とすると，A を以下のように記述できる．

$$A = \begin{bmatrix} b_1 & b_2 & b_3 & \cdots & b_{L-1} & b_L \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & \vdots \\ \vdots & \vdots & \vdots & \cdots & 0 & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix} \quad (4)$$

2.2.2 パリティ検査式の構成

まずは LFSR の構成から生成行列を導出する．$A_1^{(i)}$ を A の i 乗の第 1 行，I_L を L 行 L 列の単位行列とすると，$[x_1\, x_2 ... x_N] = [x_1\, x_2 ... x_L]G$ を満たす生成行列 G が下式のように構成できる．

$$G = [I_L A_1^{(1)} A_1^{(2)} ... A_1^{(N-L)}] \quad (5)$$

すると，生成行列 G からパリティ検査行列 $H = [P^T\, I_{N-L}]$ を導出できる．ただし，P^T は下式．

$$P^T = \begin{bmatrix} A_1^{(1)} \\ A_1^{(2)} \\ \vdots \\ A_1^{(N-L)} \end{bmatrix} \quad (6)$$

また，復号処理にはこのパリティ検査式から [定義 1] で導出されるパリティ検査式集合を使う．

[定義 1] $n = L+1, ..., N$ と $w, 1 \le w \le W$ に対して，以下のプロセスでビット n に関係するパリティ検査式集合 Ω_n を構成する．

- パリティ検査行列の $n-L$ 行目と他の w 行の和を計算する．
- これらの和の中から，ビット $i = B+1, B+2, ..., L$ が全て 0 となるものを Ω_n として記録する．

2.2.3 全数探索の仮定

LFSR の L ビットのうち，全数探索に当てる B ビットに適当な値を仮定し，その全 2^B パターン各々に対して復号処理を行う．

2.2.4 復号処理

得られたパリティ検査式集合を用いて復号処理を行う．ここでは，復号アルゴリズムに BP を使った場合の攻撃プロセスを述べる．

[初期化プロセス] $m \in \Omega_n$ をビット n に関係するパリティ検査式につけたインデックスとする．$\hat{x}_n$ を x_n の推定値とする．f_n^u は $Pr(z_n \mid x_n = u)$ であり，$z_n = 0$ ならば $f_n^0 = 1-p, f_n^1 = p$，$z_n = 1$ ならば $f_n^0 = p, f_n^1 = 1-p$ とする．q_{mn}^u は m 以外のパリティ検査式により得られた，ビット n の値が u となる確率である．

その上で，以下のように初期化する．

$$\hat{x}_n = z_n \quad (7)$$

$$q_{mn}^0 = f_n^0, q_{mn}^1 = f_n^1 \quad (8)$$

[ステップ 1] $\delta q_{mn} = q_{mn}^0 - q_{mn}^1$ として各 m, n と $u = 0, 1$ に対して以下の式を計算する．ただし，$\omega_n(m)$ はビット n に関係するパリティ検査式 m を意味する．

$$\delta r_{mn} = \prod_{n' \in \omega_n(m)} \delta q_{mn'} \quad (9)$$

$$r_{mn}^u = (1/2)(1 + (-1)^u \delta r_{mn}) \quad (10)$$

ここで，r_{mn}^u はビット n の値を u に固定し，他のビットの値が確率 $\{q_{mn'} : n' \in \omega_n(m) \setminus n\}$ で得られるとした際のパリティ検査式 m が満たされる確率である．

[ステップ 2] 各 m, n と $u = 0, 1$ に対して下式のように更新する．ただし，α_{mn} は $q_{mn}^0 + q_{mn}^1 = 1$ となるように選択される値である．

$$q_{mn}^u = \alpha_{mn} f_n^u \prod_{m' \in \Omega_n \setminus m} r_{m'n}^u \quad (11)$$

各 n と $u = 0, 1$ に対して下式のように更新する．ただし，α_n は $q_n^0 + q_n^1 = 1$ となるように選択される値である．

$$q_n^u = \alpha_n f_n^u \prod_{m \in \Omega_n} r_{mn}^u \quad (12)$$

[ステップ 3] $q_n^1 > 0.5$ ならば $\hat{x}_n = 1$，$q_n^1 \le 0.5$ ならば $\hat{x}_n = 0$ として推定系列 $\hat{X} = [\hat{x}_n]$ を生成する．$H\hat{X} = 0$ ならば $\hat{X}$ を復号結果として出力．そうでなければ，ステップ 1 へ戻る．復号結果を得られないまま定めておいた繰り返し回数を終えたら，攻撃終了．

2.2.5 最終チェック

復号結果 $\hat{X}$ が本当に正しいかどうかをチェックする．

復号フェイズで得られた系列 $\hat{x}_{L+1}, \hat{x}_{L+2}, ..., \hat{x}_N$ を用いて情報ビットの推定値 $\hat{X}_0 = \hat{x}_1, \hat{x}_2, \hat{x}_L$ を構成し，$\hat{X}_0$ から得られる符号語 $\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_N$ で下式を計算する．

$$S = \sum_{n=1}^{N} \tilde{x}_n \oplus z_n \quad (13)$$

$S \le T$ ならば (T は閾値)，$\hat{X}_0$ を LFSR 初期状態の正しい推定値として出力する．そうでなければ，全数探索の仮定に戻ってやり直す．

2.3 評価基準

攻撃法は LFSR 単体への攻撃性能と攻撃にかかる計算量で評価される．攻撃性能の評価には，Limit Noise が用いられる．これは，非線形関数を BSC と仮定した場合に，誤り率がこれより小さければ攻撃が可能であるとする評価基準である．

この基準は Mihaljevic らによる LILI-128 への攻撃 [5] 等の実際の暗号に対する攻撃でも，非線形関数を誤り率が Limit Noise の BSC と仮定した場合にある計算量で攻撃が可能というように使われている．

3 複数の LFSR への攻撃

従来法は単体の LFSR への攻撃に主眼を置いた手法であった．本章ではこれを応用し，複数の LFSR を攻撃する 2 つの手法を提案する．

3.1 準備

3.1.1 非線形関数が BSC と仮定できない場合

LFSR の出力系列はそれ自体が簡易的な擬似乱数系列であり，{0, 1} の生起確率は等しいとみなせる．しかし，非線形関数のとり方によっては出力の {0, 1} の生起確率は必ずしも等しくならないし，等しくなくても構わない．このような非線形関数の場合，これを確率モデルとみなす場合に 0 から 1，1 から 0 への誤り率が異なるため BSC にならず BAC(2 元非対称通信路) になる．しかし，従来法 [3] に少し手を加えれば，このような対象への攻撃を可能とするアルゴリズムを記述することができる．

また，同様のアルゴリズムを多次元で行うことにより，複数の LFSR 出力の集合と非線形関数出力との多次元の相関を使って複数の LFSR を同時に攻撃するようなアルゴリズムを記述することも可能である．

まず，攻撃対象の非線形関数について考察しておく．

3.1.2 非線形関数の無相関性

非線形コンバイナ型乱数生成器の非線形関数には，T. Siegenthaler によって提案された無相関な非線形関数 [4] がよく用いられている．このような非線形関数では，出力 (すなわち鍵系列) と入力 (すなわち LFSR 出力系列) に相関がない．$X^{(i)}$ を i 番目の LFSR の出力，LFSR の個数を R として，これを下式のように表現できる．

$$p^{(i)} = Pr(z_n \neq x_n^{(i)}) = 0.5 \tag{14}$$

しかし，このような非線形関数でも，複数の入力の集合と出力を見ると相関が見られる場合が考えられる．

$$Pr(z_n = j \mid x_n^{(i_1)} = j_1, x_n^{(i_2)} = j_2, ..., x_n^{(i_R)} = j_R) \neq 1/|R| \\ , j, j_1, j_2, ..., j_R \in \{0, 1\} \tag{15}$$

d 個までの入力集合と出力とが相関を持たない関数を d 次の無相関な関数という．

3.1.3 複数の LFSR を順番に攻撃する手法

Fast Correlation Attack は分割統治攻撃とも呼ばれ，LFSR を一つずつ順に攻撃していくアルゴリズムである．このとき，いくつかの LFSR を攻撃した後，残りの LFSR を攻撃する際に既に終えた分の攻撃結果を利用することは十分に考えられる．

ここで，攻撃済みの LFSR の出力系列は既知とみなせる．非線形関数の入力 R 個のうち d 個は既知であるとすると，これら d 値の入力パターンに対して，ビットごとに遷移確率 (誤り率)$Pr(z_n \mid x_n^{(i)}, x_n^{(j_1)}, x_n^{(j_2)}, ..., x_n^{(j_d)})$ を求めることができる．

このように，攻撃済みの LFSR 出力が既知の上での事後確率計算をすることで，攻撃済みの情報を新たな攻撃に利用することができる．

また，対象が d 次の無相関な非線形関数の場合でも，全数探索を行う等して d 個の LFSR を攻撃できれば，残りの LFSR を連続して攻撃することが可能である．

次節からはさらに推し進めて，複数の LFSR を同時に攻撃する手法を提案する．

3.2 多次元の相関を利用した攻撃 (確率的モデルを仮定した場合)

d 次の無相関な非線形関数でも，$d+1$ 個以上の LFSR 系列集合と非線形関数出力との間にならば多次元の相関を見出せる．本節では，このような多次元の相関を利用した攻撃手法を提案する．

ここでは，復号アルゴリズムに多次元の BP を用いた場合の攻撃プロセスを述べる．

パリティ検査式集合の構成や，全数探索の仮定，初期化プロセスは前章と同様である．復号プロセスを以下のように変更する．同時に攻撃する LFSR の個数を R，攻撃対象の LFSR の出力集合を $X_n = (x_n^{(i_1)}, x_n^{(i_2)}, ..., x_n^{(i_R)})$ とする．

複数の LFSR 出力を扱う場合でも，非線形関数が既知であるため，遷移確率 (誤り率)$Pr(z_n \mid x_n^{(i)}), i \in |R|$ は既知である．

また，LFSR の性質より $Pr(x_n^{(i)}) = 0.5$ ゆえに $Pr(X_n) = 1/2^R$ とみなせるので，非線形関数出力の 0,1 の生起確率 $Pr(z_n)$ も既知となり，下式を求められる．

$$Pr(X_n = U \mid z_n) = Pr(X_n)Pr(z_n \mid X_n = U)/Pr(z_n) \\ U = (u^{(1)}, u^{(2)}, ..., u^{(R)}) \quad u^{(i)} \in \{0, 1\} \tag{16}$$

これを q_{mn}^U の初期値とする．

[ステップ 1] 下式で r_{mn}^U を更新する．

$$r_{mn}^U = \alpha_{mn} \sum_{\sum \hat{x}_n^{(i)} = u^{(i)}, i \in |R|} \prod_{n' \in \omega_n(m)} q_{mn'}^{\hat{X}_{n'}} Pr(z_{n'} \mid \hat{X}_{n'}) \tag{17}$$

ただし，α_{mn} は $\sum r_{mn}^U = 1$ となるように選択される値である．

[ステップ 2] 各 m, n, U に対して下式のように更新する．ただし，α_{mn} は $\sum_U q_{mn}^U = 1$ となるように選択される値である．

$$q_{mn}^U = \alpha_{mn} \prod_{m' \in \Omega_n \setminus m} r_{m'n}^U \tag{18}$$

各 n, U に対して下式のように更新する．ただし，α_n は $\sum q_n^U = 1$ となるように選択される値である．

$$q_n^U = \alpha_n Pr(z_n \mid X_n = U) \prod_{m \in \Omega_n} r_{mn}^U \tag{19}$$

[ステップ 3] 最も大きい q_n^U に対して，$\hat{X}_n = U$ として推定系列 $\hat{X} = [\hat{X}_n]$ を生成する．

$H\hat{X} = 0$ ならば $\hat{X}$ を復号結果として出力．そうでなければ，ステップ 1 へ戻り，アルゴリズムを繰り返す．復号結果を得られないまま定めておいた繰り返し回数を終えたら，攻撃終了．最後に，復号結果 $\hat{X}$ が本当に正しいかどうかを従来法と同様にチェックし，正しくなければ全数探索の仮定をやり直して繰り返す．

3.3 多次元の相関を利用した攻撃 (確定的モデルを仮定した場合)

本節では，前節のように攻撃に多次元の相関を利用するときの，特別な場合を考える．

全ての LFSR 出力の集合と非線形関数出力の多次元との相関を使って擬似乱数生成器に含まれる全ての LFSR を同時に攻撃する場合，非線形関数の構成は攻撃者に既知であるため，入力集合から出力は確定的に求まる．これをネットワークモデルとしてみなすことができる．既知の $z_n = \{0, 1\}$ より，それに対応する取りうる X_n はネッ

トワークで繋がっているもののみであり，候補を絞り込むことができる．

ここでは，復号アルゴリズムに多次元の BP-BF を用いた場合について述べる．

[初期化プロセス] $\hat{X}_n = (\hat{x}_n^{(1)}, \hat{x}_n^{(2)}, ..., \hat{x}_n^{(R)})$ を各レジスタのビット n の推定値，$\hat{x}_{mn}^{(1)}, \hat{x}_{mn}^{(2)}, ..., \hat{x}_{mn}^{(R)}$ をパリティ検査式 m を使用しないでビット n を推定した値とする．

非線形関数のモデルに確率が介在しないため，X_n が $z_n = \{0, 1\}$ に繋がるどの値をとるかについて事前に得られる情報はない．よって，各 m, n, i に対して初期値はとりあえず下式のようにしておく．

$$\hat{x}_n^{(i)}, \hat{x}_{mn}^{(i)} = z_n \tag{20}$$

[ステップ 1] m 以外のパリティ検査式で推定した値を検査式 m に入れて成立するかどうかで評価する．各 n, i に対して下式を計算する．

$$\sigma_n^{(i)}(m) = \sum_{n' \in \omega(in)} \hat{x}_{mn'}^{(i)} \tag{21}$$

全ての n, m, i で $\sigma_n^{(i)}(m) = 0$ ならば，$\hat{X}_n$ を出力する．決められた繰り返し回数を終えていたら，ステップ 3 へ．

[ステップ 2] 各 n, i で以下のように推定値を更新する．

$$\sum_{m}^{\Omega_n^{(i)}} \sigma_n(m) \leq |\Omega_n^{(i)}|/2 \tag{22}$$

ならば，$\hat{X}_n^{(i)}$ のビットを反転させる．

$$\sum_{m'}^{\Omega_n^{(i)} \backslash m} \sigma_n(m') \leq |\Omega_n^{(i)} \backslash m|/2 \tag{23}$$

ならば，$\hat{X}_{mn}^{(i)}$ のビットを反転させる．

各 n, i で更新を終えたらステップ 1 へ戻る．ただし，ビット反転が 1 度も起こらなければ，ステップ 3 へ．

[ステップ 3] いずれかの次元で $x_n^{(i)}$ の推定に失敗しても，非線形関数入出力のネットワークを参照することで，各 n で $x_n^{(i)}$ の推定済みのビットから推定に失敗したビットの値，もしくはそのビットの $\{0, 1\}$ の生起確率を知ることが出来る．

ビット値を得られた場合はその値に固定，$\{0, 1\}$ の生起確率に偏りがあった場合はそれを初期値として与えることでこれらネットワークから得られた情報を利用し，再度 BP-BF のアルゴリズムを行う．

4 シミュレーションによる評価

多次元 BP-BF を用いた攻撃を例にとり，シミュレーションを行った．

ここでは攻撃対象として，下式の非線形関数を持つ Geffe 型乱数生成器を用意した．

$$Z = X^{(1)}X^{(2)} + X^{(2)}X^{(3)} + X^{(3)} \tag{24}$$

$X^{(1)}, X^{(3)}$ と Z には相関があるので多次元の BP-BF で推定を行うことが出来るが，$Pr(x_n^{(2)} = z_n) = 0.5$ のため，2 番目の LFSR の推定は失敗する．

ただし，$x_n^{(1)} = 0, x_n^{(3)} = 1$ のときは $x_n^{(2)} \neq z_n$，$x^{(1)} = 1, x^{(3)} = 0$ のときは $x_n^{(2)} = z_n$ とネットワークを参照して確定的に求まる．大まかに言って，約半分のビットが求まると考えられるだろう．これらのビット値を固定して BP-BF を行うことで $X^{(2)}$ の推定を続行できる．

なお，シミュレーションの結果 $L = 40, B = 26, N = 1024$ の条件下で攻撃が成功することを確認している．

5 考察

確率的モデルを仮定した，多次元の相関を利用した攻撃法には，計算量にアドバンテージがあると言える．特にある次数の無相関性を持つ非線形関数など，確率モデルと置いた際に大きな誤り率を持ってしまうような関数を攻撃するためには全数探索を LFSR 全体に用いる必要があるが，多次元の相関を使えばこの全数探索に当てるビット数を削減することが出来る．

確定的モデルを仮定した場合の攻撃法は，非線形関数をモデル化した際に確率的な偏りを見出しにくい場合でも攻撃済みの LFSR の情報を用いることのできるアルゴリズムである．

6 まとめ

本稿では繰り返し型の復号法を利用した Fast Correlation Attack を応用し，複数の LFSR，もしくは擬似乱数生成器全体といった従来より広い対象へ適用できる，多次元の相関を利用した攻撃手法を考案した．推定に非線形関数の情報をより効果的に用いることで，本手法は従来法をそのまま適用できないある次数の無相関な非線形関数にも適用できる上，さらに，計算量の削減を見込むことができる．

ただし，本稿では規模の小さなモデルでしかシミュレーションを行っていない．これらのアルゴリズムがその他の様々な非線形関数，特により規模の大きい，すなわち入力変数の多い非線形関数に対しても攻撃が可能であるかどうかの評価を今後の課題としたい．

謝辞

本研究を行うにあたり，数多くの御助言，ご支援を賜りました松嶋研究室の各氏に感謝いたします．なお，本研究の一部は日本学術振興会科学研究費基盤 (C) 一般 (No.15560338) の援助による．

References

[1] Willi Meier, Othmar Staffelbach, "Fast correlation attacks on certain stream ciphers," Journal of Cryptology, Vol.1 No.3, pp.159-176,1989.

[2] Miodrag J. Mihaljevic, Marc P. C. Fossorier, Hideki Imai, "A Low-Complexity and High-Performance Algorithm for the Fast Correlation Attack", FSE 2000, 196-212.

[3] Miodrag J. Mihaljevic, Marc P. C. Fossorier, Hideki Imai, "On Decoding Techniques for Cryptanalysis of Certain Encryption Algorithms," IEICE Transactions on Fundamentals, Vol.E84-A, pp.919-930, April 2001.

[4] T. Siegenthaler, "Correlation-Immunity of nonlinear combining functions for cryptographic applications," IEEE Transactions on Information theory Vol.IT-30, No.5, pp.776-780,1984.

[5] Miodrag J. Mihaljevic, Marc P. C. Fossorier, Hideki Imai, "Fast Correlation Attack Algorithm with List Decoding and an Application," FSE 2001, 196-210.

5. 各ユーザー User_i は，4で送られてきた情報から多項式補間により以下を計算，記憶する．

$$u_i = \left\{F^1(0,i,\cdots,x_t),\cdots,F^{c+1}(0,i,\cdots,x_t)\right\}. \quad (36)$$

6. 鍵共有ユーザー集合

$$\mathcal{M}_j = \{\mathrm{User}_{j_1}, \mathrm{User}_{j_2}, \cdots, \mathrm{User}_{j_t}\} \quad (37)$$

において，それぞれのユーザー User_{j_i} は，センターから送られてきた情報から計算した $t-1$ 変数の多項式の集合となる u_{j_i} に対し，

$$(x_2,\cdots,x_t) = (j_1,\cdots,j_{i-1},j_{i+1},\cdots,j_t) \quad (38)$$

として値を計算する．

7. 鍵共有ユーザー集合 $\mathcal{M}_j$ の共有鍵を，以下のように6で計算した値を連接したものとする．

$$s_j = F^1(0) \oplus \cdots \oplus F^{c+1}(0) \in \mathcal{S}. \quad (39)$$

ここで，$\oplus$ は連接を表す記号とし，

$$F^i(0) = F^i(0,j_1,\cdots,j_t),\ i = 1,\cdots,c+1 \quad (40)$$

とした． □

このプロトコルにおけるユーザーの記憶容量は定理4.1の下界を達成する．各センターが生成する $t+1$ 変数の多項式 $f_i^j(x_0,x_1,\cdots,x_t)$ は，x_0 に関して $d-1$ 次多項式となっているので，d 個のセンターの情報を集めない限り，多項式を特定することができない．また，$k+l$ 人のユーザーが結託すると，式(36)の中の l 個の多項式が特定できるので，鍵の情報が部分的に洩れていることになる．よって次の定理が成り立つ．

定理 4.2 上記のプロトコル *2* は，(c,d,k,t) 分散鍵配送方式を実現する． □

5 考察

まず，(k,t) 鍵配送方式と (c,d,k,t) 分散鍵配送方式における，ユーザーのメモリ量の下界を比較し，(c,d,k,t) 分散鍵配送方式が有効になる場合を考察する．

例 5.1 共有鍵の集合は，鍵共有ユーザー集合 $\mathcal{M}_j$ によらず，全て同じ集合 $\mathcal{S}$ で，共有鍵，ユーザーが保持している情報に関する確率分布は全て一様分布だと仮定する．秘密鍵の長さが *128* ビット，すなわち，$|\mathcal{S}| = 2^{128}$ とする．この場合の $(1,1,4,2)$ 分散鍵配送方式，$(5,2)$ 鍵配送方式，$(1,1,4,10)$ 分散鍵配送方式，$(5,10)$ 鍵配送方式に対するユーザーのメモリ量はそれぞれ，*704* ビット，*768* ビット，*173888* ビット，*256256* ビットとなる．また，式 *(15)* を安全性の尺度として用いると，攻撃者側が求めたい共有鍵 s_j の候補数として解釈でき，この値が大きいほど共有鍵の特定が困難となり，安全だといえる．

この例の場合，任意の4人以下のユーザーが結託すると，式 *(15)* の値が，全ての方式で 2^{128} となり，任意の5人のユーザーが結託すると，それぞれ 2^{64}，2^{128}，2^{64}，2^{128}，任意の6人以上のユーザーが結託してしまうと，全ての方式で1となる． □

この例から，安全性の尺度とした2つの値 2^{128} と 2^{64} を比較して，安全性に関して両者に差はない，あるいは，2^{64} という値が十分安全であるという基準に達していると考えられるのであれば，提案した (c,d,k,t) 分散鍵配送方式のほうが，ユーザーの記憶容量削減という意味で有効であるといえる．また，鍵共有ユーザー集合のサイズが大きく，結託可能なユーザー数が少ないほど，(c,d,k,t) 分散鍵配送方式がより有効となる．

次に，(k,t) 鍵配送方式，(c,d,k,t) 分散鍵配送方式，および $(k+1,t)$ 鍵配送方式の関係について考える．ここで，式(5)と式(19)の下界を，それぞれ $\alpha(k,t)$，$\beta(c,k,t)$ とすると，$\beta(c,k,t)$ は以下のような再帰的な式に書き換えられる．

$$\beta(c,k,t) = \frac{c}{c+1}\beta(c-1,k,t) + \frac{1}{c+1}\alpha(k+1,t),$$
$$c = 0,1,2,\cdots. \quad (41)$$

ここで，$\beta(-1,k,t) = 0$ とした．以上より，

$$\alpha(k,t) \le \beta(c,k,t) \le \alpha(k+1,t),\ \ c = 0,1,2,\cdots \quad (42)$$

が成り立ち，(c,d,k,t) 分散鍵配送方式においてユーザーが保持する情報量は，$0 \le c$ に対し，(k,t) 鍵配送方式と $(k+1,t)$ 鍵配送方式においてユーザーが保持する情報量の間に値をとり，c を大きくとることで，$(k+1,t)$ 鍵配送方式によるユーザーの情報量に近づくことがわかる．これにより，(c,d,k,t) 分散鍵配送方式において，c を大きくとることで，ユーザーが保持する情報量と $k+1$ 人までのユーザーの結託に対する安全性が $(k+1,t)$ 鍵配送方式とほぼ同等で，更に $k+2$ から $k+c$ 人までのユーザーの結託に対しても，ある程度の安全性が確保できる鍵配送方式となる．

6 まとめ

本稿では，ランプ型分散鍵配送方式である (c,d,k,t) 分散鍵配送方式のモデルを，しきい値型鍵分散方式である (k,t) 鍵配送方式のモデルの一般化として定義し，そのモデルのもとでユーザーが必要なメモリ量の下界を導出した．さらに，その下界を達成するプロトコルを提案し，ユーザーが必要なメモリ量の面での有効性を示した．

7 謝辞

本稿の執筆にあたり，数多くの御助言，御支援を賜りました，早稲田大学理工学部経営システム工学科松嶋研究室，平澤研究室，両研究室の先輩方，同輩達に心より感謝申し上げます．なお，本研究の一部は日本学術振興会研究費基盤研究 (C)(No.15560338) の援助による．

参考文献

[1] R. Blom, "An optimal class of symmetric key generation systems," Proceeding of EUROCRYPT '84, Lecture Notes in Computer Science, Vol.209, pp.335-338, 1984

[2] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly secure key distribution for dynamic conference," Proceeding of CRYPT '92, Lecture Notes in Computer Science, Vol.740, pp.471-486, 1993.

[3] C. Blundo, P. D'Arco and C. Padró, "A ramp model for distributed key distribution schemes," Discrete Applied Mathematics, vol.128, pp.47-64, 2003.

[4] P. D'Arco, "On the distribution of a key distribution center," Proceeding of ICTCS '01, Lecture Notes in Computer Science, Vol.2202, Springer, Berlin, 2001, pp.357-369.

[5] G. Hanaoka, T. Nishioka, Y. Zheng and H. Imai, "Optimal unconditionally secure ID-based key distribution shceme for large-scaled networks," IEICE Trans. Fundamentals, vol.E84-A, No.1, pp.222-230, 2001.

[6] 松本勉，今井秀樹，"暗号鍵を通信なしで共有する方法，" 電子情報通信学会論文誌，Vol.J71-A，No.11，pp.2046-2053，1988.

[7] D. R. Stinson, "On some methods for unconditionally secure key distribution and broadcast encryption," Designs Codes Cryptography, vol.12, No.3, pp.215-243, 1997.

The 27th Symposium on Information Theor and Its Applications (SITA2004)
Gero, Gifu, Japan, Dec. 14-17, 2004

ID情報に基づくランプ型分散鍵配送方式について
Non-perfectly secure identity based decentralized key distribution system

吉田 隆弘* Takahiro YOSHIDA
松嶋 敏泰* Toshiyasu MATSUSHIMA
平澤 茂一* Shigeichi HIRASAWA

Abstract— In this paper, we consider a ramp model for key distribution scheme. In the ramp model, the required resources can be reduced at the cost of a secerity degradation which depends on the size of users. We define a ramp model for key distribution scheme, show lower bounds on the size of the piece of a user's information and design a ramp model for key distribution scheme.

