分散システムでの時間的振る舞いの予測に関する研究

課題番号 15500053

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 4 月

研究代表者 中里秀則
（早稲田大学大学院国際情報通信研究科）

はしがき

本研究では、分散処理環境におけるアプリケーションプログラム実行の実時間性に関する検討を行った。近年、インターネットを使った、動画配信、ビデオ会議、インターネット電話、インターネットゲームといった実時間性を必要とするアプリケーションが増加している。また、電話網などのネットワークシステムはもちろんのこと、企業の基幹システムも分散処理によって実現されるようになっており、これらシステムにおいても素早い応答性が求められる。本研究では、厳密な実時間性、いわゆるハードリアルタイム性までは必要でないが、ある程度の実時間性、いわゆるソフトリアルタイム性を要求する分散処理システムにおいて、システムの時間的な振る舞いを予測可能にする分散処理システムを構築する上でのネットワーク構成方法、パケット転送方法、プログラムスケジューリング方法といったシステム構成方法を特定し、そのシステムにおけるアプリケーションソフトウェアの時間的振る舞いの予測方法を求めた。本研究の成果は、これらアプリケーションを実現する上で、その時間的振る舞いを、各アプリケーションに必要とされる時間的な制約条件に適合させるために活用することができる。

研究組織

研究代表者：中里秀則（早稲田大学大学院国際情報通信研究科）

（研究協力者：花田真樹、鎌村星平）

交付決定額（配分額）　（金額単位：千円）

	直接経費	間接経費	合計
平成 15 年度	1,000	0	1,000
平成 16 年度	1,300	0	1,300
平成 17 年度	800	0	800
総計	3,100	0	3,100

研究発表

[1] 花田真樹，中里秀則 , 非割込み型 EDF スケジューリングの近似解析 , 電子情報通信学会論文誌 (A) Vol. J87-A No.12, pp.1518-1527, 2004 年 12 月

[2] 花田真樹、中里秀則、非割込み型 EDF スケジューリングにおけるパフォーマンス予測：理論と実装、情報科学技術レターズ、vol. 4, pp. 337-340,（第 4 回情報科学技術フォーラム、LO-001）, 2005 年 9 月

[3] 花田真樹、中里秀則、ORC-GPS 方式における End-to-End 最大遅延の評価 , 電子情報通信学会技術研究報告 , vol. 106, no. , NS2006-, 2006 年 5 月

[4] 鎌村星平 , 中里秀則 , 富永英義 , 伝送時間測定に基づいたアドミッション制御方式の提案 , 2006 年電子情報通信学会総合大会講演論文集 , No. B-7-6, p. 102, 2006 年 3 月

[5] 花田真樹、中里秀則、スケジューラブル領域の最大化を目的とするスケジューリング方式 ,

電子情報通信学会技術研究報告 , vol. 105, no. 407, NS2005-122, pp. 59-62, 2005 年 11 月
[6] 鎌村星平 , 中里秀則 , 富永英義 , マルチパス環境における EDF スケジューリングの適用に関する検討 , 電子情報通信学会技術研究報告 , vol. 105, no. 178, IN2005-52, pp. 137-142, 2005 年 7 月
[7] 花田真樹 , 中里秀則 , 非割込み型 EDF スケジューリングの近似解析 , 電子情報通信学会技術研究報告，CPSY2003-12，pp.37-42，2003 年 8 月

目次

1. 背景と目的

1.1. 研究の背景

近年、インターネットを使った、動画配信、ビデオ会議、インターネット電話、インターネットゲームといったマルチメディアを使ったアプリケーションが普及し始めている。これらマルチメディアアプリケーションは、ユーザの使用に耐える応答性を確保するため、実時間性を考慮したプログラミングや通信、処理の実行を必要としている。これまでネットワークシステムやファクトリーオートメーションシステムといった制御系システムでは実時間性が重視されてきたが、一般ユーザ向けのアプリケーションや企業情報システムなど情報系システムでもある程度のリアルタイム性が必要となってきている。

1.2. 研究の目的

一般的に情報系システムはソフトリアルタイムに分類され，デッドラインミスが起きても，システムは稼動し続ける．情報系システムは、制御系システムと異なり、非決定的な環境下で稼動し，絶対的な保証より統計的（確率的）な保証を行うことが重要となる．

そこで本研究では、厳密な実時間性、いわゆるハードリアルタイム性までは必要でないが、ある程度の実時間性、いわゆるソフトリアルタイム性を要求する分散処理システムにおいて、システムの時間的な振る舞いを予測可能にする分散処理システムを構築する上でのネットワーク構成方法、パケット転送方法、プログラムスケジューリング方法といったシステム構成方法を特定し、そのシステムにおけるアプリケーションソフトウェアの時間的振る舞いの予測方法を求めた。

また、分散実時間処理システムでは、システムの構成要素として、コンピュータ、すなわち情報処理部分における時間的振舞いと、ネットワーク、すなわち通信処理部分における時間的振舞いの二つの要素があり、これらそれぞれについて検討を加えた。

2. 研究成果

2.1. 非割込み型 EDF スケジューリングの近似解析

非割込み型EDFスケジューリングの近似解析

花田　真樹[†]　　中里　秀則[†]

An approximation analysis of nonpreemptive EDF scheduling

Masaki HANADA[†] and Hidenori NAKAZATO[†]

あらまし　現在，マルチメディアの発達により，制御系システムに加えて情報系システムでもある程度のリアルタイム性が必要となってきている．一般的に情報系システムはソフトリアルタイムに分類され，デッドラインミスが起きても，システムは稼動し続ける．ソフトリアルタイムシステムでは，絶対的な保証 (デッドライン保証) より統計的な保証 (確率的な解析) を行うことが重要となる．現在，リアルタイムシステムにおいて，時間的制約を満たすために，Earliest Deadline First(EDF) などのリアルタイムスケジューリングが用いられている．EDF スケジューリングはデッドラインの早い順に優先順位を付ける非常にシンプルな手法であり，事前に起動時間がわからない非周期タスクに適用可能である．本論文では，タスクの各属性が確率分布に従うものと仮定し，非割込み型 EDF スケジューリングを適用した場合のシステム性能を数学的に解析する．

キーワード　EDF スケジューリング, 非周期タスク, 非割込み型，ソフトリアルタイム

1. まえがき

現在，マルチメディアの発達により，制御系システムに加えて情報系システムでもある程度のリアルタイム性が必要となってきている．一般的に情報系システムはソフトリアルタイムに分類され，デッドラインミスが起きても，システムは稼動し続ける．情報系システムは非決定的な環境下で稼動し，絶対的な保証 (デッドライン保証) より統計的な保証 (確率的な解析) を行うことが重要となる．現在，リアルタイムシステムにおいて，時間的制約を満たすために Earliest Deadline First (EDF) や Least Laxity First (LLF) などのリアルタイムスケジューリングが用いられている．

システムの性能評価では，システムの近似モデルを構築し，その振る舞いを解析する手法がある．待ち行列理論 [1] [2] [3] [4] では，First-Come First-Served (FCFS)，Shortest Job First (SJF) [6], Shortest Remaining Processing Time (SRPT) [2], Round Robin (RR) [7] [8] を適用した場合のシステム性能を数学的に解析している．

リアルタイムシステムにおける性能評価では，起動時間等を含めた十分なタスクの情報を必要とするハードリアルタイムが中心であったため，非周期タスクに適用した場合の解析があまり進んでいない．しかしながら，ある程度のリアルタイム性を必要とする情報系システムが増加している状況を考慮すると，リアルタイムスケジューリングを適用した場合のシステム性能を数学的に解析しておくことは重要である．

本論文では，最も単純な単一サーバモデルに対し，タスクの各属性が確率分布に従うものと仮定し，非割込み型 EDF スケジューリングを適用した場合の解析を行う．EDF スケジューリングはデッドラインの早い順に優先順位を付ける非常にシンプルな手法である．さらに，現在処理中のタスクよりも，デッドラインの早いタスクが到着した場合の処理方法として，割込み型と非割込み型が考えられる．割込み型は処理中のタスクを一時的に中断し，デッドラインの早いタスクを先に処理してから，中断した状態から再開する方法である．これに対して，非割込み型は処理中のタスクはそのまま続行する方法である．特に，割込み型の場合，非周期タスクに対して，最大遅れを最小にすることに関して最適であり，周期タスクに対しては，プロセッサ使用率が 100%までデッドラインが保証され

[†] 早稲田大学大学院国際情報通信研究科
〒 169–0051 東京都新宿区西早稲田 1–3–10 早大 29–7 号館
Graduate School of Global Information and Telecommunication Studies, Waseda University
Nishiwaseda 1–3–10, Shinjiku-ku, Tokyo, 169–0051 Japan

る [9] [10]．確率モデル (M/M/1 待ち行列モデル) に対して割込み型 EDF スケジューリングを適用した場合の研究では，平均応答時間 [5] や高負荷条件下での現在時刻からデッドラインまでの時間の分布 [11] が求められている．当然，本論文で用いる非割込み型の場合，最適ではないが，割込み型におけるコンテキスト切り替えのオーバーヘッド等を考慮すると，ソフトリアルタイムにおいては有効であると考える．

第 2 章では，システムモデルについて述べる．第 3 章では，待ち時間の条件付き分布と条件付き期待値の導出のための解析を行い，第 4 章では，第 3 章で提案した理論を用いて近似的に待ち時間の条件付き分布と条件付き期待値を求める．第 5 章では，第 4 章の結果とシミュレーションの結果を比較し，本論文の解析法を評価する．第 6 章では，本解析より明らかになった非割込み型 EDF スケジューリングの性質について述べる．第 7 章では，まとめと今後の課題について述べる．

2. システムモデル

システムモデルとして最も単純な単一サーバモデル (図 1) を対象とする．

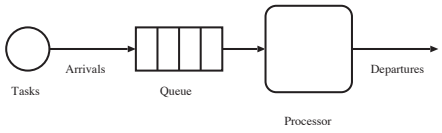

図 1　単一サーバモデル
Fig. 1　A single sever model

タスクの実行順序はデッドラインの早い順とし，キューは無限長とする．また，実行中のタスクよりも早いデッドラインをもったタスクが到着した場合でもそのまま実行を続行する非割込み型を用いる．タスクが到着した場合には，デッドライン保証の判定は行わず，即座にキューに格納される．さらに，タスクがデッドラインを守れなかった場合でもその実行を中断することはない．

タスクは実行時間と相対デッドライン時間をもつ．タスクのデッドラインは，到着時刻と相対デッドライン時間を足した値として定義される．

タスクの到着は平均到着率 λ のポアソンとし，相対デッドライン時間の確率分布関数を $F(x)$，確率密度関数を $f(x)$ とする．相対デッドライン時間 x をクラスとして考え，相対デッドライン時間が x であるタスクの到着率を $\lambda(x)$ で表す．相対デッドライン時間が $(x, x+dx)$ の間になるタスクの到着率は $\lambda(x)dx$ となるので，以下の関係式が成り立つ．

$$\lambda(x) = \lambda \frac{dF(x)}{dx} = \lambda f(x) \tag{1}$$

以下，相対デッドライン時間に対する条件付きの待ち時間に関する定義について述べる．

相対デッドライン時間，待ち時間を表す確率変数をそれぞれ X，T とする．相対デッドライン時間が x であるタスクの待ち時間の条件付き分布関数を $W(t|X=x)$，条件付き密度関数を $w(t|X=x)$ とする．

$$W(t|X=x) = P_r(T \leqq t|X=x) \tag{2}$$

$$w(t|X=x) = \frac{dW(t|X=x)}{dt} \tag{3}$$

また，相対デッドライン時間が x であるタスクの待ち時間の条件付き期待値を $W(x)$ とする．

$$W(x) = \int_0^\infty tw(t|X=x)dt \tag{4}$$

$$= \int_0^\infty (1 - W(t|X=x))dt \tag{5}$$

次に，タスクの実行時間について定義する．一般的に，相対デッドライン時間と実行時間の間には強い依存性がある．相対デッドライン時間の短いタスクは，実行時間も同様に短いと仮定する．各タスクの実行時間は，そのタスクの相対デッドライン時間の $1/n$(n 固定) とする (以下，n を実行時間比率と呼ぶ)．

3. 待ち時間の条件付き分布と期待値の導出

これまでに，優先権スケジューリングを用いた場合のクラスにおける待ち時間の条件付き期待値の導出について多くの議論が行われてきた [2] [4]．従来の枠組 (HOL 優先権) [2] では，クラス間の優先度が変化しない固定優先度を用いている．クラスの待ち時間の条件付き期待値は，再帰の方程式を解くことにより導出可能である．さらに，HOL 優先権 [2] の応用として，Shortest Job First (SJF) を適用した場合の解析 [6] も行われている．実行時間を既知とし，実行時間の短い順に優先順位を付ける手法であり，実行時間 x をもつタスクがクラスをなすと考え，HOL 優先権モデルを離散から連続に移したモデルである．しかし，EDF スケジューリングでは，優先権となる属性がデッドラインなので，到着後，時間の経過に伴い優先度が変化

することとなり，従来の枠組の適用は困難である．当然，EDF スケジューリングでは，相対デッドライン時間が優先権でないので，キュー内で相対デッドライン時間の長いタスクが相対デッドライン時間の短いタスクより先に実行される可能性が考えられる．

第 2 章にて，相対デッドライン時間に対する条件付きの定義 (到着率，待ち時間) を行った．以降では，相対デッドライン時間として一定の値 x をもつタスクについて解析を行う．特に，ここで相対デッドライン時間が x であるタスクを注目タスクと呼ぶ．また，相対デッドライン時間が t であり，注目タスクの待ちに寄与するタスクを対象タスクと呼ぶ．

