2004 年度卒業論文

相同性検索手法の組み合わせによる検索精度向上

提出日：2005 年 2 月 2 日

指導：山名早人助教授

早稲田大学理工学部情報学科
学籍番号：1G01P061-8
滝沢雅俊

概 要

ヒトをはじめとした様々な生物のゲノムの解読が完了したことによって得られた膨大な生体情報を、コンピュータを用いて有効に解析するバイオインフォマティクスの研究が行われている。バイオインフォマティクスにおいてタンパク質のアミノ酸配列比較の際、相同性は重要な基準となる。相同性とは、共通祖先に由来する子孫間の類似性を指す。共通祖先から分岐した相同タンパク質の間では、類似した構造や機能を有していることが多い。また、機能や構造未知のアミノ酸配列を問い合わせ配列とし、データベース中の既知のアミノ酸配列から、問い合わせ配列と相同なアミノ酸配列を収集する方法を相同性検索とよぶ。相同性検索によって得られた相同な配列をもとにマルチプルアラインメントを構築し、そのアラインメントから未知のアミノ酸配列の機能や構造情報を抽出することが可能となる。現在までに、BLAST や FASTA など様々な相同性検索手法が開発されてきたが、相同配列の検出精度を更に向上させることが求められている。

そこで、本研究では、BLAST、FASTA、WU-BLAST、Pattern Hunter、SCANPS、SSEARCH といった合計 6 つの相同性検索手法について、 2 手法ずつ組み合わせを行うことにより、相同配列の検出精度を向上させることを目指す。具体的に、精度の向上とは sensitivity（データセットに含まれる全ての相同なペアに対する相同性検索手法が出力した相同なペアの比率）と specificity（相同性検索手法が出力した全てのペアに対する相同性検索手法が出力した相同なペアの比率）の両方を向上させることを意味する。

E-value 閾値 3.0*E-3 において、BLAST 単独で用いた場合、sensitivity は 26.2 %、specificity は 99.7 %となった。また、FASTA 単独で用いた場合、sensitivity は 26.9 %、specificity は 99.8 %となった。それに対し、E-value 閾値 3.0*E-3 において、提案手法を適用した BLAST と FASTA の組み合わせでは、BLAST と比較して specificity を低下させることなく、相同なタンパク質ペアの検出数を 458 ペア増やし、sensitivity を 3.0 %、向上させることができた。FASTA と比較して specificity を低下させることなく、相同なタンパク質ペアの検出数を 74 ペア増やし、sensitivity を 0.47 %、向上させることができた。

目 次

第1章　はじめに

2003年3月にヒトゲノムの解読が完了したのをはじめ、マウスやイネ、シロイズナなどさまざまな生物のゲノムの解読が完了している。そして、現在では14万種類を超えるさまざまな生物のゲノム配列が、GenBank[1]やSWISS-PROT[2]などの遺伝情報データベースに登録されている[3]。それに伴い、DNAの塩基配列から、遺伝子に対応するタンパク質のアミノ酸配列のデータを大量に得ることができた。今後は、解読したアミノ酸配列をもとに、タンパク質の機能や立体構造を明らかにし、オーダーメード医療・予防医療の実現や画期的な新薬の開発が求められてくる。タンパク質の機能や立体構造の解析を行うにあたっては、タンパク質に関する膨大な情報をコンピュータを駆使して、高速かつ効率的に情報を処理・解析するバイオインフォマティクスの開発・発展が不可欠である。バイオインフォマティクスにおけるタンパク質研究の最終到達点は、タンパク質のアミノ酸配列、立体構造、機能の３者の関係を理解することである。この３者の関係を理解することにより、１本のアミノ酸配列から機能や立体構造を予測することが可能となる。現在、アミノ酸配列から機能や立体構造情報を抽出する際、最も信頼性が高く実用的な方法が、進化的情報を利用してアミノ酸配列の解析を行う、相同性検索を用いた方法である。

相同性とは、共通祖先に由来する子孫間の類似性を指す。 共通祖先から分岐した相同タンパク質の間では、 類似した構造や機能を有していることが多い。相同性検索とは、機能や構造未知のアミノ酸配列を問い合わせ配列とし、データベース中の既知のアミノ酸配列から、問い合わせ配列と相同なアミノ酸配列を収集する方法である。相同性検索によって得られた相同な配列をもとにマルチプルアラインメントを構築し、そのアラインメントから未知のアミノ酸配列の機能や構造情報を抽出することが可能となる。現在までに様々な相同性検索手法が開発されており、とりわけ、ヒューリスティックで高速な相同性検索手法であるBLASTやFASTAが研究者達の間では広く利用されているが、相同配列の検出精度を更に向上させることが求められている。

そこで、本研究では、複数の相同性検索手法について組み合わせを行うことにより、相同配列の検出精度を向上させることを目指す。英国スコットランドのDundee大学のGeoffrey.J.Barton氏らは、2003年に相同

性検索手法を union 操作や intersection 操作によって組み合わせることにより、相同な配列ペアの検出数を増やすことに成功した [4]。union 操作とは、各手法において閾値以上の配列ペア全てを出力する操作を指す。また、intersection 操作とは、両手法ともに閾値以上の配列ペアに限って出力する操作を指す。

従来研究では、sensitivity（データセットに含まれる全ての相同なペアに対する相同性検索手法が出力した相同なペアの比率）は向上したが、specificity（相同性検索手法が出力した全てのペアに対する相同性検索手法が出力した相同なペアの比率）は低下するという結果となった。本研究では、sensitivity と specificity 両方の向上を目指す。

従来研究では、各相同性検索手法が出力する全てのタンパク質ペアに対して、union 操作、または、intersection 操作を行っていた。それに対し、本研究では、各手法が出力するタンパク質ペアの E-value に応じて、union 操作や intersection 操作を使い分ける手法を提案する。E-value とは、検索に用いたデータベース中から、誤って相同であると判断されるタンパク質の数の期待値を表す。E-value が小さいほど、そのアラインメントは有意であると考えられる。相同なタンパク質ペアである確率が高くなる、E-value が 1.0*E-3 以下のペア同士を組み合わせる場合、union 操作を行い、相同なタンパク質の検出数を増やす。また、相同であるタンパク質ペアである確率が低くなる、E-value が 1.0*E-3 より大きいペア同士を組み合わせる場合、intersection 操作を行い、相同でないタンパク質の検出数を減らす。そして、BLAST、FASTA、WU-BLAST、PatternHunter、SCANPS、SSEARCH といった合計 6 つの相同性検索手法について、2 手法ずつ組み合わせを行う。

また、本論文の構成は以下の通りである。第 2 章では、相同性検索手法についての概要や本研究で用いる相同性検索手法のアルゴリズムなどについて述べる。第 3 章では、本研究で行う組み合わせの方法や実験結果について述べる。第 4 章で、全体の総括を行う。

第2章　相同性検索に関しての予備知識

本章では、相同性検索に関する予備知識として、相同性検索についての概要、配列アラインメント、および、本研究で用いる相同性検索手法について述べる。

2.1　相同性検索についての概要

相同性はタンパク質のアミノ酸配列比較を行う際、配列間の機能や構造を推定するうえで重要な基準となる。相同性、および、相同性検索の定義について以下、述べる。

2.1.1　相同性

相同性（homology、ホモロジー）とは、共通祖先に由来する子孫間の類似性を指す。相同タンパク質とは、共通の祖先遺伝子から種分化や遺伝子重複によって分岐してきた子孫遺伝子産物の一群を指す。相同タンパク質は、その共通祖先からの分岐後、アミノ酸置換や挿入/欠失などの突然変異を受け、次第にそのアミノ酸配列が変化してゆく。しかし、アミノ酸置換や挿入/欠失によって配列が大きく変化しても、相同タンパク質間では機能や立体構造が類似していることが多い。したがって、機能や構造が未知のアミノ酸配列に対し、既知のアミノ酸配列との相同性を見出すことができれば、機能や立体構造を推定するうえでの重要な手がかりとなり得る。

2.1.2　相同性検索の定義

機能や構造が未知のタンパク質のアミノ酸配列を問い合わせ配列（query sequence）として、既知のアミノ酸配列が格納されたデータベースに対して、相同なアミノ酸配列の検索を行うことを相同性検索とよぶ。相同性検索によって得られた相同な配列をもとにマルチプルアラインメントを構築

し、そのアラインメントから未知のアミノ酸配列の機能や構造情報を抽出することが可能となる。

2.2 配列アラインメント

配列アラインメントは相同性検索を行ううえで、最も必須となる技術である。配列アラインメントの良し悪しは、スコアリングモデルによって決まり、このスコアリングモデルにおいて最適なアラインメントを発見するアルゴリズムを動的計画法とよぶ。また、配列アラインメントには、１対の配列を比較するペアワイズアラインメントと、３本ないしそれ以上の配列を比較するマルチプルアラインメントがある。配列アラインメントの定義、スコアリングモデル、動的計画法、ペアワイズアラインメント、および、マルチプルアラインメントについて、以下、述べる。

2.2.1 配列アラインメントの定義

配列アラインメントとは、配列中で同じ並び方をしている連続した文字列や文字パターンを検索することにより、複数の配列を比較する手続きのことである。２本の配列を比較する手続きをペアワイズアラインメント、３本ないしそれ以上の配列を比較する手続きをマルチプルアラインメントとよぶ。また、比較する配列の領域によってアラインメントを大別すると、配列の全域にわたって行うグローバルアラインメントと配列の局所について行うローカルアラインメントの２種類に分けることができる。アラインメントの目的は、比較対象の複数の配列が突然変異（mutation）や自然淘汰（selection）のプロセスによって共通の祖先から分岐してきたという、相同性を示す痕跡を発見することである。突然変異のプロセスには、置換（substitution）、挿入（insertion）、欠失（deletion）がある。アラインメントにおいて、挿入/欠失はギャップとして表される。また、進化の過程で生じるさまざまな突然変異の中から、良さそうな変異をスクリーニングする過程が自然淘汰である。

2.2.2 スコアリングモデル

アラインメント内で対応する配列要素ペアのスコアとギャップのスコアを足したものが総スコアとしてアラインメントに与えられる。配列要素１つ１つのペアに定義される類似度のスコアは、次のように定義される。

- 配列要素 a と b を対応付けるときは、アミノ酸置換行列をもとにス

コア s(a,b) を適用する

- 配列要素とギャップを対応付けるときは、ギャップペナルティを適用する

アミノ酸置換行列、および、ギャップペナルティについて、以下、述べる。

アミノ酸置換行列

アラインメントされた各々のアミノ酸残基のペアには、スコアが与えられる。現在までに、全てのアミノ酸ペア（210 ペア）についてのスコアが考案されてきた。確率論的な視点から、それらのスコアが何を意味しているのかについて捉えていく。

配列 $x_1 \cdots x_n$ と $y_1 \cdots y_n$ のギャップを含まない大域的なペアワイズアラインメントについて考える。与えられたアラインメントについて、（配列間に何らかの関連性がある）/（配列間に何の関連性もない）で表される対数尤度の尺度に基づいたスコアを割り当てることにする。この場合、アラインメントから配列間に何らかの関連性がある確率と配列間に何の関連性もない確率を推定し、その比を考える必要がある。

まず、配列間に何の関連性もない場合のモデル、ランダムモデル（random model）R について考える。文字 a が独立に頻度 q_a で観察されると仮定すると、与えられたペアワイズアラインメントが偶然観察される確率は、アラインメントの各位置におけるアミノ酸残基の観察頻度を掛け合わせたものになる。

$$P(x, y|R) = \prod_i q_{x_i} \prod_j q_{y_j} \tag{2.1}$$

次に、配列間に何らかの関連性がある場合のモデル、一致モデル M について考える。一致モデルでは、アラインメントされたアミノ酸残基のペアは同時確率 p_{ab} で観察されると考える。この p_{ab} の値は、共通の祖先配列の残基 c から、残基 a と b が各々独立に派生した確率と見なすことができる。以上のような考えに基づくと、アラインメント全体の確立は以下のように記述できる。

$$P(x, y|M) = \prod_i p_{x_i y_i} \tag{2.2}$$

これら 2 式の尤度比は、オッズ比とよばれる。

$$\frac{P(x, y|M)}{P(x, y|R)} = \frac{\prod_i p_{x_i y_i}}{\prod_i q_{x_i} \prod_j q_{y_j}} = \prod_i \frac{p_{x_i y_i}}{q_{x_i} q_{y_i}} \tag{2.3}$$