Keywords— key predistribution system, ramp scheme

1 はじめに

大規模なネットワークにおいて，不特定多数のユーザー同士で秘密通信を行う場合に必要となるのが鍵配送方式である．本稿では，情報量的に安全なユーザーID情報に基づく鍵配送方式について考える．この鍵配送方式は，ネットワークにおいて，同一の鍵を共通するユーザーのグループの大きさが全てt人で固定されており，鍵配送センターと呼ばれている信頼のおける機関からネットワーク内の全ユーザーへ，安全な通信路を用いて各ユーザーが属するグループの共有鍵に関する情報が送信され，その情報を受け取った各ユーザーは，ユーザー同士での通信を行わずに，その情報と公開情報であるユーザーのID情報から，自分が属しているグループの共有鍵を個別に計算することができる方式である．

従来の方式は，任意のk人以下のユーザーが結託をしても，そのユーザーが属していないグループの共有鍵に関する情報は一切漏れないが，任意の$k+1$人以上のユーザーが結託してしまったら，全ての共有鍵に関する情報が完全に求まる方式となっている．本稿では，このような鍵配送方式を，オフラインのしきい値型鍵配送方式と呼ぶことにする．これまでに，この方式について，各ユーザーが共有鍵を計算するために保持すべき情報量（メモリ量）の下界が求められており[2]，この下界を達成する最適な鍵配送プロトコルとして，線形代数を利用した構成法が良く知られている[1][2][6]．このような鍵配送方式では，kとtが大きくなると，ユーザーが保持すべき情報量も増加してしまうので，通信する必要のないユーザー間の鍵配送を行わないことで，ユーザーのメモリ量の削減を図っている[5]．また，信頼できる鍵配送センターの存在が仮定されているので，センターの信頼性の確保が問題となり得るが，鍵配送センターを分散化することで対処することができる[6]．

一方，ユーザー間で秘密通信を行うときに，複数の鍵配送センターへ共有鍵の要求メッセージを各ユーザーが送信し，そのユーザーグループに応じた情報を各鍵配送センターがユーザーへ送信するというオンラインの分散鍵配送方式がある[3][4]．この方式では，鍵配送センターを複数設置しており，まず最初に，鍵配送センター間で安全な通信路を用いて通信を行う．次に，各ユーザーが，任意の鍵配送センターを一定数選んで共有鍵の要求ができるので，鍵配送センターの不正，機能停止，あるいは鍵配送センターとユーザー間が通信不能になった場合に対処できるようになっている．また，このモデルの一般化として，オンラインのランプ型分散鍵配送方式も提案されている[3]．

本研究では，上記のオンラインのランプ型分散鍵配送方式に対して，オフラインのランプ型分散鍵配送方式を考える．この方式は，ネットワーク内の任意のk人のユーザーと，$g-1$人以下の鍵配送センターが結託しても，そのユーザーが属していないグループの共有鍵の情報は全く得られず，任意の$k+l$人$(1 \leq l \leq c)$のユーザーと，$g-1$人以下の鍵配送センターが結託すると，lが大きくなるにつれて，その情報が部分的に得られるようになり，任意の$k+c+1$人のユーザーと，$g-1$人以下の鍵配送センターが結託，または，任意のg人の鍵配送センターが結託した場合に，ネットワーク内における全ての共有鍵の情報を完全に求めることができるような方式である．また，この方式では，鍵配送センター間および鍵配送センターと各ユーザー間で通信を行う必要がある．

本稿では，ランプ型分散鍵配送方式モデルを情報理論に基づいて定義し，そのモデルのもとで各ユーザーが共有鍵を求めるために保持すべきメモリ量の下界を導出し，その下界を達成するランプ型分散鍵配送方式のプロトコルの提案と，その性能解析を行う．

2 鍵配送方式

本稿で考えるオフラインの鍵配送方式（以下では単に鍵配送方式と呼ぶ．）は，各々がID（識別子／名前）を持つm人の鍵配送センター（以下では単にセンターと呼ぶ．）の集合

$$\mathcal{N} = \{\mathrm{Center}_1, \mathrm{Center}_2, \cdots, \mathrm{Center}_m\} \tag{1}$$

と，n人のユーザー（利用者）の集合

$$\mathcal{M} = \{\mathrm{User}_1, \mathrm{User}_2, \cdots, \mathrm{User}_n\} \tag{2}$$

からなるネットワークにおいて，ネットワーク内の任意のt人の鍵共有ユーザー集合$\mathcal{M}_h \subseteq \mathcal{M}, (h = 1, 2, \cdots)$が，同一の鍵を個別に生成し，共有する方式である．この鍵配送方式では，まず初期設定として，任意のd人のセンターが，ネットワーク内において公開されている情報であるセンターのIDと，非公開の情報である乱数を利用して，各センターに送る情報を生成し，安全な通信路を用いて送信する．ここで，一般性を失うことなく，初期設定で利用されるd人のセンターを$\{\mathrm{Center}_1, \mathrm{Center}_2, \cdots, \mathrm{Center}_d\}$と仮定する．次に，各ユーザーは，任意のセンターを$d$人選択し，そのセンターから情報を受け取る．このとき，選択された各センターは，初期設定で受け取った情報と，ネットワーク内において公開されている情報であるユーザーのIDを利用して，各ユーザーへ送る情報を生成し，安全な通信路を用いて送信する．また，各ユーザーは，センターから送られてきた全ての情報から保持しておく情報を生成，記憶することで，ユーザー間での通信を行わずに，記憶してある情報と，秘密通信を行いたいグループ内のユーザーIDから，個別にそのグループの共有鍵を生成する．ここで，$1 \leq i, j \leq m$と$1 \leq l \leq n$に対し，センターCenter_iからCenter_jへ送られる情報の集合を$\mathcal{R}_{i,j}$，センターCenter_iからユーザーUser_lへ送られる情報の集合を$\mathcal{U}_{i,l}$，ユーザーUser_lが記憶しておく情報の集合を$\mathcal{U}_l$，ユーザー集合$\mathcal{M}_h$における共有鍵の集合を$\mathcal{S}_h$とし，それぞれの集合の中に値をとる確率変数を$R_{i,j}$，$U_{i,l}$，U_l，S_hとする．また，それらの確率分布$P_{R_{i,j}}$，$P_{U_{i,l}}$，P_{U_l}，P_{S_h}は，センター間で通信される情報と，各センターから各ユーザーへ送る情報の生成アルゴリズムと，その際に用いる乱数によって決まる同時確率分布$P_{R_{1,2}\cdots R_{m,m-1}U_{1,1}\cdots U_{m,n}U_1\cdots U_nS_1\cdots S_Z}$の周辺確率分布として与えられる．ここで，$Z$は非負の整数とする．

*早稲田大学理工学部経営システム工学科，〒169-8555 新宿区大久保3-4-1, Dep. of Industrial and Management Systems Engineering, Waseda University, 3-4-1 Ohkubo, Shinjuku-ku, Tokyo 169-8555, E-mail:takahiro@matsu.mgmt.waseda.ac.jp

3 従来研究 ～しきい値型鍵配送方式～

しきい値型配送方式は，センターの集合の大きさが $m=1$ のときの鍵配送方式なので，初期設定で行うセンター間の通信が必要なくなり，ネットワーク内の各ユーザーは，同じセンターを1つ選択することになる．よって，センター Center_1 が，ネットワーク内において公開されている情報となるユーザーのID情報と非公開情報である乱数を用いて，各ユーザー User_l へ送る情報 $u_{1,l} \in \mathcal{U}_{1,l}\,(1 \le l \le n)$ を生成，送信する．また，各ユーザー User_l は，受信した情報 $u_{1,l}$ を $u_l \in \mathcal{U}_l$ として保持し，u_l と鍵を共有するグループ内のユーザーのIDから，個別に共有鍵を生成する．この方式を実現することで，任意の t 人の鍵共有ユーザー集合で鍵を共有でき，k 人以下のユーザーが結託しても，秘密情報が一切漏れることがなくなる．以下では，このような鍵配送方式を (k,t) 鍵配送方式と呼ぶことにする．

また，(k,t) 鍵配送方式における，各ユーザー User_i，$(i=1,2,\cdots,n)$ のメモリ量を，その集合のサイズ $|\mathcal{U}_i|$，または U_i のエントロピーで表すことにすると，これらの量の下界は以下のようになる [2].

定理 3.1 全ての鍵のエントロピーが等しい，すなわち，

$$H(S_j) = H(S),\quad j = 1,\cdots \binom{n}{t} \tag{3}$$

が成立すると仮定する．このとき，n 人のユーザー集合 $\mathcal{U}$ と，$k+t<n$ を満たす非負整数 k，t に対する，任意の (k,t) 鍵配送方式において，

$$H(U_i) \ge \binom{k+t-1}{t-1} H(S),\quad i=1,2,\cdots n, \tag{4}$$

$$\log|\mathcal{U}_i| \ge \binom{k+t-1}{t-1} \log|\mathcal{S}|,\quad i=1,2,\cdots n \tag{5}$$

が成立する．ここで，$\mathcal{S}$ は可算集合，S は $\mathcal{S}$ の中に値をとる確率変数とした． □

この下界を達成する (k,t) 鍵配送プロトコルには，対称関数を利用した算法が良く知られている [1][6][2].

4 ランプ型分散鍵配送方式

4.1 (c,d,k,t) 分散鍵配送方式モデル

前節で述べた (k,t) 鍵配送方式では，ユーザーが保持していなくてはならない情報 u_i のサイズも大きくなってしまうことが定理3.1の結果からわかる．本研究では，前節で述べた (k,t) 鍵配送方式モデルを一般化した，(c,d,k,t) 分散鍵配送方式モデルを新たに定義する．この方式は，m 人のセンター集合と，n 人のユーザー集合において，任意の k 人のユーザーと，$g-1$ 人以下の鍵配送センターが結託しても，共有鍵に関する秘密の情報は全く得られず，任意の $k+l$ 人 $(1 \le l \le c)$ のユーザーと，$g-1$ 人以下の鍵配送センターが結託すると，l が大きくなるにつれて，段階的に情報が洩れていき，任意の $k+c+1$ 人のユーザーと，$g-1$ 人以下の鍵配送センターが結託，または，任意の g 人の鍵配送センターが結託した場合に，全ての共有鍵に関する情報を完全に求めることができるような方式である．

定義 4.1 m 人のセンター集合 $\mathcal{N}$ と，n 人のユーザー集合 $\mathcal{M}$ において，$d<m$，$c+k+t<n$ を満たす非負整数 c，k，t，d に対し，以下の条件を満たすとき，その鍵配送方式を (c,d,k,t) 分散鍵配送方式という．

1. $|\mathcal{M}_j| = t$ を満たす，任意の鍵共有ユーザー集合 $\mathcal{M}_j$ における，各々の $\mathrm{User}_i \in \mathcal{M}_j$ に対し，

$$H(S_j \mid U_i) = 0, \tag{6}$$
$$H(U_i \mid U_{1,i}, U_{2,i}, \cdots, U_{d,i}) = 0 \tag{7}$$

が成立する．

2. 各々の $\mathrm{Center}_i\,(1 \le i \le m)$ と $\mathrm{User}_j\,(1 \le j \le n)$ に対し，

$$H(U_{i,j} \mid R_{1,i}, \cdots, R_{d,i}) = 0, \tag{8}$$
$$H(U_{i,j} \mid R_{1,i},, \cdots, R_{d',i}) = 0\ \ (d'<d) \tag{9}$$

が成立する．

3. $\mathcal{X} \cap \mathcal{M}_j = \phi$，$|\mathcal{X}| \le k$，$|\mathcal{Y}| < d$，$|\mathcal{M}_j| = t$ を満たす，全てのユーザー集合とセンター集合の部分集合

$$\mathcal{X} = \{\mathrm{User}_{j_1}, \cdots, \mathrm{User}_{j_{|\mathcal{X}|}}\} \subseteq \mathcal{M}, \tag{10}$$
$$\mathcal{Y} = \{\mathrm{Center}_{i_1}, \cdots, \mathrm{Center}_{i_{|\mathcal{Y}|}}\} \subseteq \mathcal{N}, \tag{11}$$

および鍵共有ユーザー集合 $\mathcal{M}_j \subseteq \mathcal{M}$ に対し，

$$H(S_j \mid U(\mathcal{X}), U(\mathcal{Y}), R(\mathcal{Y})) = H(S_j) \tag{12}$$

が成立する．ここで，

$$U(\mathcal{X}) = (U_{j_1}, U_{j_2}, \cdots, U_{j_{|\mathcal{X}|}}),$$
$$U(\mathcal{Y}) = (U_{i_1,1}, \cdots, U_{i_1,n}, \cdots, U_{i_{|\mathcal{Y}|},1}, \cdots, U_{i_{|\mathcal{Y}|},n}),$$
$$R(\mathcal{Y}) = (R_{in}(\mathcal{Y}), R_{out}(\mathcal{Y})),$$
$$R_{in}(\mathcal{Y}) = (U_{1,i_1}, \cdots, U_{d,i_1}, \cdots, U_{d,i_{|\mathcal{Y}|}}, \cdots, U_{d,i_{|\mathcal{Y}|}}),$$
$$R_{out}(\mathcal{Y}) = (U_{i_1,1}, \cdots, U_{i_1,m}, \cdots, U_{i_{|\mathcal{Y}|},1}, \cdots, U_{i_{|\mathcal{Y}|},m})$$

とした．

4. $|\mathcal{W}| = k+l$，$\mathcal{W} \cap \mathcal{M}_j = \phi$，$|\mathcal{Y}| < d$，$|\mathcal{M}_j| = t$ を満たす，全てのユーザー集合とセンター集合の部分集合

$$\mathcal{W} = \{\mathrm{User}_{j_1}, \cdots, \mathrm{User}_{j_{k+l}}\} \subseteq \mathcal{M},\ 0 \le l \le c, \tag{13}$$
$$\mathcal{Y} = \{\mathrm{Center}_{i_1}, \cdots, \mathrm{Center}_{i_{|\mathcal{Y}|}}\} \subseteq \mathcal{N}, \tag{14}$$

と，鍵共有ユーザー集合 $\mathcal{M}_j \subseteq \mathcal{M}$ に対し，

$$H(S_j \mid U_l(\mathcal{W}), U(\mathcal{Y}), R(\mathcal{Y})) = \frac{c+1-l}{c+1} H(S_j),\quad 0 \le l \le c \tag{15}$$

が成立する．ここで，

$$U_l(\mathcal{W}) = (U_{j_1}, U_{j_2}, \cdots, U_{j_{k+l}}) \tag{16}$$

とした． □

定義4.1において，$c=0$，$d=1$，$m=0$ のとき，(k,t) 鍵配送方式の定義 [2] と一致する．よって，センター集合の大きさが1の $(0,1,k,t)$ 分散鍵配送方式と，(k,t) 鍵配送方式が等価になるので，(c,d,k,t) 鍵配送方式は，(k,t) 鍵配送方式を含んだ，より一般的な鍵配送方式のモデルになっている．

4.2 ユーザーのメモリ量の下界

本節では，(c,d,k,t) 分散鍵配送方式において，共有鍵を生成するために必要なユーザーのメモリ量の下界を導出する．そのために，まず以下の補題を用意する．この補題は [2] の補題3.1に類似した結果で，ほぼ同様の手順で証明される．

補題 4.1 n 人のユーザー集合 $\mathcal{M}$ と，$c+k+t<n$ を満たす非負整数 c，k，t，l に対し，$|\mathcal{W}| = k+l$，$\mathcal{W} \cap \mathcal{M}_{j'} = \phi$，$\mathcal{W} \cap \mathcal{M}_j \ne \phi$，$|\mathcal{M}_{j'}| = |\mathcal{M}_j| = t$，$(j=1,2,\cdots,r)$ となるようなユーザー集合の部分集合を $\mathcal{W}, \mathcal{M}_{j'}, \mathcal{M}_j \subseteq \mathcal{M}$

とする．$0 \le l \le c$ に対し，任意の (c,d,k,t) 鍵配送方式において，

$$H(S_{j'} \mid S_1, S_2, \cdots, S_r) \ge \frac{c+1-l}{c+1} H(S_{j'}) \qquad (17)$$

が成立する． □

補題 4.1 を用いることで，ユーザーのメモリ量の下界が得られる．

定理 4.1 全ての鍵のエントロピーが等しい，すなわち，

$$H(S_j) = H(S), \quad j = 1, \cdots, \binom{n}{t} \qquad (18)$$

が成立すると仮定する．このとき，n 人のユーザー集合 $\mathcal{U}$ と，$c+k+t < n$ を満たす非負整数 c，k，t に対する，任意の (c,d,k,t) 分散鍵配送方式において，

$$H(U_i) \ge \sum_{l=1}^{c} \frac{c+1-l}{c+1} \binom{k+t-2+l}{t-2} H(S) + \binom{k+t-1}{t-1} H(S), \quad 1 \le i \le n, \qquad (19)$$

が成立する．ここで，$\mathcal{S}$ は可算集合，S は $\mathcal{S}$ の中に値をとる確率変数とした． □

[証明]

$$\mathcal{I}_0 \subset \mathcal{I}_1 \subset \cdots \subset \mathcal{I}_c \qquad (20)$$

となる，ユーザー集合の部分集合

$$\mathcal{I}_l = \left\{ \mathrm{User}_{j_1}, \mathrm{User}_{j_2}, \cdots, \mathrm{User}_{j_{k+l+t-1}} \right\}, \ 0 \le l \le c \qquad (21)$$

と，固定した $\mathrm{User}_i \notin I_c$ に対して，

$$\mathcal{M}_{\mathcal{I}_l} = \{ \mathcal{M}_j \mid \mathrm{User}_1, \cdots, \mathrm{User}_{t-1} \in \mathcal{I}_l, \mathrm{User}_i \} \backslash \cup_{m=0}^{l-1} \mathcal{M}_{\mathcal{I}_l}$$

のように鍵共有ユーザー集合族を定義する．ここで，

$$\mathcal{M}_j = \{ \mathrm{User}_i, \mathrm{User}_1, \cdots, \mathrm{User}_{t-1} \} \qquad (22)$$

とおき，鍵共有ユーザー集合の添字を次のように番号付けし直す．

$$\mathcal{M}_{\mathcal{I}_l} = \left\{ \mathcal{M}_{n_{l-1}+1}, \mathcal{M}_{n_{l-1}+2}, \cdots, \mathcal{M}_{n_l} \right\}, \qquad (23)$$

$$n_l = \binom{k+l+t-1}{t-1}, \quad 0 \le l \le c. \qquad (24)$$

ただし，$n_{-1} = 0$ とする．よって，$0 \le l \le c$ に対し，

$$\begin{aligned} H(U_i) &= H(S_1, \cdots, S_{n_l}) - H(S_1, \cdots, S_{n_l} \mid U_i) + H(U_i \mid S_1, \cdots, S_{n_l}) \\ &\ge H(S_1, S_2, \cdots, S_{n_l}) \\ &\ge H(S_1 \mid S_2, \cdots, S_{n_l}) + H(S_2 \mid S_1, S_3, \cdots, S_{n_l}) \\ &\quad + \cdots + H(S_{n_l} \mid S_1, \cdots, S_{n_l-1}), \end{aligned} \qquad (25)$$

が成立する．次に式 (25) において，$1 \le h \le n_l, j = 1, \cdots, h-1, h+1, \cdots, n_l$ に対し，$\mathcal{M}_{j'} = \mathcal{M}_h$，$\mathcal{W} = \mathcal{I}_l \backslash \mathcal{M}_h$，$\mathcal{M}_j$ とすると，補題 4.1 が適用でき，仮定により

$$H(U_i) \ge \frac{n_l(c+1-l)}{c+1} H(S), \quad 0 \le l \le c \qquad (26)$$

が成立する．また，$l = c$ とした場合の式 (25) の不等式の右辺第 $n_{l-1}+1$ 項から第 n_l 項までの和について，それぞれ式 (26) の不等式を用いて評価すると，式 (19) が得られ，定理が証明される． □

定理 4.1 において，$c = 0$ のとき，

$$H(U_i) \ge \binom{k+t-1}{t-1} H(S), \quad i = 1, 2, \cdots, n \qquad (27)$$

となり，[2] の定理 3.2 と同様の結果になる．したがって，定理 4.1 は，従来の結果である定理 3.1 を含んだ一般的な結果となっている．また，ユーザーの保持すべき情報量の下界は，センター集合の大きさや，初期設定で用いるセンター数に依存しない形になっている．

4.3 (c,d,k,t) 分散鍵配送プロトコル

本節では，定理 4.1 の下界を達成する (c,d,k,t) 分散鍵配送プロトコルを提案する．ここで，共有鍵の集合 S_j，$(j = 1, 2, \cdots, \binom{n}{t})$ について以下の条件を仮定する．

$$\mathcal{S} = S_j, \quad j = 1, 2, \cdots, \binom{n}{t}, \qquad (28)$$

$$\mathcal{S} = \mathcal{S}_1 \times \mathcal{S}_2 \times \cdots \times \mathcal{S}_{c+1}, \qquad (29)$$

$$|\mathcal{S}_1| = |\mathcal{S}_2| = \cdots = |\mathcal{S}_{c+1}| = q. \qquad (30)$$

また，ここで述べる (c,d,k,t) 鍵配送プロトコルは，対称多項式の性質を利用している．t 変数多項式*

$$f(x_1, x_2, \cdots, x_t) = \sum_{0 \le l_1, l_2, \cdots, l_t \le k} a_{l_1, l_2, \cdots, l_t} (x_1)^{l_1} (x_2)^{l_2} \cdots (x_t)^{l_t} \qquad (31)$$

が対称多項式であれば，任意の置換，$\sigma : \{1, 2, \cdots, t\} \to \{1, 2, \cdots, t\}$ に対して，

$$a_{l_1, l_2, \cdots, l_t} = a_{l_{\sigma(1)}, l_{\sigma(2)}, \cdots, l_{\sigma(t)}} \qquad (32)$$

が成り立つ．

【(c,d,k,t) 分散鍵配送プロトコル】

1. $1 \le i \le d$，$1 \le l \le c+1$ に対し，Center_i は $c+1$ 個の $t+1$ 変数の多項式 $f_i^l(x_0, x_1, \cdots, x_t)$ をランダムに生成する．ただし，x_0 に関しては $d-1$ 次，$x_1, \cdots, x_t$ に関しては $k+l-1$ 次で，任意の $a \in GF(q)$ に対し，$f_i^l(a, x_1, \cdots, x_t)$ が対称多項式になるように生成する．
2. Center_i $(1 \le i \le d)$ は，各センター Center_j へ

$$r_{i,j} = \left\{ f_i^1(j, \cdots, x_t), \cdots, f_i^{c+1}(j, \cdots, x_t) \right\} \qquad (33)$$

を送る．
3. $1 \le l \le c+1$ に対し，各センター Center_j は，2. で送られてきた情報から以下を計算，記憶する．

$$F^l(j, x_1, \cdots, x_t) = \sum_{i=1}^{d} f_i^l(j, \cdots, x_t). \qquad (34)$$

4. 各ユーザー User_i は，任意の d 個のセンター

$$\{ \mathrm{Center}_{j_1}, \mathrm{Center}_{j_2}, \cdots, \mathrm{Center}_{j_d} \}$$

を選択し，それらセンターから以下を受信する．

$$u_{j_h,i} = \left\{ F^1(j_h, i, \cdots, x_t), \cdots, F^{c+1}(j_h, i, \cdots, x_t) \right\}, \quad h = j_1, \cdots, j_d. \qquad (35)$$

*演算は有限体 $GF(q)$ 上の演算とする．

5. 各ユーザー User_i は，4で送られてきた情報から多項式補間により以下を計算，記憶する．

$$u_i = \left\{F^1(0,i,\cdots,x_t),\cdots,F^{c+1}(0,i,\cdots,x_t)\right\}. \quad (36)$$

6. 鍵共有ユーザー集合

$$\mathcal{M}_j = \{\mathrm{User}_{j_1},\mathrm{User}_{j_2},\cdots,\mathrm{User}_{j_t}\} \quad (37)$$

において，それぞれのユーザー User_{j_i} は，センターから送られてきた情報から計算した $t-1$ 変数の多項式の集合となる u_{j_i} に対し，

$$(x_2,\cdots,x_t) = (j_1,\cdots,j_{i-1},j_{i+1},\cdots,j_t) \quad (38)$$

として値を計算する．

7. 鍵共有ユーザー集合 $\mathcal{M}_j$ の共有鍵を，以下のように6で計算した値を連接したものとする．

$$s_j = F^1(0)\oplus\cdots\oplus F^{c+1}(0)\in\mathcal{S}. \quad (39)$$

ここで，$\oplus$ は連接を表す記号とし，

$$F^i(0) = F^i(0,j_1,\cdots,j_t),\ i=1,\cdots,c+1 \quad (40)$$

とした． □

このプロトコルにおけるユーザーの記憶容量は定理4.1の下界を達成する．各センターが生成する $t+1$ 変数の多項式 $f_i^j(x_0,x_1,\cdots,x_t)$ は，x_0 に関して $d-1$ 次多項式となっているので，d 個のセンターの情報を集めない限り，多項式を特定することができない．また，$k+l$ 人のユーザーが結託すると，式(36)の中の l 個の多項式が特定できるので，鍵の情報が部分的に洩れていることになる．よって次の定理が成り立つ．

定理 4.2 上記のプロトコル *2* は，(c,d,k,t) 分散鍵配送方式を実現する． □

5 考察

まず，(k,t) 鍵配送方式と (c,d,k,t) 分散鍵配送方式における，ユーザーのメモリ量の下界を比較し，(c,d,k,t) 分散鍵配送方式が有効になる場合を考察する．

例 5.1 共有鍵の集合は，鍵共有ユーザー集合 $\mathcal{M}_j$ によらず，全て同じ集合 $\mathcal{S}$ で，共有鍵，ユーザーが保持している情報に関する確率分布は全て一様分布だと仮定する．秘密鍵の長さが *128* ビット，すなわち，$|\mathcal{S}| = 2^{128}$ とする．この場合の $(1,1,4,2)$ 分散鍵配送方式，$(5,2)$ 鍵配送方式，$(1,1,4,10)$ 分散鍵配送方式，$(5,10)$ 鍵配送方式に対するユーザーのメモリ量はそれぞれ，*704* ビット，*768* ビット，*173888* ビット，*256256* ビットとなる．また，式 *(15)* を安全性の尺度として用いると，攻撃者側が求めたい共有鍵 s_j の候補数として解釈でき，この値が大きいほど共有鍵の特定が困難となり，安全だといえる．

この例の場合，任意の4人以下のユーザーが結託すると，式 *(15)* の値が，全ての方式で 2^{128} となり，任意の5人のユーザーが結託すると，それぞれ 2^{64}，2^{128}，2^{64}，2^{128}，任意の6人以上のユーザーが結託してしまうと，全ての方式で1となる． □

この例から，安全性の尺度とした2つの値 2^{128} と 2^{64} を比較して，安全性に関して両者に差はない，あるいは，2^{64} という値が十分安全であるという基準に達していると考えられるのであれば，提案した (c,d,k,t) 分散鍵配送方式のほうが，ユーザーの記憶容量削減という意味で有効であるといえる．また，鍵共有ユーザー集合のサイズが大きく，結託可能なユーザー数が少ないほど，(c,d,k,t) 分散鍵配送方式がより有効となる．

次に，(k,t) 鍵配送方式，(c,d,k,t) 分散鍵配送方式，および $(k+1,t)$ 鍵配送方式の関係について考える．ここで，式(5)と式(19)の下界を，それぞれ $\alpha(k,t)$，$\beta(c,k,t)$ とすると，$\beta(c,k,t)$ は以下のような再帰的な式に書き換えられる．

$$\beta(c,k,t) = \frac{c}{c+1}\beta(c-1,k,t) + \frac{1}{c+1}\alpha(k+1,t),$$
$$c = 0,1,2,\cdots. \quad (41)$$

ここで，$\beta(-1,k,t)=0$ とした．以上より，

$$\alpha(k,t)\le\beta(c,k,t)\le\alpha(k+1,t),\ \ c=0,1,2,\cdots \quad (42)$$

が成り立ち，(c,d,k,t) 分散鍵配送方式においてユーザーが保持する情報量は，$0\le c$ に対し，(k,t) 鍵配送方式と $(k+1,t)$ 鍵配送方式においてユーザーが保持する情報量の間に値をとり，c を大きくとることで，$(k+1,t)$ 鍵配送方式によるユーザーの情報量に近づくことがわかる．これにより，(c,d,k,t) 分散鍵配送方式において，c を大きくとることで，ユーザーが保持する情報量と $k+1$ 人までのユーザーの結託に対する安全性が $(k+1,t)$ 鍵配送方式とほぼ同等で，更に $k+2$ から $k+c$ 人までのユーザーの結託に対しても，ある程度の安全性が確保できる鍵配送方式となる．

6 まとめ

本稿では，ランプ型分散鍵配送方式である (c,d,k,t) 分散鍵配送方式のモデルを，しきい値型鍵分散方式である (k,t) 鍵配送方式のモデルの一般化として定義し，そのモデルのもとでユーザーが必要なメモリ量の下界を導出した．さらに，その下界を達成するプロトコルを提案し，ユーザーが必要なメモリ量の面での有効性を示した．

7 謝辞

本稿の執筆にあたり，数多くの御助言，御支援を賜りました，早稲田大学理工学部経営システム工学科松嶋研究室，平澤研究室，両研究室の先輩方，同輩達に心より感謝申し上げます．なお，本研究の一部は日本学術振興会研究費基盤研究 (C)(No.15560338) の援助による．

参考文献

[1] R. Blom, "An optimal class of symmetric key generation systems," Proceeding of EUROCRYPT '84, Lecture Notes in Computer Science, Vol.209, pp.335-338, 1984

[2] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly secure key distribution for dynamic conference," Proceeding of CRYPT '92, Lecture Notes in Computer Science, Vol.740, pp.471-486, 1993.

[3] C. Blundo, P. D'Arco and C. Padró, "A ramp model for distributed key distribution schemes," Discrete Applied Mathematics, vol.128, pp.47-64, 2003.

[4] P. D'Arco, "On the distribution of a key distribution center," Proceeding of ICTCS '01, Lecture Notes in Computer Science, Vol.2202, Springer, Berlin, 2001, pp.357-369.

[5] G. Hanaoka, T. Nishioka, Y. Zheng and H. Imai, "Optimal unconditionally secure ID-based key distribution shceme for large-scaled networks," IEICE Trans. Fundamentals, vol.E84-A, No.1, pp.222-230, 2001.

[6] 松本勉，今井秀樹，"暗号鍵を通信なしで共有する方法，"電子情報通信学会論文誌，Vol.J71-A，No.11，pp.2046-2053，1988.

[7] D. R. Stinson, "On some methods for unconditionally secure key distribution and broadcast encryption," Designs Codes Cryptography, vol.12, No.3, pp.215-243, 1997.