相対デッドライン時間が x であるタスクの待ち時間の条件付き期待値 $W(x)$ は以下の要素の合計となる．

（1） 注目タスクの到着時に実行中の対象タスクによる待ち時間 W_o

（2） 注目タスクがキューに到着した時点で，既にキューにある対象タスクの中で注目タスクより先に実行される対象タスクによる待ち時間 $W1(x)$

（3） 注目タスクがキューにいる間に到着した対象タスクの中で，注目タスクより先に実行される対象タスクによる待ち時間 $W2(x)$

これより，相対デッドライン時間が x であるタスクの待ち時間の条件付き期待値 $W(x)$ は以下となる．

$$W(x) = W_o + W1(x) + W2(x) \qquad (6)$$

3.1 待ち時間の各要素

（1） 注目タスクの到着時に実行中の対象タスクによる待ち時間 W_o

W_o は相対デッドライン時間に依存しない．実行時間だけに注目すればよい．実行時間が t/n であるタスクの実行中に，注目タスクが到着した場合，残余時間の期待値は $t/(2n)$ である．ここで，相対デッドライン時間が t であるタスクの実行中である時間割合は $\rho(t) = \lambda(t) \times (t/n)$ である．W_o は以下となる．

$$W_o = \int_0^\infty \rho(t)(\frac{t}{2n})dt \qquad (7)$$

（2） 注目タスクがキューに到着した時点で，既にキューにある対象タスクの中で注目タスクより先に実行される対象タスクによる待ち時間 $W1(x)$

待ち時間 $W1(x)$ は以下の 2 つの要素に分類される．

- 注目タスクより短い相対デッドライン時間をもつ対象タスクによる待ち時間 $W1_s(x)$

キュー内に存在している対象タスクは，注目タスクより先に到着しているので，注目タスクより短い相対デッドライン時間をもつ対象タスクは，全て先に実行される．対象タスクの到着率は $\lambda(t)$，待ち時間は $\int_0^\infty zw(z|X = t)dz$ であるので，リトルの公式より，該当する対象タスクのタスク数は $\lambda(t)\int_0^\infty zw(z|X = t)dz$ である．また，相対デッドライン時間が t であるタスクの実行時間は t/n である．したがって，該当する対象タスクが注目タスクを待たせる時間は以下となる．

$$W1_s(x) = \\ = \int_0^x \frac{t}{n}\lambda(t)\left(\int_0^\infty zw(z|X = t)dz\right)dt \qquad (8)$$

- 注目タスクより長い相対デッドライン時間をもつ対象タスクによる待ち時間 $W1_l(x)$

キュー内に存在している対象タスクは，注目タスクより先に到着しているので，注目タスクより長い相対デッドライン時間をもつ対象タスクでも，そのいくつかは先に実行される．これを求めるために，対象タスクの待ち時間の条件付き密度関数を用いる．対象タスクがキューに存在し，注目タスクが到着する場合を考える．これを図 2 に記す．対象タスクが先に実行されるのは，対象タスクのデッドラインが注目タスクのデッドラインより早い場合である．したがって，注目タスクの到着時刻と対象タスクの到着時刻の差分が両タスクの相対デッドライン時間の差分 $(t-x)$ より大きい場合である．対象タスクの待ち時間が z であるとすると，z が $(t-x)$ 以内である場合は対象タスクは注目タスク

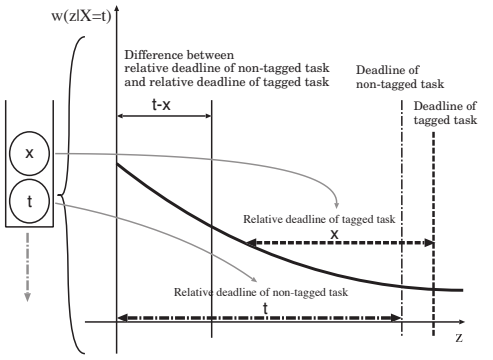

図 2 注目タスクより対象タスクが先に実行されるケース 1

Fig. 2 The first case where non-tagged are executed before the tagged task.

に影響を与えないので対象外となる．z が $(t-x)$ 以上であれば，$(t-x)$ から z の間に注目タスクが到着すると対象タスクは注目タスクを待たせる．この条件を満たす時間の期待値は $\int_{t-x}^{\infty}(z-(t-x))w(z|X=t)dz$ となる．対象タスクの到着率は $\lambda(t)$ であるので，リトルの公式より，該当する対象タスクのタスク数は $\lambda(t)\int_{t-x}^{\infty}(z-(t-x))w(z|X=t)dz$ となる．また，相対デッドライン時間が t であるタスクの実行時間は t/n である．したがって，該当する対象タスクが注目タスクを待たせる時間は以下となる．

$$W1_l(x)=\int_x^{\infty}\frac{t}{n}\lambda(t)\times\left(\int_{t-x}^{\infty}(z-(t-x))w(z|X=t)dz\right)dt \quad (9)$$

式 (8)，(9) より，

$$W1(x)=W1_s(x)+W1_l(x)=\int_0^x\frac{t}{n}\lambda(t)\left(\int_0^{\infty}zw(z|X=t)dz\right)dt+\int_x^{\infty}\frac{t}{n}\lambda(t)\times\left(\int_{t-x}^{\infty}(z-(t-x))w(z|X=t)dz\right)dt \quad (10)$$

（3） 注目タスクがキューにいる間に到着した対象タスクの中で，注目タスクより先に実行される対象タスクによる待ち時間 $W2(x)$

注目タスクより長い相対デッドライン時間をもつ対象タスクは注目タスクに影響を与えない．また，対象タスクは注目タスクより後に到着するので，注目タスクより短い相対デッドライン時間をもつ対象タスクの全てが，先に実行されるとは限らない．これを求めるために，注目タスクの待ち時間の条件付き密度関数を用いる．注目タスクがキューに存在し，対象タスクが到着する場合を考える．これを図 3 に記す．対象タスクが先に実行されるのは，対象タスクのデッドラインが注目タスクのデッドラインより早い場合である．したがって，対象タスクの到着時刻と注目タスクの到着時刻の差分が両タスクの相対デッドライン時間の差分 $(x-t)$ より小さい場合である．注目タスクの到着時刻を 0 とし，注目タスクの待ち時間が z であるとすると，z が $(x-t)$ 以内であれば，0 から z の間に対象タスクが到着すると注目タスクを待たせることとなる．また，z が $(x-t)$ 以上であれば，0 から $(x-t)$ の間に対

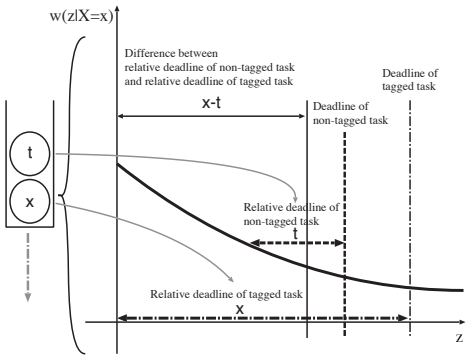

図 3 注目タスクより対象タスクが先に実行されるケース 2

Fig. 3 The second case where non-tagged are executed before the tagged task.

象タスクが到着すると注目タスクを待たせることとなる．この条件を満たす時間の期待値は $\int_0^{x-t}zw(z|X=x)dz+\int_{x-t}^{\infty}(x-t)w(z|X=x)dz$ となる．対象タスクの到着率は $\lambda(t)$ であるので，リトルの公式より，該当する対象タスクのタスク数は $\lambda(t)\int_0^{x-t}zw(z|X=x)dz+\lambda(t)\int_{x-t}^{\infty}(x-t)w(z|X=x)dz$ となる．また，相対デッドライン時間が t であるタスクの実行時間は t/n である．したがって，$W2(x)$ は以下となる．

$$W2(x)=\int_0^x\frac{t}{n}\lambda(t)\left(\int_0^{x-t}zw(z|X=x)dz\right)dt+\int_0^x\frac{t}{n}\lambda(t)\times\left(\int_{x-t}^{\infty}(x-t)w(z|X=x)dz\right)dt \quad (11)$$

3.2 待ち時間の条件付き分布と期待値

前章までに，相対デッドライン時間が x であるタスクの待ち時間の条件付き期待値 $W(x)$ の各要素 W_o，$W1(x)$，$W2(x)$ を求めた．$W(x)$ は式 (4) より，待ち時間の条件付き密度関数 $w(t|X=x)$ を用いて定義できる．よって，式 (7)，(10)，(11) より，以下の再帰の方程式となる．

$$\int_0^{\infty}zw(z|X=x)dz=\int_0^{\infty}\frac{\rho(t)t}{2n}dt+\int_0^x\frac{t}{n}\lambda(t)\left(\int_0^{\infty}zw(z|X=t)dz\right)dt+\int_x^{\infty}\frac{t}{n}\lambda(t)$$

$$\times \left(\int_{t-x}^{\infty} (z-(t-x))w(z|X=t)dz \right) dt$$
$$+ \int_0^x \frac{t}{n}\lambda(t) \left(\int_0^{x-t} zw(z|X=x)dz \right) dt$$
$$+ \int_0^x \frac{t}{n}\lambda(t) \times \left(\int_{x-t}^{\infty} (x-t)w(z|X=x)dz \right) dt \quad (12)$$

また，式 (3)，(5) より，待ち時間の条件付き分布関数 $W(t|X=x)$ を用いると式 (12) は以下のようになる．

$$\int_0^{\infty} (1-W(z|X=x))\,dz = \int_0^{\infty} \frac{\rho(t)t}{2n}dt$$
$$+ \int_0^x \frac{t}{n}\lambda(t) \left(\int_0^{\infty} 1-W(z|X=t)dz \right) dt$$
$$+ \int_x^{\infty} \frac{t}{n}\lambda(t) \left(\int_{t-x}^{\infty} 1-W(z|X=t)dz \right) dt$$
$$+ \int_0^x \frac{t}{n}\lambda(t) \times \left(\int_0^{x-t} 1-W(z|X=x)dz \right) dt \quad (13)$$

4. 待ち時間の条件付き分布の近似

式 (13) を直接解くことは困難なので，相対デッドライン時間の分布が指数分布に従うこと，および EDF スケジューリングとタスクの各属性のいくつかの特徴を利用して近似的に解くこととする．

相対デッドライン時間の分布を平均 $1/\mu$ の指数分布とすると，確率分布関数 $F(x)$，確率密度関数 $f(x)$ は以下である．

$$F(x) = 1-e^{-\mu x} \quad (14)$$
$$f(x) = \mu e^{-\mu x} \quad (15)$$

近似的に解くための 2 つのアプローチを以下に示す．

- 待ち時間の条件付き分布を近似的に定義する．
- 境界条件の特定のために式 (13) の右辺第 3 項を近似的に定義する．

4.1 待ち時間の条件付き分布の近似

EDF スケジューリングの場合の待ち時間の条件付き分布の特徴を以下に示す．

（1） ポアソン到着，指数分布の実行時間を想定した場合，待ちが生じる確率である待ち合わせ率 Π はスケジューリング手法に依存しない．したがって，EDF スケジューリングを適用した場合でも待ち合わせ率 Π は FCFS スケジューリングと同じである．FCFS スケジューリングの場合の待ち合わせ率 Π は以下である．

$$\Pi = \frac{\lambda}{n\mu} \quad (16)$$

（2） 優先権はデッドラインの早い順であるので相対デッドライン時間に条件付けを行った場合，相対デッドライン時間に依存した待ち時間の分布となる．FCFS スケジューリングの場合は，到着順なので，どの相対デッドライン時間に条件付けしても常に同じ待ち時間の分布となる．

特徴 (1)，(2) より，相対デッドライン時間が x であるタスクの待ち時間の条件付き分布関数 $W(t|X=x)$ を以下のように近似する．

$$W(t|X=x) = 1-\Pi e^{-(n\mu-\lambda)K(x)t} \quad (17)$$

ここで，$K(x)$ は条件付けした相対デッドライン時間に依存した係数である．これは，FCFS スケジューリングの場合を $1(=K(x))$ とした場合との変化を表している．当然，FCFS スケジューリングの場合では，相対デッドライン時間に依存しないので $K(x)=1$ の固定となる．EDF スケジューリングの場合は，相対デッドライン時間 x によって $K(x)$ が変動する．

両スケジューリングの場合の待ち時間の条件付き分布 $(1-W(t|X=x))$(補分布) を比較すると，図 4 となる．FCFS スケジューリングの場合は，どの相対デッドライン時間に条件付けしても常に同じ分布となるが，EDF スケジューリングの場合は，相対デッドライン時間が長いタスクは，長く待たされる確率が高くなり，相対デッドライン時間が短いタスクは，長く待たされる確率が低くなる．したがって，相対デッドライン時間が長いタスクでは 1 より小さい $K(x)$ の値をもち，短いタスクについては 1 より大きい $K(x)$ をもつことになる．$K(x)$ については，次章の再帰の方程式を解くことにより求まる．

式 (16)，式 (17) を用いると式 (13) は以下となる．

$$\int_0^{\infty} \frac{\lambda}{n\mu} e^{-(n\mu-\lambda)K(x)z}dz = \frac{\lambda}{(n\mu)^2}$$
$$+ \int_0^x \frac{t}{n}\lambda(t) \times \left(\int_0^{\infty} \frac{\lambda}{n\mu} e^{-(n\mu-\lambda)K(t)z}dz \right) dt$$
$$+ \int_x^{\infty} \frac{t}{n}\lambda(t)$$

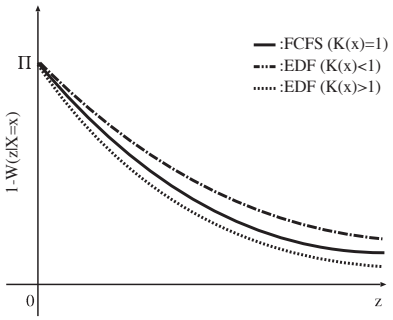

図 4　FCFS スケジューリングと EDF スケジューリングの場合の待ち時間の条件付き分布 $(1 - W(t|X = x))$

Fig. 4　Conditional waiting time distribution $(1 - W(t|X = x))$ for FCFS and EDF

$$\times \left(\int_{t-x}^{\infty} \frac{\lambda}{n\mu} e^{-(n\mu-\lambda)K(t)z} dz \right) dt$$

$$+ \int_0^x \frac{t}{n} \lambda(t) \times \left(\int_0^{x-t} \frac{\lambda}{n\mu} e^{-(n\mu-\lambda)K(x)z} dz \right) dt \quad (18)$$

4.2　境界条件の特定のための近似

式 (18) は，右辺第 3 項の性質上，解くのは困難である．$x = 0$ とした場合，式 (18) の右辺第 2 項と第 4 項は 0 になる．しかし，右辺第 3 項は特定できない．ここでは境界条件の特定のために右辺第 3 項の近似を行う．右辺第 3 項は，注目タスクがキューに到着した時点で，既にキューに存在して，注目タスクより長い相対デッドライン時間をもつ対象タスクによる待ち時間である．式 (1)，式 (18) より，右辺第 3 項は以下である．

$$\lambda \int_x^{\infty} \frac{t}{n} \times \left(\int_{t-x}^{\infty} \frac{\lambda}{n\mu} e^{-(n\mu-\lambda)K(t)z} dz \right) \mu e^{-\mu t} dt$$

$$= \lambda \int_x^{\infty} \frac{t}{n} \left(\frac{\lambda/(n\mu)}{K(t)(n\mu - \lambda)} \right) \times \left(e^{-(n\mu-\lambda)K(t)(t-x)} \right) \mu e^{-\mu t} dt \quad (19)$$

式 (19) において，x と t に依存した項を $R(x,t)$ として抽出し，以下に示す．

$$R(x,t) = \left(\frac{1}{K(t)} \right) \times \left(e^{-(n\mu-\lambda)K(t)(t-x)} \right) \left(\mu e^{-\mu t} \right) \quad (20)$$