ここで、オッズ比の対数をとることによって、対数オッズ比とよばれる加算的なスコアリングシステムを導出することができる。

$$S = \sum_i s(x_i, y_i) \tag{2.4}$$

ここで、

$$s(a, b) = log\left(\frac{p_a b}{q_a q_b}\right) \tag{2.5}$$

は、アラインメントされていない残基ペアに（a,b）に対するアラインメントされている残基ペアの対数尤度比である。

式 (2.4) は、アラインメントされた各残基ペアのスコア s(a,b) の各々を足し合わせたものとなっている。s(a,b) は行列で表され、タンパク質の場合、行列中の位置 i と j には s(a_i,a_j) が埋め込まれた 20 × 20 の行列となり、この行列をアミノ酸置換行列とよぶ。ここで、a_i と a_j は、ある番号付けに従った i 番目と j 番目のアミノ酸である。本節で述べた方法と本質的には同じ方法で導出されたアミノ酸置換行列の例としては、図 2.1 のような BLOSUM がある。また、BLOSUM 以外でよく利用されるアミノ酸置換行列には、PAM がある。

```
C  9
S -1  4
T -1  1  5
P -3 -1 -1  7
A  0  1  0 -1  4
G -3  0 -2 -2  0  6
N -3  1  0 -2 -2  0  6
D -3  0 -1 -1 -2 -1  1  6
E -4  0 -1 -1 -1 -2  0  2  5
Q -3  0 -1 -1 -1 -2  0  0  2  5
H -3 -1 -2 -2 -2 -2  1 -1  0  0  8
R -3 -1 -1 -2 -1 -2  0 -2  0  1  0  5
K -3  0 -1 -1 -1 -2  0 -1  1  1 -1  2  5
M -1 -1 -1 -2 -1 -3 -2 -3 -2  0 -2 -1 -1  5
I -1 -2 -1 -3 -1 -4 -3 -3 -3 -3 -3 -3 -3  1  4
L -1 -2 -1 -3 -1 -4 -3 -4 -3 -2 -3 -2 -2  2  2  4
V -1 -2  0 -2  0 -3 -3 -3 -2 -2 -3 -3 -2  1  3  1  4
F -2 -2 -2 -4 -2 -3 -3 -3 -3 -3 -1 -3 -3  0  0  0 -1  6
Y -2 -2 -2 -3 -2 -3 -2 -3 -2 -1  2 -2 -2 -1 -1 -1 -1  3  7
W -2 -3 -2 -4 -3 -2 -4 -4 -3 -2 -2 -3 -3 -1 -3 -2 -3  1  2 11
   C  S  T  P  A  G  N  D  E  Q  H  R  K  M  I  L  V  F  Y  W
```

図 2.1: BLOSUM62(参考文献 [5] の図 2 より引用)

ギャップペナルティ

連続したギャップにの領域には、そのギャップの長さに応じたペナルティを与えるものとする。長さ g のギャップに関するギャップペナルティは、線形スコア（linear gap score）、または、アフィンギャップスコア（affine gap score）のいずれかによって与えられる。線形スコアの場合、

$$\gamma(g) = -gd \tag{2.6}$$

によって表され、アフィンギャップスコアの場合、

$$\gamma(g) = -d - (g-1)e \tag{2.7}$$

によって表される。

(2.6) 式や (2.7) 式において、d はギャップ開始ペナルティ（gap-open penalty）、e はギャップ伸長ペナルティ（gap-extension penalty）とよばれる。一般に、d≥e の関係にある。d は使用しているアミノ酸置換行列の最大得点と同程度、e はその 1/10 程度の大きさに設定されることが多い。

2.2.3 動的計画法（dynamic programming）

2.2.2 で述べたスコアリングシステムを用いて、アミノ酸配列の最適なアラインメントを見つけるアルゴリズムは動的計画法とよばれる。動的計画法を用いたグローバルアラインメントやローカルアラインメントについての解法がすでに提案されている。グローバルアラインメントのための解法が、Needleman-Wunsch アルゴリズムであり、ローカルアラインメントのための解法が、Smith-Waterman アルゴリズムである。ここでは、グローバルアラインメントについて以下、２本の短いアミノ酸配列 HEAGAWGHEE と PAWHEAE、および、図 2.2 のアミノ酸置換行列を用いて述べる。図 2.2 は、２本のサンプル配列について、全ての残基ペアに相当する値を BLOSUM50 置換行列から抜粋したものである。また、残基ごとのギャップコストとしては d=-8 を用いる。

	H	E	A	G	A	W	G	H	E	E
P	-2	-1	-1	-2	-1	-4	-2	-2	-1	-1
A	-2	-1	**5**	0	**5**	-3	0	-2	-1	-1
W	-3	-3	-3	-3	-3	**15**	-3	-3	-3	-3
H	**10**	0	-2	-2	-2	-3	-2	**10**	0	0
E	0	**6**	-1	-3	-1	-3	-3	0	**6**	**6**
A	-2	-1	**5**	0	**5**	-3	0	-2	-1	-1
E	0	**6**	-1	-3	-1	-3	-3	0	**6**	**6**

図 2.2: BLOSUM50 より、２本のサンプル配列について、その全ての残基ペアに相当する値の抜粋

グローバルアラインメント（global alignment）：Needleman-Wunsch アルゴリズム [6]

グローバルアラインメントは配列全域にわたって行うアラインメントであり、配列全体の類似性を調べることが目的である。動的計画法を用いた Needleman-Wunsch アルゴリズムは、ギャップを許した２本の配列の最適なグローバルアラインメントを求めるように設計されている。

Needleman-Wunsch アルゴリズムの基本的なアイディアは、より小さな部分配列の最適アラインメントをひとつ前の解として、最適なアラインメントを次々と組み上げていくことである。各配列について、i と j でインデックスされた DP 行列 F を考える。ここで、F(i,j) の値は、x の x_i までのセグメント $x_1 \cdots x_i$ と y の y_j までのセグメント $y_1 \cdots y_j$ 間の最適アラインメントのスコアである。この F(i,j) を再帰的に計算していく。計算は、F(0,0) から始め、この行列の左上から右下に進みながら、各値を埋めていく。F(i-1,j-1),F(i-1,j),F(i,j-1) が計算されていれば、F(i,j) を計算することができる。x_i,y_j までのアラインメントの最適スコア F(i,j) は、以下の３通りの計算方法が考えられる。x_i と y_j をアラインメントする場合 F(i,j)=F(i-1,j-1)+s(x_i,y_j)、x_i とギャップをアラインメントする場合 F(i,j)=F(i-1,j)-d、y_j とギャップをアラインメントする場合 F(i,j)=F(i,j-1)-d の３通りである。この３通りの中で最もスコアの高いものが (i,j) までの最適アラインメントである。

以上を数式で表現すると式 (2.8) のようになる。

$$F(i,j) = max \begin{cases} F(i-1,j-1) + s(x_i, y_j) \\ F(i-1,j) - d \\ F(i,j-1) - d \end{cases} \tag{2.8}$$

この式を繰り返し適用し、F(i,j) の値を次々と求める。つまり、図 2.3 に示すように、行列における各２×２の４つのセルを考え、右下のセルの値を、左上、左、上のセルの値のひとつから計算する。

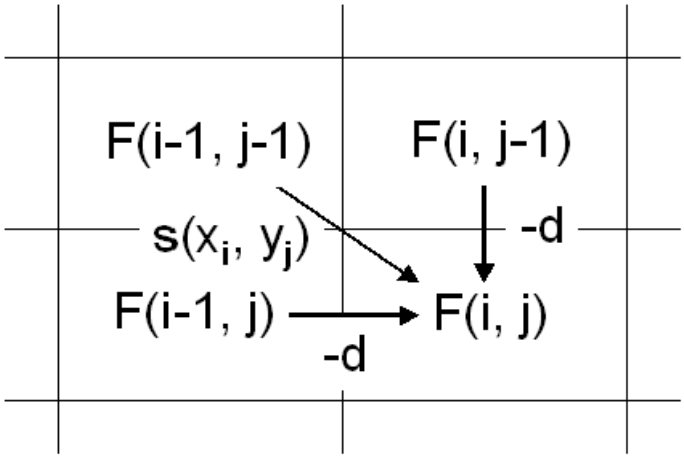

図 2.3: 最適スコアの計算方法

F(i,j) の値を計算するとともに、その値がどのセルの値から計算されたかを示すポインタを保持しておく。

DP 行列の一番上の行では（j=0）、F(i,j-1) や F(i-1,j-1) の値を定義することはできないので、F(i,0) の値を特別に扱わなければならない。この値は、x と全てギャップからなる y がアラインメントされていることを表しているため、F(i,0)=-id とする。同様に、左端の列は F(0,j)=-jd とする。

行列の一番右下のセル（最終セル）の値 F(n,m) が、$y_1 \cdots y_m$ に対する $x_1 \cdots x_n$ のアラインメントスコアの最大スコア、つまり x と y の大域アラインメントの最大スコアである。アラインメントそのものを求めるには (2.8) 式の選択により最終セルに至ったパスを求めなければならない。この操作を、トレースバック（traceback）とよぶ。トレースバックでは、行列に値を埋めながら保持してきたポインタを最終セルから逆に辿り、アラインメントを組み上げていく。トレースバックの各ステップでは、現在のセルから F(i,j) を求めた一つ前のセル、つまり (i-1,j-1),(i-1,j),(i,j-1) セルのいずれかに遡って移動していく。同時に、その時点で求められているアラインメントの先頭に文字ペアを追加していく。すなわち、このステップ

で (i-1,j-1) に移動したのであれば x_i と y_j、(i-1,j) に移動したのであれば x_i とギャップ文字'-'、(i,j-1) に移動したのであればギャップ文字'-' と y_j を追加する。トレースバックは、行列の開始点 i=j=0 に至るまで続けられる。図 2.4 は、トレースバックを行った DP 行列である。

		H	E	A	G	A	W	G	H	E	E
	0	**-8**	**-16**	-24	-32	-40	-48	-56	-64	-72	-80
P	-8	-2	-9	**-17**	**-25**	-33	-41	-49	-57	-65	-73
A	-16	-10	-3	-4	-12	**-20**	-28	-36	-44	-52	-60
W	-24	-18	-11	-6	-7	-15	**-5**	**-13**	-21	-29	-37
H	-32	-14	-18	-13	-8	-9	-13	-7	**-3**	-11	-19
E	-40	-22	-8	-16	-16	-9	-12	-15	-7	**3**	-5
A	-48	-30	-16	-3	-11	-11	-12	-12	-15	**-5**	2
E	-56	-38	-24	-11	-6	-12	-14	-15	-12	-9	**1**

図 2.4: サンプル配列のグローバルアラインメント行列（矢印はトレースバックのためのポインタを表している）

この DP 行列より、サンプル配列 HEAGAWGHEE と PAWHEAE についての最適なグローバルアラインメントは図 2.5 のようになる。

```
H E A G A W G H E - E
- - P - A W - H E A E
```

図 2.5: サンプル配列の最適グローバルアラインメント

2.2.4 E-value[14]

E-value とは、検索に用いたデータベース中から、誤って相同であると判断されるタンパク質の数の期待値を表す。E-value は小さければ小さいほど、類縁関係を見出した可能性が高くなり、そのアラインメントは有意であるといえる。

E-value の算出方法の例として、ギャップ無しローカルアラインメントの E-value について述べる。

ギャップ無しのローカルアラインメントは、同じ長さの部分配列のペアによって構成される。Smith-Waterman アルゴリズムや Seller アルゴリズムによって、比較される両配列間のギャップを含まない類似度の高い領域 (HSP) が発見される。

アルゴリズムが発見した HSP のローカルアラインメントスコアが、どのような分布になるか解析するために、ランダム配列のモデルが必要である。ランダム配列モデルでは、アミノ酸組成 P_i(i=1,2,$\cdots$,20) とスコアテーブルが与えられると HSP のローカルアラインメントスコアの分布が決まる。比較する 2 本の配列の長さを m、n とし、HSP の分布を記述するパラメータを λ と K とする。λ は分布の広がりを定める値であり、K は分布の山の位置を定めるのに関係する値である。スコア S 以上をもつ HSP が現れる本数に対する期待値は式 (2.9) のようになる。

$$E - value = Kmne^{-\lambda S} \tag{2.9}$$

2.3 相同性検索手法

2.3.1 SSEARCH[7]

SSEARCH は，米国 Virginia 大学の W.R.Pearson ら FASTA チームによって開発された相同性検索手法で，FASTA package に含まれている。相同性検索手法の中で，最も感度が高い手法であると言われている。Smith-Waterman アルゴリズムを実装することで，より厳密な検索を可能にした。SSEARCH が行う、Smith-Waterman アルゴリズムを用いた local アラインメントについて、以下に述べる。2.2.3 節で述べたグローバルアラインメントと同様に、ローカルアラインメントについて以下、2 本の短いアミノ酸配列 HEAGAWGHEE と PAWHEAE、および、図 2.2 のアミノ酸置換行列を用いて述べる。また、残基ごとのギャップコストとしては d=-8 を用いる。

ローカルアラインメント（local alignment）：Smith-Waterman アルゴリズム

ローカルアラインメントは部分配列に対して行うアラインメントであり、類似性の高い部分を局所的に調べることが目的である。比較する両方の配列において、途中から始まる類似領域の検出が可能である。ローカル

アラインメントとして、動的計画法を用いた Smith-Waterman アルゴリズムが考案されている。

Smith-Waterman アルゴリズムは、Needleman-Wunsch アルゴリズムと類似しているが、主な違いとして 2 点ある。

ひとつは、DP 行列の各セルについて、(2.8) 式に新しい選択肢を加えることである。それは、他の全ての選択肢の値が 0 以下であれば、F(i,j) の値として 0 を採用し、(2.8) 式を (2.10) 式に変更することである。

$$F(i,j) = max \begin{cases} 0 \\ F(i-1,j-1) + s(x_i, y_j) \\ F(i-1,j) - d \\ F(i,j-1) - d \end{cases} \tag{2.10}$$

F(i,j) の値として 0 を採用することは、新しいアラインメントが始まることに相当する。ある位置までの最適アラインメントが負のスコアであれば、そのアラインメントを伸長させていくより、新しいアラインメントを開始させた方が良いということである。そのため、Needleman-Wunsch アルゴリズムでは、DP 行列の一番上の行と一番左の列の境界条件として-id と-jd をそれぞれ代入していたのに対し、Smith-Waterman アルゴリズムでは、全て 0 が代入されている。