社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

再帰的組織畳み込み符号を利用した Reliability Based Hybrid ARQ についての研究

小林 直人† 小泉 大城† 松嶋 敏泰† 平澤 茂一†

† 〒169-8555 東京都新宿区大久保 3-4-1 早稲田大学理工学部経営システム工学科
E-mail: †{kobayashi,dkoizumi,toshi}@matsu.mgmt.waseda.ac.jp, ††hirasawa@hirasa.mgmt.waseda.ac.jp

あらまし Reliability Based Hybrid ARQ (RBH-ARQ) は，Hybrid ARQ 手法の一種であり，復号器は ACK/NAK 信号だけで無く，最大事後確率復号法の出力から決定される信頼度の低いビットのインデックスをフィードバック情報として用いる．送信器はフィードバック情報を元に符号化を行い，その符号語を再送する．本研究では，再帰的組織畳み込み符号に対し RBH-ARQ を用いた新たな誤り訂正手法をいくつか提案する．これらはそれぞれ，復号器において再送された受信値の用い方に明確な差異のあるものとなっている．これらの性能の比較を行うことで，RBH-ARQ 手法における再送された受信値の用い方や符号器の構成法等について検討を行う．

キーワード Reliability Based Hybrid ARQ，再帰的組織畳み込み符号，ターボ符号

A Study of Reliability Based Hybrid ARQ Schemes Using a Recursive Systematic Convolutional Code

Naoto KOBAYASHI†, Daiki KOIZUMI†, Toshiyasu MATSUSHIMA†, and Shigeichi HIRASAWA†

† Dept. of Industrial & Management Systems Engineering, School of Science and Engineering, Waseda University. Ookubo 3-4-1, Shinjyukuku, Tokyo, 169-8555 Japan.
E-mail: †{kobayashi,dkoizumi,toshi}@matsu.mgmt.waseda.ac.jp, ††hirasawa@hirasa.mgmt.waseda.ac.jp

Abstract The Reliability Based Hybrid ARQ (RBH-ARQ) is one of hybrid ARQ schemes. In the RBH-ARQ, the modified feedback is composed of both ACK/NAK signal and unreliable bit indices which is evaluted from bitwise posterior probabilities. Using the modified feedback information, a sender encodes and retransmits the codeword. In this paper, we propose several error correction procedures with the RBH-ARQ schemes based on a systematic convolutional code. These have differences in their use of received retransmitted informations. To evalute and compare the peformances of these algorithms, we consider use of received retransmitted informations and a constitution of the encoder in the RBH-ARQ schemes.

Key words Reliability Based Hybrid ARQ, Recursive Systematic Convolutional Code, Turbo Code

1. はじめに

Reliability Based Hybrid ARQ (RBH-ARQ) [1] は，Hybrid ARQ 手法の一種であり，復号器は ACK/NAK 信号だけで無く，ビット単位の事後確率復号法の出力から決定される信頼度の低いビットのインデックス情報をフィードバック情報として用いる．送信器は信頼度の低い情報ビットについて再送するために，フィードバック情報を元に符号化を行い，その符号語を送信する．ビット単位の事後確率復号法を行うアルゴリズムとしては，BCJR アルゴリズム [2] や，ターボ復号 [3] 等があり，それぞれ RBH-ARQ と組み合わせた研究が行われている [4] [5]．これらの RBH-ARQ では，再送された受信値について単純に加算を行う (白色ガウス通信路を仮定した場合) 等の処理を行っており，再送情報について考慮された手法とは言い難い．

本研究では，再帰的組織畳み込み符号に対し RBH-ARQ を用いた新たな復号アルゴリズムをいくつか提案する．これらは，それぞれ復号器において再送された受信値の用い方に明確な差異があるものになっており，これらの性能の比較を行うことで，RBH-ARQ 手法における再送された受信値の用い方や符号器の構成法等について検討を行う．なお，提案手法の一部は [6] にて提案されたものであるが，数値実験による評価が不十分であったため，提案法の評価にこれらの評価も含めて行う．

2. RBH-ARQ 手法の基本モデル

$U = u_1 u_2 ... u_K (u_i \in \{0,1\})$ を情報系列とする．$U^{(t)}$ を情報系列の部分系列とし，時点 t に符号化を行う系列とする $(t = 1,2,...)$．時点 t に送信する符号語系列を $X^{(t)} = x_1^{(t)} x_2^{(t)} ... x_{l(t)}^{(t)}$ とし，その長さを $l(t)$ と表記する．$x_j^{(t)}$ はそれぞれ $\{-1,+1\}$ を取るものとする（符号器の出力の 0 を -1 に，1 を $+1$ に対応させる）．通信路は，分散 σ の白色ガウス通信路を仮定する．

復号器は，時点 t において受信系列 $Y^{(t)} = y_1^{(t)} y_2^{(t)} ... y_{l(t)}^{(t)}$ を受け取ったもとで，各情報ビット u_i の対数尤度比

$$\lambda_i^{(t)} = \log \frac{p(u_i = 0 | Y^{(1)}, ..., Y^{(t)})}{p(u_i = 1 | Y^{(1)}, ..., Y^{(t)})}, \tag{1}$$

を周辺事後確率計算アルゴリズム等により計算する．任意に定められたしきい値 λ に対し，

$$|\lambda_i^{(t)}| < \lambda, \tag{2}$$

となる i が存在する場合(注1)，復号器は NAK 信号と共に，以下の $\Omega^{(t)}$ を送信元に送信する．

$$\Omega^{(t)} = \{i \,|\, |\lambda_i^{(t)}| < \lambda\}. \tag{3}$$

$\Omega^{(t)}$ は t 時点の復号器において，信頼度の低い推定ビットのインデックス集合である．$\Omega^{(t)}$ の要素数を $z(t)$ で表し，その各要素を $\omega_j^{(t)} (j = 1,2,..., z(t))$ とする．帰還通信路は雑音の無い通信路を仮定する．$t+1$ 時点において符号器は，$\Omega^{(t)}$ を用いて符号化を行う．時点 t において，式 (2) を満たす i が存在しない場合，復号器は ACK 信号を送信元に送信し，情報系列を式 (4) により推定する．

$$\hat{u}_i = \begin{cases} 0, & \lambda_i^{(t)} \geqq 0; \\ 1, & other. \end{cases} \tag{4}$$

また，$\Omega^{(t)}$ の要素を対数尤度比の小さい順に順序付けたものを順序付き集合 $\overline{\Omega}^{(t)}$ として定義する．

3. 再帰的組織畳込み符号

再帰的組織畳込み符号は，主にターボ符号の要素符号として用いられる符号である [3]．本研究では符号化率 1/2 の再帰的組織畳込み符号を用いる．遅延素子数を C とし，符号器のシフトレジスタが取りうる状態を $s_0, s_1, ..., s_{q-1} (q = 2^C)$ で表す．s_0 を特に遅延素子の値が全て 0 の状態（初期状態）とする．系列長 k の情報系列 $u_1 u_2 ... u_k$ を入力した時のパリティ部の出力をそれぞれ $c_1 c_2 ... c_k$ と表す．符号器の最終的な状態を s_0 とするための終端系列(注2)を，便宜的に $u_{k+1} u_{k+2} ... u_{k+C}$ と定義し，これを入力した時のパリティ部の出力を $c_{k+1} c_{k+2} ... c_{k+C}$ とする．

本稿では，再帰的組織畳込み符号の復号アルゴリズムとして BCJR アルゴリズムを用いる．アルゴリズムの詳細は [2] に従うが，受信系列として Y を受け取ったもとでの γ_i の計算については以下のように表記する．

$$\gamma_i(s', s) = \sum_{a \in \{0,1\}} \gamma_i'(s', s, a). \tag{5}$$

但し，

$$\gamma_i'(s', s, a) = \begin{cases} p(u_i) p_i(Y, s' | a), & p(s | s', a) = 1; \\ 0, & other. \end{cases} \tag{6}$$

$p(u_i)$ は，u_i の事前分布を表す．また $p(s|s', a) = 1$ は，符号器が状態 s' にあり入力が a であった時，状態が s に遷移する場合に成立することを意味する．

4. RBH-ARQ 手法を用いた提案手法

本章では，再帰的組織畳込み符号を用いた RBH-ARQ による誤り訂正手法として，以下の 4 つの手法を提案する．$\Omega^{(t)}$ の要素の決定等は，基本モデルに従うものとする．また Procedure 1-3 については [6] において提案されている手法である．本章では，k は範囲として $(1 \leqq k \leqq K)$ をとる整数値，j は範囲として $(1 \leqq j \leqq K + C)$ をとる整数値とする．

4.1 Procedure 1

【符号器】

(t = 1)

$U^{(1)} = U$ とし，符号化比率 1/2 の再帰的組織畳込み符号により符号化を行う．符号系列 $X^{(1)}$ は，全ての j に対し式 (7)(8) のように定義される．

$$x_j^{(1)} = 2u_j - 1, \tag{7}$$

$$x_{j+K+C}^{(1)} = 2c_j - 1. \tag{8}$$

符号長は $l(1) = 2(K + C)$ となる．

(t > 1)

$U^{(t)}$ を $\Omega^{(t-1)}$ に含まれるインデックスに対応する情報ビットを全て含む部分系列とする．すなわち，

$$U^{(t)} = u_{\omega_1^{(t-1)}} u_{\omega_2^{(t-1)}} ... u_{\omega_{z(t)}^{(t-1)}}, \tag{9}$$

とする．これを符号化せずそのまま送信する．すなわち全ての $i (1 \leqq i \leqq z(t))$ に対し

$$x_i^{(t)} = 2u_{\omega_i^{(t-1)}} - 1, \tag{10}$$

とする．$l(t) = z(t-1)$ となる．

【復号器】

(t = 1)

受信した $Y^{(1)}$ を元に BCJR アルゴリズムを用いて，全ての k に対し $\lambda_k^{(1)}$ を計算する．

(t > 1)

各 $l (1 \leqq l \leqq l(t))$ において，$i = \omega_l^{(t-1)}$ となる全ての i に対して，

$$\lambda_i^{(t)} = \lambda_i^{(t-1)} + \frac{2y_j}{\sigma^2}, \tag{11}$$

と更新する．更新されなかった $\lambda_i^{(t)}$ に対しては $\lambda_i^{(t)} = \lambda_i^{(t-1)}$ とする．

(注1)：$\Omega^{(t)}$ の要素数を常に定数個とする方法 [4] もあるが，本研究ではしきい値による方法を扱う．

(注2)：終端系列の生成については [8] 等を参照．

4.2 Procedure 2

【符号器】

Procedure 1 と同様.

【復号器】

(t = 1)

Procedure 1 の (t = 1) と同様.

(t > 1)

受信した $Y^{(1)}, Y^{(2)}, ..., Y^{(t)}$ を元に BCJR アルゴリズムを用いて，全ての k に対し $\lambda_k^{(t)}$ を計算する．この時全ての j に対し，γ_j 内の $p_j(Y, s'|a)$ は以下の式で計算される (Y は $Y^{(1)}, ..., Y^{(t)}$ を意味する).

$$p_j(Y, s'|a) = \exp\left(\frac{\sum_{T=1}^{t} r_j(a)^{(T)} + (y_{j+K+C}^{(1)} - \Psi(s', a))^2}{2\sigma^2}\right). \quad (12)$$

但し,

$$r_j(a)^{(T)} = \begin{cases} (y_i^{(T)} - (2a-1))^2, & (j \in \Omega^{(T-1)}, j = \omega_i^{(T-1)}) \\ 0, & (j \notin \Omega^{(T-1)}), \end{cases} \quad (13)$$

とし，$\Psi(s', a) \in \{-1, +1\}$ は，符号器が状態 s' にあり入力が a である時の符号器の出力を表す.

4.3 Procedure 3

【符号器】

(t = 1)

Procedure 1 の (t = 1) と同様.

(t > 1)

$U^{(t)}$ を式 (9) と同様に，$\Omega^{(t-1)}$ に含まれるインデックスに対応する情報ビットを全て含む部分系列とする．この時，適当なインターリーバによりこの部分系列の順序を操作を行うが，表記が煩雑となるのを避けるため，以下順序どおり並んでいるとみなした表記とする(注3)．符号化比率 1/2 の再帰的組織畳み込み符号により符号化を行い，パリティ部のみを送信する．すなわち，全ての $i(1 \leqq i \leqq z(t-1)+C)$ に対して $x_i^{(t)} = c_i$ とする．$l(t) = z(t-1) + C$ となる.

【復号器】

(t = 1)

Procedure 1 の (t = 1) と同様.

(t > 1)

各時点 t の再帰的組織畳み込み符号を要素符号とした Multiple ターボ復号 [7] を行う．時点 t に対応する復号器においては，符号長を $l(t)$ とした BCJR アルゴリズムを適用する．但し，式 (6) は各 $i(1 \leqq i \leqq l(t))$ に対し,

$$\gamma_i'(s', s, a) = \begin{cases} p(u_n) p_i(Y, s'|a), & p(s|s', a) = 1 \\ 0, & other. \end{cases} \quad (14)$$

となる．但し $n = \omega_i^{(t-1)}$ とする．事前分布値 $p(u_n)$ を任意の伝播スケジュールに従い，他の復号器の外部値より計算する (伝播スケジュールについては [6] を参照)．この時 $p_i(Y, s'|a)$ は以下の式で計算される.

$$p_i(Y, s'|a) = \exp\left(\frac{(y_n^{(1)} - (2a-1))^2 + (y_i^{(t)} - \Psi(s', a))^2}{2\sigma^2}\right). \quad (15)$$

4.4 Procedure 4

【符号器】

(t = 1)

Procedure 1 の (t = 1) と同様.

(t > 1)

$U^{(t)} = U$ とする．ここでインターリーブの方法により手法を 2 つに分類する．ランダムにインターリーブを行う手法を **Procedure 4.1** とする．付録 1. のアルゴリズムによりインターリーブを行う手法を **Procedure 4.2** とする．このインターリーブ法は，対数尤度比の低かったビットをなるべく遠くに配置するような手法であるため，Procedure 4.2 はフィードバック情報として $\overline{\Omega}^{(t)}$ を用いる．インターリーブした系列を符号化率 1/2 の再帰的組織畳み込み符号により符号化し，$\Omega^{(t-1)}$ または $\overline{\Omega}^{(t-1)}$ に含まれるインデックスに対応するパリティ部のビットのみを送信する．$l(1) = z(t-1) + C$ となる.

【復号器】

Procedure 3 と同様の Multiple ターボ復号を行う．但し各復号器は，全て符号長 K として BCJR アルゴリズムを適用し，送信されてない受信ビットについては値を 0 とする．すなわち，$y_j^{(t)} = 0\ (j \notin \Omega^{(t-1)})$ とする.

4.5 各手法の比較

Procedure 1-3 は，それぞれ再送時に，復号処理を行わない (受信値の加算のみ)，BCJR を行う，ターボ復号を行う，という点で異なり，後者ほど計算量がかかるが，復号性能は向上することが期待できる．Procedure 4.1，Procedure 4.2 は Procedure 3 にパンクチャ操作を加えた手法である．Procedure 4.2 については $\overline{\Omega}^{(t)}$ を利用し，ターボ復号器において対数尤度比の低かったビット同士を，なるべく遠くに配置するような構造としている.

5. 数値実験

5.1 実験条件

前章の各手法について以下の条件で評価を行った.

- 要素符号として $K = 256$，$C = 3$ の符号化率 1/2 の再帰的組織畳み込み符号を用いる.
- それぞれ 4000 回の実験を行い，その平均ビット誤り率が 1.0×10^{-5} (注4) となるような λ について，平均スループット及び平均再送回数を求める.
- λ は時刻 t により変化しないものとする.
- ターボ復号はその反復回数を 3 回とし，伝播スケジュールとして Serial Schedule を用いた．Procedure 3，Procedure

(注3)：同様に，復号器についてのインターリーブ操作・逆インターリーブ操作についても明記しない.

(注4)：λ を操作し平均ビット誤り率が $0.9 \times 10^{-5} \sim 2.0 \times 10^{-5}$ に入った λ を採用した.

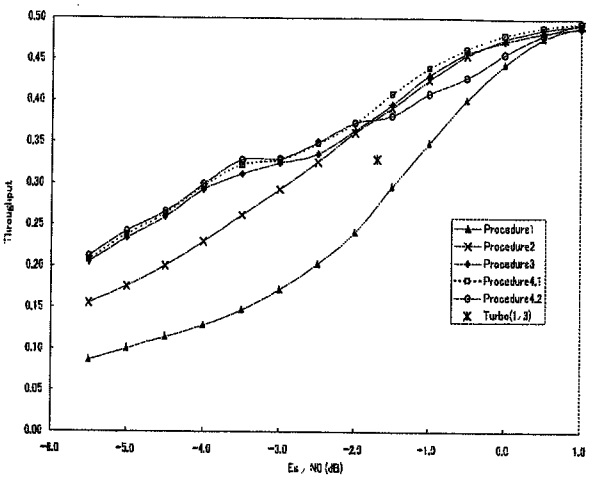


図 1　各手法における平均スループット

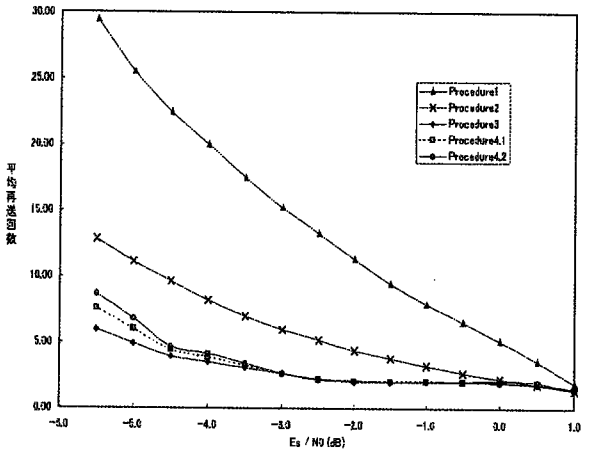


図 2　各手法における平均再送回数

表 1　平均ビット誤り率が 1.0×10^{-5} となる λ の値

Es/N0	P1	P2	P3	P4.1	P4.2
1.0	6.50	5.60	7.00	6.00	6.00
0.5	8.10	6.20	7.00	6.30	8.70
0.0	8.90	6.70	6.80	6.40	9.40
-0.5	9.00	6.90	6.70	6.70	9.60
-1.0	9.10	7.10	7.10	6.80	9.00
-1.5	9.20	7.20	7.80	7.20	9.00
-2.0	9.60	7.00	8.50	8.00	8.00
-2.5	9.80	7.08	10.50	8.80	8.60
-3.0	10.00	7.10	11.80	11.00	11.00
-3.5	10.20	7.20	10.50	8.30	8.30
-4.0	10.30	7.50	8.65	8.40	8.40
-4.5	10.30	7.80	9.35	8.90	9.10
-5.0	10.50	8.00	8.80	8.10	7.60
-5.5	11.10	8.05	8.70	8.00	7.60

4.1 のインターリーバは毎時点それぞれ乱数により生成されたものを用いた．

5.2 実験結果

各手法の平均スループットを図 1 に示す．参考として同要素符号を用いた符号化率 1/3 のターボ符号についても図示した．Procedure 4.1 については Procedure 3 より常に平均スループットの値が大きいという結果が示された．Procedure 4.2 は Es/N0 が-2.0 以下の場合は Procedure 4.1 よりも常に平均スループットの値が大きいが，それより大きい場合は，Procedure 2 より悪くなるという結果が示された．各手法の平均再送回数を図 1 に，各手法における平均ビット誤り率が 1.0×10^{-5} となる λ の値を表 1 に示す．

6. 考　　察

Procedure 1-3 については，4.5 節で述べたように，復号における計算量が増えるほど平均スループットが向上するという結果が得られた．Procedure 3 は，Es/N0 が 2.5 より小さい範囲では Procedure 2 より平均スループットがそれ以上の部分より良化している．また雑音が大きくなった場合にも，平均再送回数が他の手法よりも安定しており，Procedure 3 は，高雑音での環境及び再送回数に制限がある場合などに有効であると言える．

Procedure 4.1 については，僅かではあるが Procedure 3 より平均スループットが向上している．このことからターボ復号の各要素 BCJR 復号器において，短い符号長（トレリス線図）で復号を行うよりも，符号長は情報系列長に合わせ，パンクチャ処理を行った方が復号性能が向上すると言える．

Procedure 4.2 については，Es/N0 が-2.0 以下の範囲では Procedure 4.1 よりも平均スループットが向上している．これは対数尤度比の低かったビット同士を，なるべく遠くに配置するようなインターリーブを行うことで，復号性能が向上すると言え，RBH-ARQ においての符号の構成法にこのような処理が有効であると考えられる．逆に雑音の小さな範囲においては，Procedure 2 よりもスループットが悪くなっている．実験結果を詳細に見ると，これは再送要求ビット数が情報系列長に対して少ない場合には，対応する要素符号器の影響が非常に小さくしまうことで発生しており，Procedure 3 との組み合わせ等で対処する必要がある．

平均ビット誤り率が 1.0×10^{-5} となる λ の値については，Procedure 1-2 については，高雑音になる程その値を大きくすれば良いという結果が読み取れるが，Procedure 3-4 については，Es/N0 の値が-3.0 の周辺でその関係が逆転している．これは Procedure 3-4 の初回の復号は BCJR による復号であるが，それ以降はターボ復号となっていること，つまり初回とそれ以降では復号方法が違うことに起因していると考えられる．このことから初回とそれ以降の λ を異なる値にすることで，性能向上が期待できる．

7. ま と め

本研究では，再帰的組織畳み込み符号に対し RBH-ARQ を用いた新たな誤り訂正手法をいくつか提案し，数値実験による性能を評価を行った．本稿ではそれぞれのプロトコルを比較することを重点とした実験を行ったため，復号性能の向上については検討していない．例えば，λ を時点により可変にする等の

改良法が，従来の RBH-ARQ 手法において行われており，それらとの組み合わせによる数値実験を行い，どのような手法が有効であるかを検討することが今後の課題である．

謝　　辞

本研究を行うにあたり，数多くの御助言，御支援を賜りました松嶋研究室，平澤研究室の各氏に感謝致します．なお，本研究の一部は日本学術振興会科学研究費基盤 (C) 一般 (No.15560338) の援助による．

文　　献

[1] J. M. Shea, "Reliability-based hybrid ARQ," IEE Electronics Letter, vol. 38, pp. 644-645, June 2002.

[2] L. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, vol. 20, pp. 284-287, Mar. 1974.

[3] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in Proc. *IEEE Int. Conf. Commun.*, vol. 2, pp. 1064-1070, May. 1993.

[4] Hanjo Kim, John M. Shea, "New turbo-ARQ techniques based on estimated reliabilities," in Proc. *IEEE Wireless Communications and Networking Conf.*, pp.843-848, vol. 4, Mar. 2003.

[5] A. Roongta and J. M. Shea, "Reliability-based Hybrid ARQ using Convolutional Codes" in Proc. *IEEE Int. Conf. on Communications*, vol. 4, pp. 2889-2893, May. 2003.

[6] D. Koizumi, N. Kobayashi, T. Matsushima, S. Hirasawa, "A Study of Reliability Based Hybrid ARQ Scheme with Bitwise Posterior Probability Evaluation from Message Passing Algorithm," , 信学技報, IT2005-24, pp. 11-16, May. 2005.

[7] D. Divsalar and F. Pollara, "Multiple turbo codes," in Proc. *IEEE Military Communications Conf.*, vol. 1, pp. 279-285, Nov. 1995.

[8] B. Vucetic, J. Yuan, "TURBO CODES: Principles and Applications," Kluwer. Academic Publishers, 2000.

付　　録

1. Procedure 4.2 のインターリーブ手法

情報系列長 K がある自然数 a の 4 乗となる場合を考える．Z を要素数が K の順序付き集合とし，$Z_i = \overline{\Omega}_i^{(t)} (1 \leqq i \leqq z(t))$ とする．$z(t)+1 \leqq i \leqq K$ の範囲の要素については，$\overline{\Omega}^{(t)}$ に含まれていない情報系列のインデックスを順に並べたものとする．S^A, S^B, S^C, T を順序つき集合とし，$T_i (1 \leqq i \leqq K)$ の値を i 番目の入力に対してのインターリーバの出力順序とする．shuffle(S) を，任意の順序付き集合 S を要素順序をランダムに置換する関数とする．

```
begin
r := a random number from 1 to K;
b := (K/a) − a^2;
s_1 := s_2 := s_3 := 1;
for i := 1 to a^2 do S^A_i := Z_i;
shuffle(S^A);
for i := 1 to b do S^B_i = Z_{i+a^2};
shuffle(S^B);
for i := 1 to K − b do S^C_i = Z_{i+(K/a)} ;
shuffle(S^C);
for i := 1 to K do
  begin
  if i mod a^2 = 0 then
    T_{(i+r mod K)} := S^A_{s_1};
    s_1 := s_1 + 1;
  else if i mod a = 0 then
    T_{(i+r mod K)} := S^B_{s_2};
    s_2 := s_2 + 1;
  else
    T_{(i+r mod K)} = S^C_{s_3};
    s_3 = s_3 + 1;
  end
end
```

The 28th Symposium on Information Theory
and Its Applications (SITA2005)
Onna, Okinawa, Japan, Nov. 20-23, 2005

帰還通信路を用いた誤り制御方式に関しての研究
A Note on an Error-Correcting System with a Feedback Channel

小林 直人* 松嶋 敏泰* 平澤 茂一*
Naoto Kobayashi Toshiyasu Matsushima Shigeichi Hirasawa

Abstract— We propose a new error-correcting system with a feedback channel. A feedback channel is used in the Reliability Based Hybrid ARQ (RB-HARQ) scheme. In the RB-HARQ scheme, the modified feedback is composed of both ACK/NAK signal and unreliable bit indices which is evaluted from bitwise posterior probabilities. A sender continues to retransmit codewords until receiving ACK signal. In our proposed method, a receiver send unreliable bit indices to a sender only one time. A sender encodes information with the feedback and a receiver decodes it by turbo decoder.