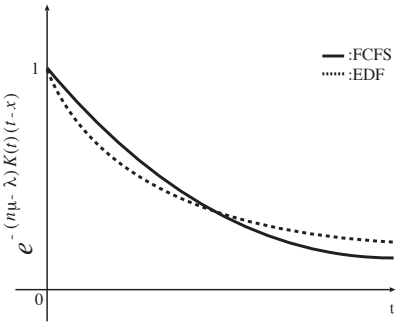

図 5　x が小さい場合の式 (20) の第 2 項

Fig. 5　The second term of (20) in the case that x is small

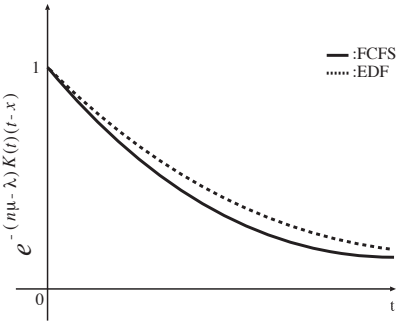

図 6　x が大きい場合の式 (20) の第 2 項

Fig. 6　The second term of (20) in the case that x is large

式 (20) の第 2 項 $(e^{-(n\mu-\lambda)K(t)(t-x)})$ の性質について考えると，注目しているのは，相対デッドライン時間が x であるタスクであり，対象としているのは，相対デッドライン時間が t であるタスクなので，積分範囲を考慮すると t が x から ∞ まで変動する．t が小さい場合は，$K(t)$ が 1 より大きい値となる．t が大きい場合は，$K(t)$ が 1 より小さい値となる．すなわち，注目タスクの相対デッドライン時間 x が小さい場合は，図 5 のように，式 (20) の第 2 項 $(e^{-(n\mu-\lambda)K(t)(t-x)})$ は FCFS スケジューリング $(K(t) = 1)$ の場合と交差する分布となる．逆に，注目タスクの相対デッドライン時間 x が大きい場合は，図 5 のように FCFS スケジューリング $(K(t) = 1)$ の場合と交差せずに常に大きい分布となる．積分すると，交差する場合は，FCFS スケジューリング $(K(t) = 1)$ の場合との誤差が打ち消される．交差しない場合は，誤差が大きくなる．し

かし，注目タスクの相対デッドライン時間 x が大きい場合，式 (20) の第 3 項 ($\mu e^{-\mu t}$) より，誤差が非常に大きくなる部分では，式 (20) の値自体が非常に低くなる．つまり，FCFS スケジューリング ($K(t) = 1$) を仮定しても，ある区間で一時的に多少の誤差が表れるが，再びその誤差は減少していくと考えられる．式 (18) の右辺第 3 項の $K(t)$ に関しては，1 に置換えてもその影響は非常に少ないと考えられる．

置換えを行うと以下の式となる．

$$\begin{aligned}
&\int_0^\infty \frac{\lambda}{n\mu} e^{-(n\mu-\lambda)K(x)z} dz = \frac{\lambda}{(n\mu)^2} \\
&+ \int_0^x \frac{t}{n}\lambda(t) \\
&\quad \times \left(\int_0^\infty \frac{\lambda}{n\mu} e^{-(n\mu-\lambda)K(t)z} dz \right) dt \\
&+ \int_x^\infty \frac{t}{n}\lambda(t) \\
&\quad \times \left(\int_{t-x}^\infty \frac{\lambda}{n\mu} e^{-(n\mu-\lambda)z} dz \right) dt \\
&+ \int_0^x \frac{t}{n}\lambda(t) \\
&\quad \times \left(\int_0^{x-t} \frac{\lambda}{n\mu} e^{-(n\mu-\lambda)K(x)z} dz \right) dt \quad (21)
\end{aligned}$$

式 (21) を解くことにより，$K(x)$ を求めることができ，式 (17) により，相対デッドライン時間が x であるタスクの待ち時間の条件付き分布関数 $W(t|X = x)$ を求めることができる．

5. シミュレーションと評価

本論文で提案した近似解析で求めた待ち時間の条件付き期待値，条件付き分布関数の結果とシミュレーションの結果を比較し，本解析法の評価を行う．

5.1 結　　果

- 待ち時間の条件付き期待値 $W(x)$

シミュレーションでは，各相対デッドライン時間の範囲に対して表 1 に示す試行回数と標本数のシミュレーションを行った．例えば，0 から 100 の相対デッドライン時間の範囲では 10 の標本間隔毎，すなわち x が 0 から 10 の間，10 から 20 の間といった 10 毎の区間に対して，標本数が 100000 個となる試行を 10 回行ったことを意味する．標本数が少なくなる相対デッドライン時間の大きい部分 (600 から 1000 の相対デッドライン時間の範囲) でも，95%の信頼区間を平均値の 10%以内で得るために 200 の標本間隔毎に標本数が 1000 個となる試行を 10 回行った．

パラメータを表 2 に示し，対応する結果を図 7，図 8，図 9，図 10，図 11 に示す．各図では待ち時間の平均値を相対デッドライン時間の範囲の中心点でプロットし，その 95%の信頼区間を縦の実線で示している．また，本解析の結果を実線で示している．

0 から 600 までの相対デッドライン時間に関しては，95%の信頼区間が平均値の 5%以内となっており，600 から 1000 までの相対デッドライン時間に関しても，95%の信頼区間が平均値の 10%以内となっている．

表 1　試行回数と標本数
Table 1　Count of trial and Sample Size

相対デッドライン時間	標本間隔	試行回数	標本数
0-100	10	10	100000
100-200	20	10	100000
200-300	50	10	100000
300-600	100	10	10000
600-1000	200	10	1000

表 2　待ち時間の条件付き期待値 ($W(x)$)
Table 2　Conditional expected waiting time($W(x)$)

λ	0.001	0.005	0.009	0.006	0.006
μ	0.01	0.01	0.01	0.01	0.01
n	1	1	1	3	6
$W(x)$	図 7	図 8	図 9	図 10	図 11

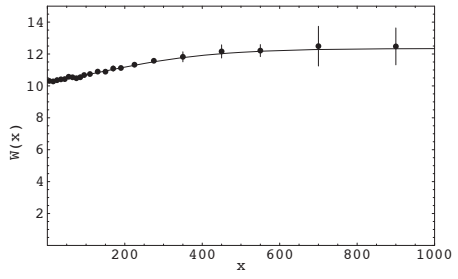

図 7　待ち時間の条件付き期待値 ($\lambda = 0.001, \mu = 0.01, n = 1$)
Fig. 7　Conditional expected waiting time($\lambda = 0.001, \mu = 0.01, n = 1$)

- 待ち時間の条件付き分布 ($1 - W(t|X = x)$)

パラメータを表 3 に示し，対応する結果を図 12，図 13 に示す．なお，各図では待ち時間の条件付き分布 ($1 - W(t|X = x)$)(補分布) を示している．図 12 では相対デッドライン時間が 95 から 105 の間にあるタスクの待ち時間の分布，図 13 においては相対デッドラ

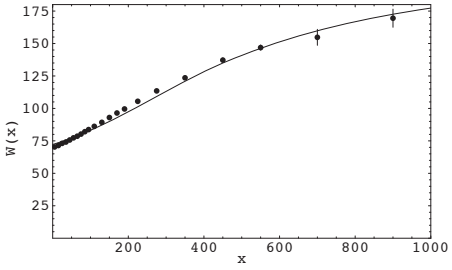

図 8　待ち時間の条件付き期待値 ($\lambda = 0.005, \mu = 0.01, n = 1$)
Fig. 8　Conditional expected waiting time($\lambda = 0.005, \mu = 0.01, n = 1$)

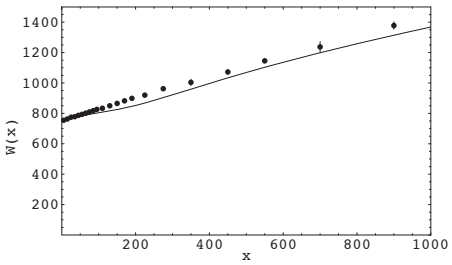

図 9　待ち時間の条件付き期待値 ($\lambda = 0.009, \mu = 0.01, n = 1$)
Fig. 9　Conditional expected waiting time($\lambda = 0.009, \mu = 0.01, n = 1$)

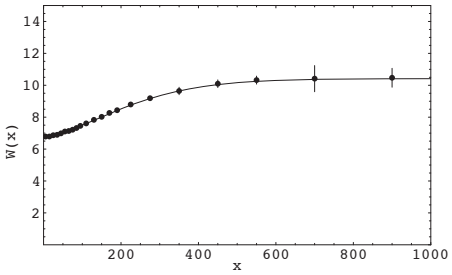

図 10　待ち時間の条件付き期待値 ($\lambda = 0.006, \mu = 0.01, n = 3$)
Fig. 10　Conditional expected waiting time($\lambda = 0.006, \mu = 0.01, n = 3$)

イン時間が 290 から 310 の間にあるタスクの待ち時間の分布をプロットしている．また，本解析の結果を実線で示している．

表 3　待ち時間の条件付き分布 ($1 - W(t|X = x)$)
Table 3　Conditional waiting time distribution($1 - W(t|X = x)$)

λ	0.005	0.005
μ	0.01	0.01
n	1	1
x	100	300
$1 - W(t\|X = x)$	図 12	図 13

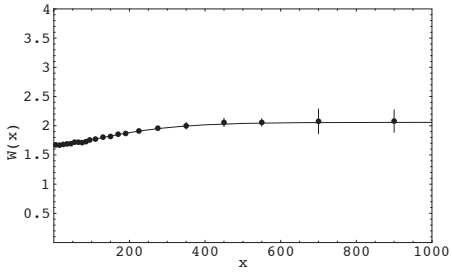

図 11　待ち時間の条件付き期待値 ($\lambda = 0.006, \mu = 0.01, n = 6$)
Fig. 11　Conditional expected waiting time($\lambda = 0.006, \mu = 0.01, n = 6$)

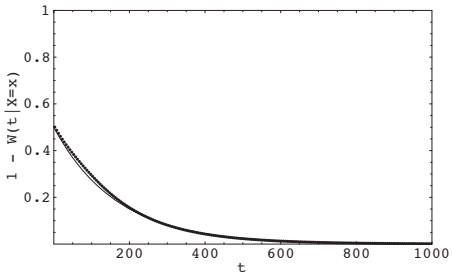

図 12　待ち時間の条件付き分布 ($x = 100$)
Fig. 12　Conditional waiting time distribution($x = 100$)

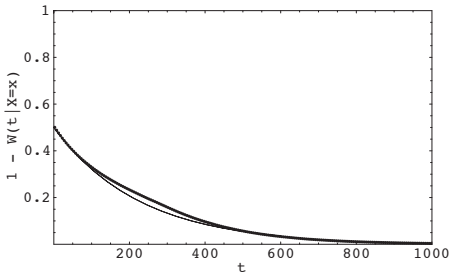

図 13　待ち時間の条件付き分布 ($x = 300$)
Fig. 13　Conditional waiting time distribution($x = 300$)

5.2 評　　価

負荷率に注目し，図 7，図 8，図 9 について評価する．低負荷 (図 7) と中負荷 (図 8) の状態では，式 (5)，(17)，(21) から求めた待ち時間の条件付き期待値とよく一致している．高負荷 (図 9) の状態では，平均実行時間の 1.5 倍あたりから 4 倍あたりまで，多少のずれが生じている．これは，式 (13) の右辺第 3 項の $K(t)$ を 1 と置換えた影響である．高負荷 (図 9) では，最大で 5%ほどの誤差が生じている．図 10，図 11 は実行

時間比率を変化させた場合である．実行時間比率を変化させても結果は一致している．しかし，高負荷になると前述と同様に多少の誤差が生じると考えられる．非常に高負荷な場合を除いては一致し，非常に厳密な精度を必要としない場合は提案した手法は有用である．

相対デッドライン時間が 100 と 300 の両待ち時間の条件付き分布に関しても，式 (17)，(21) から求めた分布とよく一致している．相対デッドライン時間が 300 の待ち時間の条件付き分布において，条件付けした相対デッドライン時間の付近で本解析の結果がシミュレーションより低めになっている．このずれは対象とする相対デッドライン時間の範囲 (290,310) を広めにとっているためと考えられる．また，厳密な待ち時間の条件付き分布を表しておらず，近似による影響が考えられる．しかし，前述と同様，非常に厳密な精度を必要としない場合には提案した手法は有用である．

6. 非割込み型 EDF スケジューリングの特性

FCFS スケジューリング，SJF スケジューリングと EDF スケジューリングを用いた場合の待ち時間の条件付き期待値に関して比較した結果を図 14 に示す．パラメータは，$\lambda = 0.005$，$\mu = 0.01$ とし，EDF スケジューリングに関しては実行時間比率 $n = 1$ である．ここで，実行時間と相対デッドライン時間が等しいとするので，ここでは相対デッドライン時間を実行時間として扱う．

FCFS スケジューリングの場合は実行時間に依存しないので，どの実行時間でも待ち時間の期待値は変わらない．

一方，SJF スケジューリングの場合は実行時間が短いタスクに対して優先権を与えるために実行時間が短いタスクは待ち時間が極端に短く，実行時間が長いタスクは待ち時間が極端に長くなる．実行時間が極めて短い場合は W_o であり，極めて長い場合は $W_o/(1-\lambda/\mu)^2$ である [2]．待ち時間の各要素については，実行時間が短いタスクに対して優先権を与えるために本解析での $W1_l(x)$ に該当する要素は存在しない．$W1_s(x)$ と $W2(x)$ については，平均実行時間の辺りから大きくなり始め，ある値に収束している．

EDF スケジューリングの場合は図 14 のように FCFS スケジューリングと SJF スケジューリングの場合の中間的な値をとりながら両スケジューリングの場合と交差する．実行時間が極めて短い場合は，$W_o + W1_l(0)$ になる．実行時間が長くなるにつれ，$W1_l(x)$ については 0 に近づき，$W1_s(x)$ と $W2(x)$ については徐々に大きくなる．理由として，$W1_l(x)$ については待ちの要素となる実行時間が x より大きい対象タスクの到着率の合計 ($\int_x^\infty \lambda(t)dt$) が徐々に少なくなり，0 に近づくためである．$W1_s(x)$ と $W2(x)$ については待ちの要素となる実行時間が 0 から x である対象タスクの到着率の合計 ($\int_0^x \lambda(t)dt$) が徐々に大きくなり，ある値に収束していくためである．第 4 章で詳細に述べているが，本解析では $W1_l(x)$ の振る舞いの予測可能性を利用し，近似する手法を用いている．x が大きいほど，$\int_x^\infty \lambda(t)dt$ が限りなく 0 に近づく性質と x が小さい場合でも誤差が打ち消される性質を利用し，$W1_l(x)$ の $W(z|X=t)$ について，FCFS スケジューリングの場合の待ち時間の分布と置き換えた手法である．