次の違いは、アラインメントが DP 行列のあらゆる所で終わる点である。つまり、Needleman-Wunsch アルゴリズムでは最大スコア F(n,m) が行列の最も右下のセルに格納されていたが、Smith-Waterman アルゴリズムでは最大スコア F(i,j) を DP 行列中から探索し、そこからトレースバックを行うという点である。トレースバックは、アラインメントの開始に相当する 0 が格納されたセルに到達するまで続けられる。トレースバックを行った DP 行列を図 2.6 に示す。

この DP 行列より、サンプル配列 HEAGAWGHEE と PAWHEAE についての最適なローカルアラインメントは図 2.7 のようになる。

2.3.2 FASTA[8]

FASTA は連続して一致する配列の断片を高速に検索し、それらの断片の中で類似度の高いものに着目して局所的なアラインメントを行い、最後にこれらをギャップを考慮し結合して、最終的なアラインメントを行う手法である。FASTA は 1980 年代後半、米国 Virginia 大学の W.R.Pearson らにより開発された。FASTA は現在、一連の更新や改善を経てバージョン 3 となり、FASTA3 とよばれる。FASTA3 では、配列をアラインメントする方法や、アラインメントの統計的有意性を計算する方法が改善され

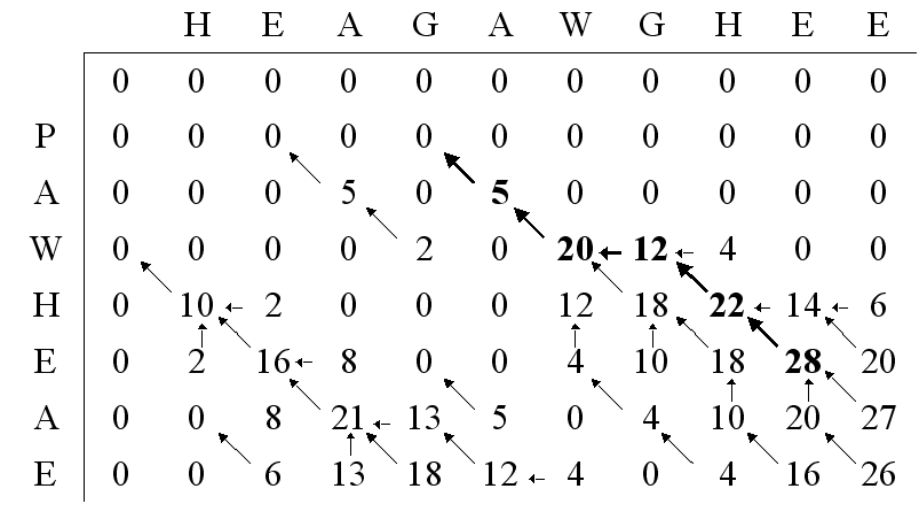

図 2.6: サンプル配列のローカルアラインメント行列（矢印はトレースバックのためのポインタを表している）

A W G H E
A W - H E

図 2.7: サンプル配列の最適ローカルアラインメント

ている。これらの変更により、FASTA3 は遠縁の配列を見つける能力が増大している。

FASTA では、以下のような手順を踏んで相同性検索を行う。

1. 問い合わせ配列をもとに、ワードのハッシュテーブルを作成する。

   ハッシュ化は、ワードに整数を割り当てることで、探索空間を小さくできる方法である。

   ハッシュテーブルは、例えば図 2.8 のように、ハッシュ関数に従ってマトリックスの各位置にワードを割り当てたものである。タンパク質の場合は、ワード長は 1 または 2 アミノ酸にすることが多い。塩基配列の場合は、ワード長は通常 4〜6 塩基にする。FASTA では、このキーにする文字数をパラメータとし、k タプル (k-tuple) とよんでいる。例えば、塩基配列を比較する際に、タプルサイズを 2 とすると、4 種類の塩基に対し、キーとして 16 種類考えられる。単純に考えると、この場合はタプルサイズが 1 のときに比べて、16 倍のス

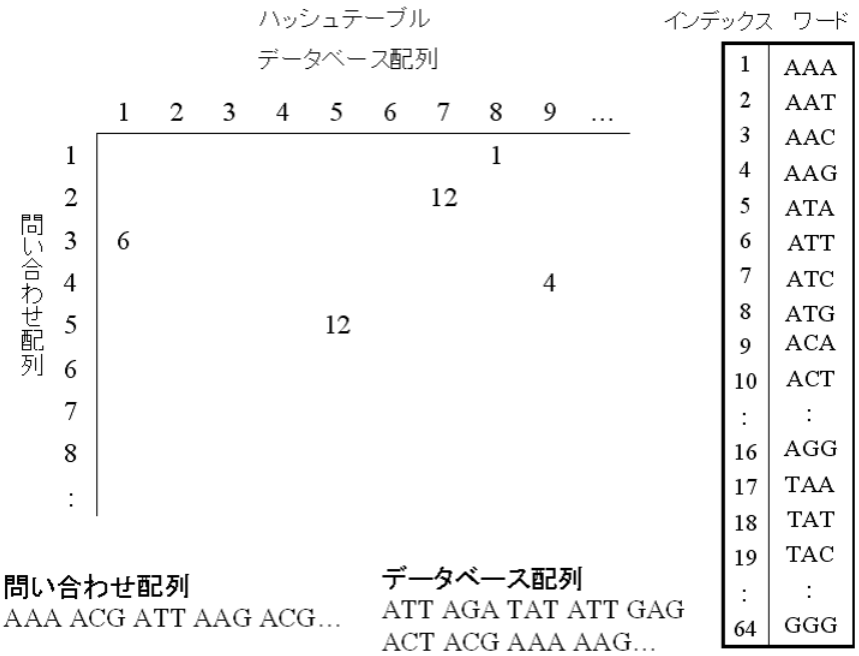

図 2.8: FASTA のハッシュテーブル

ピードアップになる。ただし、当然のことであるがキーとなる 2 文字とも一致しなければ検出できないので、例えば、AC と AG が半分一致しているといったことは分からなくなってしまう。すなわち、タプルサイズを大きくすると検索は高速になるが感度は悪くなってしまう。計算時間と検索感度にはトレードオフが存在する。

2. 問い合わせ配列をデータベース中の配列と比較する。データベース中の配列は、あらかじめ問い合わせ配列と同じ長さのワードでハッシュ化しておく。局所的に文字が連続して一致している部分は、ハッシュテーブル上で対角線の線分要素として検出することができる。このようにして検出された複数の類似度の高い領域について、同一残基の一致に対してはある一定のスコアを、非同一残基の一致に対してはある一定のギャップペナルティを与えることによって、スコアを算出する。算出されたスコアの良いものから順に 10 領域を選ぶ。さらに、それらのスコアが置換行列を用いて再計算され、最も高いスコアを与える配列断片を切り出す。この段階での最高スコアがパラメータ init1 として記録される。ギャップのない一致した領域が得られ、これを highest scoring initial region とよび、そのスコアをパラメータ init1 として記録する。

3. highest scoring initial region をギャップコストを考慮して結合する。各 highest scoring initial region のスコアの和にギャップペナルティを加えたものを、結合して得られた領域のスコアとし、これをパラメータ initn として記録する。

4. 得られた領域の周辺で、Smith-Waterman アルゴリズムを適用し、最適なアラインメントを求める。ここで得られたスコアをパラメータ opt として記録する。

2.3.3 BLAST[9]

BLAST（Basic Local Alginment Search Tool）は、1990 年に NCBI（National Center for Biotechnology Information）の S.F.Altschul らによって開発された。FASTA と同様に、BLAST もヒューリスティックな相同性検索手法であり、FASTA などといった、BLAST 開発以前のプログラムと比較して飛躍的な高速化に成功し、現在、世界中の研究者達に広く使われている。また、開発以降、改良が重ねられ、ver2.0 以降ではギャップ入りのアラインメントが可能となっている。

BLAST では以下のような手順を踏んで相同性検索を行う。

1. 問い合わせ配列を長さ k（デフォルトではアミノ酸配列で 3、塩基配列で 11）のワードに分割し、さらに各ワードに類似したワードのリストを生成する。この際、スコア行列を用いて、閾値以上の値でマッチするワードを類似ワードと定義する。一般に、閾値としては 2 ビット/残基が利用される。そして、データベース配列中に、生成されたワードに一致する部分をヒットとして検出する。

2. 検出されたヒットを起点として、問い合わせ配列とデータベース配列の局所アラインメントを、N 末側、C 末側の両方向に伸ばしていき、スコアが最大となったところでその伸長を停止させる。ここで得られたスコアが閾値 S を超える場合、その領域は HSP（High-scoring Segment Pairs）として報告される。ここで用いられる閾値 S は、ランダムな配列と比較して見つかったスコアの範囲を確かめたり、有意に大きな値を選んだりすることによって、経験的に決める必要がある。

3. 報告された複数の HSP に対し、それぞれの HSP スコアの統計的有意性を決める。ここで有意であると判断された HSP に対し、その HSP を含む配列と問い合わせ配列とのアラインメントを Smith-Waterman アルゴリズムを用いて求める。BLAST の初期のバージョ

ンでは、最初に見つかったHSPを含んだギャップなしアラインメントだけが生成されていた。もし、2つのHSPが見つかったときには、2つの領域はギャップを入れずにアラインメントできないので、2つの別々のアラインメントとして生成していた。BLAST2では、最初に見つかったHSP領域全てを含むようにギャップを入れて1つのアラインメントを生成することが可能となった。

2.3.4 WU-BLAST[10][11]

WU-BLASTは、米国Washington大学によるBLASTのバージョンである。WU-BLAST version1.4ではNCBI BLAST version1.4のいくつかのバグが修正され、1995年にweb上で公開された。また、1996年には、ギャップに統計的に対応するように実装されたWU-BLAST version2.0d1が公開された。現在のWU-BLAST version2.0における主な特徴を以下に述べる。

- BLASTP（アミノ酸配列×アミノ酸配列）やBLASTN（塩基配列×塩基配列）など、全てのBLASTの検索モードで、ギャップに対応したアラインメントを行える。また、オプションで設定すれば、ギャップ無しのアラインメントも実行可能である。
- データベース配列との複数の類似領域を認識することで、sensitivityやselectivityを向上させている。
- WU-BLAST2.0では、ギャップ有りのアラインメントを行うが、ギャップ無しのWU-BLAST1.4よりも実行時間は速く、sensitivityを損ねることもない。ただし、BLASTN（塩基配列×塩基配列）の検索モードにおいては、デフォルトのパラメータで、約10 %速度が低下した。

2.3.5 SCANPS[12]

SCANPSは1997年にG.J.Barton氏によって開発された相同性検索手法である。その後、1999年に繰り返し検索を行えるように改良された。SCANPSの特徴としては以下の通りである。

- Smith-Waterman アルゴリズムを実装した相同性検索手法である。
- プロファイルを用いた繰り返し検索を行うことができる。また、繰り返し検索にもDPを適用する。

2.3.6 PatternHunter[13]

Bioinfomatics Solutions（カナダ・ワーテルロー）で開発された PatternHunter は、相同性検索の実行速度をアップさせる spaced seed という概念が提案され、従来の BLAST の約 100 倍のスピードを実現することが可能とされている。塩基配列を比較する際、BLAST では 11 個の連続した残基が一致する領域を検索するのに対して、PatternHunter は、例えば、18 残基の部分配列において 11 箇所の一致を検索する。その結果、PatterHunter による検索は、より感度が高く、驚くほど速い。PatternHunter は、multiple seed 方式を用いることで、Smith-Waterman アルゴリズムと同等の感度を、最大で Smith-Waterman アルゴリズムの 1000 倍以上の速度で実現することが可能となっている。

speced seed 法

BLAST の場合、塩基配列では 11 残基、アミノ酸配列では 3 残基の連続した類似度の高い領域をベース（コア）とし、そのコア領域を中心として相同性領域の適合度の計算を行う。それに対し、PatternHunter で採用されている spaced seed 法では、連続して一致する領域ではなく特定の一致パターンをコアとして利用する。つまり、塩基配列の場合、例えば 18 残基の部分配列において 11 箇所の一致を検索する。比較する配列間において、残基の一致を要求する位置を 1 で表現し、一致しても一致しなくても構わない位置を 0 で表現すると、塩基配列を比較する際、PatternHunter では、例えば、111010010100110111 となるような領域を検索する。