Keywords— Feedback channel, Reliability Based Hybrid ARQ scheme (RB-HARQ), Turbo decoding

1 はじめに

本稿では帰還通信路を用いた新たな誤り制御方式を提案する．帰還通信路を用いる誤り制御方式としては，ARQ（Automatic Repeat Request）方式が一般によく知られている [1]．ARQ は誤り検出符号を用いて符号化を行い，受信器で誤りを検出した場合には，帰還通信路を介して送信器に対し否定応答（NAK）信号を送出し，同一の情報を再送させる方式である．一般的に，送信器は肯定応答（ACK）信号を受信するまで再送を繰り返す．また ARQ を拡張した方式として，Reliability Based Hybrid ARQ (RB-HARQ) という方式がある [2]．RB-HARQ は誤り訂正符号と最大事後確率（MAP）復号法を用いた誤り制御方式で，受信器は ACK/NAK 信号だけで無く，MAP 復号器の出力から決定される信頼度の低いビットのインデックス情報を，帰還通信路を介して送出する．送信器は，その信頼度の低い情報ビットについて再送するためにフィードバックを元に再度符号化を行い，その符号語を送信する．RB-HARQ においても，一般的に送信器は ACK 信号を受信するまで再送を繰り返す．従って，これらの方式において送信器が送出する総シンボル数は不定であり，符号化比率（スループット）もまた不定となる．

これらに対し本稿で提案する方式は，受信器は RB-HARQ のように信頼度の低いビットのインデックス情報を帰還通信路を介して送出するが，再送回数は必ず 1 回とする誤り制御方式である．すなわち，送信器は必ず 1 回だけフィードバックを受信し，それを元に送信手順を変更する．受信器は 2 回目の受信で必ず情報系列を推定し，処理を終了する．従って，提案法の符号化比率（スループット）は一般的な前方向誤り訂正方式と同様に一定となる．提案法では，誤り訂正符号として再帰的組織畳込み符号を用い，受信器は 1 回目の復号を BCJR アルゴリズム [3]，2 回目の復号をターボ復号 [4] によって行う．送信器は 2 回目の符号化を行う際，どのビットにパンクチャ操作を行うか，またどのようなインターリーブ法を行うかを，フィードバックを元に決定する．本研究は，提案法と帰還通信路を用いないほぼ同等の符号（ターボ符号）を比較することにより，フィードバックを用いることで誤り訂正能力をどれだけ向上させることができるかを評価することを目的とする．

本稿では，これらの性能をシミュレーションにより評価し，その結果，フィードバックを用いることによりビット誤り率が改善できることを示した．また，初回の復号結果において信頼度の低かったビット同士を，再送時の符号化の際に（トレリス線図上において）ある程度の間隔をもって配置するようなインターリーブ法が有効であることを示した．

2 Reliability Based Hybrid ARQ

Reliability Based Hybrid ARQ (RB-HARQ) は，誤り訂正符号と最大事後確率（MAP）復号法を用いた誤り制御方式である．受信器は，MAP 復号器の出力から決定される信頼度の低いビットのインデックス情報を，送信器に対し帰還通信路を介して送出する [2]．MAP 復号アルゴリズムとしては，BCJR アルゴリズム [3] やターボ復号 [4] 等があり，それぞれ RB-HARQ と組み合わせた研究が行われている [5] [6] [7]．本稿で提案する誤り制御方式において，フィードバックは RB-HARQ のものに則する．以下にその基本手順を記述する．

$U = u_1 u_2 ... u_K (u_i \in \{0,1\})$ を情報系列とする．時点 t に送信する符号語系列を $X^{(t)} = x_1^{(t)} x_2^{(t)} ... x_{l(t)}^{(t)}$ とし，その長さを $l(t)$ と表記する $(t = 1, 2, .., T)$．

受信器は，時点 t において受信系列 $Y^{(t)} = y_1^{(t)} y_2^{(t)} ... y_{l(t)}^{(t)}$ を受け取ったもとで，各情報ビット u_i の対数尤度比

$$\lambda_i^{(t)} = \log \frac{p(u_i = 0 | Y^{(1)}, ..., Y^{(t)})}{p(u_i = 1 | Y^{(1)}, ..., Y^{(t)})}, \qquad (1)$$

* 〒 169-8555 東京都新宿区大久保 3-4-1 早稲田大学理工学部経営システム工学科 Dept. of Industrial & Management Systems Engineering, School of Science and Engineering, Waseda University. Ookubo 3-4-1, Shinjyukuku, Tokyo, 169-8555 Japan. E-mail: kobayashi@matsu.mgmt.waseda.ac.jp

を MAP 復号アルゴリズム等により計算する．

任意に定められた整数値 δ に対し，$\Omega^{(t)}$ を情報系列のインデックス $1 \leq i \leq K$ のうち異なる δ 個を要素として持つ順序付き集合と定義する．$\Omega^{(t)}$ は，各インデックスに対応する $|\lambda_i^{(t)}|$ の値の小さいものから順に δ 個選んだ要素により構成され，その要素に対応する $|\lambda_i^{(t)}|$ が小さい順に各インデックスが並んでいるものとする．再送終了条件を任意に定め，これを満たしていない場合は NAK 信号と共に，$\Omega^{(t)}$ を送信器に送出する[1]．$t+1$ 時点において送信器は，$\Omega^{(t)}$ を用いて符号化を行う．ある時点 t において再送終了条件を満たした場合，受信器は ACK 信号を送信器に送出した上で，情報系列を式 (2) により推定し処理を終了する．

$$\hat{u}_i = \begin{cases} 0, & \lambda_i^{(t)} \geq 0; \\ 1, & other. \end{cases} \tag{2}$$

3 提案手法

帰還通信路を用いた新たな誤り訂正手法を提案する．提案法の手順は主に前章で記述した RB-HARQ の基本手順に従うが，$t=1$ の場合は必ず再送を行い，$t=2$ の場合は再送を行わないものとする（$T=2$ とする）．すなわち，符号化を 2 回に分け，2 回目の符号化はフィードバック（1 回目の復号結果）を利用した符号化を行う動的な符号化法と表現できる．従って提案法の符号化比率は，一般的な前方向誤り訂正方式と同様に一定となる．通信路は分散 σ の白色ガウス通信路を，帰還通信路は雑音の無い通信路を仮定する．以下に提案手法の詳細を記載する．

$(t=1)$

1. 送信器は，符号化比率 1/2，拘束長 C の再帰的組織畳込み符号 [4] により情報系列を符号化する．パリティ部のうち任意の ρ 個のシンボルをパンクチャし送信する $(0 \leq \rho < K)$．$l(1) = 2(K+C) - \rho$ となる．
2. 受信器は，受信した系列 $Y^{(1)}$ を元に BCJR アルゴリズム [3] により，各情報系列ビットの対数尤度比を計算する．
3. 基本手順に従い $\Omega^{(1)}$ を決定し，送信器に帰還通信路を介して送出する．但し $\delta = K - \rho$ とする．

$(t=2)$

1. 送信器は，情報系列にインターリーブ操作を行った系列を，再帰的組織畳込み符号により符号化する．インターリーブ操作については後述する．$\Omega^{(1)}$ に含まれないビットをパンクチャし，パリティ部のみを送信する．$l(2) = K - \rho$ となる（終端処理部は送信しない）．
2. 受信器は $Y^{(1)}$，$Y^{(2)}$ を元に，ターボ復号 [4] を行い情報系列の推定をする．$t=1$ 時に送信した符号に対応する要素復号器を「要素復号器 1」，$t=2$ 時に送信した符号に対応する要素復号器を「要素復号器 2」と呼ぶ．

[1] $\Omega^{(t)}$ の要素数を不定とし，一定のしきい値以下の対数尤度比を持つインデックスを全て送出する手法もある [5] [7].

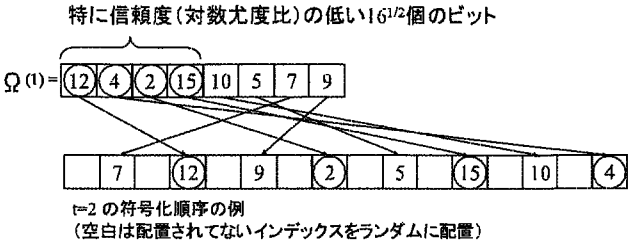


図 1: Procedure2 のインターリーブ法の具体例

符号化比率は $K/(3K+2(C-\rho))$ となる．$t=2$ 時に行うインターリーブ操作の方法により手法を 2 つに分類する．ランダムにインターリーブ操作を行う手法を Procedure 1 とし，付録のアルゴリズムによるインターリーブ操作を行う手法を Procedure 2 とする[2]．

このインターリーブ法は，$t=1$ 時の復号の際に対数尤度比の絶対値の小さい $K^{\frac{1}{2}}$ 個のビットが，$(K^{\frac{1}{2}}-1)$ の間隔ごとに均等に配置され，次に対数尤度比の絶対値の小さい $(K^{\frac{3}{4}}-K^{\frac{1}{2}})$ 個のビットが，すでに配置されている箇所を除いた $(K^{\frac{1}{4}}-1)$ の間隔ごとに均等に配置される手法である（配置位置の基準点及び，各要素がどの位置に配置されるかはランダム）．この手法により，初回の復号において信頼度の低いビット同士は，要素復号器 2 のトレリス線図上において，必ずある程度の間隔が保たれることとなる．$K=16$，$\rho=8$ とした場合のこのインターリーブ法の具体例を 1 に示す．

4 シミュレーション

提案法の性能について評価を行うために，以下の条件でシミュレーションを行う．比較対象として同じ要素符号を用いたターボ符号についても同様に評価を行う．

- $K=256, 1296$，$C=3, 4$ のそれぞれについて実験を行う．
- $\rho = K/2$ とする．従って符号化比率は約 1/2 となる．
- ターボ符号においては，一方の符号器は偶数インデックスをもつ符号語シンボルをパンクチャし，もう片方の符号器は奇数インデックスをもつ符号語シンボルをパンクチャする（インターリーブ操作を行う前のインデックスに対して）．

[2] 付録のインターリーブ法は [7] にて提案されたものである．

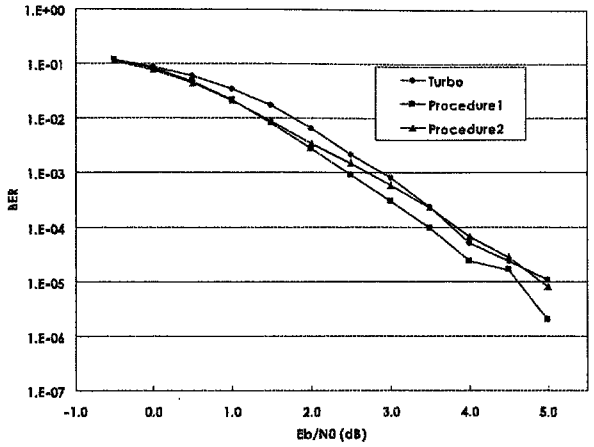


図 2: K=256, C=3 のビット誤り率

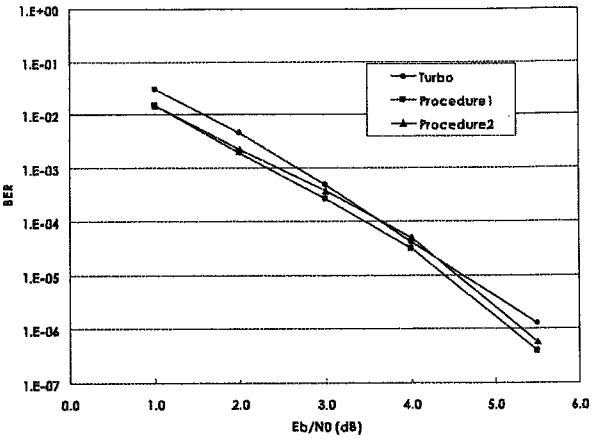


図 4: K=1296, C=3 のビット誤り率

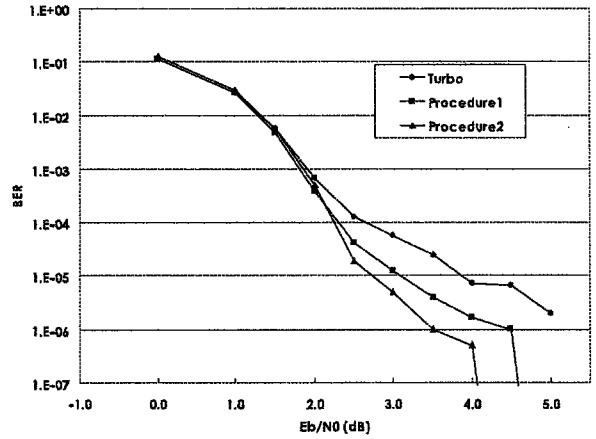


図 3: K=256, C=4 のビット誤り率

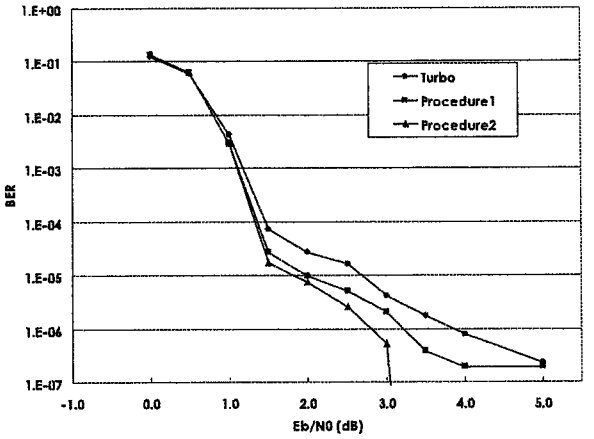


図 5: K=1296, C=4 のビット誤り率

- Procedure 1,2 において，$t = 1$ 時の符号器は偶数インデックスをもつ符号語シンボルをパンクチャする．
- それぞれ 24000 回の実験を行い，各 SNR(E_b/N_0) についてビット誤り率 (BER) を評価する．
- ターボ復号はその反復回数を 10 回 ($K = 256$)，15 回 ($K = 1296$) とする．
- ターボ符号，Procedure 1 のインターリーバは，実験ごとにそれぞれ乱数により生成されたものを用いる．

実験結果を図 2〜図 5 に示す．なお，グラフにて BER の値が 10^{-7} 以下となっているデータについては，今回の実験中にはビット誤りが発生しなかったことを表す．

5 考察

Procedure 1 は，$K = 256, 1296$，$C = 3, 4$ のどの組み合わせの場合でも，ターボ復号と比較すると，特にエラーフロア領域においては，ビット誤り率が低くなっていることがわかる．Procedure 2 は，拘束長が 4 の符号においては Procedure1 に比べビット誤り率が改善されていることがわかる．これについては，以下のターボ符号の性質から考えることができる．畳込み符号及び BCJR アルゴリズムの性質により，復号ビット誤りはトレリス線図上での近接ビットに連続して生じることが多い．この点で考えた場合，ターボ復号器において要素復号器 1 で隣接しているビットは，要素復号器 2 ではある程度間隔をもって配置されている方が望ましい．本研究では RB-HARQ 手法を用いることで，どのビットに誤りが発生しているかをインターリーブ操作を行う前に，$t = 1$ 時の復号器の出力による信頼度（対数尤度比）から推測することが可能である[3]．これを用いて信頼度の低いビッ

[3] 逆に考えると，2 つめの符号器の符号化の際に，1 つめの符号器の誤り箇所があらかじめ推測できない（帰還通信路を用いない）通常のターボ符号の場合は，ランダムなインターリーブ手法が平均的に良いというと言うことが出来る．

ト間に一定の間隔をもって配置することで，その符号構造がターボ復号器として望ましい形になり，ビット誤り率が改善されたと言える[4]．しかし拘束長が3の場合においては，Procedure2による明確な改善は見られない．これは，要素符号として拘束長が3の符号をもつターボ符号は，符号としての性能（重み分布等）があまり良くないため，このような復号アルゴリズムを改良を行っても，誤り率の改善にはなりづらいためと考えられる．

6 まとめ

本稿では，帰還通信路を用いた新たな誤り制御方式を提案した．提案法はターボ符号よりもビット誤り率が低くなることをシミュレーションによって示し，フィードバックにより，誤り訂正性能が改善できることを示した．今回はシミュレーションによって，このインターリーブ操作の有効性を示したが，理論的な解釈を与えることが今後の課題として挙げられる．

謝辞

本研究を行うにあたり，数多くの御助言，御支援を賜りました松嶋研究室，平澤研究室の各氏に感謝致します．なお，本研究の一部は日本学術振興会科学研究費基盤(C)一般(No.15560338)の援助による．

参考文献

[1] S. Lin, D. Costello, Jr. "Error Control Coding: Fundamentals and Applications," Englewood Cliffs, NJ: Prentice-Hall, 1983.

[2] J. M. Shea, "Reliability-based hybrid ARQ," IEE Electronics Letter, vol. 38, pp. 644-645, June 2002.

[3] L. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, vol. 20, pp. 284-287, Mar. 1974.

[4] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in Proc. *IEEE Int. Conf. Commun.*, vol. 2, pp. 1064-1070, May. 1993.

[5] Hanjo Kim, John M. Shea, "New turbo-ARQ techniques based on estimated reliabilities," in Proc. *IEEE Wireless Communications and Networking Conf.*, pp.843-848, vol. 4, Mar. 2003.

[6] A. Roongta, J. M. Shea, "Reliability-based Hybrid ARQ using Convolutional Codes" in Proc. *IEEE Int. Conf. on Communications*, vol. 4, pp. 2889-2893, May. 2003.

[7] 小林直人, 小泉大城, 松嶋敏泰, 平澤茂一, "再帰的組織畳込み符号を利用したReliability Based Hybrid ARQについての研究," 信学技報, Sep. 2005.

[4] 今回提案したインターリーブのアルゴリズムは，信頼度が低い，という基準のみを考慮しており，要素復号器1で隣接しているかどうかは考慮していない．しかしながら，畳込み符号及びBCJRアルゴリズムの性質により（特に通信路の雑音が比較的小さい場合は）信頼度の低いビット集合に隣接したビットが含まれるケースは多く，結果的に「要素復号器1において隣接している信頼度の低いビットが，要素復号器2においては必ずある程度の間隔をもって配置される」ことが多くなる．

A Procedure 2のインターリーブ手法

以下，任意の順序付き集合Aに対し，その要素を$A_i(1 \leq i \leq |A|)$と表記する．情報系列長Kがある自然数aの4乗となる場合を考える．Zを要素数Kの順序付き集合とし，その要素を$Z_i = \Omega_i^{(t-1)}(1 \leq i \leq \delta)$とする．$\delta+1 \leq i \leq K$の範囲の要素については，$\Omega^{(t-1)}$に含まれていない情報系列のインデックスを順に並べたものとする．S^A, S^B, S^C, Tを順序つき集合とし，$T_i(1 \leq i \leq K)$の値をi番目の入力に対してのインターリーバの出力順序とする．shuffle(S)を，任意の順序付き集合Sを要素順序をランダムに置換する関数とする．

```
begin
r := a random number from 1 to K;
b := (K/a) − a^2;
s_1 := s_2 := s_3 := 1;
for i := 1 to a^2 do S^A_i := Z_i;
shuffle(S^A);
for i := 1 to b do S^B_i = Z_{i+a^2};
shuffle(S^B);
for i := 1 to K − b do S^C_i = Z_{i+(K/a)} ;
shuffle(S^C);

for i := 1 to K do
  begin
  if i mod a^2 = 0 then
    T_{(i+r mod K)} := S^A_{s_1};
    s_1 := s_1 + 1;
  else if i mod a = 0 then
    T_{(i+r mod K)} := S^B_{s_2};
    s_2 := s_2 + 1;
  else
    T_{(i+r mod K)} = S^C_{s_3};
    s_3 = s_3 + 1;
  end
end
```

The 28th Symposium on Information Theory
and Its Applications (SITA2005)
Onna, Okinawa, Japan, Nov. 20–23, 2005

統計的決定理論に基づく有限受信バッファSR ARQ方式
A Selective-Repeat ARQ Scheme with Finite Receiver Buffer Based on Statistical Decision Theory

雨宮 康二* 小林 直人* 松嶋 敏泰*
Koji Amemiya Naoto Kobayashi Toshiyasu Matsushima

Abstract— The Selective-Repeat ARQ scheme is capable of providing superior throughput performance independent of round trip delay, but requires excessively large receiver buffers. There are many researches about Selective-Repeat ARQ schemes with finite-length receiver buffer. We formulate a Selective-Repeat ARQ scheme based on statistical decision theory, and derive ARQ scheme which maximizes throughput with receiver buffer size, round trip delay, block error rate.

Keywords—ARQ, Statistical Decision Theory, Selective-Repeat, Finite Receiver Buffer, Round Trip Delay

1 はじめに

ARQ（Automatic Repeat reQuest；自動再送要求）方式は誤り検出符号を用いて伝送誤りを検出し，再送要求信号を送信者へ送り返すことで，同一のメッセージブロックを再度受け取ることにより誤り訂正を行う．ARQ方式は高い信頼性を必要とするデータ通信において広く用いられている．ARQの一方式であるSR（Selective-Repeat；選択再送）ARQ方式は最も効率の良い方式として知られているが，完全な動作を得るためには理論上無限容量の受信バッファを必要とする [1]．そのため，有限バッファ上での動作を保証する有限バッファSR方式が多数研究されている [1]-[4]．

バッファ容量を有限に制限するとバッファオーバーフローが生じ伝送効率が悪くなる．バッファオーバーフローの発生を抑えるため，同一メッセージを連続して多重送信する，連続多重送信選択再送（Continuous Multiple Transmissions；CMT）方式が考案されている [2]．CMT方式ではブロック毎の再送要求回数に応じて多重送信数を変化させているが，これらの多重送信数についての理論的な保証はない．

本稿では統計的決定理論の立場から，有限バッファSR方式の定式化をし，与えられた受信バッファ容量 M，往復伝搬遅延 N，ブロック誤り率 P_B のもとでスループット最大化に対して最適な送信ブロック制御方式を提案し，スループットの理論的な評価を行う．さらにシミュレーションにより従来研究，無限バッファSR方式との比較により提案方式の性能評価を行う．

2 準備

2.1 有限バッファSR ARQ方式

本稿で扱う有限バッファSR ARQ方式の問題設定について説明する．送信者は誤り検出符号を用いて符号化したメッセージブロックを送信する．送信するブロックには送受信者間で管理が行いやすいように通し番号がつけられている．メッセージの伝送中，送信ブロックにはブロック誤り率 P_B の確率で誤りが生じると仮定する．受信者は受信したブロックに誤りが検出されない場合にはACK，誤りが検出された場合には再送要求を表すNAKの確認信号を返信する．確認信号を受信した後，送信者は次に送信するブロックを選択し送信を行う．往復伝搬遅延 N は，ブロックの送信が終わってからそのブロックに対する確認信号を受信するまでの間に送信するブロック数を表す．受信者は有限の容量 M の受信バッファを用意しており，受信したブロックをバッファに格納する．バッファに格納されたブロックはそれより小さな番号のブロックが全て正しく受信されるとバッファから開放され上位層へ送られる．ここで，伝送効率を表す評価基準としてスループット η を次式のように定義する．

$$\eta = \frac{\text{上位層へ送られたブロック数}}{\text{全送信ブロック数}}. \qquad (1)$$

ブロック誤り率 P_B が大きく再送が起きやすい状況やバッファ容量 M が小さい場合にはバッファオーバーフローが起こりやすく，スループットが悪化しやすい．

有限バッファSR ARQ方式の動作例を図1に示す．

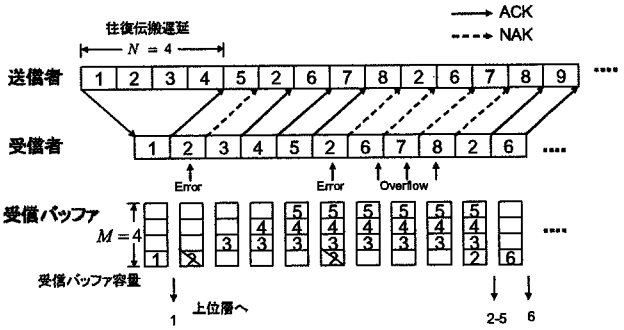


図 1: 有限バッファSR ARQ方式の動作例

2.2 連続多重送信選択再送方式

バッファオーバーフローによるスループットの悪化を軽減するため，再送を要求されたブロックについて多重送信（同一ブロックを連続して数回送る）を行う連続多重送信選択再送（Continuous Multiple Transmissions；CMT）方式が考案されている [2]．

受信バッファ容量が $M = q \cdot N$（q は正整数）であるとき，CMT方式では同一メッセージブロックに対する多重送信数をそのブロックの再送が何回目であるかによって次のように変化させる．各ブロックの初めての送信時はブロックを n_0 個連続して送信する．送信した n_0 個のブロック全てに対してNAKが返信された場合には，送信者は同じブロックを n_1 個連続して送信する．以下，$i(i \leq q)$ 回目の再送要求に対して n_i 個のブロックを多重送信する．$i > q$ 回を超える再送要求に対しては n_q 個のブロックを多重送信する．送信したブロックに対して一つでもACKが受信された場合には次のブロックを n_0 個送信する．$q = 1, n_0 = 1, n_1 = 2$ としたCMT方式の動作例を図2に示す．文献 [2] では多重送信数 n_i をシミュレーションにより決定しており，決定された n_i には理論的な保証はない．

* 早稲田大学理工学部経営システム工学科，東京都 Dept.of Industrial & Management Systems Engineering, School of Science and Engineering, Waseda University, Tokyo, 169-8555 Japan. Email : amemiya@matsu.mgmt.waseda.ac.jp

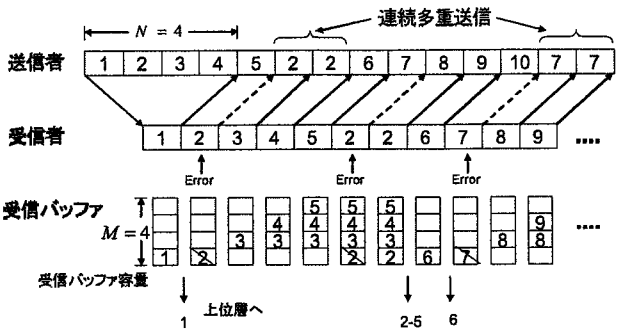


図 2: CMT 方式 ($n_0 = 1, n_1 = 2$)

3 統計的決定理論にもとづく定式化

本稿では統計的決定理論にもとづいて，受信バッファ容量 M，往復伝搬遅延 N，ブロック誤り率 P_B を用いて，有限バッファSR方式の定式化を行う．さらにスループットを評価基準としたとき，最適な送信ブロック選択方式を提案し，スループットの理論的な評価式を与える．

本稿では簡単のため次のような仮定を用いる．ブロックの誤りは必ず検出できるものとする．確認信号の信頼性は十分高く，誤って判別されることはないとする．バッファオーバーフローが生じたときには，バッファが用意されていないブロックに対しては誤りの有無にかかわらずNAKが返信され，ブロックは破棄される．

3.1 状態表現と状態遷移の仕方

単位時間辺りに1個のメッセージブロックの送信を行うとし，ブロックを送信する全回数を T とする．有限バッファSR ARQ方式において，時刻 $t \in T$ までに送信者が確認したブロック毎の確認信号と，時刻 $t-N+1 \sim t-1$ の間に送信したブロックの番号を時刻 t における送信者の状態 $s_t \in S$ として表現する．各時刻 t において，送信者は時刻 $t-N+1$ に送信したブロックに対する確認信号を受信後，時刻 t に送信するブロック $a_t \in A$ を決定する．a_t を便宜上，送信者の時刻 t における行動と呼ぶ．

時刻 $t+1$ の状態 s_{t+1} は時刻 t における状態 s_t と行動 a_t によって状態遷移確率 $p(s_{t+1}|s_t, a_t)$ に従い確率的に定まる．また，時刻 t の状態 s_t，行動 a_t，時刻 $t+1$ の状態 s_{t+1} が定まれば，時刻 $t+1-\lfloor \frac{N-1}{2} \rfloor$ に，受信バッファから開放され上位層へ送られたブロック数 $r(s_t, a_t, s_{t+1}) \in \{0, 1, \ldots, M\}$ を求めることが可能である．取りうる状態の集合 S と行動の集合 A，状態の遷移の仕方についての詳細は付録に記す．

3.2 効用関数と期待効用関数

時刻 $t = 1, \ldots, T+N$（ただし，時刻 $t = T+1$ 以降はブロックの送信は行わないものとする）における状態が定まれば，各時刻に何番のブロックが送信されたのか，どのような確認信号が返信されたのか，何番のブロックが受信バッファから開放され上位層へ送られたのかが再現できる．時刻 $t = 1, \ldots, T+N$ までの状態の系列 s^{T+N} と行動の系列 a^{T+N} が与えられたもとで，受信バッファから上位層へ送られたブロックの総数を効用関数と呼び次式により定義する．

$$u_1(s^{T+N}, a^{T+N}) = \sum_{t=1}^{T+N} r(s_t, a_t, s_{t+1}). \tag{2}$$

このとき，スループットは (1) 式を用いて，

$$\eta = u_1(s^{T+N}, a^{T+N})/T, \tag{3}$$

で計算できる．

時刻 $t = 1$ における状態 s_1 が与えられたもとで，各時刻 t における状態 s_t に遷移する確率により効用関数の期待値を取ったものを期待効用関数と呼び次式で定義する．

$$v_1(s^{T+N}, a^{T+N}) = \sum_{s_2, \ldots, s_{T+N}} p(s_2, \ldots, s_{T+N}|s_1, a^{T+N}) \times u_1(s^{T+N}, a^{T+N}). \tag{4}$$

(4) 式の右辺は再帰的構造を持ち，状態 s_t において行動 a_t を選択したときの時刻 t 以降の期待効用関数 $V_t(s_t, a_t)$ は次式のように再帰式で表せる [5]．

$$V_t(s_t, a_t) = \sum_{s_{t+1}} p(s_{t+1}|s_t, a_t) \times \{r(s_t, a_t, s_{t+1}) + V_{t+1}(s_{t+1}, a_{t+1})\}. \tag{5}$$

3.3 最適送信ブロック選択方式

時刻 t の状態 s_t において期待効用関数を最大化する行動を選択した場合の時刻 t 以降の期待効用関数を次式で定義する．

$$V_t^*(s_t) = \max_{a_t \in A} \Big\{ \sum_{s_{t+1}} p(s_{t+1}|s_t, a_t) \times \{r(s_t, a_t, s_{t+1}) + V_{t+1}^*(s_{t+1})\} \Big\}. \tag{6}$$

状態 s_t において期待効用関数を最大化する行動を状態 s_t における最適行動と呼び，次式で定義する．

$$a_t^*(s_t) = \arg\max_{a_t \in A} \Big\{ \sum_{s_{t+1}} p(s_{t+1}|s_t, a_t) \times \{r(s_t, a_t, s_{t+1}) + V_{t+1}^*(s_{t+1})\} \Big\}. \tag{7}$$

最適送信ブロック選択方式では時刻 t の状態 s_t において，(7) 式の最適行動を選択する．

【定理 1】 最適送信ブロック選択方式は，受信バッファ容量 M，往復伝搬遅延 N，ブロック誤り率 P_B，ブロック送信数 T の有限バッファSR ARQ方式において，(1) 式で表されるスループットの期待値を最大化する目的に対し最適な送信ブロック選択方式となる．また，このときスループットの期待値 η^* は

$$\eta^* = V_1^*(s_1)/T, \tag{8}$$

により計算することができる． □

(証明) (6), (7) 式より明らか．

期待効用関数は動的計画法のアルゴリズムで計算される [5]．全ての $s_{T+N+1} \in S$ について $V_{T+N+1}(s_{T+N+1}) = 0$ とし，(6) 式の再帰式を用い，$t = 1$ まで計算することにより各時刻 t の状態 s_t における期待効用関数，最適行動が求められる．