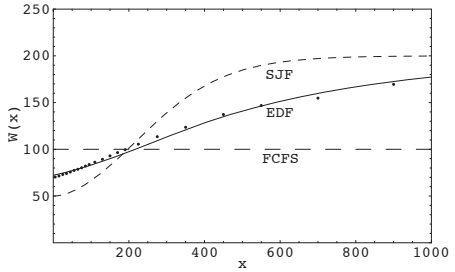

図 14　各スケジューリング手法における待ち時間の条件付き期待値 $W(x)$
Fig. 14　Conditional expected waiting time $W(x)$ under each scheduling

7. む す び

本論文では，最も単純な単一サーバモデルを対象とし，タスクの各属性が確率分布に従うものと仮定し，非割込み型 EDF スケジューリングを適用した場合のシステム性能を数学的に解析した．EDF スケジューリングを用いた場合，一般的な優先権スケジューリングの平均待ち時間導出の枠組みを適用できないために，新たな理論的枠組みを提案した．さらに，その枠組みを用いて，近似的に解く手法を提案した．評価では本解析の結果とシミュレーションの結果の比較により本解析法が有用であることを示し，さらに，本解析より明らかになった非割込み型 EDF スケジューリングの性質を示した．本論文では示していないが，待ち時間の分布より，リアルタイムシステムの性能指標である平均遅延時間，デッドラインミス率の算出も可能であ

る．解析の対象としているモデルは最も単純ではあるが，システム設計時，あるいはシステム変更時などに，どのような効果があるかを把握するための十分な指標となり得る．

本解析の拡張としては，相対デッドライン時間と実行時間の間の静的な関係を確率的な関係にすることや待ち時間，デッドラインなどを考慮した最適化問題などが考えられる．また，現実のシステムでは，様々なタスクの到着，相対デッドライン時間，実行時間も考えられる．特に，すべてのタスクの到着間隔，相対デッドライン時間 (あるいは実行時間) が指数分布であると仮定できないシステムも多く存在する．今後はタスクの到着間隔，相対デッドライン時間，実行時間が一般分布に従う場合の解析も必要である．また，実システムに対する適用事例として，本論文で提案した解析の結果と現実のシステムの性能の比較を行う予定である．

文　　献

[1] L. Kleinrock, Queueing Systems Volumne I, John Wiley & Sons Inc., New York, 1975.

[2] L. Kleinrock, Queueing Systems Volumne II, John Wiley & Sons Inc., New York, 1975.

[3] 森村英典，大前義次, 応用待ち行列理論，日科技連出版社，東京, 1975.

[4] 亀田尚夫, 紀一誠, 李頡, 性能評価の基礎と応用，共立出版, 東京, 1998.

[5] 中里秀則，“EDF スケジューリングでの応答時間予測，”電子情報通信学会 コンピュータシステム研究会 RTP'99, 1999.

[6] T.E. Phipps, “Machine Repair as a Priority Waiting Line Problem,” Oper. Res., Vol.4, pp.76-85, 1956.

[7] M. Sakata, S. Noguchi and J. Oizumi, “An Analysis of the M/G/1 Queue Under Round-Robin Discipline,” Oper. Res., Vol.19, pp.371-385, 1971.

[8] E.G. Coffman, Jr., R.R. Muntz, and H. Trotter, “Waiting Time Distributions for Processor-Sharing Systems,” J. Assoc. Comput. Mach., Vol.17, No.1, pp.123-130, Jan 1970.

[9] J.R. Jackson, “Scheduling a production line to minimize maximum tardiness,”Research Report 43, Management Science Research Project, University of California, Los Angeles, 1955.

[10] C.L. Liu and J.W. Layland, “Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment,”J. Assoc. Comput. Mach., Vol.20, No.1, pp.41-46, Jan 1973.

[11] John. P. Lehoczky, “Real-Time Queueing Theory,” Proc. IEEE Real-Time Systems Symposium, pp.186-195, Dec 1996.

2.2. 非割込み型 EDF スケジューリングにおけるパフォーマンス予測：理論と実装

非割込み型 EDF スケジューリングにおけるパフォーマンス予測：理論と実装
Theory and Experimental Results of Non-Preemptive EDF Scheduling Performance

花田真樹 †　　中里秀則 †
Masaki HANADA　　**Hidenori NAKAZATO**

1　まえがき

近年，計算機の高性能化やネットワークの高速化に伴い，リアルタイムシステムの適用分野が，ロボット制御やプラント制御などの制御系システムのみでなく，VOD やテレビ会議システムなどのマルチメディア情報を扱う情報系システムへと広がっている．一般的に，これらの情報系システムはソフトリアルタイムに分類され，デッドライン (実行終了期限) ミスが起きても，システムは稼動し続ける．さらに，ユーザ要求のタイミングを事前に把握することが困難なために非決定的な環境下で動作することが多い．そのため，絶対的な保証を行うのは困難であり，統計的な保証が必要となる．

リアルタイム性を確保するためには，どのタスクをいつ実行するかを決定するスケジューリング手法が重要な要素である．これまでに，タスクのデッドラインを守るために，Earliest Deadline First(EDF)[1, 2, 3] などのリアルタイムスケジューリングが提案されている．EDF スケジューリングは，デッドラインの早い順に優先順位を付ける手法であり，事前に起動時間がわからない非周期タスクに適応可能である．

我々は，既にタスクがランダムに到着する仮定下で，非割込み型 EDF スケジューリングを用いた場合のシステム性能を待ち行列理論を用いて解析している [3]．しかしながら，我々の提案している解析ではタイマの精度，割込み処理やコンテキストスイッチのオーバーヘッドなどを考慮しておらず，システム性能に関して，実システムの実測データとは一致しない可能性がある．そこで，本論文では，非割込み型 EDF スケジューリングを行うタスクスケジューラをリアルタイム OS 上に実装し，システム性能を理論解析と実測データの両面から評価する．

現在，リアルタイムアプリケーション構築の際のプラットフォームとして，様々なリアルタイム OS が用いられている．リアルタイム OS を用いる主な利点は，マルチタスクによるリアルタイム性の確保にある．特に，本論文ではスケジューラの変更を行うため，フリーで変更が比較的容易なタイムシェアリング系 OS をリアルタイム拡張した RTLinux[4, 5, 6] を用いる．

2 では既に提案している解析によるシステム性能の算出式について述べる．**3** では RTLinux について述べる．**4** ではタスクモデルと非割込み型 EDF スケジューリングを行うスケジューラの実装について述べ，実験結果を示し，システム性能に関して理論解析と実測データの両面から評価する．**5** ではまとめと今後の課題について述べる．

2　理論解析

(1) システムモデル

システムモデルは，単一サーバモデル (図 1) とする．

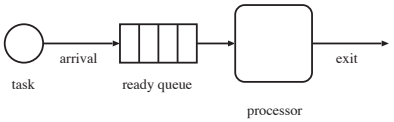

図 1　単一サーバモデル

タスクの実行順序はデッドラインの早い順とし，キューは無限長とする．実行中のタスクよりも早いデッドラインをもったタスクが到着した場合でもそのまま実行を続行する非割込み型である．タスクが到着した場合には，デッドライン保証の判定は行わず，即座にキューに格納され，さらに，タスクがデッドラインを守れなかった場合でもその実行を中断することはないとする．

タスクは実行時間と相対デッドラインをもつ．タスクのデッドラインは，到着時刻と相対デッドラインを足した値として定義される．

(2) 性能解析 [3]

タスクの到着は平均到着率 λ のポアソンとし，相対デッドラインを平均 $1/\mu$ の指数分布とする．

一般的に，相対デッドラインと実行時間の間には強い依存性があると考えられるので，相対デッドラインの短いタスクは，実行時間も同様に短いと仮定する．これより，各タスクの実行時間は，そのタスクの相対デッドライン時間の $1/n(n$ は固定) とする (以下，n を実行時間比率と呼ぶ).

① 待ち時間の条件付き期待値の各要素

システム性能評価指標として，相対デッドラインが x であるタスクの待ち時間の条件付き期待値 $W(x)$ を導出する．

$W(x)$ は以下の 3 つの要素で構成される．ここで，相対デッドライン時間が x であるタスクを注目タスク，注目タスクの待ちに寄与する相対デッドライン時間が t であるタスクを対象タスクと呼ぶ．

i. 注目タスクの到着時に実行中の対象タスクによる待ち時間 W_o
ii. 注目タスクがキューに到着した時点で，既にキューにある対象タスクの中で注目タスクより先に実行される対象タスクによる待ち時間 $W1(x)$
iii. 注目タスクがキューにいる間に到着した対象タスクの中で，注目タスクより先に実行される対象タスクによる待ち時間 $W2(x)$

これより，$W(x)$ は以下である．

$$W(x) \quad = \quad W_o + W1(x) + W2(x) \qquad (1)$$

② 待ち時間の条件付き期待値

待ち時間の条件付き期待値の各要素を求め，条件付き分布関数 $W(t|X = x)$ を用いると式 (1) は以下の

† 早稲田大学大学院 国際情報通信研究科
〒 367-0032 本庄市大字西富田字大久保山 1101-3
GITS,Waseda University
1101-3 Ohkuboyama,Nishitomida,Honjo-shi,
Saitama 367-0032,Japan

ようになる．

$$
\begin{aligned}
W(x) &= \int_0^{\infty} (1 - W(z|X = x))\, dz \\
&= \int_0^{\infty} \frac{\rho(t)t}{2n} dt \\
&+ \int_0^{x} \frac{t}{n}\lambda(t) \left(\int_0^{\infty} 1 - W(z|X = t) dz \right) dt \\
&+ \int_x^{\infty} \frac{t}{n}\lambda(t) \left(\int_{t-x}^{\infty} 1 - W(z|X = t) dz \right) dt \\
&+ \int_0^{x} \frac{t}{n}\lambda(t) \left(\int_0^{x-t} 1 - W(z|X = x) dz \right) dt.
\end{aligned} \tag{2}
$$

また，$W(t|X = x)$ は依存係数 $K(x)$ を用いることにより，近似的に以下の式で表せる．

$$
W(t|X = x) \quad = \quad 1 - \frac{\lambda}{n\mu} e^{-(n\mu - \lambda)K(x)t}. \tag{3}
$$

式 (3) を式 (2) に代入することにより，$K(x)$ が求まる．
また，$W(x)$ は式 (3) を用いると以下の式となる．

$$
W(x) \quad = \quad \int_0^{\infty} \frac{\lambda}{n\mu} e^{-(n\mu - \lambda)K(x)z} dz. \tag{4}
$$

式 (2)，式 (3) より求めた $K(x)$ を式 (4) に代入することにより $W(x)$ が求まる．

3 RTLinux

(1) 概要

RTLinux は，Linux を修正してリアルタイム処理をできるようにした OS である．Linux ではタイムシェアリングを用いており，タスクが CPU への割り当てられた時間枠を超えるとカーネルがそれを停止し，他のタスクに割り当てるので，リアルタイム性の必要なタスク (以下，リアルタイムタスクと呼ぶ) も他のタスクと平等に扱われることになる．そこで，RTLinux では，新しいレイヤ (リアルタイムカーネル) をハードウェアと Linux カーネルの間に導入し，リアルタイムカーネルの RT スケジューラにより，Linux カーネルを優先順位の低い一つのタスクとして実行し，リアルタイムタスクが CPU を必要としたときは，横取りするように設計されている．図 2 に RTLinux の構造を示す．

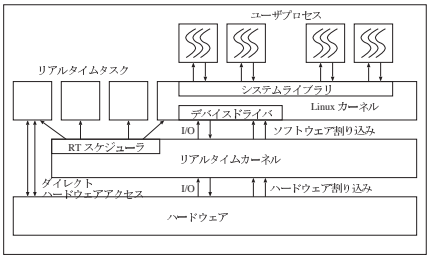

図 2　RTLinux カーネルの構造

(2) スケジューリング方式

RTLinux では，割込み型の固定優先度スケジューリング方式を提供している．同一優先順位内での方式として，先入れ先出しを行うスケジューリング方針とシステムごとのタイムスライス (割り当てられた時間の長さ) を使用するラウンドロビン スケジューリングがある．

(3) 割込み処理

RTLinux では，特定の割込みに対してのみ，適切な割り込みサービスルーチンが動作する．それ以外の割込みは全て一旦保留され，RTLinux カーネルがアイドル状態になり，Linux カーネルが稼働したときにソフトウェア割り込みとしてカーネルにわたる仕組みとなっている．ハードウェア割り込みに関しては，最悪のケースで割り込み発生から 15 マイクロ秒の遅れで実行され，周期的なプロセスに対してはスケジューリングされた時刻から 25 マイクロ秒以内の遅れで実行される仕様となっている [7]．

4 実装と実験

この章では，まず，実システムにおける割込みハンドラとタスクについて述べ，次に，タスクモデルとスケジューラの実装について述べる．次に，実験結果を示し，理論解析と実測データの両面より評価する．

(1) 実システムにおける割込みハンドラとタスク

リアルタイムシステムの実行単位は，タスクとハンドラに分類される．一般的に，CPU は通常の実行状態と割込み/例外処理実行状態をもっており，タスクは通常の実行状態に対応し，ハンドラは割込み/例外処理実行状態に対応する動作単位である．実システムでは，割込みハンドラは，最小限の割込み応答処理と，イベントを検出し，タスクへ通知する処理を行うのが一般的である (図 3)．

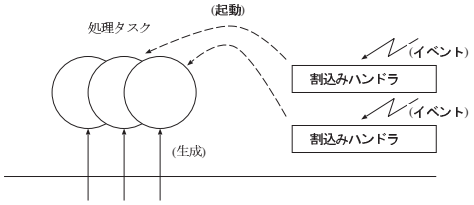

図 3　実システムにおける割込みハンドラとタスク

(2) 実装

① タスクモデル

上記で述べたように，実システムでは一般的に割込みハンドラがイベントを検出し，対応するタスクを起動するという実装が多く行われる．割込みハンドラはある特定のタスクと対応している場合が多い．本実験では，タスクのポアソン到着を擬似するために，割込みハンドラを用いず，処理を振り分けるタスク (以下，振り分けタスクと呼ぶ) から，処理を行うタスク (以下，処理タスクと呼ぶ) を起動する (図 4)．振り分けタスクはタイマ割込みにより起動する．また，各タスクは，リアルタイム性が要求されるため，リアルタイムカーネル上のリアルタイムタスクとして扱う (図 2)．