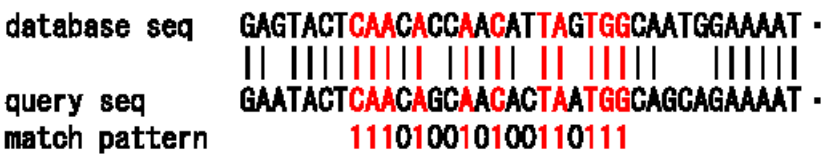

図 2.9: spaced seed 法

BLAST では、長く連続した類似度の高い領域を見つけようとすると、弱い相同性が検知できず、一方、短く連続した類似度の高い領域を見つけようとすると、たくさんヒットしてしまい実行速度が遅くなってしまうというジレンマがあった。PatternHunter では、spaced seed 法により、相同な領域に対するヒットを増やし、相同でない領域に対するヒットを減らすことができ、BLAST のジレンマに対処し、BLAST より高感度で高速な相同性検索を実現することが可能となった。

optimized multiple spaced seed 法

PatternHunter version2.0 では、最適な spaced seed を複数選択して、相同性検索を行うことができるように改良された。

spaced seed セット A が $\{a_1,\cdots,a_k\}$ といった k 個の spaced seed で構成されるとする。a_1∈A が、比較する配列ペアにヒットする場合、$\{a_1,\cdots,a_k\}$=A がヒットするといえる。DP を用いて、spaced seed a_i∈A(i=1,$\cdots$,k) が、比較する配列ペアにヒットする確率を算出する。ヒットする確率が最大となる spaced seed を a_x とする。次に、DP を用いて spaced seed セット B=$\{a_x,a_j\}$(j=1,$\cdots$,k) がヒットする確率を算出する。spaced seed セット B がヒットする確率が最大となるように、a_y を選択する。以上に述べたことを spaced seed セット B の要素数が、設定した spaced seed の個数となるまで繰り返す。

spaced seed の数を増やすことによって、検索感度を向上させることができる。しかし、spaced seed の数を増やすと、ヒットの数やコンピュータのメモリの使用量が増え、ヒットを検証するのに時間がかかり、検索速度が遅くなってしまう。

PatternHunter version2.0 ではアミノ酸配列比較の場合、spaced seed の数は、最大で 4 個まで設定することができる。

2.3.7 まとめ

表 2.1 は 2.3 節で述べた相同性検索手法についてまとめを行った表である。

表 2.1: 相同性検索手法についてのまとめ

SSEARCH [7]	長所	Smith-Watermanアルゴリズムを利用した高感度な検索が可能
	短所	検索時間が遅い
FASTA [8]	長所	ヒューリスティック手法を用いた高速な検索が可能
	短所	SSEARCHよりも精度が落ちる
BLAST [9]	長所	ヒューリスティック手法を用いた高速な検索が可能
	短所	SSEARCHやFASTAよりも精度が落ちる
WU-BLAST [10][11]	長所	BLASTを改良し、ギャップに対応させることやヒューリスティック手法を用いた高速な検索が可能
	短所	SSEARCHよりも精度が落ちる
SCANPS [12]	長所	Smith-Watermanアルゴリズムを利用した高感度な検索やプロファイルを利用した繰り返し検索が可能
	短所	検索時間が遅い
PatternHunter [13]	長所	spaced seed法を利用した高速かつ高感度な検索が可能
	短所	メモリ使用量が多い

第3章　相同性検索手法の組み合わせ

3.1　関連研究

3.1.1　概要

2003年、英国スコットランドのDundee大学のGeoffrey J.Barton氏らは、相同性検索手法を組み合わせることによって、相同な配列の検出数、すなわち、coverageを増やすことに成功した。実験では、ローカルアラインメントを用いるPRSS、SSEARCH、SCANPS、グローバルアラインメントを用いるGSRCH、AMPS、ヒューリスティックな手法を用いる、BLAST、FASTAの７つの相同性検索手法を組み合わせに用いた。７つの相同性検索手法はデフォルトのパラメータに設定されたうえで実験が行われた。また、GSRCHの置換行列としてBLOSUM50とBLOSUM62の２通りを用いたため、合計８手法に対し、２手法ずつ同じP-value閾値における出力結果についてunion、intersection操作を行った。union操作とは、各手法において閾値以上の配列ペア全てを出力する操作を指す。また、intersection操作とは、２つの手法において閾値以上の配列ペアに限って出力する操作を指す。

3.1.2　結果

56通りの組み合わせを行った結果、表3.1に示す19通りの組み合わせで、組み合わせの元となる手法に比べてcoverageが増加した。特に、相同でない配列の検出数が少ない場合、coverageが著しく増加した組み合わせもあった。その例としては、SSEARCHとGSRCH（BLOSUM62）の組み合わせでは、相同でないタンパク質の検出数が5ペアのとき、組み合わせの元となる手法のSSEARCHに比べて12.4％、coverageの増加に成功した。

表 3.1: 相同でないタンパク質ペアの検出数が 5 における coverage（参考文献 [4] より引用

Method A	Method B	Set Operation	True positives hit	%increase over parent methods	*p*-value cut-off
GSRCH62	SSEARCH	Intersection	444	12.4/14.4	3.5e−05
AMPS	SCANPS	Intersection	435	22.2/8.5	1.3e−04
AMPS	SSEARCH	Intersection	426	19.7/9.8	1.7e−04
GSRCH62	SCANPS	Intersection	426	7.8/6.2	1.3e−04
GSRCH50	SCANPS	Union	424	7.6/5.7	3.5e−06
GSRCH50	SCANPS	Intersection	423	7.4/5.5	9.3e−05
GSRCH62	SCANPS	Union	423	7.1/5.5	3.5e−06
GSRCH50	SSEARCH	Intersection	423	7.4/9.0	1.74e−04
BLAST	GSRCH50	Union	421	12.6/6.9	6.2e−06
BLAST	GSRCH62	Union	421	12.6/6.6	5.8e−06
GSRCH50	SSEARCH	Union	413	4.8/6.4	4.2e−06
AMPS	PRSS	Intersection	412	15.7/7.9	1.3e−04
GSRCH62	SSEARCH	Union	410	3.8/5.7	4.2e−06
FASTA	GSRCH50	Union	407	17.6/3.3	5.9e−06
GRSCH50	PRSS	Union	407	3.3/6.5	4.7e−06
AMPS	BLAST	Intersection	403	13.2/7.8	2.5e−04
BLAST	SSEARCH	Union	403	7.8/3.9	1.1e−05
BLAST	FASTA	Union	388	3.7/12.1	1.3e−05
AMPS	FASTA	Union	368	3.4/6.4	1.0e−06

3.1.3 関連研究と本研究との位置づけ

関連研究では、union 操作や intersection 操作を行うことによって相同な配列の検出数の向上に成功した。本研究では、E-value 閾値によって union 操作や intersection 操作を効果的に使い分けることで、相同な配列の検出数や検出された配列ペアのうち相同な配列ペアが占める割合も増やすことを目指す。

3.2 組み合わせについての概要

3.2.1 相同についての定義

タンパク質データベース SCOP では構造既知のタンパク質に対し、構造的、進化的類似を描写するために、ファミリー、スーパーファミリー、フォールド、クラスといった階層的な分類を用いている。２本の配列を比べるとき、ファミリー、スーパーファミリーなどの低位の階層において同じものに属する場合は、その２本の配列は機能、構造的に類似していると判定できる。そして、本研究では、スーパーファミリーが同じものに属するタンパク質ペアを相同であると定義する。相同性検索手法が相同だと判断したペアが実際に相同であるペアの場合、そのタンパク質ペアを true positive とよぶ。また、フォールドが異なるものに属するタンパク質ペアを相同でないと定義する。相同性検索手法が相同だと判断したペアが実際には相同でないペアの場合、そのタンパク質ペアを false positive とよぶ。

3.2.2 sensitivity、specificity についての定義

本研究では、複数の相同性検索手法について組み合わせを行うことにより、相同な配列ペアの検出数を増やすといった sensitivity（感度）や、検出された配列ペアのうち相同な配列ペアが占める割合といった specificity（特異性）を向上させることを目指す。

sensitivity、specificity は以下の式に基づいて算出する。

$$sensitivity = \frac{\text{相同性検索手法が見つけてきた相同なタンパク質ペア}}{\text{データセットに含まれる全ての相同なタンパク質ペア}}$$

$$specificity = \frac{\text{相同性検索手法が見つけてきた相同なタンパク質ペア}}{\text{相同性検索手法が拾った全てのタンパク質ペア}}$$

3.2.3　組み合わせ方法

まず、相同性検索手法を組み合わせるに際し、手法を単独で用いる。相同性検索を行い、タンパク質ペアの E-value について閾値を定め、閾値以下のタンパク質ペアを出力する。E-value とは、検索に用いたデータベース中から、誤って相同であると判断されるタンパク質の数の期待値を表す。E-value が小さいほど、そのアラインメントは有意であると考えられる。そして、２種類の相同性検索の手法に対し、同じ E-value 閾値における出力結果について union、intersection 操作を行い、手法の組み合わせを図る。union 操作とは、各手法において閾値以上の配列ペア全てを出力する操作を指す。また、intersection 操作とは、２種類全ての手法において閾値以上の配列ペアに限って出力する操作を指す。組み合わせの結果として出力されたタンパク質ペアに対し、true positive または false positive のどちらであるか確認する。

3.2.4　組み合わせ手法の有用性

２つの相同性検索の出力結果に対し、union、intersection 操作を行うことには、以下のような有用性があると考えられる。

union 操作

union 操作は下図のように、true positive となったタンパク質ペアについては、同じペアを共通してはあまり含まず、false positive となったタンパク質ペアについては、共通してたくさんのペアを含むような場合、有効であると考えられる。

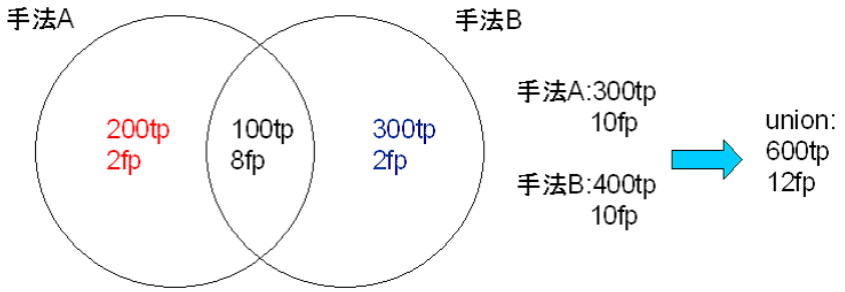

図 3.1: union 操作の有用性

また、一般に union 操作の長所・短所として、

- 単独で用いる場合と比べて、sensitivity を確実に向上させることが

できる

- 単独で用いる場合と比べて、specificity を低下させてしまう可能性がある

といったことが考えられる。

intersection 操作

intersection 操作は下図のように、true positive となったタンパク質ペアについては、共通した同じペアをたくさん含み、false positive となったタンパク質ペアについては、共通したペアをあまり含まない場合、有効であると考えられる。

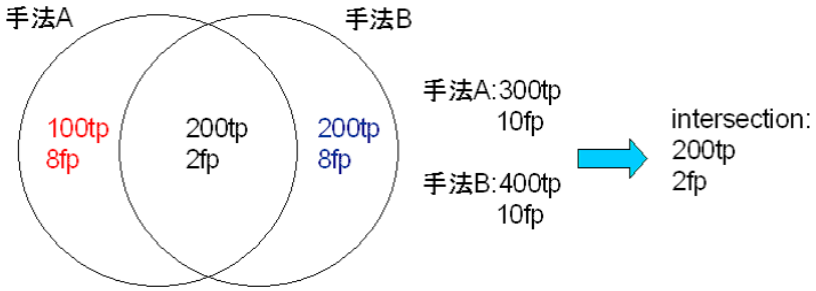

図 3.2: intersection 操作の有用性

また、一般に intersection 操作の長所・短所として、

- 手法単独で用いる場合と比べて、specificity を向上させる可能性が高い
- 手法単独で用いる場合と比べて、sensitivity を確実に低下させてしまう

といったことが考えられる。

3.2.5 本研究で用いる相同性検索手法

本研究では、BLAST、FASTA、SSEARCH、SCANPS、WU-BLAST、PatternHunter の合計 6 つの手法を用いて組み合わせを行う。また、ギャップペナルティやスコアテーブルなどのパラメータはデフォルトのものを用いて実験を行う。ただし、相同配列の検出精度を向上させるために、PatternHunter については、spaced seed を 4 に設定、FASTA については

k-tuple を 1 に設定したうえで、実験を行う。PatternHunter version2.0 ではアミノ酸配列同士の比較を行う際、spaced seed の数を最大で 4 個まで設定できる。PatternHunter では、spaced seed を増やすことによって検索時間は遅くなるが、検索感度を向上させることができる。また、FASTA では k-tuple を小さくすることによって検索時間は遅くなるが、検索感度を向上させることができる。

また、表 3.2 は本研究で用いる相同性検索手法について、アルゴリズムやパラメータについてまとめた表である。

表 3.2: 相同性検索手法についての比較表

相同性検索手法	アルゴリズム	アミノ酸置換行列	ギャップペナルティ		備考
			開始	伸長	
BLAST	ヒューリスティック手法	BLOSUM62	11	1	デフォルト
FASTA	ヒューリスティック手法	BLOSUM50	10	2	k-tuple=1
SSEARCH	Smith-Waterman アルゴリズム	BLOSUM50	10	2	デフォルト
SCANPS	Smith-Waterman アルゴリズム	BLOSUM50	12	2	デフォルト
WU-BLAST	ヒューリスティック手法	BLOSUM62	9	2	デフォルト
PatternHunter	ヒューリスティック手法	BLOSUM62	11	4	spaced seeds=4

3.2.6 本研究で用いるデータセット

タンパク質立体構造データベース ASTRAL から入手した sequence identity が 40 %未満の配列データセットを用いて、実験を行う。表 3.3 は、本データセットについて取った統計である。

また、本データセットに含まれる配列の本数は 5674 本であった。その全 16094301 ペアのうち、スーパーファミリーが同じものに属しているタンパク質ペアは 58853 ペア、フォールドが同じものに属しているタンパク質ペアは 139728 ペア、フォールドが異なるものに属しているタンパク質ペアは 15954573 ペアであった。