4 性能評価

本章では，シミュレーションにより提案手法と CMT 方式，無限バッファ容量 SR ARQ 方式の比較を行う．

4.1 シミュレーション条件

往復伝搬遅延 $N = 3$ とし，提案手法においてはバッファ数 M を $3 \sim 9$ に変えて実験を行う．CMT 方式においては M を N の整数倍の $M = 3, 6, 9$ とする．ブロック誤り率を $P_B = 0.01 \sim 0.99$ に設定して各誤り率において $T = 1{,}000{,}000$ ブロック送信を行い (1) 式のスループットを用いて評価をする．スループットの上界となる無限バッファSR 方式については $\eta = 1 - P_B$ を用いて計算する．なお，CMT 方式の多重送信数 n_i については各誤り率において $n_0 = n_1 = \ldots = n_q = 1$ から $n_0 = \ldots = n_q = 8$ まで全ての組み合わせを実験し，スループットが最大となるパラメータを用いた．

4.2 シミュレーション結果

バッファ数 $M = 3, 6, 9$ とした場合の提案手法（理論値，実験値），CMT 方式，無限バッファSR 方式（理論値）の実験結果を示す（図 3-5）．なお，計算量を軽減するため提案手法の理論値は $T = 1000$ として (8) 式を計算した値を用いている．

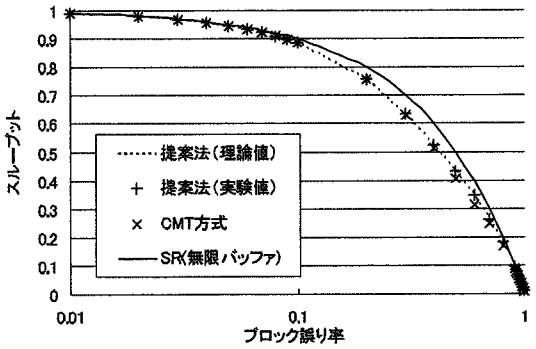


図 3: 提案手法，CMT 方式の比較 ($N = 3, M = 3$)

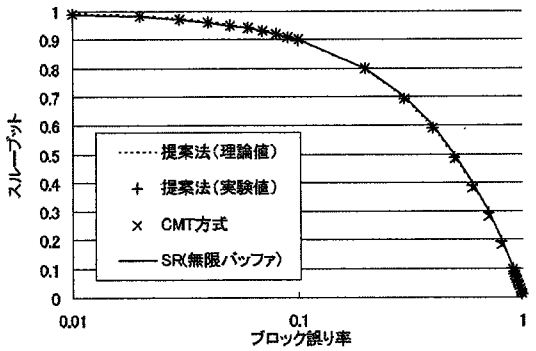


図 4: 提案手法，CMT 方式の比較 ($N = 3, M = 6$)

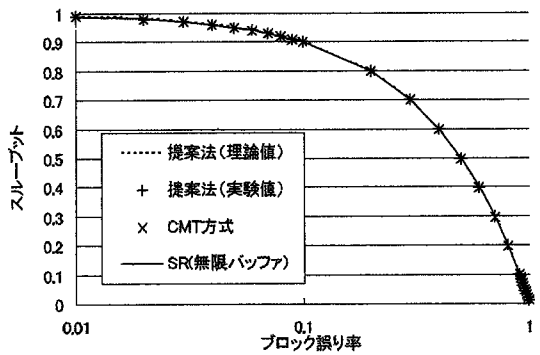


図 5: 提案手法，CMT 方式の比較 ($N = 3, M = 9$)

提案手法において，ブロック誤り率 $P_B = 0.1, 0.5, 0.9$ で，バッファ数を $M = 3 \sim 9$ に変えたときの理論値と実験値のスループットの変化を示す（図 6-8）．

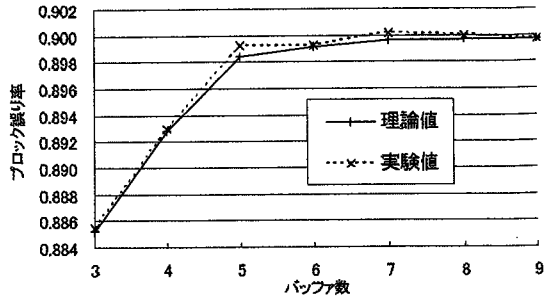


図 6: 提案手法のバッファ数による比較 ($P_B = 0.1$)

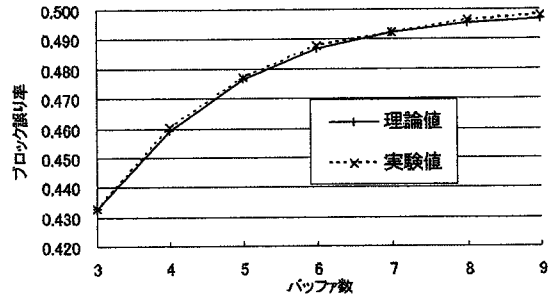


図 7: 提案手法のバッファ数による比較 ($P_B = 0.5$)

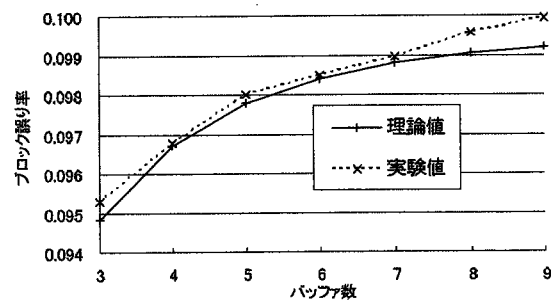


図 8: 提案手法のバッファ数による比較 ($P_B = 0.9$)

4.3 考察

提案手法と CMT 方式 [2] を比較する．図 3〜5 において，提案手法は，CMT 方式よりも高いスループットを示しているが，その差はわずかとなっている．これは往復伝搬遅延 N の値が小さく，方式によるスループットの差が出難い条件であったためと，CMT 方式はスループットの期待値最大化に対し最適となっている提案手法と似通ったブロック選択を行っているためと考えられる．バッファ容量 $M = 3$，ブロック誤り率 $P_B \geq 0.4$ の条件では CMT 方式と提案手法のスループットの差が大きくなっている．CMT 方式は受信バッファの状態を考慮せず，各ブロックの再送要求回数により，多重送信数を決定するため，受信バッファにブロックが溜まりブロックの多重送信数を増やした方が良い状態でも決まった多重送信数でしか送信がなされないためと考えられる．

バッファ数を変えた場合の提案手法の評価については，バッファ数が増えるに従いスループットの上界である無限バッファSR 方式の理論式 $\eta = 1 - P_B$ に漸近して行く様子が分かる．バッファ数は往復伝搬遅延の倍程度用意されていれば，無限バッファSR 方式と比較しても十分なスループットが得られることが分かった．また提案手法の理論値と実験値は良好な一致をみた．

5 まとめ

本稿では，有限バッファSR ARQ方式を統計的決定理論の立場から定式化を行い，与えられたバッファ容量M，往復伝搬遅延N，ブロック誤り率P_Bからスループットの期待値を最大化する最適送信ブロック選択方式を提案し，理論的な評価を行った．

本稿で行った定式化を利用して，行動集合に制約を加えることにより，有限バッファSR ARQ方式についての従来研究のスループットの理論的な評価を行うことが可能である．

通信路のブロック誤り率が時間によって変化する場合や誤り訂正符号と組み合わせたHybrid ARQ方式への拡張などが今後の課題としてあげられる．

謝辞

本研究を行うにあたり，数多くの御助言，御支援を賜りました松嶋研究室，平澤研究室の各氏に感謝致します．なお，本研究の一部は日本学術振興会科学研究費基盤(C)一般(No.15560338)の援助によ る．

参考文献

[1] S. Lin and D. Costello, Jr.,"Error Control Coding: Fundamentals and Applications,"Englewood Cliffs, NJ: Prentice-Hall, 1983.

[2] 楠堂忠夫，岡育生，藤原値賀人,"連続多重送信選択再送方式に関する考察,"電子情報通信学会論文誌 B-I, Vol.J81-B-I, No.8, pp.531-539, 1998.

[3] 西田梯彦，岡育生，藤原値賀人,"適応型フレーム分割多重再送方式,"電子情報通信学会論文誌 B-I, Vol.J85-B-I, No.4, pp.453-461, 2002.

[4] G. Benelli,"A Selective ARQ Protocol with a Finite-Length Buffer,"IEEE Trans. Commun., Vol.41, No.7, pp.1102-1111, 1993.

[5] M.L. Puterman,"Markov Decision Processes: Discrete Stochastic Dynamic Programming,"John Wiley & Sons, 1994.

付録

A 有限バッファSR ARQ方式の定式化

A.1 状態表現と状態集合について

時刻tの送信者の状態s_tは，送信者が受信した時刻$t-N+1$までに送信したブロックに対する確認信号の情報（時刻$t-\lfloor\frac{N-1}{2}\rfloor$の受信バッファの状態；これを確認信号履歴と呼ぶ）と時刻$t-N+1\sim t-1$の間に送信したブロックの情報（送信番号履歴と呼ぶ）により表現される．

$$s_t=(x_0,x_1,\ldots,x_{M-1}|a_{t-1},a_{t-2},\ldots,a_{t-N+1}). \quad (9)$$

ここで，確認信号履歴$x_i\in\{0,1\},(i=0,\ldots,M-1)$は$x_0$を送信者がACKを受け取っていない最小番号のブロックと考え，x_0+i番のブロックに対してACKを確認している場合には$x_i=1$，ACKを確認していない場合には$x_i=0$とする．x_0は常に0の値を取る．送信番号履歴$a_j\in\{-M,-M+1,\ldots,M+N-2\},(j=t-1,\ldots,t-N+1)$は時刻$t-N+1\sim t-1$の間に送信したブロック番号を$x_0$を基準に表したものである．図9を例に用いて説明する．時刻$t=8$の状態は$s_8=(x_0,x_1,x_2,x_3|a_7,a_6,a_5)=(0,1,1,0|0,0,3)$となる．確認信号履歴は，基準となる（送信者がACKを受け取っていない最小のブロック番号）x_0のブロック番号は2であり，x_1,x_2に該当するブロック番号3,4については既にACKを受け取っているので$x_1=1$，$x_2=1$となる．ブロック番号5についてはまだACKを確認していないので$x_3=0$となる．送信番号履歴は時刻$t=7,6$に送ったブロック番号は2であるから，x_0のブロック番号を基準にし，$a_7=a_6=2-2=0$となる．時刻$t=5$に送ったブロック番号は5であるから$a_5=5-2=3$となる．

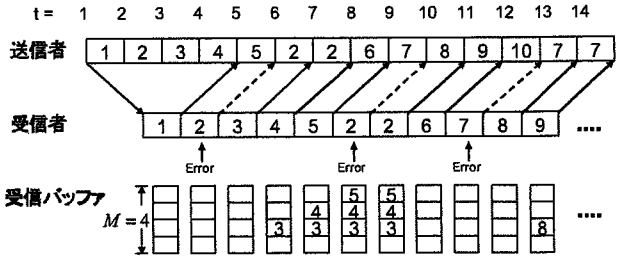


図9: 有限バッファSR ARQ方式$(N=4,M=4)$

状態集合Sは$S=\{(0,0^{M-1}|(-M)^{N-1}),\ldots,(0,1^{M-1}|(M+N-2)^{N-1})\}$となり，状態数は最大で$2^{M-1}\times(2M+N-1)^{N-1}$となる．ただし，$y_j\leq -1$は同じ状態と見なせることや，送信ブロック番号の選択に制約を設けることで状態数は大幅に削減できる．また行動集合Aは$A=\{0,1,\ldots,M+N-2\}$と定義される．

A.2 状態遷移の仕方

状態s_tにおける確認信号履歴$x_i,(i=0,\ldots,M-1)$を$s_t(x_i)$，送信番号履歴$a_j,(j=t-1,\ldots,t-N+1)$を$s_t(a_j)$と表現する．時刻$t+1$の始めに$s_t(a_{t-N+1})$に対する確認信号を受信する．

以下に状態s_tから状態s_{t+1}への状態遷移の仕方と状態遷移確率，利得関数を示す．

- $s_t(a_{t-N+1})=0$の場合
  - $s_t(a_{t-N+1})$に対する確認信号がACK
    受信バッファから上位層へ送られたブロック数rを計算し，次のように状態遷移する．
    $r=1$;
    for $(i=1;\ i<M;\ i{+}{+})$ {
    if $(s_t(x_i)==1)$ $r{+}{+}$;
    else if $(s_t(x_i)==0)$ break;
    }
    for $(i=0;i<M-r;\ i{+}{+})$ $s_{t+1}(x_i)=s_t(x_{i+r})$;
    for $(i=M-r;\ i<M;\ i{+}{+})$ $s_{t+1}(x_i)=0$;
    $s_{t+1}(a_{t-1})=a_t-r$;
    for $(j=t-N+1;j<t-1;j{+}{+})$ {
    $s_{t+1}(a_j)=s_t(a_{j+1})-r$;
    }
    状態遷移確率は$p(s_{t+1}|s_t,a_t)=1-P_B$，
    利得関数は$r(s_t,a_t,s_{t+1})=r$となる．
  - $s_t(a_{t-N+1})$に対する確認信号がNAK
    状態遷移の仕方は[操作1]．$p(s_{t+1}|s_t,a_t)=P_B$，
    $r(s_t,a_t,s_{t+1})=0$.
- $s_t(a_{t-N+1})\in\{1,\ldots,M-1\}$かつ$s_t(x_{s_t(a_{t-N+1})})=0$の場合
  - $s_t(a_{t-N+1})$に対する確認信号がACK
    状態遷移の仕方は[操作1]を行った後，
    $s_{t+1}(x_{s_t(a_{t-N+1})})=1$とする．
    $p(s_{t+1}|s_t,a_t)=1-P_B$，$r(s_t,a_t,s_{t+1})=0$.
  - $s_t(a_{t-N+1})$に対する確認信号がNAK
    状態遷移の仕方は[操作1]．
    $p(s_{t+1}|s_t,a_t)=P_B$，$r(s_t,a_t,s_{t+1})=0$.
- その他の場合
  状態遷移の仕方は[操作1]．
  $p(s_{t+1}|s_t,a_t)=1$，$r(s_t,a_t,s_{t+1})=0$.

[操作1]
for $(i=0;\ i<M;\ i{+}{+})$ $s_{t+1}(x_i)=s_t(x_i)$;
$s_{t+1}(a_{t-1})=a_t$;
for $(j=t-N+1;\ j<t-1;\ j{+}{+})$ $s_{t+1}(a_j)=s_t(a_{j+1})$;

The 28th Symposium on Information Theory
and Its Applications (SITA2005)
Onna, Okinawa, Japan, Nov. 20–23, 2005

単一ループをもつグラフィカルモデルにおける確率伝播型アルゴリズムに関する一考察

A Note on Belief Propagation Algorithm in Graphical Models with a Single Loop

岡野 洋平* 小泉 大城* 松嶋 敏泰*
Youhei Okano Daiki Koizumi Toshiyasu Matsushima

Abstract— In the field of probabilistic reasoning, the peformance of Belief Propagation Algorithm(BP) in graphical models with a single loop is analyzed to some degree. The approximate posterior probability as the output of BP can be calculated by the largest eigenvalue of a matrix which is product of all transition matrices, whereas the exact posterior probability can be calculated by the all eigenvalues of the matrix. By using this fact, we examine the factors related to the second eigenvalue against the largest eigenvalue, and analyze the performance of BP in Tailbiting(TB) codes whose graphical models have a single loop.

Keywords— Probabilistic Reasoning, Belief Propagation Algorithm, Tailbiting Codes, Single Loop, Eigenvalue

1 はじめに

確率モデルをグラフィカルモデルで表現したときに，ノード間で確率情報を伝播して周辺事後確率を計算する確率伝播型アルゴリズム (BP) がある [1]．BP は，グラフィカルモデルが木構造の場合は真の事後確率を計算するが，ループをもつ場合は近似事後確率を計算する．複数のループをもつ場合，BP の収束は保証されず近似精度の解析は非常に困難であるが，単一ループをもつ場合，収束が保証され収束後の近似精度は解析されている [2]．

畳み込み符号器の初期状態と終了状態を等しく符号化した Tailbiting(TB) 畳み込み符号は，確率モデルをグラフィカルモデルで表現すると単一ループとなる [3]．復号アルゴリズムとして Tailbiting BCJR アルゴリズム (TB-BCJR) が知られ，これは BP と等価である [4]．

[2] の結果を TB 畳み込み符号に適用すると，BP による近似事後確率は，全受信系列を得たもとでの畳み込み符号器の状態遷移確率行列の第 1 固有値・ベクトルだけを用いて計算していることが分かる．真の事後確率は，全固有値・ベクトルを用いれば計算できることが分かるので，当然，第 2 固有値・ベクトルを用いれば BP による近似事後確率の近似精度は上がると考えられる．

そこで本研究では，まず第 2 固有値・ベクトルを求める方法，及びそれを用いた BP による近似事後確率の修正法を示し，シミュレーションによりいくつかの条件のもとでの第 1 固有値に対する第 2 固有値の影響を調べ，TB 畳み込み符号における BP の復号性能を評価する．

* 〒 169-8555 東京都新宿区大久保 3-4-1 早稲田大学理工学部経営システム工学科 Dept. of of Industrial and Management Systems Engineering, School of Science and Engineering, Ookubo3-4-1, Shinjyukuku, Tokyo, 169-8555 Japan. E-mail: okano@matsu.mgmt.waseda.ac.jp

2 従来研究

2.1 Talibiting 畳み込み符号 [3]

本研究では符号化率 1/2 の Feedforward 畳み込み符号器を用いる [5]．遅延素子数を v とおくと状態数は 2^v となり，t 時点の状態は $S_t = \{0, 1, \cdots, 2^v - 1\}$ となる $(t = \{1, 2, \cdots, N\})$．TB 畳み込み符号は，0 時点の符号器に情報記号系列 $u_1 u_2 \cdots u_N$ の末尾 $u_{N-v+1} u_{N-v+2} \cdots u_N$ を入力する．その結果，情報記号系列を送った後，$S_N = S_0$ が成立する．

2.2 確率伝播型アルゴリズム [1][2][4]

u_t の推定には最大事後確率復号法を用いる．Feedforward 畳み込み符号器では u_t は S_t により一意に定まる．よって，TB 畳み込み符号における BP は全受信系列 $Y_1^N = Y_1 Y_2 \cdots Y_N$ を得たもとでの t 時点の状態の事後確率 $P(S_t = i|Y_1^N)$ の近似値 $P^{(1)}(S_t = i|Y_1^N)$ を計算する．

まず以下の関数を定義する．

$$\alpha_t(i) \equiv P\left(S_t = i, Y_1^t\right), \quad (1)$$

$$\beta_t(j) \equiv P\left(Y_{t+1}^N | S_t = j\right), \quad (2)$$

$$\Gamma_t(i,j) \equiv P\left(S_t = j, Y_t | S_{t-1} = i\right). \quad (3)$$

(1)(2)(3) を便宜上，以下のようにベクトル・行列表記する．

$$\alpha_t \equiv (\alpha_t(0), \alpha_t(1), \cdots, \alpha_t(2^v - 1)), \quad (4)$$

$$\beta_t \equiv (\beta_t(0), \beta_t(1), \cdots, \beta_t(2^v - 1)), \quad (5)$$

$$\Gamma_t \equiv \begin{bmatrix} \Gamma_t(0,0) & \cdots & \Gamma_t(0, 2^v - 1) \\ \vdots & \ddots & \vdots \\ \Gamma_t(2^v - 1, 0) & \cdots & \Gamma_t(2^v - 1, 2^v - 1) \end{bmatrix}. \quad (6)$$

以下に近似事後確率の計算アルゴリズムを示す．但し，c は正規化定数とし，$t \geq N$ の場合は $\Gamma_t \equiv \Gamma_{t \bmod N}$ とする．

Step0: 初期化

$$\alpha_0 = (1/2^v, \cdots, 1/2^v), \quad (7)$$

$$\beta_N = (1/2^v, \cdots, 1/2^v). \quad (8)$$

Step1: 前向き計算

α_t が終了条件を満たすまで以下の操作を繰り返す．

$$\alpha_t = c\alpha_{t-1}\Gamma_t, \quad t = 1, \cdots, N, N+1, \cdots. \quad (9)$$

終了条件: $\|\alpha_t - \alpha_{t-N}\| \ll \epsilon$.

Step2: 後向き計算

β_t が終了条件を満たすまで以下の操作を繰り返す.

$$\beta_t = c\Gamma_{t+1}\beta_{t+1}, \quad t = N-1, \cdots, 0, -1, \cdots . \tag{10}$$

終了条件: $\|\beta_t - \beta_{t+N}\| \ll \epsilon$.

Step3: 近似事後確率計算

α_t,β_t の収束値をそれぞれ $\alpha_t^{(1)}$,$\beta_t^{(1)}(t \equiv t \mod N)$ とし, 以下のように状態の近似事後確率を計算する.

$$c\alpha_t^{(1)}(i)\beta_t^{(1)}(i) = P^{(1)}\left(S_t = i|Y_1^N\right). \tag{11}$$

3 Tailbiting 畳み込み符号における確率伝播型アルゴリズムの性能

3.1 アルゴリズムの収束値と固有値との関係

TB 畳み込み符号は単一ループからなるグラフィカルモデルで表現できるので, BP によるノード間の確率情報の伝播は N 回後グラフィカルモデル上を一周する.

$$\begin{aligned} c\alpha_{t-N}\Gamma_{t+1} &= \alpha_{t-N+1}, & (12)\\ c\alpha_{t-N+1}\Gamma_{t+2} &= \alpha_{t-N+2}, & (13)\\ &\vdots \\ c\alpha_{t-1}\Gamma_t &= \alpha_t. & (14) \end{aligned}$$

ここで, $C_{t+1,t} \equiv \Gamma_{t+1}\Gamma_{t+2}\cdots\Gamma_N\Gamma_1\cdots\Gamma_t$ を定義すると, 以下が成立する (β_t についても同様に成立する).

$$\begin{aligned} c\alpha_{t-N}\Gamma_{t+1}\cdots\Gamma_N\Gamma_1\cdots\Gamma_t &= c\alpha_{t-N}C_{t+1,t} & (15)\\ &= \alpha_t. & (16) \end{aligned}$$

このとき, 以下の定理が成立する.

定理 1: $\alpha_t^{(1)}$ は $C_{t+1,t}$ の第 1 左固有ベクトルに等しい.

定理 2: $\beta_t^{(1)}$ は $C_{t+1,t}$ の第 1 右固有ベクトルに等しい.

定理 3: $C_{t+1,t}$ の固有値を $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_{2^v}|$ とするとアルゴリズムの収束の早さは $|\lambda_2/\lambda_1|$ に依存する.

証明 以下の補題より明らかである.

Power Method Lemma[2]:

t 回目の再帰計算による行ベクトルを b_t, $|\lambda_1| > |\lambda_2| > \cdots$ となる固有値, それに一致する固有ベクトルをもつ行列を A とする. このとき,

$$b_t A = b_{t+1}, \tag{17}$$

における b_t は, 初期値 b_0 に 0 がなければ $t \to \infty$ で A の第 1 固有ベクトルに収束し, その収束の早さは $|\lambda_2/\lambda_1|$ に依存する.

3.2 真の事後確率の計算方法

$C_{t+1,t}$ の対角要素を正規化した値は $P(S_t = i|Y_1^N)$ の真の値に等しい. 但し, $\sum_{S_t/S_1,\cdots,S_N}$ は S_t 以外の和, e_i は i 列目の要素が 1 でそれ以外は 0 の行ベクトルを意味する.

$$\begin{aligned} &P(S_t = i|Y_1^N)\\ &= c\sum_{S_t/S_1,\cdots,S_N} P(S_1,\cdots,S_t = i,\cdots,S_N,Y_1^N) & (18)\\ &= c\sum_{S_t/S_1,\cdots,S_N} P(S_{t+1},Y_{t+1}|S_t = i)\\ &\quad \cdots P(S_t = i, Y_t|S_{t-1}) & (19)\\ &= ce_i\Gamma_{t+1}\cdots\Gamma_N\Gamma_1\cdots\Gamma_t e_i^T & (20)\\ &= ce_i C_{t+1,t}e_i^T & (21)\\ &= cC_{t+1,t}(i,i) & (22)\\ &= C_{t+1,t}(i,i)/trace(C_{t+1,t}). & (23) \end{aligned}$$

3.3 アルゴリズムによる近似事後確率と真の事後確率との関係

$C_{t+1,t}$ を対角化すると以下が成立する.

$$C_{t+1,t} = R\Lambda R^{-1}. \tag{24}$$

行列 Λ:絶対値が大きい順に固有値を対角要素に並べた行列で対角要素以外は 0.

行列 R:固有値に一致する固有ベクトル行を並べた行列.

これを利用して, (21) 式に続く形で式展開を行うと以下が成立する.

$$\begin{aligned} &P(S_t = i|Y_1^N)\\ &= ce_i C_{t+1,t}e_i^T & (25)\\ &= ce_i R\Lambda R^{-1}e_i^T & (26)\\ &= c\sum_j R(i,j)\lambda_j R^{-1}(i,j) & (27)\\ &= c(\lambda_1 R(i,1)R^{-1}(i,1) + \lambda_2 R(i,2)R^{-1}(i,2)\\ &\quad + \cdots + \lambda_{2^v} R(i,2^v)R^{-1}(i,2^v)). & (28) \end{aligned}$$

ここで, $R(i,1)R^{-1}(i,1) = \alpha_t^{(1)}\beta_t^{(1)}$ であることから, BP による近似事後確率 $P^{(1)}\left(S_t = i|Y_1^N\right)$ は $C_{t+1,t}$ の第 1 固有値・ベクトルだけを用いて計算していることが分かる. さらに真の事後確率は $C_{t+1,t}$ の全固有値・ベクトルを用いれば計算できることも分かる.

4 近似事後確率の修正

4.1 第 2 固有値・ベクトルを用いた近似事後確率の修正法

第 2 固有値・ベクトルを用いれば, BP による近似事後確率を修正して近似精度をさらに高めることができる. 第 2 固有ベクトルを $\alpha_t^{(2)}$,$\beta_t^{(2)}$ とおくと, (28) 式から修正

した近似事後確率 $P^*(S_t = i|Y_1^L)$ は以下のようになる.

$$P^*(S_t = i|Y_1^L) \tag{29}$$
$$= c\left(\lambda_1 \alpha_t^{(1)}(i)\,\beta_t^{(1)}(i) + \lambda_2 \alpha_t^{(2)}(i)\,\beta_t^{(2)}(i)\right). \tag{30}$$

4.2 第 2 固有値・ベクトルの求め方

べき乗法を用いる [6]. 行列 $B = C_{t+1,t} - \lambda_1 E$ を定義すると以下が成立する.

$$\alpha_0 B = c_1(\lambda_1 - \lambda_1)\alpha_t^{(1)} + c_2(\lambda_2 - \lambda_1)\alpha_t^{(2)} + \cdots + c_{2^v}(\lambda_{2^v} - \lambda_1)\alpha_t^{(2^v)}. \tag{31}$$

$C_{t+1,t}$ を用いて第 1 固有値を求めた方法と同様に, B を用いて収束するまで再帰計算を行うと, $\alpha_t^{(2)}$ に収束し, λ_2 が求められる ($\beta_t^{(2)}$ についても同様に求められる).

5 シミュレーションによる性能評価

5.1 実験内容

TB 畳み込み符号における BP の性能を, 様々な条件のもとでの第 1 固有値に対する第 2 固有値の影響をみることで評価する.

5.2 実験条件

通信路は白色ガウス通信路を仮定し, 符号器は [5] による最小の復号誤り率を出力するものを用いる. 情報記号系列長 $N = \{5, 10, 20\}$, SN 比 $= \{0.00, 3.37, 6.00\}$, 遅延素子数 $v = \{2, 3, 4, 5\}$ の場合で実験を行う. 各実験, 実験回数を 10,000 とし, 復号誤り率, 第 1 固有値に対する第 2 固有値の比はともに実験回数で平均をとる.

5.3 実験 1

真の事後確率による復号誤り率と BP による復号誤り率を調べる. 表 1 に真の事後確率による復号誤り率, 表 2 に BP による復号誤り率の結果を示す.