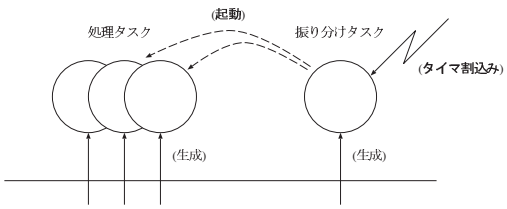

図 4　タスクモデル

振り分けタスクでは，必要な前処理 (全タスクに共通な処理など) が行われ，デッドラインなどの必要なパ

ラメータを付加して，対応するタスクを起動し，処理を依頼する．

② スケジューラ

リアルタイムカーネルの RT スケジューラを，既に実装されている割込み型の固定優先度スケジューリング方式から固定優先度非割込み型 EDF スケジューリング方式に変更する．タスクは固定の優先度とデッドラインをもっている．固定優先度非割込み型 EDF スケジューリング方式では，まず，優先度の高いタスクが先に実行される．さらに，タスクが同優先度を持つ場合は，タスクのデッドラインの早い順に実行される．

振り分けタスクは他の処理タスクへの通知を行う必要があるので，タイマ割込みがあると他の処理タスクが実行中であっても，処理タスクへ割込みを行えるものとする．処理タスクは，振り分けタスクからのみ割込みを許可し，他の処理タスクからの割込みを不可とする．図 5 は，処理タスク A が振り分けタスクに割り込まれ，処理タスク A が処理タスク C に割り込まれない状況を示している．

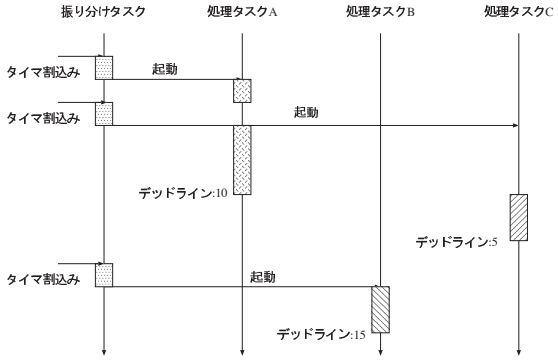

図 5　タスク実行の例

(3) 実験

実験では，Linux カーネル (version2.4.4) と RTLinux(version3.1pre3) を使用する．また，前提として，**2** の条件と同様に，タスクの到着は平均 λ のポアソン，相対デッドラインは平均 $1/\mu$ の指数分布とする．各タスクの実行時間は，簡易的に，そのタスクの相対デッドラインと等しいとする．

実験における振り分けタスクと処理タスクの処理シーケンスの概要を図 6 に示す．

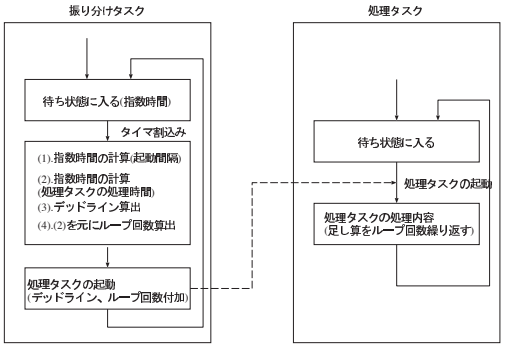

図 6　実験における処理シーケンス

- 処理タスク

  処理内容は，簡易的に足し算 (1+1) を繰り返す処理とする．

- 振り分けタスク

  タイマ割込みにより起動する．処理タスクの指数時間の処理を実現するために，処理時間に対応する繰り返し回数を求め，デッドラインと共に処理タスクに通知する．

RTLinux 上で足し算の繰り返し回数に対する計測した処理時間 (ナノ秒) を表 1 に示す．

表 1　ループ回数に対する処理時間

回数	100	1000	10000	100000	1000000	10000000
処理	337	3376	33498	333371	3334283	33341903

ループ回数を l，処理時間を s とすると，1 次式への近似は以下となる．

$$3.33 \times l + 41.19 = s$$

これより，指数分布の乱数 (処理時間 s) から繰り返し回数 l を算出する．

(4) 実験結果

パラメータと対応する実測データを表 2 に示す．$1/\lambda$ と $1/\mu$ に関しては，単位はマイクロ秒である．図中における実線は **2** で述べた我々の提案している解析結果であり，プロットは実験により得られた実測データである．また，横軸は相対デッドライン (処理時間) であり，縦軸は実行可能キューにおける待ち時間である．図中の単位はナノ秒である．

表 2　パラメータ

$1/\lambda$	1000	1000	1000	10000	10000	10000
$1/\mu$	200	500	800	2000	5000	8000
W(x)	図 7	図 8	図 9	図 10	図 11	図 12

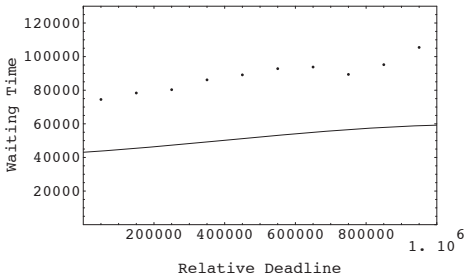

図 7　平均待ち時間 W(x)($1/\mu = 200$)

(5) 評価

図 10，図 11，図 12 においては，実測データと解析結果がほぼ一致しているが，図 7，図 8，図 9 においては，大きな誤差が表れている．また，高負荷 (図 12 と図 9) の時と比べて，低負荷 (図 10 と図 7) の方が誤差が大きい．これらは，パラメータに対して，コンテキストスイッチにかかる時間が大きいためである．図 13 に，コンテキストスイッチを含めた各処理にかかる時間を示す．

振り分けタスクには，2 回の指数時間算出，デッドラインとループ回数算出の処理時間 (図 13 の①) が含まれる．さらに，それに加えて，タスク起動 (図 13 の②) とコンテキストスイッチ (図 13 の③) の時間が必要となる．

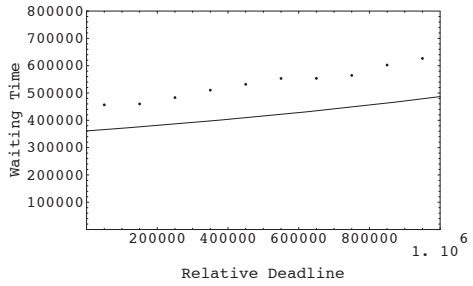

図 8　平均待ち時間 W(x)($1/\mu = 500$)

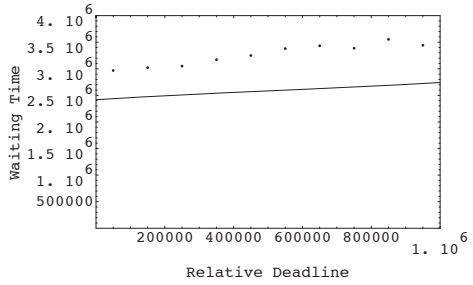

図 9　平均待ち時間 W(x)($1/\mu = 800$)

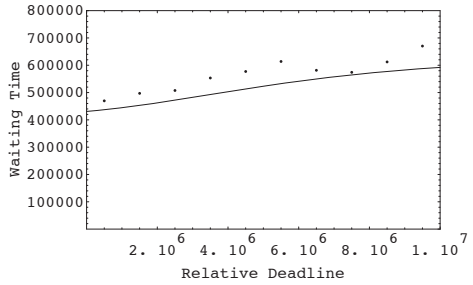

図 10　平均待ち時間 W(x)($1/\mu = 2000$)

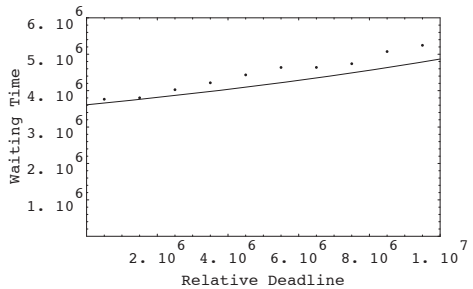

図 11　平均待ち時間 W(x)($1/\mu = 5000$)

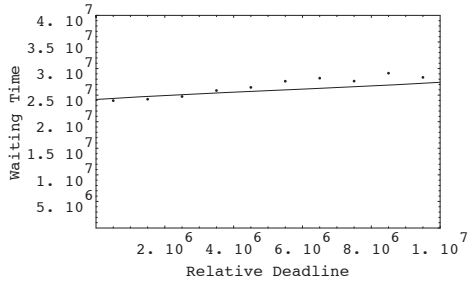

図 12　平均待ち時間 W(x)($1/\mu = 8000$)

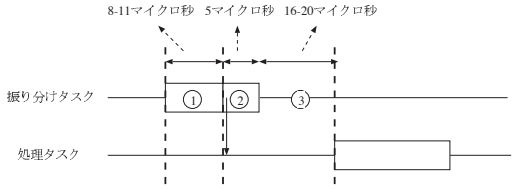

図 13　各処理にかかる時間

これより，パラメータに対して，コンテキストスイッチにかかる時間が大きい場合は，我々の提案している解析法の適用は困難である．しかし，パラメータがコンテキストスイッチにかかる時間に比べて，十分に小さい場合は，我々の提案している解析法は有効である．

5　おわりに

本論文では，非割込み型 EDF スケジューリングを行うスケジューラをリアルタイム OS 上に実装し，システム性能を実測データと理論的解析の両面から評価した．パラメータに対して，タスク起動やコンテキストスイッチにかかる時間が大きい場合，解析結果とは一致しない．しかし，パラメータがタスク起動やコンテキストスイッチにかかる時間に比べて，十分に小さい場合は，我々の提案している解析法は有効であることが分かった．

解析の課題としては，コンテキストスイッチは遊休期間と考えられるので，遊休期間を含めた解析が必要である．

今後の課題としては，本論文では待ち時間に注目したが，待ち時間に処理時間を加えた応答時間では，振り分けタスクが処理タスクに割り込む可能性が高いので，割込み回数を含めた算出が必要である．また，本論文では実験のために擬似的にポアソン到着，指数分布の処理時間を生成している．システムによっては到着や処理の分布が異なる可能性があるので，その場合の解析や実験も必要である．さらに，本論文ではリアルタイム OS を用いたが，予測が困難な一般的な OS での実験も必要であると考える．

参考文献

[1] J.R. Jackson, “Scheduling a production line to minimize maximum tardiness,” Research Report 43, Management Science Research Project, University of California, Los Angeles, 1955.

[2] C.L. Liu and J.W. Layland,“Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment,”J. Assoc. Comput. Mach., Vol.20, No.1, pp.41-46, Jan 1973.

[3] 花田 真樹，中里 秀則，“非割込み型 EDF スケジューリングの近似解析,”電子情報通信学会論文誌, VOL.J87-A No.12, pp.1518-1527, Dec 2004.

[4] FSMLabs, http://www.fsmlabs.co.jp/.

[5] Matt Sherer，FSMLabs Technical Staff, RTLinux テキストブック, CQ 出版社, 2003 年 4 月.

[6] 船木陸議，“RT-Linux の構造と実装の詳細,”CQ 出版社インターフェース増刊 Vol.5 第 7 章, 2000 年 7 月.

[7] 石井和男，“リアルタイム Linux「RTLinux」の導入とロボットへの適用,”日本ロボット学会第 61 回講習会ロボット工学セミナーテキスト，pp.1-23，2000.

2.3. ORC-GPS 方式における End-to-End 最大遅延の評価

[奨励講演] ORC-GPS 方式における End-to-End 最大遅延の評価

花田　真樹†　　中里　秀則†

† 早稲田大学大学院 国際情報通信研究科
〒 367-0032 埼玉県本庄市大字西富田字大久保山 1101-3
E-mail: †hanada@fuji.waseda.jp, nakazato@waseda.jp

あらまし　近年，電話システムやビデオ会議システムなどの普及に伴い，パケット交換ネットワークにおいて，決定的あるいは統計的に遅延を保証することが求められている．我々は，既に決定的遅延保証を行うためのスケジューリング方式である Output Rate-Controlled Generalized Processor Sharing (ORC-GPS) を提案し，EDF と比較して同等以上のスケジューラブル領域が得られることを示した．本稿では，これに加え，数値例を用いて，EDF と GPS の両方式と比較してどの程度 End-to-End 最大遅延を削減できるかを示す．
キーワード　決定的遅延保証，End-to-End 最大遅延，スケジューラブル領域，EDF，GPS

[Encouragement Talk] Evaluation of End-to-End Delay Bounds for ORC-GPS

Masaki HANADA† and Hidenori NAKAZATO†

† Graduate School of Global Information and Telecommunication Studies, Waseda University
1101-3 Ohkuboyama, Nishitomida, Honjo-shi, Saitama, 367-0032 Japan
E-mail: †hanada@fuji.waseda.jp, nakazato@waseda.jp

Abstract　Recently telephone and video applications have become widely used in packet-switching networks. These applications require deterministic or statistical delay guarantees. We have proposed a scheduling algorithm called Output Rate-Controlled Generalized Processor Sharing (ORC-GPS) to guarantee deterministic delay bounds. We proved that the schedulable region of ORC-GPS is larger than or almost the same as EDF in our previous paper. We compare End-to-End delay achievable by ORC-GPS, EDF, and GPS, and show the superiority of ORC-GPS.
Key words　deterministic delay guarantees, End-to-End delay, schedulable region, EDF, GPS

1. まえがき

近年，電話システムやビデオ会議システムなどの普及に伴い，パケット交換ネットワークにおいて，決定的あるいは統計的に遅延を保証することが求められている．決定的遅延保証とは決められた遅延を完全に保証することであり，統計的遅延保証とは決められた遅延をある確率以内で保証することである．

我々は，既に音声や映像などのリアルタイムトラヒックに対する決定的遅延保証に注目し，スケジューラブル領域の最大化を目的とするスケジューリング方式 Output Rate-Controlled Generalized Processor Sharing (ORC-GPS) を提案している [4]．ここで，スケジューラブル領域とは，決定的遅延保証が可能な遅延の範囲のことである．N 個の End-to-End トラヒック (以下，セッションと呼ぶ) が与えられ，セッション i の最大遅延を d_i と表記すると，スケジューラブル領域は保証可能な最大遅延のベクトル $\vec{d} = (d_1, d_2, \ldots, d_N)$ の集合として定義される．スケジューリング方式によって生じる最大遅延が小さくなれば，そのスケジューラブル領域は大きくなることになる．

これまでに，決定的遅延保証を行うために，Generalized Processor Sharing (GPS) [1], [2], [5] や Earliest Deadline First (EDF) [3] に基づく多くのスケジューリング方式が提案され，Leaky Bucket による制約の条件下で最大遅延が求められている．

GPS では各セッションに重みを割り当て，その重みに従ってサービスされる．しかし，常に重みに従ってサービスされるために，各セッションの最大遅延が不必要に増加してしまう可能性がある．一方，各ノードにおいて最悪ケースのセッション毎のサービス曲線を厳密に特定することが可能であり，セッションの経由するノードが複数の場合 (以下，複数ノードと呼ぶ) では，このサービス曲線を用いることにより，厳密な End-to-End 最大遅延が求められる．(以下，サービス累積量を表す曲線をサービス曲線と呼ぶ．)