3.3 手法単独で用いた場合の結果

3.3.1 true positive および false positive の推移

図 3.3 は、各相同性検索手法が E-value 閾値 1.0*E-15〜1.0*E-1 において、false positive が 50 ペア検出されるまでの、true positive の検出数の推移を表した図である。

表 3.3: データセットについての統計

Class	Number of folds	Number of superfamilies	Number of families
All alpha proteins	179	299	480
All beta proteins	126	248	462
Alpha and beta proteins (a/b)	121	199	542
Alpha and beta proteins (a+b)	234	348	566
Multi-domain proteins	38	38	53
cell surface proteins	36	66	73
Small proteins	66	95	146
Total	800	1293	2322

また、表 3.4 は、各相同性検索手法の Error level が 5、20、50 における、true positive の検出数を表した表である。

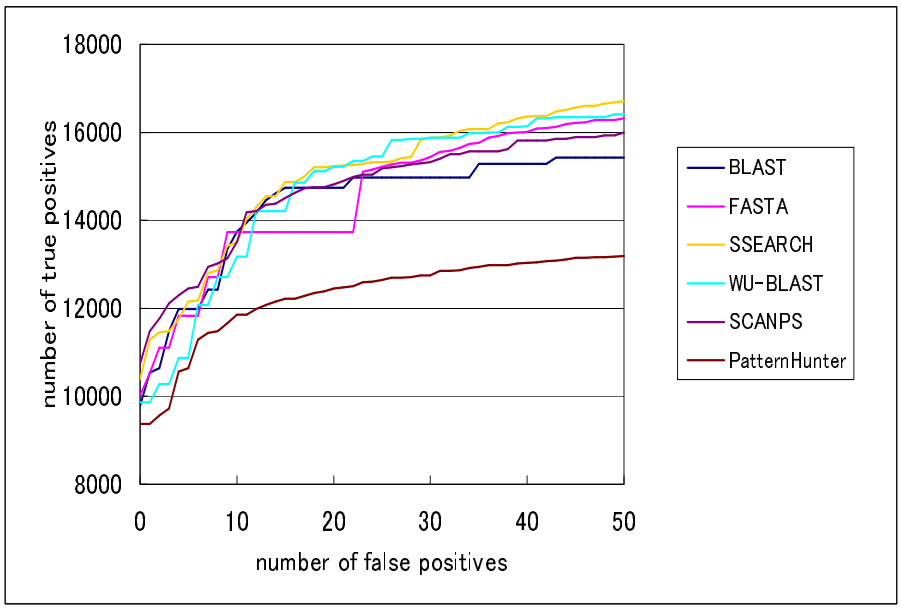

図 3.3: true positive および false positive の推移

表 3.4: false positive が 5、20、50 ペア検出される際の true positive

	Error level (false positives)		
Method	5	20	50
BLAST	11989	14746	15422
FASTA	11829	13736	16314
SSEARCH	12152	15229	16709
WU-BLAST	10872	15218	16408
SCANPS	12445	14820	15994
PatternHunter	10634	12448	13185

SSEARCH や SCANPS は、SmithWaterman アルゴリズムを用いた local アラインメントを実装したことにより、相同配列の検出感度が高いとされている。今回の実験結果では、Error Level（false positive の検出数）が 0〜8 までの間では、SCANPS が最も多くの true positive を検出していた。また、Error Level が 12〜13、15〜21、32〜50 までの間では、SSEARCH が最も多くの true positive を検出していた。逆に、PatternHunter に関しては、どの Error Level においても、他の手法と比較すると、true positive の検出数が少なかった。PatternHunter version2.0 では、アミノ酸配列比較の際、spaced seed の数を最大で 4 個まで設定することができる。Pat-

ternHunter では、spaced seed の数を増やすことによって、検索速度は遅くなるが、検索感度は向上する。図 3.3 や表 3.4 において PatternHunter は、検出精度を向上させるために、spaced seed を 4 に設定した結果である。それに対し、図 3.4 は spaced seed の数を、1 個、2 個、4 個と変更することで、true positive の検出数にどのような影響を及ぼすか観察した図である。spaced seed の数を少なく設定すると、高速に検索を行えるが、true positive の検出数は少ない結果となった。また、逆に spaced seed の数を多く設定すると、検索に時間がかかってしまうが、true positive の検出数は多い結果となった。

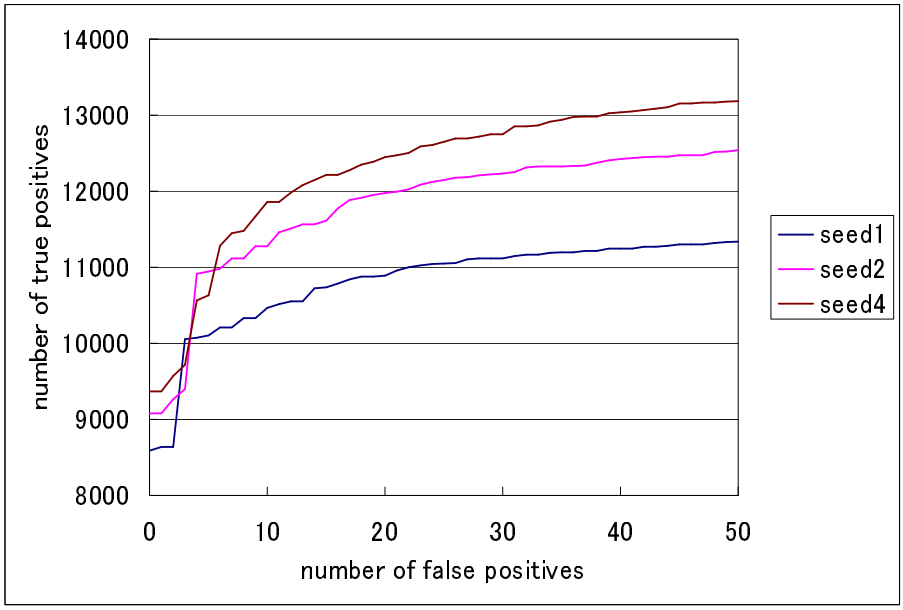

図 3.4: PatternHunter の spaced seed の数による true positive の検出数の変化

また、図 3.3 や表 3.4 の結果からでは E-value 閾値が各相同性検索手法の sensitivity、specificity にどのような影響を及ぼしたのかといったことを読み取ることができない。したがってこの結果のみから、各相同性検索手法についての優劣を判断することはできない。E-value 閾値によって、各相同性検索手法の sensitivity、speicifity がどのように変わっていくか、以下、述べる。

3.3.2 sensitivity

sensitivity は、以下の式で算出される。

$$sensitivity = \frac{\text{相同性検索手法が見つけてきた相同なタンパク質ペア}}{\text{データセットに含まれる全ての相同なタンパク質ペア}}$$

図 3.5 は、E-value 閾値（1.0*E-50〜1.0）によって、各手法の sensitivity がどのように変化したのかを表す図である。

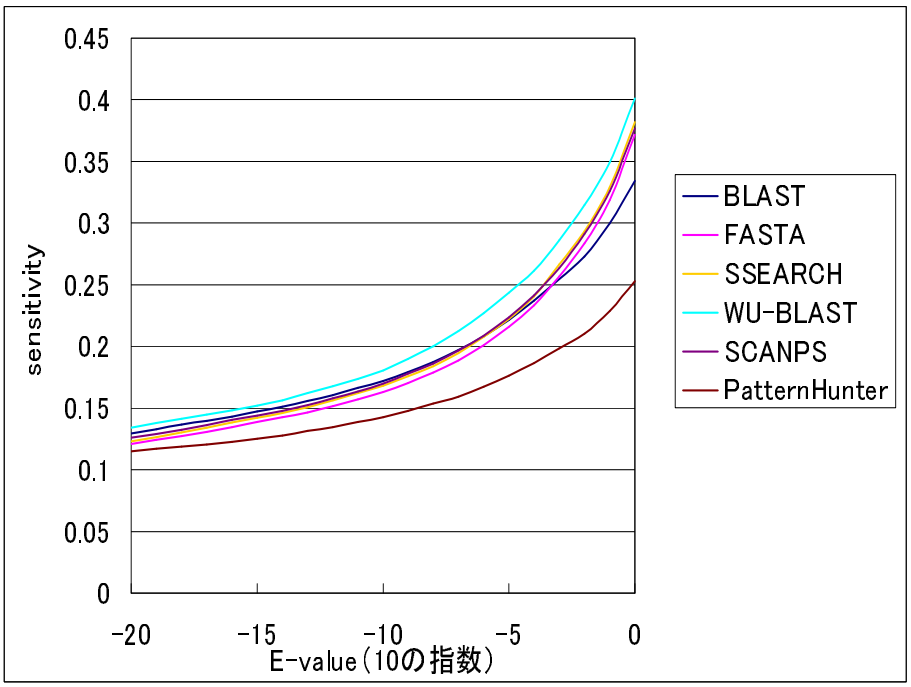

図 3.5: E-value が 1.0*E-50〜1.0 間の各手法の sensitivity の推移

各手法ともに、E-value が 1.0*E-10 より大きくなったあたりから、sensitivity が著しく上昇していることがわかる。すなわち、E-value が 1.0*E-10 より大きくなったあたりから、より多くの相同なタンパク質ペアを検出していることがわかる。ただし、E-value が大きくなるにつれて、相同でないタンパク質の検出数が増えている可能性があるということも留意しなければならない。

また、E-value が 1.0*E-50〜1.0 の範囲において、WU-BLAST の sensitivity が高く、PatternHunter の sensitivity は低いという結果となった。

3.3.3 specificity

specificity は、以下の式で算出される。

$$specificity = \frac{\text{相同性検索手法が見つけてきた相同なタンパク質ペア}}{\text{相同性検索手法が拾った全てのタンパク質ペア}}$$

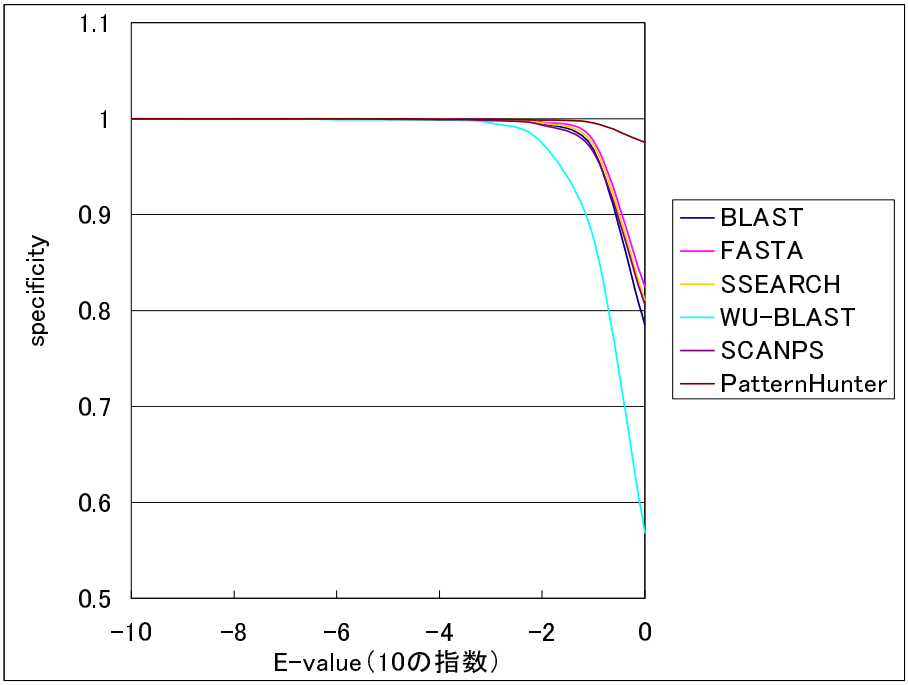

図 3.6: E-value が 1.0*E-10〜1.0 間の各手法の specificity の推移

図 3.6 より、各手法とも、E-value が 1.0*E-3 より大きくなったあたりから、specificity が下降し始めていることがわかる。すなわち、E-value が 1.0*E-3 より大きくなったあたりから、より多くの相同でないタンパク質ペアが検出されていることがわかる。

また、sensitivity が最も高かった WU-BLAST は、specificity においては、低い結果となった。そして、WU-BLAST と全くの結果となったのが PatternHunter である。また、sensitivity が最も低かった PatternHunter では、specificity においては、高い結果となった。PatternHunter では、E-value 閾値が 0.1 の場合、99.5 %を、そして、E-value 閾値が 1.0 の場合でも、97.5 %の specificity を記録した。

また、表 3.5 は E-value 閾値（1.0*E-15〜1.0）毎の false positive の検出数を記録した表である。

表 3.5: E-value 閾値毎の false positive の検出数

E-value(10の指数)	BLAST	FASTA	SSEARCH	WU-BLAST	SCANPS	PatternHunter
-15	0	0	0	0	0	0
-14	0	0	0	0	0	0
-13	0	0	0	0	0	0
-12	0	0	0	2	0	0
-11	0	0	0	2	0	0
-10	1	0	0	4	0	0
-9	1	0	0	6	0	0
-8	3	1	1	6	1	0
-7	4	2	2	8	2	0
-6	7	4	7	12	4	4
-5	9	7	9	16	10	4
-4	11	9	12	24	13	6
-3	22	23	29	81	32	9
-2	81	63	76	485	116	19
-1	596	448	550	2947	698	65
0	5390	4640	5252	17930	5334	378