表 1. 真の事後確率による復号誤り率

		SNR		
N	v	0	3.37	6
5	2	0.06922	0.00754	0.00002
	3	0.07056	0.00752	0.00014
	4	0.0584	0.0046	0.0002
10	2	0.07024	0.00367	0.00002
	3	0.07436	0.00393	0
	4	0.07274	0.00391	0
	5	0.07372	0.00264	0
20	2	0.07156	0.00367	0.00002
	3	0.071445	0.00301	0
	4	0.07002	0.000895	0
	5	0.0707	0.00126	0

表 2. BP による復号誤り率

		SNR		
N	v	0	3.37	6
5	2	0.06922	0.00754	0.00002
	3	0.07056	0.00752	0.00014
	4	0.0584	0.0046	0.0002
10	2	0.07024	0.0036	0.00002
	3	0.07465	0.00391	0
	4	0.07469	0.00395	0
	5	0.08126	0.00412	0
20	2	0.07156	0.00376	0.00002
	3	0.071445	0.00301	0
	4	0.070065	0.000955	0
	5	0.07095	0.001155	0

5.4 実験 2

遅延素子数と第 1 固有値に対する第 2 固有値の比の関係について SN 比を変えて調べる. 図 1 に $N = 5$, 図 2 に $N = 10$, 図 3 に $N = 20$ の結果を示す.

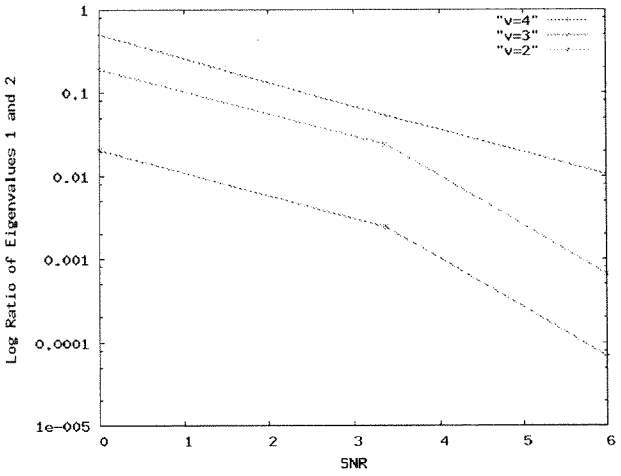


図 1: 実験 2 の結果 (情報系列長=5)

5.5 考察

実験 1 の結果より, 符号器の構成にもよるが, 遅延素子数 (v) が増えるに従い, 真の事後確率による復号誤り率と BP による復号誤り率とに差が出やすいことが分かる. 表 1,2 中の灰色で塗りつぶした部分は, 両者の誤り率に差が出たことを示している. このことから遅延素子数が増えるに従い, 第 2 固有値の影響が大きい傾向にあると考えられる. 実際, 実験 2 の結果により, 遅延素子数が大きくなるに従い, 第 1 固有値に対する第 2 固有値の比が大きくなっていることが分かる. また、系列長の長さ・SN 比が大きくなるに従い, 第 1 固有値に対する第 2 固有値の比は顕著に小さくなっている. このことから系列長の長さ・SN 比が大きい場合の BP の復号性能は非常に良いと考えられる.

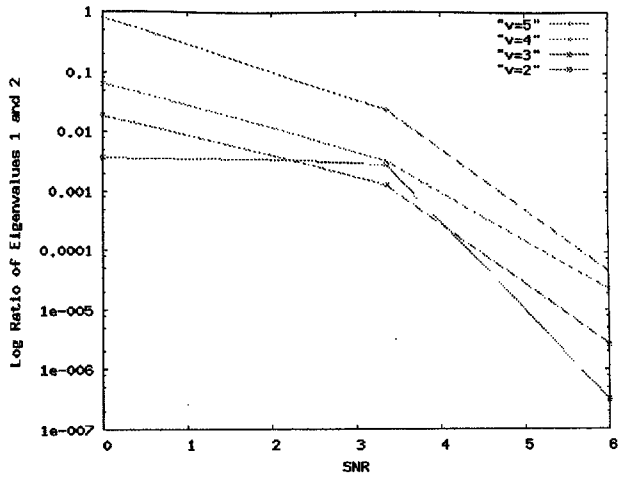


図 2: 実験 2 の結果 (情報系列長=10)

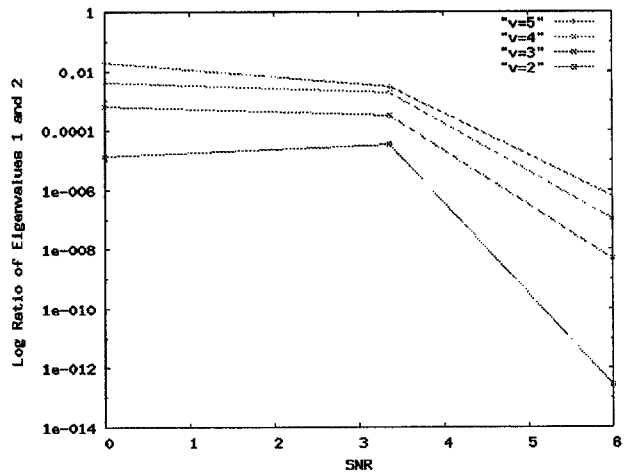


図 3: 実験 2 の結果 (情報系列長=20)

6 まとめ

本稿では，TB 畳み込み符号における BP の復号性能を第 1 固有値に対する第 2 固有値の影響を調べることで評価した．その結果，系列長の長さ・SN 比が大きい場合，遅延素子数が増えるに従い，第 2 固有値の影響は顕著に小さくなり，BP の復号性能は良いことが分かる．

今後の課題として，本稿で評価した実験回数による平均だけでなく，個々の系列についての固有地の比および誤り率等を調べ，どのような状態の遷移確率行列が与えられたときにこれらに差が出るのかを調べることが考えられる．またその際，本稿で提案した第 2 固有値・ベクトルを用いた近似事後確率の修正法による復号性能を検討することも挙げられる．

謝辞

本研究を行うにあたり，数多くの御助言，御支援を賜りました松嶋研究室，平澤研究室の各氏に感謝致します．なお，本研究の一部は日本学術振興会科学研究費基盤 (C) 一般 (No.15560338) の援助による．

文献

[1] T. Matsushima, and S. Hirasawa, "A Formalization of Generalized Probabilistic Reasoning and Its Procedures on Junction Trees," submitted to *IEEE Trans. Inf. Theory.*

[2] Y. Weiss, "Correctness of Local Probability Propagation in Graphical Models with Loops," *Neural Computation*, 12, pp. 1-41, 2000.

[3] H. H. Ma and J. K. Wolf, "On Tail Biting Convolutional Codes," *IEEE Trans. Commun.*, vol. COM-34, pp. 104-111. Feb. 1986.

[4] John B. Anderson and Stephen M. Hladik, "Tailbiting MAP Decoders," *IEEE Journal on Selected Areas in Commun.*, vol. 16, No. 2, pp. 297-302. Feb. 1988.

[5] John B. Anderson, "Best Short Rate 1/2 Tailbiting Codes for the Bit-Error Rate Criterion," *IEEE Trans. Commun.*, vol. 48, No. 4, pp. 597-610. Apr. 2000.

[6] B. Carnahan, H. A. Luther and J. O. Wilkes 著, 藤田宏 訳, "計算機による数値計算法," 科学技術出版社, 1982.

The 28th Symposium on Information Theory
and Its Applications (SITA2005)
Onna, Okinawa, Japan, Nov. 20–23, 2005

使用ユーザが変化するDS/CDMAシステムにおける ベイズ最適なマルチユーザ検出について

Bayes Optimal Multi User Detection for DS/CDMA Systems with Time-Varying Group of Active Users

堀井 俊佑* 須子 統太 * 松嶋 敏泰 *
Shunsuke Horii Tota Suko Toshiyasu Matsushima

Abstract— In this paper we consider multiuser detection (MUD) scheme for direct sequence code division multiple access (DS/CDMA) systems. A change in group of active users in a mobile communication environment will degrade the performance of MUD if it cannnot adapt quickly to take into account the new set of interference parameters. We propose an optimal MUD scheme with reference to the Bayes criterion in DS/CDMA systems which consider change in groupe of active users. Computer simulation is used to obtain the efficiency of this scheme.

Keywords— CDMA,multiuser detection,Bayesian decision theory,Time-Varying Group of Active Users

1 はじめに

CDMA 環境では，複数ユーザが同一の通信路を共有するため，すべてのユーザの情報ビットを同時に推定することで受信復調性能を向上できる．このような復調受信機をマルチユーザ受信機 (MUD) と呼ぶ [1]．移動体通信環境では使用するユーザの集合が変化し，MUD の性能に大きく影響を与えることがある [2]．使用ユーザの集合が変化する DS/CDMA システムに対する従来研究では，使用ユーザの集合を推定して一意に決定し，決定された使用ユーザの集合のもとで最適な MUD を構成している [2][3][4]．しかし，実際に使用しているユーザの集合と推定されたユーザの集合との間には誤差が生じるため，一般には推定されたユーザの集合に対して最適な MUD が，ビット誤り率等の目的関数を最良にするとは限らない．

本研究では，考えられるユーザの集合すべてについて事後確率でビット誤り率の期待値をとり，それを最小にする MUD を提案する．この方式は，統計的決定理論の立場からビット誤り率に対し，ベイズ基準のもとで最適な方式となっている．またシミュレーションにより従来研究との比較を行い提案方式の性能評価を行い，その有効性を示す．

* 〒 169-8555 東京都新宿区大久保 3-4-1 早稲田大学理工学部経営システム工学科 Dept. of Industrial & Management Systems Engineering, School of Science and Engineering, Ookubo 3-4-1, Shinjyukuku, Tokyo, 169-8555 Japan. E-mail: horii@matsu.mgmt.waseda.ac.jp

2 準備

2.1 使用ユーザの変化について

本研究では，時間とともに使用ユーザの集合が変化する．最大で K 人が使用可能な CDMA システムを考える．一般性を失わず，ユーザ 1 は常に存在するものと仮定し，ユーザ 1 が送信した情報ビットを推定するものとする．今，z_t^i を t 時点でユーザ i がシステムを利用していれば 1，そうでなければ 0 をとる変数とする．ユーザ 1 以外のユーザの存在を表すベクトルを $\boldsymbol{z}_t = [z_t^2 \quad \cdots \quad z_t^K] \in \mathcal{Z}$ で表す．$\boldsymbol{z}_t$ は状態数が 2^{K-1} のマルコフモデルにより表現される．以下，$\boldsymbol{z}_t$ を状態と呼ぶ．本研究では，ユーザ 1 は状態遷移確率行列を知っているものと仮定する．この仮定は，各ユーザが使用を開始する確率・使用を終える確率を知っていると仮定することと同値である．

2.2 DS/CDMA システム

DS/CDMA システムの一般的な構造は図 1 のようになる．

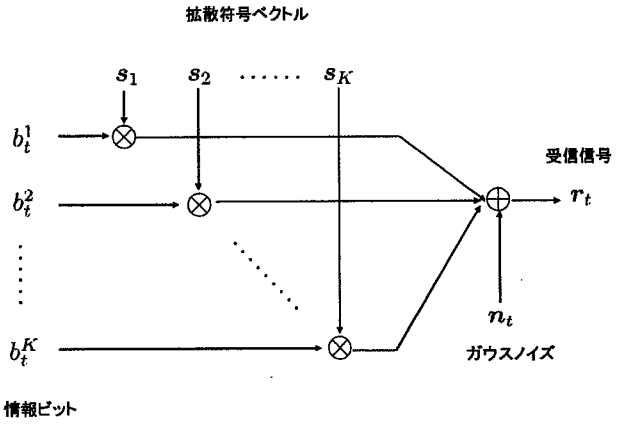


図 1: DS/CDMA システムの構成

単純化のための仮定として，すべてのユーザは互いに完全に同期したタイミングで通信を行っているものとする．ある時点 t に注目し，その時点においてユーザ i が情報ビット $b_t^i \in \{\pm 1\}$ を等確率で送信するものとする．情報ビットひとつ当たりの符号チップ数を N とおく．各ユーザの拡散符号を簡単のため，互いに独立な $\{-1, 1\}$

のランダムビット列として扱う．ユーザ i の拡散符号ベクトルを $\boldsymbol{s}_i = [s_{i1} \quad \cdots \quad s_{iN}]$ とする．DS/CDMA システムでは，各ユーザは情報ビットを拡散符号によって変調したものを搬送波変調を施して送信する．ユーザ i の信号電力を a_i とし，白色ガウス通信路を想定すると，受信側では搬送波復調の結果として，信号，

$$\boldsymbol{r}_t = a_1 b_t^1 \boldsymbol{s}_1 + \sum_{i=1}^{K} z_t^i a_i b_t^i \boldsymbol{s}_i + \sigma \boldsymbol{n}_t, \tag{1}$$

が受信される．$\boldsymbol{r}_t$ は N 次元のベクトルである．ここで，$\boldsymbol{n}_t$ は平均 0，共分散行列 $\boldsymbol{I}_N$(I_N は $N \times N$ の単位行列)の正規分布に従う白色ガウス雑音ベクトルである．MUD に関する従来研究は，復調を行うユーザが持っている情報により以下のように分類することができる．

Case1 他ユーザの拡散符号・振幅が既知
Case2 他ユーザの拡散符号が既知，振幅が未知
Case3 他ユーザの拡散符号が未知，振幅が既知
Case4 他ユーザの拡散符号・振幅が未知

本研究では，特に Case1，Case2 の問題を扱う．

3 ベイズ基準に基づく最適な MUD

3.1 Case1 における最適な MUD

$\boldsymbol{r}_t$ に対する尤度関数は以下で与えられる．

$$P(\boldsymbol{r}_t|\boldsymbol{z}_t, \boldsymbol{b}_t) = \frac{1}{(2\pi\sigma^2)^N} \times \exp\left(-\frac{1}{2\sigma^2}||\boldsymbol{r}_t - a_1 b_t^1 \boldsymbol{s}_1 - \sum_{i=2}^{K} z_t^i a_i b_t^i \boldsymbol{s}_i||\right). \tag{2}$$

ここで，$||\cdot||$ は二乗ノルムを表し，$\boldsymbol{b}_t = \{b_t^i\}$ である．t 時点でユーザ 1 が送信した情報ビット b_t^1 の推定方法として，$\boldsymbol{r}^t = \boldsymbol{r}_1, \cdots, \boldsymbol{r}_t$ が与えられたもとでの推定値 $\hat{b}_t^1(\boldsymbol{r}^t)$ を求める問題を扱う．推定に対する損失を 0-1 損失として以下で定義する．

$$Loss = (b_t^1 - \hat{b}_t^1(\boldsymbol{r}^t)) = \begin{cases} 0 & \text{if} \quad b_t^1 = \hat{b}_t^1(\boldsymbol{r}^t) \\ 1 & \text{if} \quad b_t^1 \neq \hat{b}_t^1(\boldsymbol{r}^t). \end{cases} \tag{3}$$

この損失関数は実際に得られるデータに依存した値をとる．そのため，損失関数にデータの出現確率で期待値をとったものを危険関数として以下で定義する．

$$Risk = \int Loss \times P(\boldsymbol{r}^t|\boldsymbol{z}^t) d\boldsymbol{r}^t. \tag{4}$$

ここで，$\boldsymbol{z}^t = \boldsymbol{z}_1, \cdots, \boldsymbol{z}_t$ であり，

$$P(\boldsymbol{r}^t|\boldsymbol{z}^t) = \prod_{j=1}^{t} P(\boldsymbol{r}_j|\boldsymbol{z}_j), \tag{5}$$

$$P(\boldsymbol{r}_j|\boldsymbol{z}_j) = \sum_{\boldsymbol{b}_j} P(\boldsymbol{r}_j|\boldsymbol{z}_j, \boldsymbol{b}_j) P(\boldsymbol{b}_j), \tag{6}$$

である．この危険関数を最小にする $\hat{b}_t^1$ を求めたいが，$\boldsymbol{z}^t$ により危険関数を最小にする推定値は異なるため，任意の $\boldsymbol{z}^t$ について最小にする $\hat{b}_t^1$ は存在しない [5]．そこで危険関数を状態の系列 $\boldsymbol{z}^t$ の事前分布 $P(\boldsymbol{z}^t)$ で期待値をとったベイズリスクを以下で定義する．

$$BR = \sum_{\boldsymbol{z}^t} (Risk \times P(\boldsymbol{z}^t)). \tag{7}$$

ベイズ基準のもとで最適な推定はこのベイズリスクを最小にする推定を行うものである．本研究におけるベイズ最適な推定値は次式で表される．

$$\hat{b}_t^{1*} = \arg \max_{b_t^1 \in \{\pm 1\}} P(b_t^1|\boldsymbol{r}^t). \tag{8}$$

ここで，

$$P(b_t^1|\boldsymbol{r}^t) = \sum_{\boldsymbol{z}_t} P(b_t^1|\boldsymbol{z}_t, \boldsymbol{r}^t) P(\boldsymbol{z}_t|\boldsymbol{r}^t), \tag{9}$$

である．

3.2 Case2 における最適な MUD

尤度関数 $P(\boldsymbol{r}^t|\boldsymbol{z}^t, \boldsymbol{b}_t, \boldsymbol{a})$ は以下で与えられる．

$$P(\boldsymbol{r}^t|\boldsymbol{z}^t, \boldsymbol{b}_t, \boldsymbol{a}) = \frac{1}{(2\pi\sigma^2)^N} \times \exp\left(-\frac{1}{2\sigma^2}||\boldsymbol{r}_t - a_1 b_t^1 \boldsymbol{s}_1 - \sum_{i=2}^{K} z_t^i a_i b_t^i \boldsymbol{s}_i||\right). \tag{10}$$

他ユーザの振幅が未知の場合も，既知の場合同様に危険関数・ベイズリスクを以下のように定義する．

$$Risk = \int Loss \times P(\boldsymbol{r}^t|\boldsymbol{z}^t, \boldsymbol{a}) d\boldsymbol{r}^t, \tag{11}$$

$$BR = \sum_{\boldsymbol{z}^t} \int Risk \times P(\boldsymbol{a}|\boldsymbol{z}^t) P(\boldsymbol{z}^t) d\boldsymbol{a}. \tag{12}$$

ここで，

$$P(\boldsymbol{r}^t|\boldsymbol{z}^t, \boldsymbol{a}) = \sum_{\boldsymbol{b}_t} P(\boldsymbol{r}^t|\boldsymbol{z}^t, \boldsymbol{b}_t, \boldsymbol{a}) P(\boldsymbol{b}_t). \tag{13}$$

である．ベイズリスクを最小にする推定値は以下で与えられる．

$$b_t^{1*} = \arg \max_{b_t^1 \in \{\pm 1\}} P(b_t^1|\boldsymbol{r}^t). \tag{14}$$

この時，

$$P(b_t^1|\boldsymbol{r}^t) = \sum_{\boldsymbol{z}^t} P(\boldsymbol{z}^t|\boldsymbol{r}^t) P(b_t^1|\boldsymbol{z}^t, \boldsymbol{r}^t) = \sum_{\boldsymbol{z}^t} P(\boldsymbol{z}^t|\boldsymbol{r}^t) \int P(b_t^1|\boldsymbol{z}_t, \boldsymbol{r}_t, \boldsymbol{a}) P(\boldsymbol{a}|\boldsymbol{z}^t, \boldsymbol{r}^t) d\boldsymbol{a}, \tag{15}$$

である．(15) 式のうち，z^t の事後確率を表す $P(z^t|r^t)$ は，

$$P(z^t|r^t) \propto P(z^t)P(r^t|z^t) = P(z^t)\int P(r^t|z^t,a)P(a|z^t)da, \quad (16)$$

により求まり，状態 z^t のもとでの b_t^1 の事後確率を表す $P(b_t^1|z^t,r^t)$ は，

$$P(b_t^1|z^t,r^t) = \int P(b_t^1|z_t,r_t,a)P(a|z^t,r^t)da$$
$$= \int \sum_{b_t^2\cdots b_t^K} P(b_t^1,b_t^2,\cdots,b_t^K|z_t,r_t,a)P(a|z^t,r^t)da$$
$$\propto \sum_{b_t^2\cdots b_t^K} \int P(r_t|b_t^1,b_t^2,\cdots,b_t^K,z_t,a)P(a|z^t,r^t)da, \quad (17)$$

とすることで求めることができる．

3.3 Case1 におけるアルゴリズム

今回は受信信号を観測シンボルとした隠れマルコフモデルを考え，隠れマルコフモデルにおける逐次的アルゴリズムを用いる．まず観測した受信信号と，1 時点前までのデータによって計算した状態の事前確率をもとに状態の事後確率を求める．そして，その状態のもとでユーザ 1 の情報ビットの事後確率を計算する．この値を全ての状態について事後確率で重みをとることで，ビット誤り率の期待値を計算できる．Case1 におけるアルゴリズムの詳細を以下に示す．

Case1 におけるベイズ最適な推定アルゴリズム

step1:

初期分布 $P(z_0)$ が全ての状態に対して与えられる．

step2:

現在を t 期とすると，z_t の事後確率はベイズの定理より，以下のように計算できる．

$$P(z_t|r^t) = \frac{P(r_t|z_t)P(z_t|r_1^{t-1})}{\sum_{z_t\in\mathcal{Z}} P(r_t|z_t)P(z_t|r_1^{t-1})}. \quad (18)$$

ここで，$P(r_t|z_t)$ は以下で与えられる．

$$P(r_t|z_t) = \sum_{b_t} P(r_t|z_t,b_t)P(b_t|z_t). \quad (19)$$

step3:

状態 z_t の元での b_t^1 の事後確率 $P(b_t^1|z_t,r^t)$ を求める．

$$P(b_t^1|z_t,r^t) = \sum_{b_t^2,\cdots,b_t^K} P(b_t^1,b_t^2,\cdots,b_t^K|z_t,r^t). \quad (20)$$

step4:

b_t^1 の事後確率 $P(b_t^1|r^t)$ を求め，b_t^1 の推定を行う．

$$P(b_t^1|r^t) = \sum_{z_t} P(b_t^1|z_t,r^t)P(z_t|r^t), \quad (21)$$
$$\hat{b}_t^{1*} = \arg\max_{b_t^1} P(b_t^1|r^t). \quad (22)$$

step5:

r^t を観測したもとでの状態 z_{t+1} の事後確率を求める．

$$P(z_{t+1}|r^t) = \sum_{z_t} P(z_{t+1}|z_t)P(z_t|r^t). \quad (23)$$

$t+1$ 時点の情報ビット推定に移る (t を $t+1$ にして step2 に戻る)．

上記のアルゴリズムを，各時点で繰り返すことが，各時点においてビット誤り率の期待値を最小化する目的に対し，ベイズ基準のもとで最適となっている．

4 シミュレーション

4.1 シミュレーション目的

Case1 において，ベイズ最適な推定法と従来の推定法の性能をシミュレーションにより比較する．ユーザ数が変化するモデルに対する MUD に関する研究として，

- 状態の事後確率を計算し，現在の状態を一意に決定する [3].
- 各時点で得られる受信信号から，状態の尤度を計算し，尤度比検定により現在の状態を一意に決定する [4].

が挙げられる．これらの研究はすべて，何らかの基準のもとで状態を推定し，推定された状態のもとで MUD を構成している．これらの文献ではアルゴリズムの評価を状態推定の誤り確率や，状態変化の検出遅れにより行っている．そのため，提案法と従来研究を直接比較するのは難しい．そこで本研究では，事後確率最大の状態を現在の状態とし，そのもとで b_1 の事後確率を求めるアルゴリズム (以下，状態推定方式と呼ぶ) と比較を行う．また限界の指標として，使用しているユーザの集合を知ったもとで b_1 の事後確率を求めるアルゴリズムとの比較を行う．

両方式の基準を以下に整理し，その違いを明確にする．

(a) 状態推定方式：状態を推定することに関して最適化する．

$$\hat{b}_t^1 = \arg\max_{b_t^1} P(b_t^1|z_t^*),\ z_t^* = \arg\max_{z_t} P(z_t|r^t). \quad (24)$$

(b) 提案方式：ビット誤り率について最適化する．

$$\hat{b}_t^1 = \arg\max_{b_t^1} \left[P(b_t^1|z_t)P(z_t|r^t)\right]. \quad (25)$$

状態推定方式のアルゴリズムは，提案方式と同様に，各状態にいる事後確率を逐次的に更新していき，各時点で事後確率が最大の状態のもとで，情報ビットの事後確率を計算する方式である．これは，その時点にいる状態がどの状態であるかを一意に推定するという目的に対して最適な推定となっている．一方提案方式は，ビット誤り率を最小化するという目的に対して最適な推定方式である．どちらも，すべての状態の事後確率を計算するため，計算量のオーダは同じである．

4.2 シミュレーション条件

- 拡散符号の長さ：$N = 21$
- E_b/N_0：15dB
- 各ユーザの振幅：$a_1 = 1$，$a_i = 2, i \neq 1$
- 各ユーザが使用を開始･終了する確率 $(i = 2, \cdots, K)$:
  $P(z_{t+1}^i = 1 | z_t^i = 0) = P(z_{t+1}^i = 0 | z_t^i = 1) = 0.95$
- 送信ビット数：1000 ビット
- 実験回数：100 回

4.3 結果

最大使用可能ユーザ数が $K = 3, 4, 5, 6$ の場合における，各方式のビット誤り率の比較を行った．

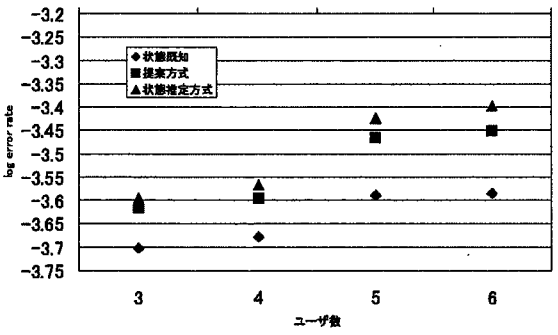


図 2: ビット誤り率による各手法の比較

5 考察

提案方式が，ビット誤り率を最小化する目的に対し，ベイズ基準のもとで理論的に最適であることは明らかであるが，実験結果を見ても，それを確認することが出来る．しかし，全ての実験で提案方式と状態推定方式の差は小さい．これは，各時点で全状態の事後確率を計算した時に，ある 1 つの状態の事後確率が，他の状態の事後確率に比べて極端に大きくなることが原因であると考えられる．これにより，(25) 式のように全ての状態について事後確率で重み付けを行っても，事後確率最大の状態のみが寄与してくるため，事後確率最大の状態を推定した場合との差が殆ど無くなり，情報ビットの推定結果に違いが出てこなくなるものと考えられる．1 つの状態の事後確率のみが大きくなる原因として，Case1 では未知パラメータが無いため，状態の推定が行いやすい環境になっていることが考えられる．

6 まとめ

使用ユーザの集合が時点により変化する DS/CDMA システムにおいて，ベイズ基準のもとで最適な MUD を提案した．提案方式は従来の推定方式と比較して，同程度の計算量でより小さいビット誤り率を達成する．今回は Case1 でシミュレーションを行い，両者の差は殆ど無かった．しかし，Case2 や他の Case では全ての状態で重みをとる場合と，状態を 1 つに推定する場合の差が出てくると考えられる．Case2 において，ベイズ最適なアルゴリズムを構成するためには，振幅の事後分布の計算を行なう必要性から，計算量が増加する．そこで，今後の課題として提案方式の計算量削減が挙げられる．本研究より，重み付けに寄与する状態の数は少ない (他の状態に比べて事後確率の高い状態が少数存在する) ことが予想されるため，計算量削減方法として，事後確率の大きい状態を上手く見つけることができれば，全ての状態の重み付けをしなくてもベイズ最適に近い推定が可能になると予測される．

謝辞

本研究を行うにあたり，数多くの御助言，御支援を賜りました松嶋研究室，平澤研究室の各氏に感謝致します．なお，本研究の一部は日本学術振興会科学研究費基盤 (C) 一般 (No.15560338) の援助による．

文献

[1] 丸林元，中川正雄，河野隆二，"スペクトル拡散通信とその応用，" コロナ社, 1998.

[2] T. Oskiper and H.V. Poor, "On line activity detection in a multiuser environment using the matrix CUSUM algorithm,"IEEE Trans. Inf.Theory, Vol.48, No.2, pp.477-491, 2002.

[3] T.N. Bui, V. Krishnamurthy and H.V. Poor, "Online Bayesian Activity Detection in DS/CDMA Networks,"IEEE Trans. Signal Processing, Vol.53, NO.1, pp.371-375, 2005.

[4] K.W. Halford and M.B. Pearce, "New-User Identification in a CDMA System,"IEEE Trans. Communication, Vol.46, N.1, pp.144-155, 1998.

[5] J.O. Berger, "Statistical Decision Theory and Bayesian Analysis,"Springer, 1985.