EDF では各セッションにデッドラインを割り当て，そのデッドラインの早い順にサービスされる．EDF は単一ノードでは全てのスケジューリング方式の中で最大のスケジューラブル領域を持つ [3]．しかし，セッション間の依存性が大きいために，各ノードにおいてセッション毎のサービス曲線を厳密に特定することが困難であり，最悪ケースではサービス曲線をデッドライン曲線と同一と見なさなければならない．複数ノードにおいて，

これは入力トラヒック曲線がその前のノードのデッドライン曲線と同一になることを意味する (入力トラヒック曲線は入力トラヒックの流入量を表す曲線，デッドライン曲線はデッドラインを守るために必要なサービス累積量を表す曲線のことであり，詳細については **2.2** で述べる)．これにより，End-to-End 最大遅延は各ノードでの最大遅延の単純な合計となり，他のスケジューリング方式よりスケジューラブル領域が小さくなる可能性がある．

一方，ORC-GPS は，レートベースでサービスを行い，さらに，各セッションの最大遅延が不必要に増加しないように，各セッションにおいてサービスの制限を行い，サービスレートを変化させる方式である．これにより，ORC-GPS は EDF と同等以上のスケジューラブル領域となる結果が得られている [4]．本稿では，これに加え，数値例を用いて，EDF と GPS の両方式と比較してどの程度 End-to-End 最大遅延を削減できるかを検証する．

2. Leaky Bucket および各定義

2.1 Leaky Bucket

ネットワークに流入するトラヒック特性を保証するために，Leaky Bucket モデル [1] を用いる．

時間 $(\tau, t]$ でのセッション i のネットワークへの流入量 $A_i(\tau, t)$ は以下となる．

$$A_i(\tau, t) \leqq \sigma_i + \rho_i(t - \tau), \quad \forall t \geqq \tau \geqq 0. \tag{1}$$

この制限より，平均レートが ρ_i，最大バースト長が σ_i となる．$A_i(\tau, t)$ が式 (1) に従う場合，$A_i \sim (\sigma_i, \rho_i)$ と表記する．

2.2 各 定 義

時間 $(\tau, t]$ でのセッション i のサービス累積量を $S_i(\tau, t)$，セッション i がデッドラインを守るために必要となる時間 $(\tau, t]$ でのサービス累積量を $D_i(\tau, t)$ と表記する．$A_i(\tau, t)$，$S_i(\tau, t)$，$D_i(\tau, t)$ をそれぞれセッション i の入力トラヒック曲線，サービス曲線，デッドライン曲線と呼び，これらの各曲線を図 1 に示す．デッドライン曲線は，入力トラヒック曲線を許容遅延分だけ遅延させた曲線となる．

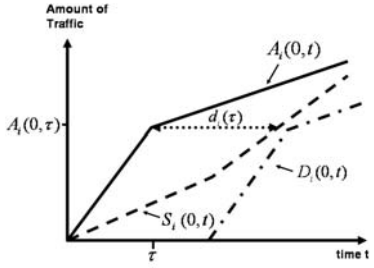

図 1 $d_i(\tau)$，$A_i(0,t)$，$S_i(0,t)$，$D_i(0,t)$

3. ORC-GPS の概要

3.1 サービス制御法

ORC-GPS では，N 個のセッションがサービスされるサーバは，N 個の重み ϕ_i $(i = 1, 2, ..., N)$ で特徴付けられる．さらに，サービスを制限するためのバケット (以下，サービス制限バケットと呼ぶ) を各バッファの後方に導入する (図 2)．ここで，サービス制限バケットの平均レートを ρ_i，最大バースト長を $\sigma_{srv,i}$ $(0 \leqq \sigma_{srv,i} < \sigma_i)$ とする．なお，ここでは全てのセッションは以下の制約を満たすと仮定する．

$$C\frac{\phi_i}{\sum_{j=1}^{N}\phi_j} > \rho_i. \tag{2}$$

セッションが開始した時点で，サービス制限バケットが満杯と仮定し，パケットはサービス制限バケットから当該パケットに相当するトークンを取り出した後に次のノードに流入する．ORC-GPS サーバはサービス制限バケットが空でないセッションを重みに従ってサービスする．サービス制限バケットが空の時は，そのセッションはサービスされず，バッファで待たされる．

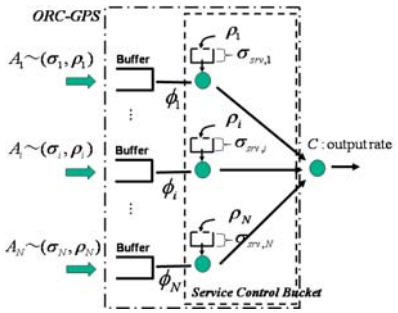

図 2 ORC-GPS

複数ノードでは，セッション i に注目し，そのセッションの経路全体を考慮する．ノード m におけるセッション i の重みを ϕ_i^m，サービス制限バケットの最大バースト長を $\sigma_{srv,i}^m$ $(0 \leqq \sigma_{srv,i}^m < \sigma_i^m)$ と表記する．

(1) セッション i は $A_i^m \sim (\sigma_i^m, \rho_i)$ に従って，ノード m にトラヒックが流入する．

(2) セッション i はその経路において以下が成り立つ．

$$A_i^m(\tau, t) = S_i^{m-1}(\tau, t). \tag{3}$$

ここで，$A_i^m(\tau, t)$ はノード m における時間 $(\tau, t]$ でのセッション i の流入量であり，$S_i^m(\tau, t)$ はノード m における時間 $(\tau, t]$ でのセッション i のサービス量である．

3.2 単一ノードでの最大遅延

全てのセッションが時刻 0 でグリディ (全グリディシステム) となる場合を仮定し，この時のセッション i のサービス曲線 (以下，最悪ケースのサービス曲線と呼ぶ) を $\hat{S}_i(t)$ と表記する．ここで，時刻 0 でグリディとはセッション i の入力トラヒック曲線 $A_i(0, t)$ が $\sigma_i + \rho_i t$ の場合であり，全グリディシステムとは，全てのセッションが時刻 0 でグリディとなる場合である．

全グリディシステムにおいて，x 番目にサービス制限バケットによる制限を受けるセッションをセッション $K(x)$ とし，そのサービス制限バケットによる制限を受ける時刻を t_x とする．また，セッション $K(x)$ の時間 $[t_{y-1}, t_y]$ $(y \leqq x)$ におけるレートを $r_{K(x)}^y$ とする．

以下，セッション i が x 番目にサービス制限バケットによる制限を受けると仮定する．つまり，$i = K(x)$ である．セッション i の最悪ケースのサービス曲線 $\hat{S}_i(t)$ を図 3 に示す．

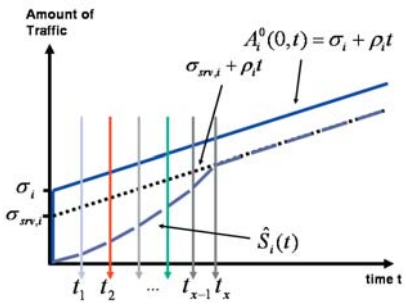

図 3 全グリディシステムにおけるセッション i のサービス曲線 $\hat{S}_i(t)$

セッション i の時間 $[t_{y-1}, t_y]$ におけるレート r_i^y $(i = K(x) \wedge y \leqq x)$ は，セッション $K(l)$ $(l \leqq y - 1)$ が順番にサービス制限バケットにより制限を受けるので以下となる．

$$r_i^y = \frac{\phi_i(C - \sum_{l=1}^{y-1} \rho_{K(l)})}{\sum_{m=1}^{N} \phi_m - \sum_{n=1}^{y-1} \phi_{K(n)}}. \quad (4)$$

また，セッション i は時刻 t_x にサービス制限バケットによる制限を受けるので以下が成立する．

$$\sigma_{srv,i} + \rho_i t_x = \sum_{l=1}^{x} r_i^l (t_l - t_{l-1}). \quad (5)$$

式 (5) より，t_x は以下となり，$x = 1$ から順番に求められる．

$$t_x = \frac{\sigma_{srv,i} - \sum_{l=1}^{x-1} r_i^l (t_l - t_{l-1}) + r_i^x t_{x-1}}{r_i^x - \rho_i}. \quad (6)$$

最悪ケースのサービス曲線 $\hat{S}_i(\tau)$ $(0 \leqq \tau \leqq t_x)$ は以下の傾きと時間のペアのリストで定義される．

$$(r_i^1, z_i^1), (r_i^2, z_i^2), ..., (r_i^x, z_i^x). \quad (7)$$

ここで，z_i^y は時間 $[t_{y-1}, t_y]$ $(y \leqq x \wedge i = K(x))$ である．

式 (7) より構成される曲線を $L_i(\tau)$ とすると，最悪ケースのサービス曲線 $\hat{S}_i(\tau)$ は以下となる．

$$\hat{S}_i(t) = \begin{cases} L_i(t), & \text{for } t \leqq t_x \\ L_i(t_x) + \rho_i(t - t_x), & \text{for } t > t_x \end{cases} \quad (8)$$

セッション i の最大遅延 $\tilde{d}_i$ は，入力トラヒック曲線 $\sigma_i + \rho_i t$ と最悪ケースのサービス曲線 $\hat{S}_i(t)$ の水平距離の最大値となるので，以下で求められる．

$$\tilde{d}_i = t. \text{ ただし，} \sigma_i = \hat{S}_i(t). \quad (9)$$

3.3 End-to-End 最大遅延

End-to-End 最大遅延は次の手順で求められる．まず，各ノードにおいてセッション毎の最悪ケースのサービス曲線を構築する．次に，全ノードにおけるこれらのサービス曲線から，傾きの小さいセグメントを順番に選び，それらを組み合わせて曲線を構築する (以下，この曲線をユニバーサルサービス曲線と呼ぶ)．ユニバーサルサービス曲線より，各セッションの End-to-End 最大遅延が求められる．

始めに，各ノードにおける最悪ケースの入力トラヒック曲線の最大バースト長を求める．式 (3) より，各ノードにおける入力トラヒック曲線の最大バースト長はその前ノードにおけるサービス曲線に依存するので以下が成立する．

$$\sigma_i^m = \sigma_{srv,i}^{m-1}. \quad (10)$$

ノード m において，セッション i が x^m 番目にサービス制限バケットによる制限を受けるものとする．つまり，$i = K^m(x^m)$ である．最悪ケースのサービス曲線 $\hat{S}_i^m(\tau)$ $(0 \leqq \tau \leqq t_{x^m})$ は以下のペアのリストとなる．

$$(r_i^{1,m}, z_i^{1,m}), (r_i^{2,m}, z_i^{2,m}), ..., (r_i^{x^m,m}, z_i^{x^m,m}). \quad (11)$$

さらに，ノード 1 から K_i までのペアの集合 $E_i^{K_i}$ は以下となる．

$$E_i^{K_i} = \bigcup_{m=1}^{K_i} \bigcup_{j=1}^{x^m} \{(r_i^{j,m}, z_i^{j,m})\} \quad (12)$$

各セッションは最終的に経路中で最も小さい最大バースト長をもつサービス制限バケットによって制限をされる．式 (10) より，サービス制限バケットの最大バースト長 $\sigma_{srv,i}$ の最も小さいノードは，経路中の最後のノードであり，ユニバーサルサービス曲線 $U_i(t)$ は以下で定義される．

$$U_i(t) = min\{G_i^{K_i}(t), \sigma_{srv,i}^{K_i} + \rho_i t\}. \quad (13)$$

ここで，K_i はセッション i の経由するノード数である．また，$G_i^{K_i}(t)$ は以下の手順により求めることができる．

（1） $G_i^{K_i}(0) = 0$，$Glist = \phi$ とする．

（2） $E_i^{K_i}$ から傾きの最も小さい要素 $e^{new} = (r^{new}, z^{new})$ を取り出し，$Glist$ へ挿入する．これを $E_i^{K_i}$ の要素がなくなるまで繰り返す．

（3） $Glist$ の要素から，$G_i^{K_i}(t)$ を区分線形凸関数として定義する．

End-to-End 最大遅延 $\bar{d}_i$ は，入力トラヒック曲線 $\sigma_i + \rho_i t$ とユニバーサルサービス曲線 $U_i(t)$ の水平距離の最大値であり，以下で求められる．

$$\bar{d}_i = t. \text{ ただし，} \sigma_i = U_i(t). \quad (14)$$

4. 従来方式との比較

4.1 準　　備

[4] では，単一ノードにおいて各セッションの最大遅延を小さくするために，全てのセッション i についてそのサービス量が σ_i に達すると同時にサービス制限バケットの制限を受ける場合を仮定した (図 4)．この場合のサービス制限バケットの最大バースト長を $\sigma_{srv,i}^*$ (以下，単一ノード最適値と呼ぶ) と表記すると以下が成り立つ．

$$\sigma_{srv,i}^* = \sigma_i - \rho_i t_x^* \text{ and } t_x^* = \tilde{d}_i. \quad (15)$$

また，この時のセッション i の最大遅延 $\tilde{d}_i$ は式 (16) で求められる．なお，式 (16) において $i = K(i)$ を仮定し，変形させた式が [4] の式 (10) である．

$$\tilde{d}_i = \frac{\sum_{m=1}^{N} \phi_m - \sum_{n=1}^{x-1} \phi_{K(n)}}{\phi_i(C - \sum_{l=1}^{x-1} \rho_{K(l)})} (\sigma_i - \frac{\phi_i}{\phi_{K(x-1)}} \sigma_{K(x-1)}) + \tilde{d}_{K(x-1)}. \quad (16)$$

しかしながら，複数ノードの場合にサービス制限バケットを単一ノード最適値に設定してしまうと，小さくなりすぎ，End-to-End 最大遅延がサービス制限バケットによって制限された状態のときに達成され，大きな値となってしまう可能性がある．最悪の場合，End-to-End 最大遅延が Leaky Bucket の最大バースト長を最低保証レートで割った値となる．

そこで，**4.2** の数値例では，現実的なパラメータを想定し，各ノード m でのサービス制限バケットの最大バースト長 $\sigma_{srv,i}^m$ を $\sigma_i^m > \sigma_{srv,i}^m \geqq \sigma_{srv,i}^{*,m}$ の範囲で変化させ，従来の両方式と比較してどの程度最大遅延を削減できるかを検証する．

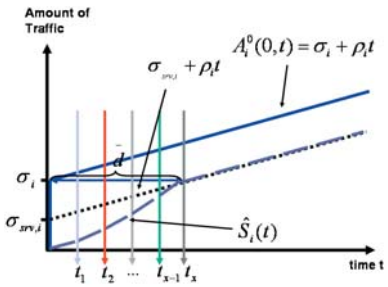

図 4　End-to-End 最大遅延 (サービス量が σ_i に達すると同時にサービス制限バケットの制限を受けるケース)