表 3.5 からは、前述したように、E-value が 1.0*E-3 より大きくなったあたりから、相同でないタンパク質ペアの検出数が増えてきていることがわかる。

3.3.4 まとめ

図 3.7 は、E-value 閾値が 1.0*E-50〜1.0 において、各相同性検索手法の sensitivity、specificity の推移を表した図である。

図 3.7 より、WU-BLAST の sensitivity は最も高いが、specificity は最も低いことがわかる。また、PatternHunter に関しては、sensitivity は最も低いが、specificity は最も高いことがわかる。また、図 3.7 より、SSEARCH が図の右上にもっとも近く、sensitivity、specificity ともにバランスのとれた結果となった。

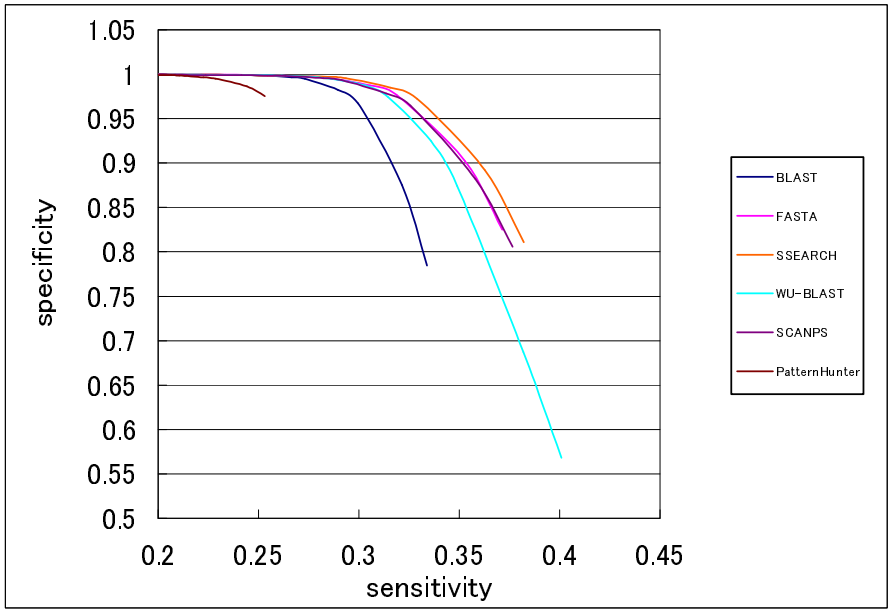

図 3.7: E-value が 1.0*E-50〜1.0 間の各手法の sensitivity、specificity の推移

3.4 union、intersection 操作を行った結果

3.4.1 true positive および false positive の推移

表 3.6 は Error level（false positives の検出数）が 10 において、true positive の検出数が、組み合わせの元となる 2 つの手法に比べて増加していた組み合わせである。Error level が 10 においては、30 通りの組み合わせのうち、12 通りの組み合わせで、true positive の検出数が、組み合わせの元となった 2 つの手法に比べて増加した。

表 3.6: Error level10 において、true positive の検出数が、組み合わせの元となる 2 つの手法に比べて増加していた組み合わせ

Method A	Method B	Set Operation	True positives	Coverage %increase over parent	E-value cut-off
BLAST	FASTA	intersection	13779	0.26 / 0.31	4.3*E-4.0
BLAST	FASTA	union	13908	1.2 / 1.3	3.9*E-5.0
BLAST	SSEARCH	union	13918	1.3 / 3.1	2.2*E-5.0
BLAST	WU-BLAST	union	14339	4.3 / 8.8	8.6*E-6.0
BLAST	SCANPS	intersection	13747	0.029 / 1.7	2.2*E-5.0
BLAST	SCANPS	union	13950	1.5 / 3.2	2.4*E-5.0
FASTA	WU-BLAST	intersection	13931	1.4 / 5.7	1.8*E-4.0
FASTA	SCANPS	intersection	13867	0.95 / 2.6	1.8*E-4.0
FASTA	PatternHunter	union	13939	1.5 / 17.5	1.3*E-4.0
SSEARCH	SCANPS	union	13766	2.0 / 1.8	2.2*E-5.0
SSEARCH	PatternHunter	union	13722	1.6 / 15.7	2.7*E-5.0
SCANPS	PatternHunter	union	13585	0.50 / 14.6	2.4*E-5.0

また、表 3.7 は Error level（false positives の検出数）が 20 において、true positive の検出数が、組み合わせの元となった 2 つの手法に比べて増加していた組み合わせである。Error level が 20 においては、30 通りの組み合わせのうち、17 通りの組み合わせで、true positive の検出数が、組み合わせの元となった 2 つの手法の手法に比べて増加した。

表 3.7: Error level20 において、true positive の検出数が、組み合わせの元となる 2 つの手法に比べて増加していた組み合わせ

Method A	Method B	Set Operation	True positives	Specificity %increase over	E-value cut-off
BLAST	FASTA	intersection	14963	1.5 / 8.9	4.9*E-3.0
BLAST	FASTA	union	15494	5.1 / 12.8	7.5*E-4.0
BLAST	SSEARCH	union	15498	5.1 / 1.8	5.0*E-4.0
BLAST	WU-BLAST	union	15359	4.2 / 0.93	7.3*E-5.0
BLAST	SCANPS	union	15165	2.8 / 2.3	2.6*E-4.0
BLAST	PatternHunter	union	14880	0.91 / 19.5	9.0*E-4.0
FASTA	SSEARCH	union	15455	12.5 / 1.5	7.1*E-4.0
FASTA	WU-BLAST	union	15286	11.3 / 0.45	7.3*E-5.0
FASTA	SCANPS	intersection	14897	8.5 / 0.52	1.0*E-3.0
FASTA	SCANPS	union	14971	9.0 / 1.0	3.1*E-4.0
FASTA	PatternHunter	union	14963	20.2 / 8.9	7.7*E-4.0
SSEARCH	WU-BLAST	union	15419	1.2 / 1.3	7.3*E-5.0
SSEARCH	PatternHunter	union	15428	1.3 / 23.9	6.8*E-4.0
WU-BLAST	SCANPS	intersection	15282	0.42 / 3.1	9.0*E-4.0
WU-BLAST	SCANPS	union	15367	1.0 / 3.7	8.9*E-5.0
WU-BLAST	PatternHunter	union	15222	0.026 / 22.3	7.3*E-5.0
SCANPS	PatternHunter	union	14892	0.49 / 19.6	3.1*E-4.0

表 3.6 や表 3.7 から、組み合わせの元となった手法と同じ Error level において、true positive の検出数を増やすためには、intersection 操作よりも union 操作の方が有効であることがわかる。

また、図 3.8 は BLAST と FASTA について、union、intersection 操作を行った結果である。

図 3.8: BLAST と FASTA の組み合わせ

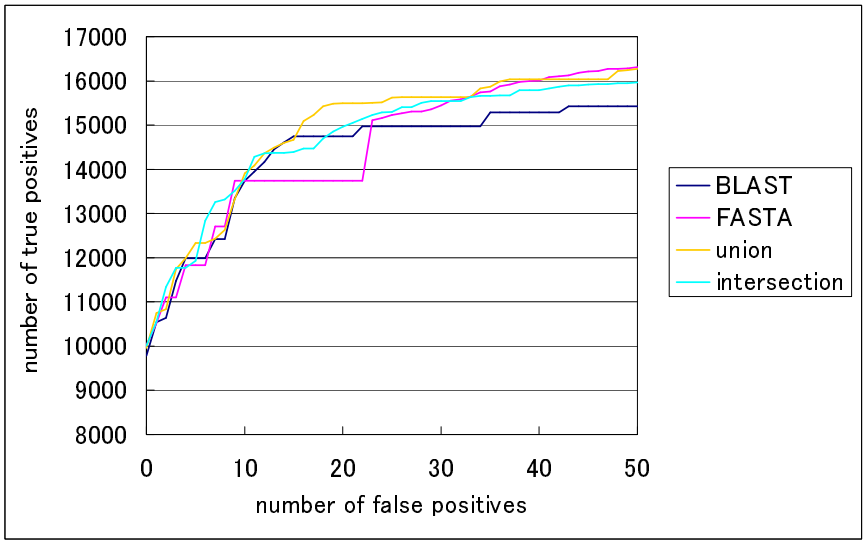

図 3.8 のように、Error level（false positives の検出数）が 20 までの範囲では、true positive の検出数が、組み合わせの元となる手法に比べて増えていたという組み合わせが多く見受けられた。しかし、Error level（false positives の検出数）が 20 を超える範囲では、true positive の検出数の増加幅は、減っていくという現象が、全 30 通りの組み合わせの結果に共通して起こった。

3.4.2 union

union 操作を行った結果、sensitivity が向上し、speicficity が低下する結果が得られた。、E-value 閾値 1.0*E-3 において、15 通りの union 操作のうち、表 3.8 に示す 9 通りが組み合わせの元となる 2 つの手法と比べて、sensitivity が 1.0 %以上向上した。

表 3.8 では、sensitivity が上昇しているものの、若干、specificity が下降していることがわかる。しかし、E-value 閾値が 1.0*E-3 の場合、６つの各相同性検索手法ともに 99.8 %以上の specificity を記録している。この E-value 閾値が 1.0*E-4 においては、各手法単独での specificity は十分に

表 3.8: E-value 閾値が 1.0*E-4 における union 操作の sensitivity、specificity の増減率

Method A	Method B	Set Operation	Sensitivity %increase over parent methods	Specificity %increase over parent methods
BLAST	FASTA	union	3.9 / 5.5	-0.011 / -0.024
BLAST	SSEARCH	union	5.5 / 4.0	-0.016 / -0.010
BLAST	WU-BLAST	union	11.6 / 1.1	-0.075 / 0.0017
BLAST	SCANPS	union	5.7 / 3.7	-0.023 / -0.010
FASTA	SSEARCH	union	4.4 / 1.4	-0.018 / 0.0012
FASTA	SCANPS	union	4.5 / 1.0	-0.025 / 0.00089
SSEARCH	WU-BLAST	union	10.2 / 1.2	-0.075 / -0.0045
SSEARCH	SCANPS	union	2.7 / 2.2	-0.012 / -0.0049
SSEARCH	PatternHunter	union	1.6 / 30.9	0.0013 / -0.029

高く、若干下げたとしても問題ないと考えられる。したがって、表 3.8 から、E-value が 1.0*E-3 よりも小さい範囲、つまり、相同でないタンパク質ペアの検出数が少ない範囲では、union 操作が有効であることがわかる。

また、E-value 閾値 0.001 においては、組み合わせの元となる 2 つの手法と比べて sensitivity が 1.0 %以上向上したものは 15 通りの union 操作のうち、表 3.9 に示す 7 通りであった。

表 3.9: E-value 閾値が 0.001 における union 操作の sensitivity、specificity の増減率

Method A	Method B	Set Operation	Sensitivity %increase over parent methods	Specificity %increase over parent methods
BLAST	FASTA	union	5.5 / 4.6	-0.068 / -0.063
BLAST	SSEARCH	union	7.0 / 2.1	-0.084 / -0.046
BLAST	WU-BLAST	union	13.6 / 1.1	-0.370 / -0.036
BLAST	SCANPS	union	7.0 / 3.4	-0.096 / -0.037
FASTA	SCANPS	union	4.0 / 1.4	-0.070 / -0.016
SSEARCH	WU-BLAST	union	8.9 / 1.5	-0.33 / -0.040
SSEARCH	SCANPS	union	1.5 / 2.8	0.060 / -0.038

また、表 3.8 と表 3.9 の specificity を比較してみることにより、E-value が大きくなるにつれて、union 操作による specificity の減少幅が増えてきていることがわかる。これは、3.2.3 節で述べた、E-value 閾値が 1.0*E-3 より大きくなると相同でないタンパク質の検出数が増えるといったことを、反映した結果といえる。

3.4.3 intersection

intersection 操作を行った結果、specificity が上昇し、sensitivity が下降するといった union 操作とは全く逆の結果が得られた。E-value 閾値 1.0*E-3 において、15 通りの intersection 操作のうち、表 3.10 に示す 5 通りが組み合わせの元となる 2 つの手法と比べて、specificity が 1.0 %以上向上した。

表 3.10: E-value 閾値が 0.1 における intersction 操作の sensitivity、specificity の増減率

Method A	Method B	Set Operation	Sensitivity %increase over parent methods	Specificity %increase over parent methods
BLAST	FASTA	intersection	-4.7 / -10.4	2.4 / 1.5
BLAST	SSEARCH	intersection	-2.9 / -11.6	2.4 / 1.9
BLAST	WU-BLAST	intersection	-2.1 / -15.9	1.6 / 12.4
BLAST	SCANPS	intersection	-3.7 / -11.4	2.5 / 2.7
WU-BLAST	SCANPS	intersection	-8.4 / -1.9	11.5 / 1.0

また、E-value が大きくなるにつれて、intersection 操作による specificity の上昇幅が大きくなるという結果になった。6 つの各相同性検索手法ともに、specificity が低下しはじめる E-value が 1.0*E-3 より大きくなったあたりから、intersection 操作による specificity の上昇幅が増え始める結果となった。したがって、E-value が大きく、各手法ともに単独で用いると、相同でないタンパク質ペアを多数検出してしまうような場合、intersection 操作によって specificity を上昇させることができ、有用であると考えられる。