The 28th Symposium on Information Theory
and Its Applications (SITA2005)
Onna, Okinawa, Japan, Nov. 20-23, 2005

BW変換を用いたユニバーサル符号化アルゴリズムに関する研究
A Note on Universal Coding Algorithm with the BWT

須子 統太* 松嶋 敏泰* 平澤 茂一*
Tota Suko Toshiyasu Matsushima Shigeichi Hirasawa

Abstract— In this paper, we discuss the source coding algorithm for FSMX sources. Recently, there are many researches about the universal coding algorithm for FSMX sources using Barrows-Wheeler transform(BWT). BWT has the character to convert FSMX sources into p.i.i.d. sources. We propose an efficient Bayes coding algorithm for FSMX sources using this character . We also show asymptotic code length of this codes.

Keywords— Burrows-Wheeler transform (BWT), universal coding, FSMX sources, tree sources, Bayes coding

1 はじめに

本研究では，FSMX 情報源に対するユニバーサル符号化アルゴリズムを扱う．近年，FSMX 情報源に対して Barrows-Wheeler transform（BW 変換）を用いた符号化アルゴリズムが提案されている．[3][5][6] これらは，FSMX 情報源から出力された系列を BW 変換すると，p.i.i.d. 情報源と呼ばれる情報源からの出力とみなせるという性質を利用している．

p.i.i.d. 情報源とは，情報源系列が一定のパラメータに従って出力するが，ある時点でそのパラメータが突然変化する情報源である．そのため区分的には定常な情報源とみなせるが，全体的にはパラメータが不連続に変化するため非定常な情報源であると言える．[2]

FSMX 情報源のモデルが未知である場合，情報源系列を BW 変換すると，変化時点が未知の p.i.i.d. 情報源からの出力系列とみなすことができる．[4] そのため，従来では Baron らによる MDL 基準を用いて変化時点を推定し符号化するアルゴリズム [5] や，Effros らによる全ての変化パターンを重み付ける符号化アルゴリズム [3] などが提案されている．

本研究では，まず FSMX 情報源に対して，BW 変換を用いたベイズ符号の構成法を示す．そのもとで，効率的なベイズ符号化アルゴリズムを提案する．また，提案した符号の平均符号長の漸近式を示す．

* 〒 169-8555 東京都新宿区大久保 3-4-1 早稲田大学理工学部経営システム工学科．Dept. of Industrial & Management Systems Engineering, School of Science and Engineering, Ookubo 3-4-1, Shinjyukuku, Tokyo, 169-8555 Japan. E-mail: suko@matsu.mgmt.waseda.ac.jp

2 FSMX 情報源

FSMX 情報源とは過去の有限系列から現在のシンボルの発生確率が決まる情報源で，マルコフ過程の一種である．また，FSMX 情報源は階層型モデルであるため，モデル $m \in M$ とそのモデルのもとでのパラメータ $\theta^m \in \Theta^m$ により定まる．

1時点目から n 時点目までの情報源系列を $x_1^n : x_1 x_2 \cdots x_n$ とすると，t 時点での状態は過去の系列 x_1^{t-1} により定まる．モデル m における状態の集合を S_m とし，x_1^{t-1} から状態 $s \in S_m$ への写像を $s(x_1^{t-1})$ とする．今，情報源アルファベットを $a \in A = \{a|0 \le a \le l\}$ とすると，各状態 s でのシンボルの出現確率は l 次元パラメータベクトル $\theta^s = \{\theta_1^s, \theta_2^s, \cdots, \theta_l^s\}$ によって決まる．この時，x_1^{t-1} が出力されたもとでの x_t の出現確率は $P(x_t|\theta^{s(x_1^{t-1})}, s(x_1^{t-1}))$ で定義される．l 次元パラメータベクトル θ^s は各状態 s に対応しているので，あるFSMX情報源モデル m のパラメータ θ^m は $\theta^m = \{\theta^s|s \in S_m\}$ で表すことができる．

FSMX 情報源は完全木で表現することができる．木におけるそれぞれの枝にシンボル $a \in A$ を割り当てる，葉ノードは状態と一対一で対応するため葉ノードに s を割り当てることができる．モデル m における状態の集合 S_m は木表現における葉ノードの集合であると言える．また，木における葉ノードから根ノードまでのパスをコンテクストともしくはポストフィクスと呼ぶ．

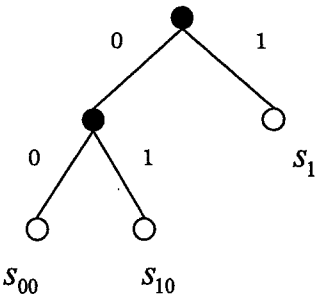


図 1: 2 元 FSMX 情報源モデルの例（木表現）

3 BW 変換

BW 変換は近年数多くのユニバーサル符号化アルゴリズムに用いられている手法である．以下で BW 変換の

アルゴリズムと，FSMX 情報源に対する BW 変換の性質について述べる．

3.1 BW 変換アルゴリズム

【BW 変換アルゴリズム】

step-1. x_1^n を逆順に並べた系列 $\bar{x}_1^n$ を生成し，系列の最後に終端記号 \$（但し，$\$ > l$ とする）を加えた系列 $\bar{x}_1^n\$$ を作る．

step-2. $\bar{x}_1^n\$$ を左巡回シフトさせた系列を n 個生成し縦に並べ，$(n+1)\times(n+1)$ の行列 $M(\bar{x}_1^n\$)$ を作る．

step-3. 行列 $M(\bar{x}_1^n\$)$ の各行を，左端のシンボルから順に辞書順にソートし，ソート後の行列を $\tilde{M}(\bar{x}_1^n\$)$ とする．

step-4. $\tilde{M}(\bar{x}_1^n\$)$ の右端の列を \$ を除いて上から $y_1, y_2, \cdots$ とし系列 y_1^n を出力．(この時，\$ の列番号を保持し，復号に用いるが詳細については省略する．)

例 3.1 $x_1^9 = 101001101$ の場合．

$\bar{x}_1^9\$ = 101100101\$$

$y_1^9 = 101110001$

$M(\bar{x}_1^9\$)$ $\qquad$ $\tilde{M}(\bar{x}_1^9\$)$

$$\begin{pmatrix} 101100101\$ \\ 01100101\$1 \\ 1100101\$10 \\ 100101\$101 \\ 00101\$1011 \\ 0101\$10110 \\ 101\$101100 \\ 01\$1011001 \\ 1\$10110010 \\ \$101100101 \end{pmatrix} \longrightarrow \begin{pmatrix} 00101\$1011 \\ 0101\$10110 \\ 01100101\$1 \\ 01\$1011001 \\ 100101\$101 \\ 101100101\$ \\ 101\$101100 \\ 1100101\$10 \\ 1\$10110010 \\ \$101100101 \end{pmatrix}$$

図 2: BW 変換の例

3.2 FSMX 情報源に対する BW 変換

FSMX 情報源からの出力系列 x_1^n を BW 変換すると，変換後の系列 y_1^n は p.i.i.d. 情報源からの出力系列とみなせることが Effros によって示されている．[4]

p.i.i.d. 情報源はパラメータの変化パターン $\omega \in \Omega$ と，その変化パターンのもとでのパラメータ θ^ω によって定まる確率分布で表される．今，c 回目にパラメータの変化が起きた時点を u_c とする．但し，$u_0 = 1$ とする．この時，パラメータの変化パターン ω を u_c を用いて $\omega = \{u_0, u_1, \cdots, u_{C_\omega}\}$ と表す．また，$\theta^\omega = \{\theta^{u_0}, \theta^{u_1}, \cdots, \theta^{uC_\omega}\}$ とする．但し，θ^{u_c} は u_c 時点から $u_{c+1}-1$ 時点までの系列が従う l 次元パラメータベクトルとする．このとき，y_1^N を p.i.i.d. 情報源からの出力系列とすると y_1^n の出現確率は以下で表される．

$$P(y_1^n|\theta^\omega, \omega) = \prod_{c=0}^{C_\omega} P(y_{u_c}^{u_{c+1}-1}|\theta^{u_c}). \qquad (1)$$

但し，$u_{C_\omega+1} = n+1$ とする．

例 3.2 例 *3.1* で用いた $x_1^9 = 101001101$ が図 *2* の *FSMX* 情報源モデルからの出力であるとする．

$\tilde{M}(\bar{x}_1^9\$)$ の一番右上のシンボルに注目する．このシンボルに対するコンテンクストが一番上の行の左から順に 00101 と現れていることが分かる．同様に，右一列のシンボルは全て同じ行の左から順にコンテクストが現れている．そのため，図 *2* の *FSMX* 情報源モデルを仮定した場合，y_1 は状態 s_{00},y_2, y_3, y_4 は状態 s_{01}，$y_5, \cdots, y_8$ は状態 s_1 からの出力であると言える．このように，*BW* 変換後の系列 y_1^n は同じ状態からの出力シンボルが連続して出力するため，*p.i.i.d.* 情報源とみなすことができる．

FSMX 情報源 $m \in M$ から BW 変換による p.i.i.d. 情報源 $\omega \in \Omega$ への写像を $\phi : M \to \Omega$ とする．また，$\Phi_M = \{\phi(m)|m \in M\}$ とすると，$\Phi_M \subset \Omega$ となり，Φ_M に含まれない $\omega \in \Omega$ が存在することが分かる．

4 BW 変換を用いたベイズ符号

情報源系列の出現確率を仮定すれば算術符号を用いることで符号化が可能になる．そのためユニバーサル情報源符号化の問題は系列の出現確率を決定する問題に帰着される．ベイズ符号ではベイズ基準のもとで冗長度を最小にする出現確率（以下，符号化確率と呼ぶ）を決定する．

以降，FSMX 情報源モデル m および，パラメータ θ^m を未知とし，m, θ^m の事前分布 $P(m), P(\theta^m)$ は既知であるとする．

4.1 ベイズ符号化確率

FSMX 情報源に対する，一括型，逐次型のベイズ符号の符号化確率はそれぞれ以下の式で求められる．[1]

$$P_c(x_1^n) = \sum_{m\in M} \int_{\theta^m} P(x_1^n|\theta^m, m) P(\theta^m|m) d\theta^m P(m). \qquad (2)$$

$$P_c(x_t|x_1^{t-1}) = \sum_{m\in M} \int_{\theta^m} P(x_t|x_1^{t-1}, \theta^m, m) P(\theta^m|m, x_1^{t-1}) d\theta^m \times P(m|x_1^{t-1}). \qquad (3)$$

系列 x_1^n に対して一括型ベイズ符号の符号長と逐次型ベイズ符号の符号長は等しくなることが知られている．[1]

$$-\log P_c(x_1^n) = -\sum_{t=1}^{n} \log P_c(x_t|x_1^{t-1}). \qquad (4)$$

今，x_1^n を BW 変換して得られた系列 y_1^n を逐次符号化することを考える．y_1^n は前述の通り，p.i.i.d. 情報源

からの出力とみなすことができる．その為，y_t に対するベイズ符号化確率は (3) 式を変形することで次式で与えられることが分かる．

$$P_c(y_t|y_1^{t-1}) = \sum_{\omega\in\Phi_M}\int_{\theta^\omega} P(y_t|y_1^{t-1},\theta^\omega,\omega)P(\theta^\omega|\omega,y_1^{y-1})d\theta^\omega \times P(\omega|y_1^{t-1}). \quad (5)$$

BW 変換後の系列を上式の符号化確率を用いて符号化することでベイズ符号が構成可能であるが，y_1^n だけでは Φ_M に含まれる ω を特定することができない．また，仮に分かったとしても $|\Phi_M|$ 自体 n に対して指数的に増えていくため計算量が膨大になる．そこで，BW 変換の出力に新たな情報を加えることで効率的に上式を計算するアルゴリズムを次に示す．

4.2 効率的符号化アルゴリズム

効率的符号化アルゴリズムを示す為に，FSMX 情報源モデル m の事前分布に対し以下の仮定を置く．

仮定 4.1 *FSMX* 情報源モデル m の事前分布 $P(m)$ もしくは，$P(S_m)$ を次式で仮定する．

$$P(m) = \prod_{s\in N_m\setminus S_m}(1-q(s))\prod_{s'\in S_m} q(s'). \quad (6)$$

この時，N_m は *FSMX* 情報源 m を木表現した際の全てのノード集合とする．また，

$$q(s) = \frac{P(S_{m_s})}{\sum_{S\in S_s^{comp}} P(S_{m_s}\cup S)}, \quad (7)$$

とする．但し，$m_s \in \{m|s\in S_m\}$ とし，S_s^{comp} は完全木を構成する s の子ノード集合のクラスであるとする．

仮定 4.2 $q(s) = 1-\gamma$ とし，全ての s において一定であるとする．

また，Φ_M を特定するための情報を付加するために，BW 変換における step-4 を以下の step-4′ に置き換える．(以降，step-4′ に置き換えたものを修正 BW 変換と呼ぶ．)

step-4′. $\tilde{M}(\bar{x}_1^n\$)$ の右端の列を \$ を除いて上から $y_1, y_2, \cdots$ とし系列 y_1^n を出力．$\tilde{M}(\bar{x}_1^n\$)$ の $i-1$ 番目の行と i 番目の行を左からみて，初めて一致しないシンボルが出た列番号を z_i とし，系列 z_1^n を出力．(但し，$z_1 = 0$ とする) また，i 行目の z_i 番目のシンボルが \$ の時 $d_i = 1$，それ以外は $d_i = 0$ とし，d_1^n を出力．(但し，$d_1 = 0$ とする)

次に，アルゴリズムに用いる変数として τ_t を定義する．τ_t は y_1^t において最後にパラメータの変化が起きた時点を表す変数とし，その取り得る集合を B_t とする．

以上を用いて，効率的符号化アルゴリズムを示す．

【効率的符号化アルゴリズム】

step-1. x_1^n に対し修正 BW 変換を行い，y_1^n, z_1^n, d_1^n を得る．

step-2. $t = 1, B_1 = \{1\}, P(\tau_t = 1) = 1$ とし，step-4 へ．

step-3. $d_t = 0$ の時は step-3a へ，$d_t = 1$ の時は step-3b へ．

step-3a $B' = \{\tau_{t-1} : z_{\tau_{t-1}} < z_t, \tau_{t-1}\in B_{t-1}\}$ とし，B' の中で最大の値を持つ要素を τ_{max} とする．
$B_t = B'\cup\{t\}$ とし，全ての $\tau_t \in B_t$ について次式を計算する．

$$P(\tau_t|y_1^{t-1}) = \begin{cases} \gamma^{z_t - z_{max}}P(\tau_{t-1}=\tau_{max}|y_1^{t-1}) & \tau_t = t, \\ (1-\gamma^{z_t-z_{max}})P(\tau_{t-1}=\tau_{max}|y_1^{t-1}) & \tau_t = \tau_{max}, \\ P(\tau_{t-1}|y_1^{t-1}) & \text{otherwise.} \end{cases} \quad (8)$$

step-3b $B_t = \{\tau_{t-1} : z_{\tau_{t-1}} < z_t, \tau_{t-1}\in B_{t-1}\}$，とし，全ての $\tau_t \in B_t$ について，

$$P(\tau_t|y_1^{t-1}) = P(\tau_{t-1}|y_1^{t-1}), \quad (9)$$

とする．

step-4. 次式で符号化確率を計算する．

$$P_c(y_t|y_1^{t-1}) = \sum_{\tau_t\in B_t} P_c(y_t|y_1^{t-1},\tau_t)P(\tau_t|y_1^{t-1}). \quad (10)$$

但し，

$$P_c(y_t|y_1^{t-1},\tau_t) = \int_{\theta^{\tau_t}} P(y_t|y_1^{t-1}\theta^{\tau_t}.\tau_t)P(\theta^{\tau_t}|\tau_t, y_1^{t-1})d\theta^{\tau_t}, \quad (11)$$

とする．上式はパラメータの事前分布にディリクレ分布を仮定した場合，次式で解析的に求める事ができる．

$$P_c(y_t|y_1^{t-1},\tau_t) = \frac{\nu(y_t|\tau_t)+\beta(y_t|\tau_t)}{\sum_{a=0}^{l}\{\nu(a|\tau_t)+\beta(a|\tau_t)\}}. \quad (12)$$

ここで，$\nu(a|\tau_t)$ は $y_{\tau_t}^{t-1}$ におけるシンボル a の生起回数とし，$\beta(a|\tau_t)$ はディリクレ分布のパラメータで既知とする．

step-5. $P(\tau_t|y_1^t)$ を次式で計算する．

$$P(\tau_t|y_1^t) = \frac{P_c(y_t|y_1^{t-1},\tau_t)P(\tau_t|y_1^{t-1})}{P_c(y_t|y_1^{t-1})}. \quad (13)$$

$t = t+1$ として step-3 へ．

4.3 効率的符号化アルゴリズムの計算量

効率的符号化アルゴリズムは大きく分けで，step-1 の修正 BW 変換部分と step-2 以降の符号化確率の計算部分に分けられる．修正 BW 変換部分にかかる計算量は，ソー

トにクイックソートを用いた場合，平均で $O(n\log n)$，最大で $O(n^2)$ かかる．次に，符号化確率の計算部分について考える．t 番目の記号を符号化する時にかかる計算量は $|B_t|$ に依存するため $O(|B_t|)$ となり，長さ n のブロックを符号化する時の計算量は $O(\sum_{t=1}^{n}|B_t|)$ となる．今，$|B_t| \leq t$ となるため，長さ n のブロックを符号化する時に最大で $O(n^2)$ の計算量がかかることが分かる．しかし，各時点で $|B_t| = t$ となることはまれであるため，平均的にはより少ない計算量しかかかっていないと考えられる．そこで $\sum_{t=1}^{n}|B_t|$ のサイズについての実験を行った．時点 $n = 1, 51, 101, \cdots, 1001$ において，事前分布 $P(m)$，$P(\theta^m)$ に従い m, θ^m をそれぞれ 10000 回発生させ，各 m, θ^m ごとに系列 x_1^n をランダムに 1 本出力した．ブロック n における $\sum_{i=1}^{t}|B_i|$ の平均値と最大の値 $((1+n)n/2)$ を図 3 に示す．図 3 より，平均的な計算量は最大の場合に比べ大幅に少ないことが分かる．

また従来，FSMX 情報源に対する効率的な符号化法として Matsushima らのアルゴリズムがあるが，FSMX 情報源モデルの最大次数が未知の場合，長さ n のブロックを符号化するのに必ず $O(n^2)$ の計算量がかかる．[8] それに対し，本研究で提案したアルゴリズムは最大の計算量が $O(n^2)$ であり，平均的には従来手法よりも計算量が削減されていると考えられる．

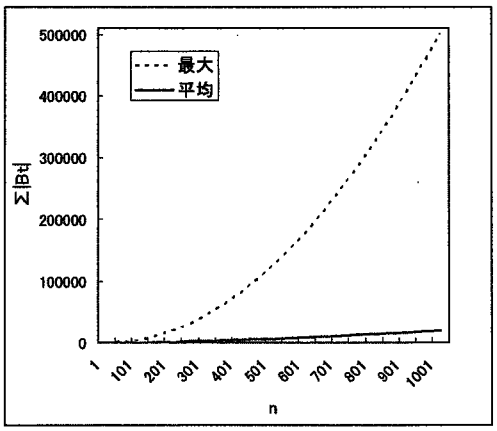


図 3: $\sum_{t=1}^{n}|B_t|$ のサイズ

5 平均符号長

効率的符号化アルゴリズムは，FSMX 情報源に対するベイズ符号化確率を計算している．その為，平均符号長の漸近式については，Goto らの結果 [7] をそのまま適応することができ，以下の式で表される．

$$\begin{aligned} & E_{Y_1^n}\left[-\log P_c(Y_1^n)\right] \\ &= E_{X_1^n}\left[-\log P_c(X_1^n)\right] \\ &= H(X_1^n|\theta^*_{m^*}, m^*) \\ &+ \frac{k}{2}\log\frac{n}{2\pi e} + \log\frac{\sqrt{\det I_X(\theta^*_{m^*})}}{P(\theta^*_{m^*})} + o(1). \quad (14) \end{aligned}$$

6 まとめ

本研究では，FSMX 情報源に対する BW 変換を用いたベイズ符号の構成法について示した．そのもとで，効率的な符号化アルゴリズムを提案した．また，提案したアルゴリズムにおける，平均符号長の漸近式について示した．

謝辞

本研究を行うにあたり，数多くの御助言，御支援を賜りました，松嶋研究室，平澤研究室の各氏に感謝致します．なお，本研究の一部は日本学術振興会科学研究費基盤 (C) 一般 (No.15560338) の援助による．

参考文献

[1] T. Matsushima, H. Inazumi and S. Hirasawa, "A Class of Distortionless Codes Designed by Bayes Decision Theory" IEEE Trans. Inf. Theory, vol.37, No.5, page 1288, 1991.

[2] N. Merhav, "On the minimum description length principle for sources with piecewise constant parameters." *IEEE Trans. Inf. Theory*, Vol. 39, No. 6, page 1962, 1993.

[3] M. Effros, K. Visweswariah, R. Kulkarni and S. Verdú, "Universal Lossless Source Coding With the Barrows Wheeler Transform," *IEEE Trans. Inf. Theory*, Vol. 48, No. 5, page 1061, 2002.

[4] M. Effros, "Universal lossless source coding with the Barrows Wheeler Transform," *Proc. Data Compression Conf.*, Snowbird, UT, Mar, page 178, 1999.

[5] D. Baron and Y. Bresler, "An $O(n)$ Semipredictive Universal Encoder via the BWT," *IEEE Trans. Inf. Theory*, Vol. 50, No. 5, page 928, 2004.

[6] H. Cai, S. R. Kulkarni and S. Verdú, "Universal Entropy Estimation Via Block Sorting," *IEEE Trans. Inf. Theory*, Vol. 50, No. 7, page 1551, 2004.

[7] M. Gotoh, T. Matsushima and S. Hirasawa, "A Generalization of B. S. Clarke and A. R. Barron's Asymptotics of Bayes Codes for FSMX Sources," *IEICE Trans. Fundamentals*, Vol. E81-A, No. 10, page 2123, 1998.

[8] T. Matsushima, and S. Hirasawa, "A bayes coding using context tree." In Proc. Int. Symp. on. Inf. Theory, page 386, 1994.

SCIS 2006 The 2006 Symposium on Cryptography and Information Security
Hiroshima, Japan, Jan. 17-20, 2006
The Institute of Electronics, Information and Communication Engineers

ID情報に基づくランプ型非対称鍵配送方式について

Non-Perfectly Secure Identity Based Asymmetric Key Distribution Scheme

海上 勇二 * 齋藤 友彦 * 松嶋 敏泰 *
Yuji UNAGAMI Tomohiko SAITO Toshiyasu MATSUSHIMA

あらまし 本稿では，複数のユーザと鍵配送センタからなるネットワークにおいて，センタからユーザへ秘密鍵に関する情報を送り，ユーザはユーザ間での通信をせず，秘密鍵を共有したい相手のIDから自分が属する鍵共有ユーザグループの秘密鍵を個別に求めることができる鍵配送方式について考える．従来，しきい値型鍵配送方式のユーザの記憶容量を削減するため，不必要な鍵共有機能を削減し，ユーザの記憶容量の削減を行っている方式がある．本研究では，不必要な鍵共有機能を削減した鍵配送方式に対して，一般化したランプ型鍵配送方式モデルを定義し，そのプロトコルを提案し，各ユーザが秘密鍵を求めるために必要な記憶容量の評価を行う．

キーワード 鍵事前配送方式，ランプ型，記憶容量，エントロピー

1 はじめに

大規模なコンピュータネットワーク上で多くの情報をやり取りをする場合，情報が第3者に漏れたり，悪用されないためには情報の暗号化が必要となる．このとき，通信を行うユーザ間で秘密鍵を共有するために必要となるのが鍵配送方式である．鍵配送方式とは，複数のユーザと鍵配送センタという信頼できる機関からなるネットワークにおいて，鍵配送センタがユーザへ安全な通信路を用いて各ユーザが属するグループの秘密鍵に関する情報を送信する．情報を受け取った各ユーザは，公開されているユーザのID情報から自分の属しているグループの秘密鍵を個別に計算できる方式である．

従来方式では，秘密鍵を共有するユーザグループの大きさをt人とし，k人以下のユーザが結託しても，そのユーザが属していないグループの秘密鍵の情報は全く得られないが，$k+1$人以上のユーザが結託すると，すべての秘密鍵の情報が完全に求められる．本稿では，この方式をしきい値型鍵配送方式と呼ぶ．この方式について各ユーザが秘密鍵を計算するために必要な記憶容量の下界とその下界を達成するプロトコルも示されている [1]．しきい値型鍵配送方式において，ユーザ間で通信を行わない場合，鍵を共有する必要がないため，不必要な鍵共有機能を削除することでユーザの記憶容量を削減する方式が提案されている [2]．また，しきい値型鍵配送方式を一般化し，k人以下のユーザが結託してもすべての秘密鍵に関する情報は一切得られずに，$k+1$人以上$k+c$人以下のユーザが結託すると結託するユーザ数に対して段階的に秘密鍵に関する情報が部分的に得られ，$k+c+1$人以上のユーザが結託するとすべての秘密鍵の情報を完全に求めることができるランプ型鍵配送方式が提案されている [4]．

本研究では，上記のユーザ間で鍵共有を必要としない場合の鍵配送方式に対して，一般化したランプ型鍵配送方式モデルを定義する．この方式はユーザの部分集合$\mathcal{P}_j$において，ψ_j人以下のユーザが結託してもすべての秘密鍵に関する情報は一切得られずに，ψ_j+1人以上ψ_j+c_j人以下のユーザが結託すると結託するユーザ数に対して段階的に秘密鍵に関する情報が部分的に得られ，ψ_j+c_j+1人以上のユーザが結託するとすべての秘密鍵の情報を完全に求めることができる方式である．ユーザが秘密鍵を求めるためのプロトコルを提案し，ユーザが秘密鍵を求めるために必要な記憶容量の評価を行う．

2 鍵配送方式

本稿で考える鍵配送方式では，信頼できる鍵配送センタ（以下で単にセンタとする）とID（識別子）を持つn人のユーザの集合$\mathcal{P}=\{P_1,P_2,\cdots,P_n\}$からなるネットワークにおいて，ユーザの任意の$t$人の鍵共有ユーザ集合$\mathcal{M}_j\subseteq\mathcal{P},(j=1,2,\cdots)$が同一の鍵をユーザ個別に

* 早稲田大学理工学部経営システム工学科，〒169-8555 東京都新宿区大久保3-4-1，Dept. of Industrial and Management Systems Engineering, School of Science and Engineering, Waseda University, 3-4-1 Okubo, Shinjuku-ku, Tokyo, 169-8555, JAPAN. E-mail: unagami@matsu.mgmt.waseda.ac.jp

計算し，共有する方式である．まず，センタが公開されている各ユーザのIDと非公開の情報である乱数を用いて，各ユーザP_iへ送る情報u_iを生成し，安全な通信路を用いて送信する．各ユーザはセンタから送られてきた情報を記憶し，ユーザ間での通信を行わずに，記憶してある情報と秘密鍵を共有したいグループ内のユーザIDから，個別にそのグループの秘密鍵を生成する．ここで，$1 \leq i \leq n$に対し，ユーザP_iが記憶しておく情報の集合を$\mathcal{U}_{P_i}$，鍵共有ユーザ集合$\mathcal{M}_j$における秘密鍵の集合を$\mathcal{K}_j$とし，それぞれの集合の中に値をとる確率変数をU_{P_i}, K_jとする．

3 しきい値型鍵配送方式

3.1 (k,t)鍵配送方式モデル

しきい値型鍵配送方式は，任意のt人のグループで秘密鍵を共有することができ，k人以下のユーザが結託をし，ユーザがセンタから受け取った情報を共有しても秘密鍵に関する情報は全く得られないが，$k+1$人以上のユーザが結託をしてしまうとすべての秘密鍵の情報が完全に求めることができる．このような鍵配送方式を(k,t)鍵配送方式という．

定義 3.1 n人のユーザ集合$\mathcal{P}$と$k+t \leq n$を満たす非負整数k,tに対し，以下の条件を満たすとき，(k,t)鍵配送方式という．
(1) $|\mathcal{M}_j| = t$を満たす，任意の鍵共有ユーザ集合$\mathcal{M}_j \subseteq \mathcal{M}, j = 1,2,\cdots,\binom{n}{t}$における各ユーザ$P_i \in \mathcal{P}_j$に対し，

$$H(K_j|U_{P_i}) = 0 \tag{1}$$

が成り立つ．
(2) $\mathcal{W} \cap \mathcal{M}_j = \phi, |\mathcal{W}| \leq k, |\mathcal{M}_j| = t$を満たす，すべてのユーザ集合の部分集合

$$\mathcal{W} = \{P_{j_1}, P_{j_2}, \cdots, P_{j_{|\mathcal{W}|}}\} \tag{2}$$

と$\mathcal{M}_j \subseteq \mathcal{P}$に対し，

$$H(K_j|U_{P_{j_1}}, \cdots, U_{P_{j_{|\mathcal{W}|}}}) = H(K_j) \tag{3}$$

が成り立つ．

3.2 ユーザの記憶容量の下界

(k,t)鍵配送方式における各ユーザP_iの記憶容量をU_{P_i}のエントロピーまたは集合のサイズ$|\mathcal{U}_{P_i}|$で表すことにすると記憶容量の下界は以下の定理で与えられる[1]．