4.2 数　値　例

ネットワークに 4 つのノードが存在し，各セッションが 2 つのノードを経由する場合について考える．各リンクの帯域は 100Mbps とする．想定するネットワークモデルを図 5 に示し，表 1 に各集約セッションのパラメータを示す．ここでは，同一

のパラメータをもつ複数セッションを一括して集約セッションと呼んでいる．図と表において，AS は集約セッション，N はノードを表す．なお，集約セッション AS0，AS2，AS5，AS8 は蓄積ビデオ型，集約セッション AS1，AS4，AS7，AS9 はオーディオ型，集約セッション AS3，AS6 はビデオ会議型のトラヒックを想定している．

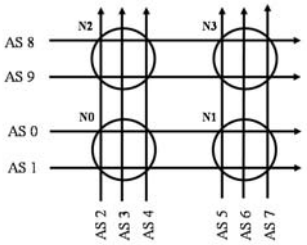

図 5　ネットワークモデル

表 1　パラメータ

AS #	σ Kbits	ρ Mbps	セッション数
0	800	3	1
1	10	0.0064	15
2	800	3	5
3	80	0.5	45
4	10	0.0064	245
5	800	3	5
6	80	0.5	40
7	10	0.0064	100
8	800	3	1
9	10	0.0064	15

表 2　各ノードにおける EDF での最大遅延

AS	最大遅延 (msec)			
#	N0	N1	N2	N3
0	120	100	-	-
1	60	40	-	-
2	120	-	120	-
3	80	-	80	-
4	60	-	60	-
5	-	100	-	100
6	-	60	-	60
7	-	40	-	40
8	-	-	120	100
9	-	-	60	40

各方式の End-to-End 最大遅延は以下の方法で求める．結果を表 3 に示す．

（1） EDF

EDF において，全てのセッション i が遅延 d_i^E 以内でサービスが完了するための条件式は以下である．

$$d_i^E \geq \frac{\sigma_i + \sum_{l=1}^{i-1}(\sigma_l - \rho_l d_l^E)}{C - \sum_{l=1}^{i-1}\rho_l}. \tag{17}$$

ここで，$d_i^E < d_j^E$ である時は，$i < j$ となるように順序付けされているものとする．

式 (17) より，各ノード m における最大遅延のベクトルを 1 つ求める．これを表 2 に示す．また，End-to-End 最大遅延は各ノードの最大遅延の合計となる．

（2） ORC-GPS

サービス制限バケットの最大バースト長 $\sigma_{srv,i}^m$ を単一ノード最適値 $\sigma_{srv,i}^{*,m}$ と等しい値に設定する．式 (16) より，各ノード m において表 2 の EDF における最大遅延を超えないように，各ノードにおける ORC-GPS の重み $(\phi_i^m, ..., \phi_j^m)$ を求める．

次に，各ノード m において重みを上記の値に固定し，サービス制限バケットの最大バースト長 $\sigma_{srv,i}^m$ を $\sigma_i^m > \sigma_{srv,i}^m \geqq \sigma_{srv,i}^{*,m}$ の範囲で変化させる．この範囲における割合を p で表す．$\sigma_{srv,i}^m$ は p を用いて以下で表される．これにより，End-to-End 最大遅延を求める．

$$\sigma_{srv,i}^m = \sigma_{srv,i}^{*,m} + p(\sigma_i^m - \sigma_{srv,i}^{*,m}) \tag{18}$$

表 3　End-to-End 最大遅延 (EDF，ORC-GPS，GPS)

AS #	EDF	ORC-GPS (msec)				GPS
	(msec)	p=0	p=0.3	p=0.5	p=0.8	(msec)
0	220	192	170	178	183	184
1	100	98	69	60	60	60
2	240	196	178	191	200	201
3	160	127	95	95	95	95
4	120	116	83	60	60	60
5	200	167	140	148	161	167
6	120	101	75	68	68	68
7	80	78	55	40	40	40
8	220	151	157	169	189	199
9	100	102	76	60	60	60

（3） GPS

GPS の重みを (2) の ORC-GPS の重みと同じ値に設定し，End-to-End 最大遅延を求める．

表 3 では，p の値にかかわらず ORC-GPS における各集約セッションの最大遅延は EDF より小さくなっている．p=0 の時は集約セッション AS8 を除いて，GPS より大きくなっているが，これは，各ノードにおけるサービス制限バケットの最大バースト長が小さすぎるために，End-to-End 最大遅延がサービス制限バケットによって制限された時に達成され，比較的大きな値になるためである．p=0.5 の時，ORC-GPS における最大遅延が EDF と GPS の両方式より全ての集約セッションで等しいかあるいは小さくなっている．EDF と比較して集約セッション毎の平均で 49msec，最大で 65msec 程小さく，GPS と比較して集約セッション毎の平均で 7msec，最大で 30msec 程小さくなっている．この差は品質に大きな影響を与えるので，ORC-GPS の有用性はかなり高いと考えられる．

5. まとめと今後の課題

本稿では，決定的遅延保証を行うためのスケジューリング方式である Output Rate-Controlled Generalized Processor Sharing (ORC-GPS) と従来の両方式 (GPS と EDF) を比較して，実際のトラヒックを想定した場合に，どの程度 End-to-End 最大遅延が削減できるかを検証した．これにより，ORC-GPS は従来の両方式と比較して有用な性質を持っていることが分かった．

今後の課題として，制約 $C\frac{\phi_i}{\sum_{j=1}^{N}\phi_j} > \rho_i$ を満たさない場合の検証が必要である．また，本論文では流体モデルを前提としているのでパケットを前提とした手法を提案し，ORC-GPS を用いた受付制御の提案が必要である．

文　献

[1] A. Parekh, R. Gallager, “A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case,” *IEEE/ACM Trans. Networking*, pp. 344-357, June 1993.

[2] A. Parekh, R. Gallager, “A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Multiple Node Case,” *IEEE/ACM Trans. Networking*, pp. 137-150, April 1994.

[3] J. Liebeherr, D.E.Wrege, D.Ferrari, “Exact admission control for networks with a bounded delay service,” *IEEE/ACM Trans. Networking*, pp. 885 - 901, Dec 1996.

[4] 花田真樹, 中里秀則, “スケジューラブル領域の最大化を目的とするスケジューリング方式,“ 電子情報通信学会技術研究報告書 (NS2005-122)，pp. 59-62, November 2005.

[5] J.-Y. Le Boudec and P. Thiran, Network Calculus, Springer, 2001.

2.4. 伝送時間測定に基づいたアドミッション制御方式の提案

伝送時間測定に基づいたアドミッション制御方式の提案

Admission control scheme based on measuring transmission time

鎌村 星平　　中里　秀則　　富永 英義
Shohei KAMAMURA　Hidenori NAKAZATO　Hideyoshi TOMINAGA
早稲田大学 大学院 国際情報通信研究科
Graduate School of Global Infomation and Telecommunication Studies, Waseda University

1　はじめに

近年の IP ネットワークの広帯域化に伴い，リアルタイム性を要求するアプリケーションが急増してきた．そこで， End-to-End での QoS (Quality of Service) 保証技術の実現の為には， IP コアネットワーク上での遅延保証技術が必要不可欠となることが予想される．本稿では，遅延保証という観点から，特にユーザとの間の所定の契約を満たす伝送時間を保証するバックボーンアーキテクチャの実現を目的とした提案を行う．

2　基本方針

本稿では，ユーザの要求する伝送時間がパケットに付加された状態で，コアネットワーク上のエッジノードに到着する事を前提とし，その上での制御方式を提案する．

単一の経路に着目すると，要求伝送時間から算出されるパケットのデッドラインに基づいたスケジューリングを適用する事で，ユーザの要求に合わせた伝送時間保証の提供を行う．

また，エッジノードにおいては，複数の経路が選択可能な環境を想定する．ここで，エッジノードにおいて，背景トラヒックの影響によるネットワーク負荷を考慮した，フロー単位での測定ベースによる CAC (Connection Admission Control) 及び，経路割当て制御を実行する事で，高負荷時の可用性を向上させる制御を実行する．

3　提案方式

3.1　単一経路での制御

エッジノードでは，流入パケット p が持つ要求伝送時間 T から，p が各ノードに到着し，送出されるまでの，相対締切時間 $D^i_{relative}$ を決定する．この $D^i_{relative}$ を，エッジノードは制御チャネルを通して，各コアノードへと通知する．各コアノード，及び，出側エッジノードにおいては，それぞれ $D^i_{relative}$ を元に EDF (Earliest Deadline First) スケジューリング [1] を実行することで，要求伝送時間 T の保証を行う．

なお，相対締切時間 $D^i_{relative}$ は以下の通り算出する．フロー F に属するパケット p の要求伝送時間を T，ノード i における遅延重みを w^i_d とすると，ノード i における $D^i_{relative}$ は，

$$D^i_{relative} = T \times w^i_d \tag{1}$$

となる．ここで，遅延重みとは，パス k 上の全ノード n に対して，ノード i 及びノード i に隣接するリンクが占める遅延の比率を意味しており，

$$w^i_d = \frac{d^i_{bb}}{\sum_{k=1}^{n} d^k_{bb}} \tag{2}$$

により求める．ただし，d^i_{bb} は，

$$d^i_{bb} = W_i + 1/\mu_i + d_i \tag{3}$$

で表わされ，それぞれ，リンクの出力バッファにおける平均待ち時間 W_i，リンク速度 μ_i，リンク遅延 d_i により構成される．

3.2　複数経路での制御 (D.P.AC)

複数経路を利用する事による，負荷の影響の軽減を目的とした制御を行う．

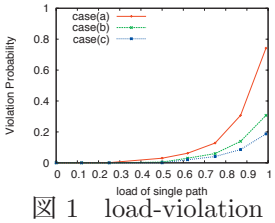

図 1　load-violation

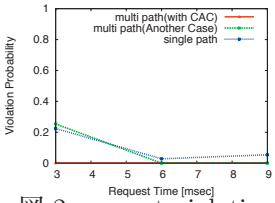

図 2　requet-violation

複数経路の管理は，エッジノードにおいて実行する事で，コアノードに対するオーバヘッドを軽減する．また，経路の選択方法としては，ホップ数をメトリックとした，ダイクストラ法を複数回適用することにより，k 個の経路を決定する．

次に，選択された k 個の経路に対し，フローを効果的に分散させるための制御方式である， D.P.AC (Delay Prediction Admission Control) を実行する．

D.P.AC では，フロー l が，時刻 t^{in} に入エッジノードに到着し，時刻 t^{out} に出エッジノードに到着した際に，フロー l の D.P.(Delay Prediction) 値を，

$$D.P._l = t^{out}_l - t^{in}_l \tag{4}$$

と定義する．t^{out} が，出エッジノードから，入エッジノードへと通知され，入エッジノードでは，それを契機にフロー毎の D.P. 値を算出し，これを時系列に記録する．この D.P. 値を参照することで，新規入力フローに対し，フローの持つ要求伝送時間に対して適当な経路を選択する．ここで適当な経路とは，フローの要求伝送時間に近い D.P. 値を持った経路を言う．

次に，選択された経路において，入力フローが以下の基準を満たす場合，フローの参加を認め，そうでない場合は，拒否する．

1. 要求伝送時間 T が D.P. 瞬時値を満たす
2. 時間間隔 τ の間，D.P. 値が安定している (安定度 α の検定)
3. フローの新規参加による遅延増加分を考慮しても，要求伝送時間を満たす

4　評価

$D_{relative}$ の違反率，及び， D.P.AC の有効性を計算機シミュレーションにより評価する．図 1 に，使用経路数を 1-3 本 (case(a)-(c)) に変化させ，かつ入力フローを経路数に応じて均等に分散させた場合の，同量のネットワーク負荷に対する違反率の特性を示す．高負荷時においても，経路数が多い程違反率が低下する事が分かるが，(a)-(c) において，経路コスト (ホップ数) がそれぞれ異なるケースで同様の評価を行った場合では，複数経路利用の有効性が低下した．そこで，図 2 において，D.P.AC に従った配置を行うケース，均等に配置したケース，単一経路を使用するケースを比較した所，D.P. 値に従うケースにおいて，最も違反率が低下し，提案する D.P.AC の有効性が確認出来た．

5　まとめ

本稿では，契約伝送時間を保証するための，バックボーンアーキテクチャについて提案した．今後の課題として， D.P.AC の特性評価を，実測値に基づいて行う必要がある．

参考文献

[1] M.Andrews, “Probablistic end-to-end delay bounds for earliest deadline first scheduling,” Proc IEEE INFOCOM, 2000.