3.5 提案手法

単独で用いた結果や組み合わせを行った結果を踏まえ、sensitivity、specificity ともに向上させる方法を考える。

3.5.1 提案手法についての概要

単独で用いた結果や組み合わせを行った結果についてのまとめ

単独で用いた結果や組み合わせを行った結果について、まとめると以下のようになる。

手法単独で用いた結果

- E-value 閾値が 1.0*E-10 よりも大きくなると、各手法ともに sensitivity が著しく上昇する（相同なタンパク質の検出数が増える）
- E-value 閾値が 1.0*E-3 よりも小さい場合、各手法ともに specificity は非常に高い（相同でないタンパク質ペアの検出数が少ない）
- E-value 閾値が 1.0*E-3 より大きくなると、各手法ともに specificity が下降し始める（相同でないタンパク質の検出数が増える）

union 操作

- 各手法ともに specificity が高く、99.8 %以上（E-value 閾値が 1.0*E-3 以下）の場合、union 操作によって specificity をほとんど下げることなく、sensitivity を上昇させることが可能
- 各手法ともに specificity が低い（E-value 閾値が 1.0*E-3 よりも大きい）場合、union 操作によって、specificity をさらに低下させてしまう。

intersection 操作

- sensitivity は確実に低下させてしまうものの、各手法ともに specificity が低い（E-value 閾値が 1.0*E-4 よりも大きい）場合、speicificity を向上させることが可能

以上に述べた、単独で用いた結果や組み合わせを行った結果の特徴を踏まえ、タンパク質ペアの E-value によって、union、intersection 操作を使い分け、sensitivity、specificity ともに向上させることを目指す。

3.5.2 提案手法（union、intersection 操作の使い分け）

union、intersection を以下のように使い分けることにより、手法単独で用いると specificity が低下する（E-value 閾値が 1.0*E-3 よりも大きい）範囲で、sensitivity、specificity ともに向上させることを目指す。

- E-value が 1.0*E-3 以下のペア同士を組み合わせる場合、union 操作を行う
  相同でないタンパク質ペアの検出数をほとんど増やさずに、相同なタンパク質ペアの検出数の増加が望める
- E-value が 1.0*E-3 よりも大きいペア同士を組み合わせる場合、intersection 操作を行う
  相同でないタンパク質ペアの検出数の減少が望める

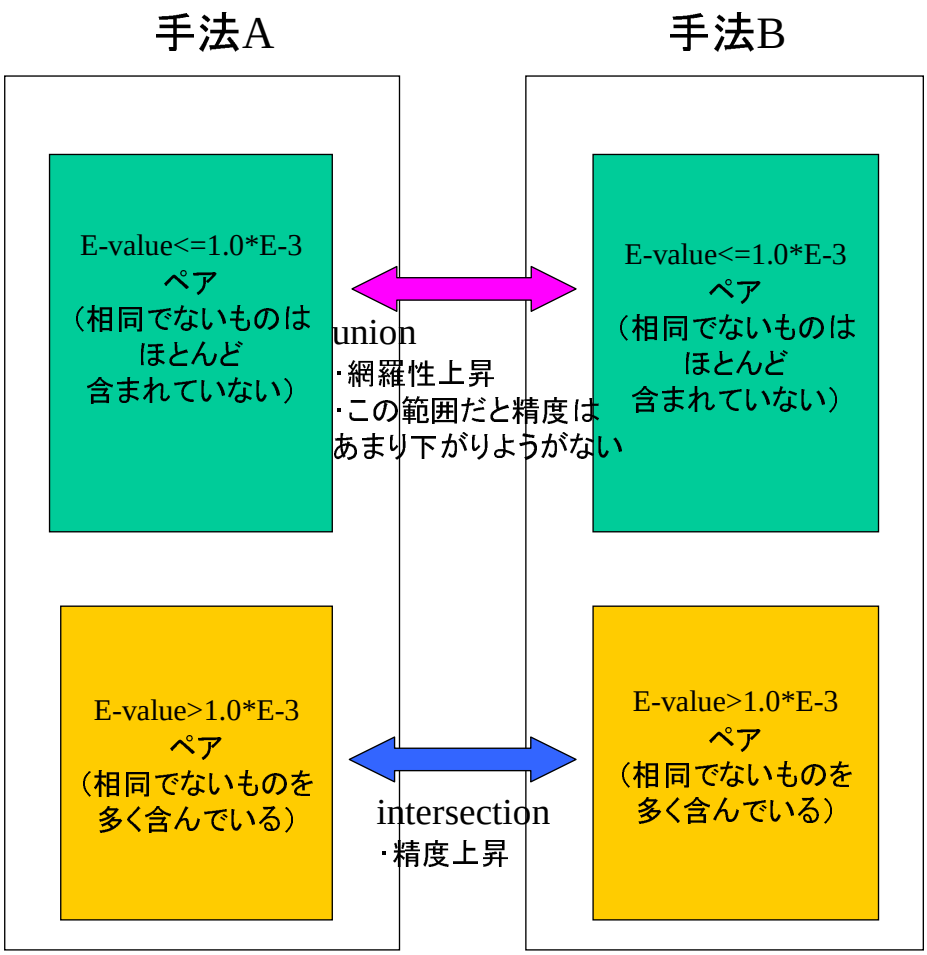
手法A
手法B
E-value<=1.0*E-3
ペア
（相同でないものは
ほとんど
含まれていない）
E-value<=1.0*E-3
ペア
（相同でないものは
ほとんど
含まれていない）
union
・網羅性上昇
・この範囲だと精度は
あまり下がりようがない
E-value>1.0*E-3
ペア
（相同でないものを
多く含んでいる）
E-value>1.0*E-3
ペア
（相同でないものを
多く含んでいる）
intersection
・精度上昇

図 3.9: 提案手法

3.5.3 結果

E-value 閾値が 3.0*E-3 の場合、新たに提案した組み合わせを行った全 15 通りのうち、BLAST と FASTA の組み合わせで sensitivity、specificity ともに組み合わせの元となる 2 つの手法に比べて向上させることができた。また、全 15 通りのうち、表 3.11 に示す BLAST と SSEARCH、BLAST と SCANPS、FASTA と SSEARCH の 3 通りで sensitivity、specificity ともに組み合わせの元となる 2 つのの手法のうちのひとつに比べて向上させることができた。

表 3.11: E-value 閾値が 3.0*E-3 における提案手法の結果

Method A	Method B	Sensitivity %increase over parent methods	Specificity %increase over parent methods
BLAST	FASTA	3.0 / 0.47	0.052 / 0.0011
BLAST	SSEARCH	4.4 / -1.6	0.043 / 0.014
BLAST	WU-BLAST	10.5 / -2.7	-0.27 / 0.39
BLAST	SCANPS	4.4 / -0.86	0.018 / 0.096
BLAST	PatternHunter	-1.9 / 26.2	0.13 / -0.052
FASTA	SSEARCH	1.7 / -1.8	0.0038 / 0.026
FASTA	WU-BLAST	7.7 / -2.8	-0.31 / 0.39
FASTA	SCANPS	1.1 / -1.6	-0.023 / 0.11
FASTA	PatternHunter	-3.5 / 27.3	0.070 / -0.057
SSEARCH	WU-BLAST	4.8 / -2.1	-0.30 / 0.38
SSEARCH	SCANPS	-1.0 / -0.14	-0.015 / 0.092
SSEARCH	PatternHunter	-3.7 / 31.5	0.066 / -0.084
WU-BLAST	SCANPS	-2.4 / 5.3	0.36 / -0.21
WU-BLAST	PatternHunter	-3.9 / 40.6	0.45 / -0.38
SCANPS	PatternHunter	-3.8 / 30.4	0.15 / -0.10

また、図 3.10 は、E-value 閾値 3.0*E-3 において、BLAST と FASTA に union、intersection 操作を行った結果と提案手法を適応した結果を比較した図である。

図 3.10 より、E-value 閾値 3.0*E-3 において、union 操作では BLAST と FASTA を単独で用いる場合に比べて、sensitivity を向上させているものの、specificity を低下させていることがわかる。また、intersection 操作では、specificity は向上させているものの、sensitivity は低下させていることがわかる。一方、提案手法では、sensitivity、specificity の両方を、BLAST と FASTA 単独で用いた場合に比べて向上させていることが

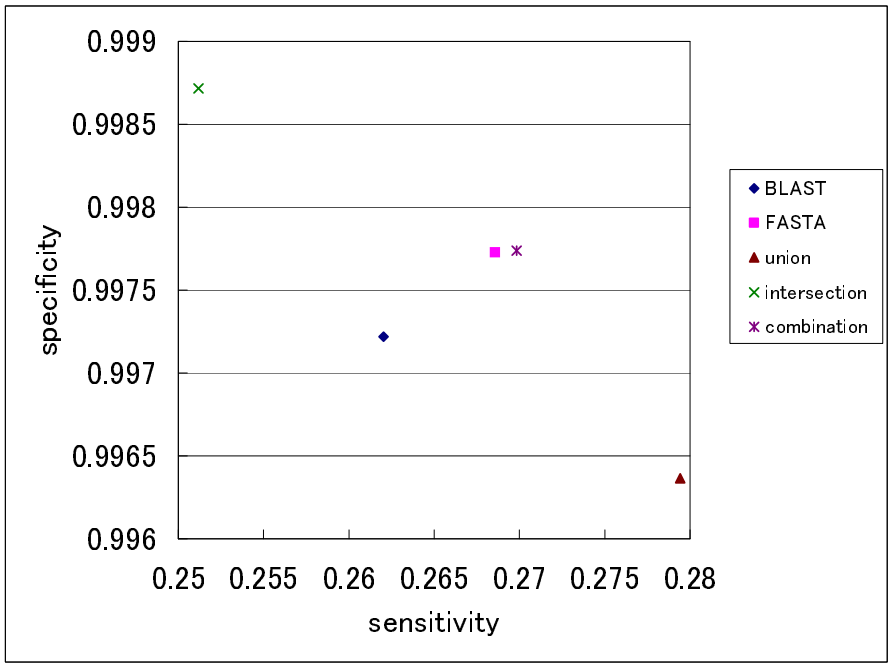

図 3.10: E-value 閾値 3.0*E-3 における BLAST と FASTA の組み合わせ結果

わかる。

また、図 3.11 は、BLAST と FASTA の組み合わせ（E-value 閾値 1.0*E-10〜1.0 間）における sensitivity、specificity の推移を表した図である。なお、提案手法に関しては、union、intersection 操作の使い分けに用いる E-value 閾値は 1.0*E-3 に設定した結果である。

図 3.10 や図 3.11 より、E-value 閾値 3.0*E-3 のように、union、intersection 操作の使い分けに用いる E-value 閾値（1.0*E-3）に近い範囲では、sensitivity、specificity 共に、組み合わせの元となる 2 つの手法に比べて向上させることができたことがわかる。しかし、E-value 閾値が大きくなると、specificity は向上させることができたものの、sensitivity は低下させてしまったことがわかる。また、提案手法は、union 操作を行う場合と比べて、sensitivity は低下させてしまったものの、specificity は向上させることができた。さらに、提案手法は intersection 操作を行う場合と比べて、E-value 閾値が 1.0*E-1 以下の範囲では specificity は低下させてしまったものの、sensitivity は向上させることができた。E-value 閾値が 1.0*E-1 より大きい範囲では sensitivity、specificity 共に、intersection 操作を行う場合に比べて向上させることができた。

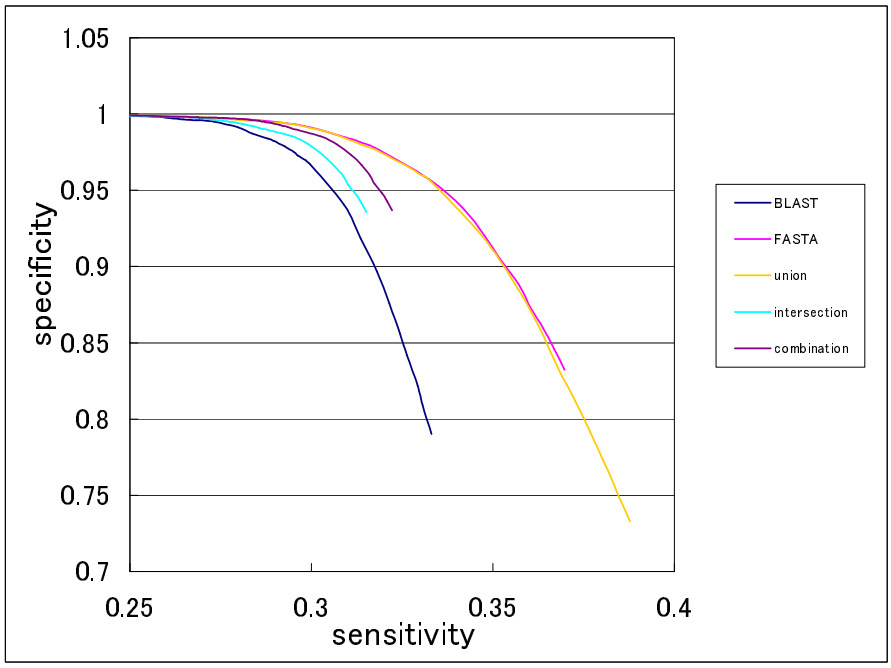

図 3.11: E-value 閾値 1.0*E-10〜1.0 における BLAST と FASTA の組み合わせ結果

3.5.4 考察

提案手法を行った結果として、BLAST と FASTA の組み合わせで、E-value 閾値 3.0*E-3 において、BLAST と比較して sensitivity を 3.0 %、specificity を 0.0052 %、FASTA と比較して sensitivity を 0.47 %、specificity を 0.0011 %向上させることができた。提案手法の結果についての考察として、BLAST と FASTA の組み合わせで、sensitivity、specificity ともに向上できたものの、specificity の増加幅は小さいという結果になったことについての検証を行う。