定理 3.1 n人のユーザ集合を$\mathcal{P}$と$k+t \leq n$を満たす非負整数k,tに対し，すべての鍵のエントロピーが等しい，つまり

$$H(K_j) = H(K), \quad j = 1,2,\cdots,\binom{n}{t} \tag{4}$$

が成り立つと仮定する．このとき，

$$H(U_{P_i}) \geq \binom{k+t-1}{t-1} H(K), \quad i = 1,2,\cdots,n, \tag{5}$$

$$\log|\mathcal{U}_{P_i}| \geq \binom{k+t-1}{t-1} \log|\mathcal{K}|, \quad i = 1,2,\cdots,n, \tag{6}$$

が成り立つ．ここで，$\mathcal{K}$は可算集合，Kは$\mathcal{K}$の中に値をとる確率変数とした．□

3.3 (k,t)鍵配送方式プロトコル

(k,t)鍵配送方式は，t変数対称多項式を用いたプロトコルが示されている[1]．ここでt変数対称多項式[1]とは

$$f(x_1, x_2, \cdots, x_t) = \sum_{i_1=0}^{k} \cdots \sum_{i_t=0}^{k} a_{i_1 \cdots i_t} x^{i_1} \cdots x^{i_t} \tag{7}$$

が，任意の置換$\sigma : \{1,2,\cdots,t\} \to \{1,2,\cdots,t\}$に対して，$a_{i_1 \cdots i_t} = a_{\sigma(i_1 \cdots i_t)}$が成り立つ．

【(k,t)鍵配送方式プロトコル】

1. センタはt変数対称多項式$f(x_1,\cdots,x_t)$をランダムに生成する．
2. センタは，$u_i = f(id_i, x_2, \cdots, x_t)$を計算し，ユーザ$P_i$へ$u_i$を送信する．
3. 鍵共有ユーザ集合$\mathcal{M}_j = \{P_{j_1}, P_{j_2}, \cdots, P_{j_t}\}$において，ユーザ$P_{j_i}$は$u_{j_i} = f(id_{j_i}, x_2, \cdots, x_t)$を$(x_2, \cdots, x_t) = (id_{j_1}, \cdots, id_{j_{i-1}}, id_{j_{i+1}}, \cdots, id_{j_t})$として計算する．
4. 鍵共有ユーザ集合$\mathcal{M}_j$それぞれの秘密鍵は

$$k_j = f(id_{j_1}, id_{j_2}, \cdots, id_{j_t}) \tag{8}$$

となる．

このプロトコルは定理3.1の下界を達成することが示されている[1]．

4 非対称t鍵配送方式

4.1 非対称t鍵配送方式モデル

大規模なネットワークを構築するとき，ユーザ間で通信を行わないため，鍵を共有する必要がない場合が考えられる．このような場合，不必要な鍵共有機能を削除することでユーザの記憶容量を削減する方式がある[2]．ユーザの集合をN個の部分集合$\{\mathcal{P}_1, \mathcal{P}_2, \cdots, \mathcal{P}_N\}$ $(\mathcal{P}_i \cap \mathcal{P}_j = \phi)$

[1] 演算は有限体$GF(q)$上の演算とする

に分割する．それぞれのユーザの部分集合内で鍵を共有しない場合において，(k,t) 鍵配送方式を用いると，ユーザ $P_i \in \mathcal{P}_i$ の記憶容量は (6) 式より，$|\mathcal{P}_i|+|\mathcal{P}_j|-1$ に比例する．しかし，$P_i \in \mathcal{P}_i$ が $|\mathcal{P}_j|$ と通信をするために $|\mathcal{P}_i|+|\mathcal{P}_j|-1$ に比例した記憶容量が必要となり効率が悪い．そこで，このような場合には次の非対称 t 鍵配送方式を用いることでユーザの記憶容量を削減する．非対称 t 鍵配送方式は，$P_i \in \mathcal{P}_i$ $(i=1,\cdots,t)$ の t 人のユーザが鍵を共有し，ユーザの部分集合 $\mathcal{P}_j$ において，ψ_j 人以下のユーザが結託しても秘密鍵に関する情報は一切得られずに，ψ_j+1 人以上結託すると秘密鍵の情報が完全に求まる方式である．

定義 4.1 n 人のユーザ集合 $\mathcal{P}$ を t 個の部分集合 $\mathcal{P}=\{\mathcal{P}_1,\mathcal{P}_2,\cdots,\mathcal{P}_t\}$ とし，$\psi_j+1 \le |\mathcal{P}_j|$ を満たす非負整数 ψ_j, t に対し，以下の条件を満たすとき，非対称 t 鍵配送方式という．
(1) $\mathcal{M}_r=\{P_1,P_2,\cdots,P_t\}$ $(r=1,2,\cdots)$, $P_1\in\mathcal{P}_1, P_2\in\mathcal{P}_2,\cdots,P_t\in\mathcal{P}_t$ において，$P_i\in\mathcal{M}_r$ に対し，

$$H(K_{P_i}|U_{P_i})=0 \tag{9}$$

が成立する．
(2) $\psi_1,\psi_2,\cdots,\psi_t$ を $\mathcal{P}_1,\mathcal{P}_2,\cdots,\mathcal{P}_t$ の結託のしきい値とし，$\mathcal{W}_j\subset\mathcal{P}_j, \mathcal{W}_j\cap\mathcal{M}_r=\phi$, $|\mathcal{W}_j|\le\psi_j$ を満たす，ユーザの部分集合

$$\mathcal{W}_j=\{P_j^{k_j}|k_j=0,1,\cdots,|\mathcal{W}_j|, k_j\neq k_j^*\}\subset\mathcal{P}_j,\quad j=1,\cdots,t,\ j\neq i, \tag{10}$$

および $P_i\in\mathcal{P}_i$ と $P_j^{k_j^*}\in\mathcal{P}_j$ $(j=1,2,\cdots,t,\ j\neq i)$ に対し，

$$H\Big(K_{P_i,P_1^{k_1^*},\cdots,P_{i-1}^{k_{i-1}^*},P_{i+1}^{k_{i+1}^*},\cdots,P_t^{k_t^*}}\Big|U_{\mathcal{W}_1},\cdots,U_{\mathcal{W}_{i-1}},U_{\mathcal{W}_{i+1}},\cdots,U_{\mathcal{W}_t}\Big) = H\Big(K_{P_i,P_1^{k_1^*},\cdots,P_{i-1}^{k_{i-1}^*},P_{i+1}^{k_{i+1}^*},\cdots,P_t^{k_t^*}}\Big) \tag{11}$$

が成立する．

4.2 ユーザの記憶容量の下界

非対称 t 鍵配送方式における各ユーザ P_i の記憶容量の下界は以下の定理で与えられる [2].

定理 4.1 n 人のユーザ集合 $\mathcal{P}$ を t 個の部分集合 $\mathcal{P}=\{\mathcal{P}_1,\mathcal{P}_2,\cdots,\mathcal{P}_t\}$ とし，$\psi_j+1\le|\mathcal{P}_j|$ を満たす非負整数 ψ_j, t に対し，すべての鍵のエントロピーが等しい，すなわち

$$H\left(K_{P_i,P_1^{k_1^*},\cdots,P_{i-1}^{k_{i-1}^*},P_{i+1}^{k_{i+1}^*},\cdots,P_t^{k_t^*}}\right)=H(K) \tag{12}$$

と仮定する．このとき，非対称 t 鍵配送方式において，ユーザ $P_i\in\mathcal{P}_i$ の記憶容量の下界は，

$$H(U_{P_i})\ge\left(\prod_{j\in\{1,\cdots,t\},j\neq i}(\psi_j+1)\right)H(K) \tag{13}$$

となる． □

4.3 非対称 t 鍵配送方式プロトコル

非対称 t 鍵配送方式のプロトコルは (7) 式の対称多項式を t 変数でそれぞれの変数の次数が $\psi_1,\psi_2,\cdots,\psi_t$ となる非対称多項式に置き換えることで構成できることが示されている [2]．非対称多項式とは

$$f(x_1,\cdots,x_t)=\sum_{i_1=0}^{\psi_1}\cdots\sum_{i_t=0}^{\psi_t}a_{i_1\cdots i_t}x_1^{i_1}\cdots x_t^{i_t} \tag{14}$$

が，$a_{i_1\cdots i_t}$ は任意の置換 σ に対して $a_{i_1\cdots i_t}=a_{\sigma(i_1\cdots i_t)}$ でなくてよい．

【非対称 t 鍵配送方式プロトコル】

1. センタは t 変数多項式 $f(x_1,\cdots,x_t)$ をランダムに生成する．
2. センタは，$u_i=f(x_1,x_2,\cdots,x_t)|_{x_i=id_i}$ を計算し，ユーザ $P_i\in\mathcal{P}_i$ へ u_i を送信する．
3. ユーザ $P_i\in\mathcal{P}_i$ は

   $$u_i|_{x_1=id_1,\cdots,x_{i-1}=id_{i-1},x_{i+1}=id_{i+1},\cdots,x_t=id_t} \tag{15}$$

   を計算する．
4. 鍵共有ユーザ集合 $\mathcal{M}_r$ それぞれの秘密鍵は，

   $$k_r=f(id_{j_1},id_{j_2},\cdots,id_{j_t}) \tag{16}$$

   となる．

このプロトコルは定理 4.1 の下界を達成することが示されている [2].

5 ランプ型鍵配送方式

5.1 (c,k,t) 鍵配送方式モデル

(k,t) 鍵配送方式では，ネットワーク内の任意の $k+1$ 人以上のユーザが結託した場合に秘密鍵に関する情報を完全に求められる方式となっているが，ネットワークの安全性をより高くするために k の値を大きくするとユーザの記憶容量も大きくなってしまうことが定理 3.1 からわかる．本節では，(k,t) 鍵配送方式を一般化し，k 人以下のユーザが結託しても秘密鍵に関する情報は一切得られず，$k+1$ 人以上 $k+c$ 人以下のユーザが結託すると結託するユーザ数に対して段階的に秘密鍵に関する情報

が部分的に得られ，$k+c+1$ 人以上のユーザが結託すると秘密鍵の情報を完全に求めることができる (c,k,t) 鍵配送方式という [4].

定義 5.1 n 人のユーザ集合 $\mathcal{P}$ と $c+k+t \leq n$ を満たす非負整数 c,k,t に対し，以下の条件を満たすとき，(c,k,t) 鍵配送方式という．
(1) $|\mathcal{M}_j| = t$ を満たす，任意の鍵共有ユーザ集合 $\mathcal{M}_j \subseteq \mathcal{P}, j = 1,2,\cdots,\binom{n}{t}$ における各ユーザ $P_i \in \mathcal{M}_j$ に対し，

$$H(K_j|U_{P_i}) = 0 \tag{17}$$

が成り立つ．
(2) $\mathcal{W} \cap \mathcal{M}_j = \phi, |\mathcal{W}| \leq k, |\mathcal{M}_j| = t$ を満たす，すべてのユーザ集合の部分集合

$$\mathcal{W} = \{P_{j_1}, P_{j_2}, \cdots, P_{j_{|\mathcal{W}|}}\} \tag{18}$$

と $\mathcal{M}_j \subseteq \mathcal{P}$ に対し，

$$H(K_j|U_{P_{j_1}}, \cdots, U_{P_{j_{|\mathcal{W}|}}}) = H(K_j) \tag{19}$$

が成り立つ．
(3) $\mathcal{V} \cap \mathcal{M}_j = \phi, |\mathcal{V}| = k+l, |\mathcal{M}_j| = t$ を満たす，すべてのユーザ集合の部分集合

$$\mathcal{V} = \{P_{j_1}, P_{j_2}, \cdots, P_{j_{k+l}}\}, \quad 0 \leq l \leq c \tag{20}$$

と $\mathcal{M}_j \subseteq \mathcal{P}$ に対し，

$$H(K_j|U_{P_{j_1}}, \cdots, U_{P_{j_{k+l}}}) = \frac{c+1-l}{c+1} H(K_j), \quad 0 \leq l \leq c \tag{21}$$

が成り立つ．
定義 5.1 において，$c=0$ のとき (k,t) 鍵配送方式の定義 3.1 と同じになる．つまり，(c,k,t) 鍵配送方式は，(k,t) 鍵配送方式を含んだ一般的な鍵配送方式となっていることがわかる．

5.2 ユーザの記憶容量の下界

(c,k,t) 鍵配送方式における各ユーザ P_i の記憶容量の下界は以下の定理で与えられる [4].

定理 5.1 n 人のユーザ集合を $\mathcal{P}$ と $c+k+t \leq n$ を満たす非負整数 k,t に対し，すべての鍵のエントロピーが等しい，つまり

$$H(K_j) = H(K), \quad j = 1,2,\cdots,\binom{n}{t} \tag{22}$$

が成り立つと仮定する．このとき，

$$H(U_{P_i}) \geq \sum_{l=1}^{c} \frac{c+1-l}{c+1} \binom{k+t-2+l}{t-2} H(K) + \binom{k+t-1}{t-1} H(K), \; i = 1,2,\cdots,n \tag{23}$$

が成り立つ． □

5.3 (c,k,t) 鍵配送方式プロトコル

(c,k,t) 鍵配送方式プロトコルは，(k,t) 鍵配送方式プロトコルを利用した構成法が示されている [4].
秘密鍵の集合 $\mathcal{K}_j$ について以下の条件を仮定する．

$$\mathcal{K}_j = \mathcal{K}, \quad j = 1,2,\cdots,\binom{n}{t}, \tag{24}$$

$$\mathcal{K} = \mathcal{K}^{(0)} \times \mathcal{K}^{(1)} \times \cdots \times \mathcal{K}^{(c)}, \tag{25}$$

$$|\mathcal{K}^{(0)}| = |\mathcal{K}^{(1)}| = \cdots = |\mathcal{K}^{(c)}| = q. \tag{26}$$

【(c,k,t) 鍵配送方式プロトコル】

1. センタは $0 \leq l \leq c$ に対し，$c+1$ 個の t 変数対称多項式を $(k+l,t)$ 鍵配送方式を用いて，ランダムに生成する．

$$f^{(l)}(x_1,\cdots,x_t) = \sum_{i_1=0}^{k+l} \cdots \sum_{i_t=0}^{k+l} a_{i_1\cdots i_t} x_1^{i_1} \cdots x_t^{i_t}. \tag{27}$$

ここで，$x_1,\cdots,x_t$ の次数は $k+l$ 次 $(l=0,1,\cdots,c)$ になっている．

2. センタは

$$u_i = \{f^{(l)}(id_i,\cdots,x_t) | l = 0,1,\cdots,c\} \tag{28}$$

を計算し，ユーザ P_i へ u_i を送信する．

3. 鍵共有ユーザ集合 $\mathcal{M}_j = \{P_{j_1}, P_{j_2}, \cdots, P_{j_t}\}$ において，ユーザ P_{j_i} は $u_{j_i} = f^{(l)}(id_{j_i}, x_2, \cdots, x_t)$ を $(x_2,\cdots,x_t) = (id_{j_1},\cdots,id_{j_{i-1}},id_{j_{i+1}},\cdots,id_{j_t})$ として計算する．

4. 鍵共有ユーザ集合 $\mathcal{M}_j$ それぞれの秘密鍵は，以下のように 3 で計算した値を連接したものとする．つまり，

$$k_j = f^{(0)}(id_{j_1},\cdots,id_{j_t}) \oplus \cdots \oplus f^{(c)}(id_{j_1},\cdots,id_{j_t}) \tag{29}$$

$$= k_j^{(0)} \oplus k_j^{(1)} \oplus \cdots \oplus k_j^{(c)}, \quad k_j^{(l)} \in \mathcal{K}^{(l)}, l = 0,\cdots,c, \tag{30}$$

となる．ここで，$\oplus$ は連接を表す記号とした．

このプロトコルは定理 5.1 の下界を達成することが示されている [4].

6 ランプ型非対称 t 鍵配送方式

6.1 ランプ型非対称 t 鍵配送方式モデル

本研究では，非対称 t 鍵配送方式を一般化したランプ型非対称 t 鍵配送方式モデルを定義する．この方式は，ユー

ザ集合 $\mathcal{P}$ を t 個の部分集合 $\{\mathcal{P}_1, \mathcal{P}_2, \cdots, \mathcal{P}_t\}$ $(\mathcal{P}_i \cap \mathcal{P}_j = \phi)$ に分割する．$\{\mathcal{P}_1, \mathcal{P}_2, \cdots, \mathcal{P}_t\}$ の結託のしきい値をそれぞれ $\psi_1, \psi_2, \cdots, \psi_t$ としたとき，$\mathcal{P}_j$ の任意の ψ_j 人以下のユーザが結託しても秘密鍵に関する情報は一切得られずに，$\psi_j + 1$ 人以上 $\psi_j + c_j$ 人以下のユーザが結託すると結託するユーザ数に対して段階的に秘密鍵に関する情報が部分的に得られ，$\psi_j + c_j + 1$ 人以上のユーザが結託すると秘密鍵の情報を完全に求めることができる方式である．

定義 6.1 ユーザ集合 $\mathcal{P}$ を t 個の部分集合 $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \cdots, \mathcal{P}_t\}$ とし，$c_j + \psi_j + 1 \leq |\mathcal{P}_j|$ を満たす非負整数 c_j, ψ_j, t に対し，以下の条件を満たすとき，ランプ型非対称鍵配送方式という．

(1) $\mathcal{M}_r = \{P_1, P_2, \cdots, P_t\}$ $(r = 1, 2, \cdots)$, $P_1 \in \mathcal{P}_1, P_2 \in \mathcal{P}_2, \cdots, P_t \in \mathcal{P}_t$ において，$P_i \in \mathcal{M}_r$ に対し，

$$H(K_{P_i}|U_{P_i}) = 0 \tag{31}$$

が成立する．

(2) $\psi_1, \psi_2, \cdots, \psi_t$ を $\mathcal{P}_1, \mathcal{P}_2, \cdots, \mathcal{P}_t$ の結託のしきい値とし，$\mathcal{W}_j \subset \mathcal{P}_j, \mathcal{W}_j \cap \mathcal{M}_r = \phi$, $|\mathcal{W}_j| \leq \psi_j$ を満たす，ユーザの部分集合

$$\mathcal{W}_j = \{P_j^{k_j} | k_j = 0, 1, \cdots, |\mathcal{W}_j|, k_j \neq k_j^*\} \subset P_j, \quad j = 1, \cdots, t,\ j \neq i, \tag{32}$$

および $P_i \in \mathcal{P}_i$ と $P_j^{k_j^*} \in \mathcal{P}_j$ $(j = 1, 2, \cdots, t,\ j \neq i)$ に対し，

$$H\Big(K_{P_i, P_1^{k_1^*}, \cdots, P_{i-1}^{k_{i-1}^*}, P_{i+1}^{k_{i+1}^*}, \cdots, P_t^{k_t^*}} \Big| U_{\mathcal{W}_1}, \cdots, U_{\mathcal{W}_{i-1}}, U_{\mathcal{W}_{i+1}}, \cdots, U_{\mathcal{W}_t}\Big) = H\Big(K_{P_i, P_1^{k_1^*}, \cdots, P_{i-1}^{k_{i-1}^*}, P_{i+1}^{k_{i+1}^*}, \cdots, P_t^{k_t^*}}\Big) \tag{33}$$

が成立する．

(3) $\mathcal{V}_j \subset \mathcal{P}_j, \mathcal{V}_j \cap \mathcal{M}_r = \phi$, $|\mathcal{V}_j| = \psi_j + l_j$ $(0 \leq l_j \leq c_j)$ を満たす，ユーザの部分集合

$$\mathcal{V}_j = \{P_j^{k_j} | k_j = 0, 1, \cdots, \psi_j + l_j, k_j \neq k_j^*\} \subset P_j, \quad j = 1, \cdots, t,\ j \neq i, 0 \leq l_j \leq c_j, \tag{34}$$

および $P_i \in \mathcal{P}_i$ と $P_j^{k_j^*} \in \mathcal{P}_j$ $(j = 1, 2, \cdots, t,\ j \neq i)$ に対し，

$$H\Big(K_{P_i, P_1^{k_1^*}, \cdots, P_{i-1}^{k_{i-1}^*}, P_{i+1}^{k_{i+1}^*}, \cdots, P_t^{k_t^*}} \Big| U_{\mathcal{V}_1}, \cdots, U_{\mathcal{V}_{i-1}}, U_{\mathcal{V}_{i+1}}, \cdots, U_{\mathcal{V}_t}\Big) = \frac{\prod_{j,\ j \neq i}(c_j + 1) + 1 - \prod_{j,\ j \neq i}(l_j + 1)}{\prod_{j,\ j \neq i}(c_j + 1)} H\Big(K_{P_i, P_1^{k_1^*}, \cdots, P_{i-1}^{k_{i-1}^*}, P_{i+1}^{k_{i+1}^*}, \cdots, P_t^{k_t^*}}\Big) \tag{35}$$

が成立する．

定義 6.1 において，$c_j = 0$ $(j = 1, \cdots, t,\ j \neq i)$ のとき非対称 t 鍵配送方式の定義 4.1 と同じになる．つまり，ランプ型非対称鍵配送方式は非対称 t 鍵配送方式を含んだ一般的な鍵配送方式となっていることがわかる．

6.2 ランプ型非対称鍵配送方式プロトコル

本節では，ランプ型非対称鍵配送方式のプロトコルを提案する．

ここで，秘密鍵の集合 $\mathcal{K}_{P_i, P_1^{k_1}, \cdots, P_{i-1}^{k_{i-1}}, P_{i+1}^{k_{i+1}}, \cdots, P_t^{k_t}}$ について以下の条件を仮定する．

$$\mathcal{K}_{P_i, P_1^{k_1}, \cdots, P_{i-1}^{k_{i-1}}, P_{i+1}^{k_{i+1}}, \cdots, P_t^{k_t}} = \mathcal{K}, \quad k_j = 0, 1, 2, \cdots, \psi_j + c_j, \tag{36}$$

$$\mathcal{K} = \mathcal{K}^{(1)} \times \mathcal{K}^{(2)} \times \cdots \times \mathcal{K}^{(\prod_{j \in \{1, \cdots, t\}}(c_j + 1))}, \tag{37}$$

$$|\mathcal{K}^{(1)}| = |\mathcal{K}^{(2)}| = \cdots = |\mathcal{K}^{(\prod_{j \in \{1, \cdots, t\}}(c_j + 1))}| = q. \tag{38}$$

【ランプ型非対称鍵配送方式プロトコル】

1. センタは $0 \leq l_j \leq c_j$ に対し，$\prod_{j \in \{1, \cdots, t\}}(c_j + 1)$ 個の t 変数対称多項式を $(\psi_j + l_j, t)$ 鍵配送方式を用いて，ランダムに生成する．

$$f^{(l_1, \cdots, l_t)}(x_1, \cdots, x_t) = \sum_{i_1 = 0}^{\psi_1 + l_1} \cdots \sum_{i_t = 0}^{\psi_t + l_t} a_{i_1 \cdots i_t} x_1^{i_1} \cdots x_t^{i_t}. \tag{39}$$

ここで，それぞれの変数 $x_1, \cdots, x_t$ の次数は $\psi_j + l_j$ 次 $(l_j = 0, 1, \cdots, c_j)$ になっている．

2. センタは

$$u_i = \{f^{(l_1, \cdots, l_t)}(id_i, \cdots, x_t) \Big| l_j = 0, 1, \cdots, c_j\}, \tag{40}$$

を計算し，ユーザ P_i へ u_i を送信する．

3. 鍵共有ユーザ集合 $\mathcal{M}_r = \{P_{j_1}, P_{j_2}, \cdots, P_{j_t}\}$ において，ユーザ P_{j_i} は $u_{j_i} = f^{(l_1, \cdots, l_t)}(id_{j_i}, x_2, \cdots, x_t)$ を $(x_2, \cdots, x_t) = (id_{j_1}, \cdots, id_{j_{i-1}}, id_{j_{i+1}}, \cdots, id_{j_t})$ として計算する．

4. 鍵共有ユーザ集合 $\mathcal{M}_r$ それぞれの秘密鍵は，以下のように 3 で計算した値を連接したものとする．つまり，

$$k_j = f^{(0, \cdots, 0)}(id_{j_1}, \cdots, id_{j_t}) \oplus \cdots \oplus f^{(c_1, \cdots, c_t)}(id_{j_1}, \cdots, id_{j_t}) \tag{41}$$

$$= k_j^{(1)} \oplus k_j^{(2)} \oplus \cdots \oplus k_j^{(\prod_{j \in \{1, \cdots, t\}}(c_j + 1))}, \quad k_j^{(l)} \in \mathcal{K}^{(l)}, l = 1, \cdots, \prod_{j \in \{1, \cdots, t\}} (c_j + 1), \tag{42}$$

となる．ここで，$\oplus$ は連接を表す記号とした．

7 考察

非対称 t 鍵配送方式とランプ型非対称鍵配送方式のプロトコルを比較する．ランプ型非対称鍵配送方式に対して $c_j = 0$ $(j = 1, 2, \cdots, t)$ としたときのプロトコルと，非対称 t 鍵配送方式に対するプロトコルは同様の計算を行っていることがわかる．つまり，ランプ型非対称鍵配送方式は非対称鍵配送方式を含んだ一般的な鍵配送方式になっている．次に，非対称 t 鍵配送方式とランプ型非対称鍵配送方式のプロトコルを用いたときのユーザの記憶容量を比較する．

例 秘密鍵の集合は鍵共有ユーザ集合によらず，全て同じ集合 $\mathcal{K}$ とし，秘密鍵，ユーザへ送信する秘密情報に関する確率分布は一様分布であると仮定する．秘密鍵の長さが 256 ビット，すなわち $|\mathcal{K}| = 2^{256}$ とする．この場合，非対称 t 鍵配送方式において，$(\psi_2, \psi_3, t) = (4, 3, 3)$ としたときと，ランプ型非対称鍵配送方式において $(c_2, c_3, \psi_2, \psi_3, t) = (1, 1, 3, 2, 3)$ とを比較すると，ユーザへの秘密情報がそれぞれ，5120 ビット，4032 ビットとなる．この量は，可能性のある秘密鍵の個数として解釈ができ，値が大きいほど安全である．

この例の場合，$\mathcal{P}_2$ に関して任意の 3 人以下のユーザと $\mathcal{P}_3$ に関して任意の 2 人以下のユーザが結託すると (35) 式が 2 つの方式で $2^{256}, 2^{256}$ となり，$\mathcal{P}_2$ に関して任意の 4 人以下のユーザで，$\mathcal{P}_3$ に関して任意の 2 人以下のユーザが結託する，または，$\mathcal{P}_2$ に関して任意の 3 人以下のユーザで，$\mathcal{P}_3$ に関して任意の 3 人以下のユーザが結託すると $2^{256}, 2^{192}$，$\mathcal{P}_2$ に関して任意の 4 人以下のユーザで，$\mathcal{P}_3$ に関して任意の 3 人以下のユーザが結託すると $2^{256}, 2^{64}$ となる．ここで，安全性の尺度として，上の例で用いた 2 つの値 2^{256} と 2^{64} を比較して，2^{64} で安全性が十分であり基準に達していると考えられるのであれば，ユーザの記憶容量を削減することができ，提案したランプ型非対称鍵配送方式が有効であるといえる．

8 まとめ

本稿では，ランプ型非対称鍵配送方式のモデルを，非対称 t 鍵配送方式のモデルの一般化として定義し，そのモデルのもとで，プロトコルを提案した．そして，ユーザが秘密鍵を求めるために必要な記憶容量の評価をし，ユーザが必要な記憶容量の面で有効性を示した．

謝辞

本研究を行うにあたり，数多くの御助言，御支援を賜りました松嶋研究室，平澤研究室の各氏に感謝致します．なお，本研究の一部は日本学術振興会科学研究費基盤 (C) 一般 (No.15560338) の援助による．

参考文献

[1] C.Blundo, A.De.Santis, A.Herzberg, S.Kutten, U.Vaccaro, and M.Yung, "Perfectly-Secure Key Distribution for Dynamic Conferences," Proc. of CRYPT 92, LNCS,vol.470,pp.471-486, 1993.

[2] G.Hanaoka, T. Nishioka, H. Imai, "Optimal Unconditionally Secure ID-Based Key Distribution Scheme for Large-Scaled Networks," IEICE Trans. Fundamentals, volE84-A, no.1, pp.222-230, 2001.

[3] R.Blom,"An Optimal Class of Symmetric Key Generation Systems," Proc. of Eurocrypt 84, LNCS vol.209,pp.335-338,1985.

[4] 吉田隆弘，松嶋敏泰，平澤茂一，"ID 情報に基づくランプ型分散鍵配送方式について，" SITA2004 , pp327-330, 2004.