図 3.12 や図 3.13 は、BLAST 単独の結果に対する組み合わせを行った結果の比率を表す。

図 3.12 や図 3.13 より、union 操作を行うことによって、sensitivity が向上し、specificity が低下することがわかる。また、intersection 操作によって、specificity が向上するが、sensitivity が低下することがわかる。

また、提案手法では、E-value 閾値が 1.0*E-3 以下では、union 操作を行うため、sensitivity が向上し、specificity が低下する。また、E-value 閾値が 1.0*E-3 より小さい場合では、intersection を行うことによって、E-value 閾値が 1.0*E-3 以下で union 操作を行うため向上していた sensitivity が

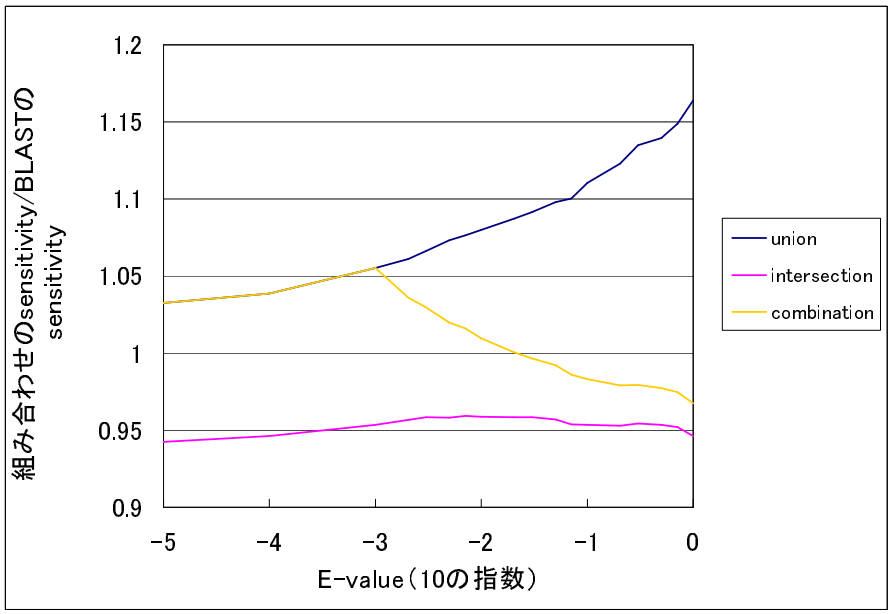

図 3.12: BLAST 単独の結果の sensitivity に対する組み合わせを行った結果の sensitivity の比率

低下し、specificity が向上する。図 3.14 は、BLAST 単独の結果に対する提案手法を行った結果の比率を表す。

図 3.14 より、提案手法によって E-value 閾値 3.0*E-3($1.0*10^{-2.69}$)～3.0*E-2($1.0*10^{-1.69}$) の間で、BLAST の sensitivity、specificity の両方を向したことが確認できる。また、E-value 閾値 3.0*E-3($1.0*10^{-2.69}$)～3.0*E-2($1.0*10^{-1.69}$) の間では、BLAST 単独で用いた結果の sensitivity が 27.9 %以上だったのに対し、specificity は 99.1 %以上だった。したがって、E-value 閾値 3.0*E-3($1.0*10^{-2.69}$)～3.0*E-2($1.0*10^{-1.69}$) の間では、specificity は sensitivitiy と比較し、上昇する余地が少なかったため、sensitivity に比べ増加幅も小さくなったといえる。

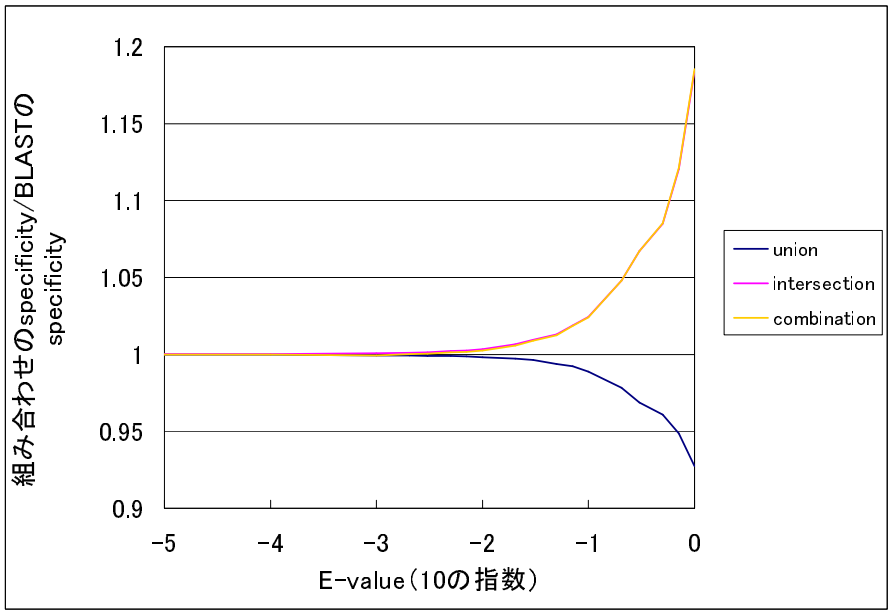

図 3.13: BLAST 単独の結果の specificity に対する組み合わせを行った結果の specificity の比率

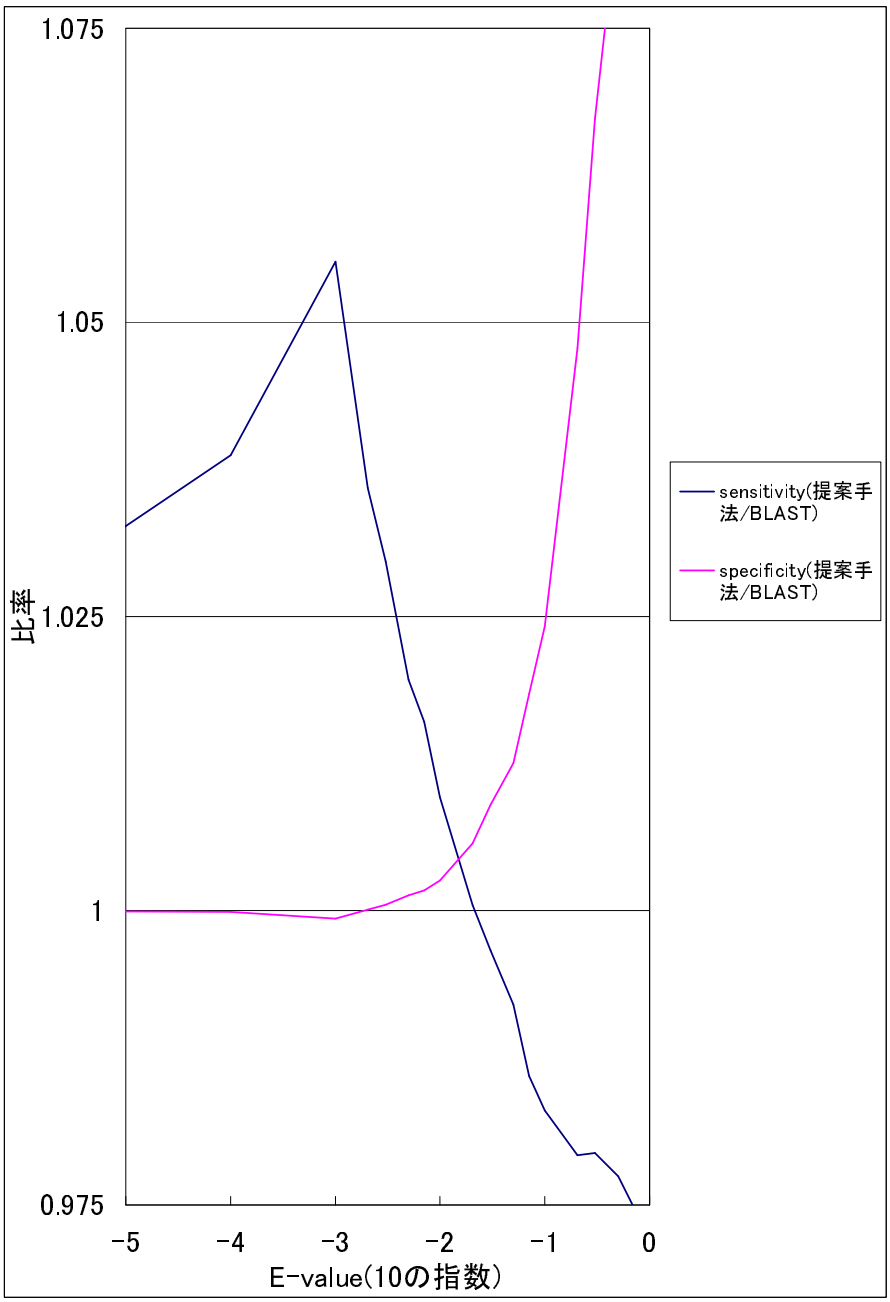

図 3.14: BLAST 単独の結果に対する提案手法を行った結果の比率

第4章　おわりに

本研究では、相同性検索手法を、タンパク質ペアの E-value によって union、intersection 操作を効果的に使い分けて、組み合わせることにより、sensitivity、specificity 両方の向上を目指した。その結果、BLAST と FASTA の組み合わせでは、E-value 閾値 3.0*E-3 のように、union、intersection 操作の使い分けに用いる E-value 閾値（1.0*E-3）に近い範囲では、sensitivity、specificity 共に、BLAST と FASTA 単独で用いる場合に比べて向上させることができた。しかし、E-value 閾値が大きくなると、specificity は向上したものの、sensitivity は低下するといった結果となった。

BLAST と FASTA のように、手法単独で用いる場合と比較して sensitivity や specificity が向上した組み合わせがあった背景には、タンパク質のアミノ酸配列のアラインメントを行うアルゴリズムや E-value の計算方法が各手法で異なるということが考えられる。

また、組み合わせを行ったが、sensitivity を E-value 閾値全域にわたって、向上するには至らなかったことから、各手法が拾ってくる相同なタンパク質ペアは似通っていると考えられ、ペアワイズアラインメントで相同性検索を行う点に限界を感じる。

今後は、手法の組み合わせによって通常の Gapped-BLAST より多くの配列を拾うことで、PSI-BLAST のプロファイルの質や PSI-BLAST の精度にどのような影響を及ぼすのか検証を行う予定である。

謝辞

本研究を行うにあたり、適切な助言やご指導を頂いた山名早人助教授に深く感謝致します。また、日頃から常に熱心かつ的確に指導をして下さった山田真介先輩に心から感謝致します。そして、PC の設定や相同性検索手法のインストールなどを手伝って下さった山田晃太郎先輩、岩橋永悟先輩、斉藤純先輩をはじめ、研究室の先輩方、同輩にも御礼申し上げます。

参考文献

[1] GenBank
http://www.ncbi.nlm.nih.gov/

[2] SWISS-PROT
http://www.expasy.ch/sprot/

[3] Benson.DA, Karsch.MI, Lipman.DJ, Ostell.J, Wheeler.DL(2004) GenBank: update, *Nucleic Acid Research*, vol.32, D23-D26

[4] C.Webber, G.J.Barton(2003)Increased coverage obtained by combination for protein sequence database searching, *Bioinfomatics*, vol.19, 1397-1403

[5] Henikoff,S. and Henikoff,J.G.(1992)Amino acid substitution matrices from protein blocks, *Proc. Natl. Acad. Sci. USA*, vol.89, 10915-10919

[6] S.B. Needlman and C.D. Wunsch(1970)A general method applicable to the search for similarities in the amino acid sequence of two proteins, *J.Mol.Biol.*, vol.48, 443-453

[7] T.F. Smith and M.S. Waterman(1981)Identification of common molecular subsequences, *J.Mol.Biol.*, vol.147, 195-197

[8] Altschul.SF, Gish.W, Miller.W, Myers.EW, Lipman.DJ (1990) Basic Local Alignment Search Tool, *J. Mol. Biol*, vol.215,403-410

[9] Pearson.WR (1990) Rapid and Sensitive Sequence Comparison with FASTP and FASTA, *Methods in Enzymology*, vol.183, 63-98

[10] Altschul.SF, Gishu.W (1996) Local alignment statistics, *Methods in Enzymology*, vol.266, 460-480

[11] WU-BLAST2.0 TOPICS
http://blast.wustl.edu/blast/README.html

[12] Barton.GJ (1992) Computer Speed and Sequence Comparison, *Science*, vol.257, 1609

[13] Bin.M, John.T, Ming.L (2004) PatternHunter : Highly Sensitive and Fast Homology Search, *Journal of Bioinfomatics and Computational Biology*

[14] The statistics of Sequence Similarity Scores
http://www.ncbi.nih.gov/BLAST/tutorial/Altschul-1.html/