2.5. スケジューラブル領域の最大化を目的とするスケジューリング方式

スケジューラブル領域の最大化を目的とするスケジューリング方式

花田　真樹†　　中里　秀則†

† 早稲田大学大学院 国際情報通信研究科
〒 367-0032 埼玉県本庄市大字西富田字大久保山 1101-3
E-mail: †hanada@fuji.waseda.jp, nakazato@waseda.jp

あらまし　本稿では，スケジューラブル領域の最大化を目的とするスケジューリング方式である Output Rate-Controlled Generalized Processor Sharing(ORC-GPS) を提案する．スケジューラブル領域とは決定的遅延保証が可能な遅延の範囲のことである．近年，決定的あるいは統計的遅延保証を行うために，Generalized Processor Sharing(GPS) や Earliest Deadline First(EDF) に基づいた多くのスケジューリング方式が提案されている．本稿で提案する ORC-GPS は，GPS と同様にレートベースでサービスを行う．さらに，他のフローの最大遅延に影響を与えないように，そのサービス量の制御を行う．これより，シングルノードでは EDF とほぼ同一のスケジューラブル領域となり，さらに，マルチノードでも EDF より大きなスケジューラブル領域となる．
キーワード　決定的遅延保証，スケジューラブル領域，EDF，GPS

Scheduling Algorithm for Maximizing Schedulable Region

Masaki HANADA† and Hidenori NAKAZATO†

† Graduate School of Global Information and Telecommunication Studies, Waseda University
1101-3 Ohkuboyama, Nishitomida, Honjo-shi, Saitama, 367-0032 Japan
E-mail: †hanada@fuji.waseda.jp, nakazato@waseda.jp

Abstract　In this paper, we propose a scheduling algorithm for maximizing schedulable region, called Output Rate-Controlled Generalized Processor Sharing(ORC-GPS). The schedulable region is defined in terms of delay bounds that can be guaranteed. Recently many packet scheduling algorithms based on Generalized Processor Sharing(GPS) and Earliest Deadline First(EDF) have been proposed in order to guarantee deterministic or statistical delay bound. ORC-GPS is a rate-based scheduling like GPS and controls the output rate using bucket. As a result, ORC-GPS has almost the same schedulable region as EDF in single-node setting and the schedulable region larger than EDF in multiple-node setting.
Key words　Deterministic Delay Guarantees, Schedulable Region, EDF, GPS

1. まえがき

近年，音声や映像を用いたリアルタイムアプリケーションの普及に伴い，パケット交換ネットワークにおいて，決定的 (Deterministic)，あるいは統計的 (Statistical) に遅延を保証することが求められている．決定的遅延保証とは決められた遅延を完全に保証することであり，統計的遅延保証とは決められた遅延をある確率以内で保証することである．

本稿では，スケジューラブル領域の最大化を目的とするスケジューリング方式である Output Rate-Controlled Generalized Processor Sharing(ORC-GPS) を提案する．なお，スケジューラブル領域とは，決定的遅延保証が可能な遅延の範囲を意味する．

決定的遅延保証を行うためには，トラヒックの性質を特徴づけるモデル化とスケジューリング方式が重要な要素となる．

トラヒックの性質を特徴づけるモデル化に関しては，パケット交換ネットワークにおいて，様々なトラヒックが混在する可能性を考慮すると，正確なモデル化は困難な場合が多い．そこで，本稿では代表的なトラヒックシェーピングである Leaky Bucket [2] を用いる．

スケジューリング方式に関しては，ハードリアルタイム [6] や待ち行列 [1], [7] の分野で多くの研究が行われてきたが，近年，高速ネットワークでの性能保証の目的とした研究も多く行われている．Generalized Processor Sharing(GPS) [2], [3] に基づいたスケジューリング方式として，Packetized GPS(PGPS) [2]〜[4], Worst-case Fair Weighted Fair Queueing(WF^2Q) [5] があり，Earliest Deadline First(EDF) [6], [11], [12] に基づいたスケジューリング方式として，Delay Earliest Due Date(Delay-EDD) [8], [9]，Rate-Controlled EDF(RC-EDF) [10] などがある．

ここで，ネットワーク上でフローの経由するノードが 1 つの場合をシングルノード，2 つ以上の場合をマルチノードと呼ぶ．また，横軸に時間，縦軸に到着の累積量とする曲線を到着曲線，縦軸にサービスの累積量とする曲線をサービス曲線と呼ぶ．

GPS [2], [3] では，各フローに重みを割り当て，その重みに

従ってサービスされる．これより，他のフローの振舞いにかかわらず最低サービスレートを保証することが可能となる (以下，フローの分離性と呼ぶ)．しかし，常に重みに従ってサービスされるために，他のフローの遅延を不当に増加させてしまう可能性がある．これが，シングルノードでのスケジューラブル領域を小さくする要因となっており，スケジューラブル領域の最大化の観点からは望ましくない．その反面，フローの分離性より，最悪ケースのサービス曲線をレートベースで予測することが可能であり，マルチノードでは，このサービス曲線を用いることにより，タイトな遅延保証が可能である．タイトな遅延保証を行うことがスケジューラブル領域を大きくすることにつながるが，シングルノードでスケジューラブル領域が最大とならないので，マルチノードでも同様となる．

EDF [8]〜[10] では，各フローにデッドラインを割り当て，そのデッドラインの早い順にサービスされる．一般的に，デッドラインは到着時刻にある一定の遅延を加えた時刻として定義される．EDF はシングルノードでは全てのスケジューリング方式の中でスケジューラブル領域が最大となる [11], [12]．よって，シングルノードではスケジューラブル領域の最大化の観点からは最適である．しかし，デッドラインまでにサービスされることが保証される一方で，フロー間の影響が大きいために，サービス曲線をレートベースで予測することができず，最悪の場合，サービス曲線をデッドライン曲線と同一と見なさなければならない．ここで，デッドライン曲線とは，到着曲線に対して一定の遅延を加えた曲線である．したがって，マルチノードでは，シングルノードでの最大遅延の合計となり，ルーズな遅延保証となる．つまり，シングルノードでは，スケジューラブル領域が最大となるが，マルチノードではルーズな遅延保証となり，スケジューラブル領域の最大化の観点から望ましくない．

本稿で提案する ORC-GPS では，サービス曲線の予測性を高めるために，デッドラインベースではなくレートベースのサービスを行う．また，各フローが他のフローの遅延を不当に増加させないように，そのサービス量を制限する．これにより，シングルノードでは EDF とほぼ同一のスケジューラブル領域となり，さらに，マルチノードでは EDF より大きなスケジューラブル領域となる．

2. Leaky Bucket と各定義

2.1 Leaky Bucket

Leaky Bucket [2] では，トークンは固定レート ρ_i で生成される．パケットはバケットから要求されたトークンを取り出した後にネットワークに流入する．

時間 $(\tau, t]$ で，ネットワークに流入するセッション i の総量 $A_i(\tau, t)$ は以下となる．

$$A_i(\tau, t) \leqq \sigma_i + \rho_i(t - \tau), \quad \forall t \geqq \tau \geqq 0. \tag{1}$$

この制限より，平均レートが ρ_i，最大バースト長が σ_i となり，ネットワークに流入するトラヒック特性を保証することが可能である．また，$A_i(\tau, t)$ が式 (1) に従う時，$A_i \sim (\sigma_i, \rho_i)$ と表記する．また，$l_i(t)$ は，時刻 t における Leaky Bucket のトークンの総量を表す．

2.2 グリディセッション

Leaky Bucket から可能な限りの最大レートで流入することをグリディと呼ぶ．セッション i が時刻 τ でグリディになる場合の流入量 $A_i^\tau(\tau, t)$ は以下となる．

$$A_i^\tau(\tau, t) = l_i(\tau) + \rho_i(t - \tau), \quad \forall t \geqq \tau. \tag{2}$$

また，全てのセッションが時刻 0 でビジー区間が始まり，同時にグリディになる場合を全グリディシステムと呼ぶ．すなわち，全てのセッション i に対して，ネットワークに流入するセッションの総量 $A_i^0(0, \tau)$ が以下の時である．

$$A_i^0(0, \tau) = \sigma_i + \rho_i\tau, \quad \tau \geqq 0. \tag{3}$$

2.3 スケジューラブル領域

N 個のセッションがあり，セッション i の最大遅延を d_i とする．ここで，d_i が満たさなければならない条件をスケジューラブル条件と呼ぶ．遅延ベクトルは $\overrightarrow{d} = (d_1, d_2, ..., d_N)$ で記述され，スケジューラブル領域は遅延ベクトルの集合として定義される．

3. Output Rate-Controlled Generalized Processor Sharing(ORC-GPS)

ORC-GPS は，サービス曲線の予測性を高めるために，デッドラインベースではなくレートベースのサービスを行う．さらに，スケジューラブル領域を最大にするために，他のセッションの遅延を不当に増加させることを防ぐ．

3.1 定　　義

ORC-GPS では，N 個のセッションがサービスされるサーバは，N 個の正の実数 $\phi_i (i = 1, 2, ..., N)$ で特徴付けられる．セッション i は重み $\phi_i (i = 1, 2, ..., N)$ に従ってサービスされる．さらに，サービス量制限のためのバケット (以下，サービス量制限バケットと呼ぶ) を導入する (図 1)．よって，ORC-GPS は，GPS サーバとサービス量を制限するサービス量制限バケットで構成される．ここで，サービス量制限バケットの平均レートを ρ_i，最大バースト長を $\sigma_{srv,i}$ と定義する．

セッションが到着した時点で，Leaky Bucket と同様に，バケットが満杯と仮定し，パケットはバケットから要求されたトークンを取り出した後に次のサーバに流入する．サービス量制限バケットが空の時は，サービスを行わず，バッファで待たされる．以下，サービス量制限バケットの残りのトークンがない場合を「サービス量が制限されている状態」と呼び，サービス量制限バケットの残りのトークンが残っている場合を「サービス量が制限されていない状態」と呼ぶ．サービス量が制限されている状態の時は，サービス量が制限されていない状態の残りのセッションが重み ϕ_i に従ってサービスされる．また，平均レート ρ_i は最低保証レート $C\phi_i / \sum_{j=1}^{N} \phi_j$ より小さいと仮定する．ここで，C はサーバの出力レートである．ORC-GPS のサービス曲線を図 2 に示す．

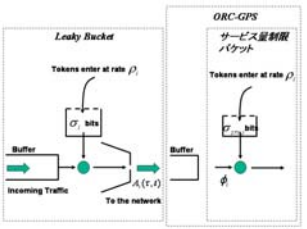

図 1　ORC-GPS

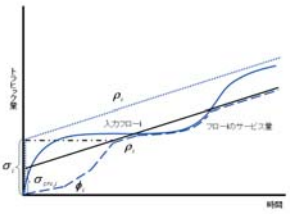

図 2　ORC-GPS のサービス曲線

サービス量制限バケットを用いた場合のサービス量 $S_i(\tau, t)$ は以下に制限される．

$$S_i(\tau, t) \leqq \sigma_{srv,i}(\tau) + \rho_i(t - \tau). \tag{4}$$

ここで，$\sigma_{srv,i}(\tau)$ は時刻 τ におけるサービス制限バケットの残りのトークンである．

3.2 最 大 遅 延

GPS では，各セッションの最大遅延は，全てのセッションが同時にグリディになる場合に達成されることが証明されている [2], [3].

[定理 1]　全てのセッション $i(1, ..., N)$ において，最大遅延が達成されるのは，全てのセッションが同時にグリディになる (全グリディシステム) 場合である．

ORC-GPS でも，GPS におけるサービス量の制限の式が式 (4) となるだけなので，同様に証明可能である．証明は省略する．

全グリディシステムについて考える．サーバの出力レートを C とし，j 番目にサービス量がそのセッションの最大バースト長に達するセッションを j とする．さらに，セッション j が最大バースト長 σ_j をサービスするのに要する時間を d_j とする (図 3)．

$[d_{k-1}, d_k]$ の間にセッション i がサービスされるレートを r_k^i

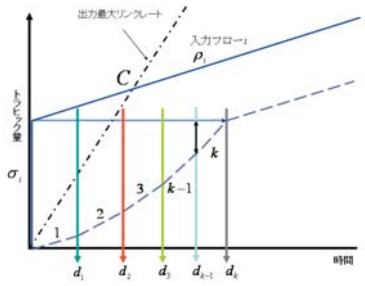

図 3　ORC-GPS の最大遅延

とする．

$[0, d_1]$ では，全てのセッションが最大バースト長に達していないので，r_1^i は以下となる．

$$r_1^i = C\frac{\phi_i}{\sum_{j=1}^{N}\phi_j}. \tag{5}$$

セッション 1 は d_1 で最大バースト長 σ_1 に達するので，$[d_1, d_2]$ の間にセッション i がサービスされるレート r_2^i は以下となる．

$$r_2^i = \frac{(C-\rho_1)\phi_i}{\sum_{l=1}^{N}\phi_l - \phi_1}. \tag{6}$$

再帰的に適用すると以下が成り立つ．

$$r_k^i = \frac{(C-\sum_{l=1}^{k-1}\rho_l)\phi_i}{\sum_{l=1}^{N}\phi_l - \sum_{l=1}^{k-1}\phi_l}, k < j, k = 1, ..., N. \tag{7}$$

また，r_k^k は平均レート ρ_k より大きい必要があるので，以下が成り立つ．

$$\rho_k < \frac{(C-\sum_{l=1}^{k-1}\rho_l)\phi_k}{\sum_{l=1}^{N}\phi_l - \sum_{l=1}^{k-1}\phi_l}. \tag{8}$$

平均レート $\rho_{L(k)}$ に関して式 (8) に従う順序を Feasible Ordering と呼ぶ．

セッション i が最大バースト長 σ_i に達する時間 d_i を求める．セッション i については以下が成り立つ．

$$\sigma_i = \sum_{k=1}^{i} r_k^i(d_k - d_{k-1}). \tag{9}$$

これを解くと以下で与えられる．

$$d_i = \sum_{k=1}^{i}\frac{\sum_{l=k}^{N}\phi_l}{(C-\sum_{l=1}^{k-1}\rho_l)\phi_k}(\sigma_k - \frac{\phi_k}{\phi_{k-1}}\sigma_{k-1}). \tag{10}$$

4. ORC-GPS と EDF のスケジューラブル領域の比較

4.1 シングルノードにおけるスケジューラブル領域の比較

ORC-GPS の最大遅延は式 (10) で与えられ，EDF の最大遅延のスケジューラブル条件は以下の式で与えられる．

$$d_j \geqq \frac{\sigma_j + \sum_{i=1}^{j-1}(\sigma_i - \rho_i d_i)}{C - \sum_{i=1}^{j-1}\rho_i}. \tag{11}$$

スケジューラブル領域の比較に関して，EDF が最適であることを考慮すると，ORC-GPS が EDF と全く同一のスケジューラブル領域を持つことは困難である．そこで，以下の定理を示す．

［定理 2］ シングルノードでは，ORC-GPS は EDF とほぼ同等のスケジューラブル領域を持つ．

([定理 2] の証明)

EDF の最大遅延のスケジューラブル条件式 (11) に，ORC-GPS の最大遅延の式 (10) を代入し，セッション i の重み ϕ_i を決定することができれば，ORC-GPS は EDF と同一のスケジューラブル領域を持つことになる．

式 (11) に式 (10) を代入すると以下となる．

$$\sum_{k=1}^{i}\frac{\sum_{l=k}^{N}\phi_l}{(C-\sum_{l=1}^{k-1}\rho_l)\phi_k}(\sigma_k - \frac{\phi_k}{\phi_{k-1}}\sigma_{k-1}) \geqq \frac{\sigma_i + \sum_{n=1}^{i-1}(\sigma_n - \rho_n d_n)}{C - \sum_{n=1}^{i-1}\rho_n}. \tag{12}$$

ここで，d_n は以下である．

$$d_n = \sum_{k=1}^{n}\frac{\sum_{l=k}^{N}\phi_l}{(C-\sum_{l=1}^{k-1}\rho_l)\phi_k}(\sigma_k - \frac{\phi_k}{\phi_{k-1}}\sigma_{k-1}).$$

式 (12) を満たすセッション i の重み ϕ_i を求める．

式 (12) を展開し，式 (12) の両辺に $\phi_1 \cdot\cdot \phi_i C(C-\rho_1)(C-\rho_1-\rho_2) \cdot\cdot (C-\rho_1-\rho_2-\cdot\cdot-\rho_{i-1})$ を掛ける．それを整理すると以下となる．

$$\phi_1 \cdot\cdot \phi_{i-2} C(C-\rho_1) \cdot\cdot (C-\rho_1-\cdot\cdot-\rho_{i-2}) \times \sum_{l=i+1}^{N}\phi_l(\phi_{i-1}\sigma_i) \geqq 0. \tag{13}$$

ここで，任意の i について，$C - \sum_{l=1}^{i-2}\rho_l > 0$，$\sigma_i > 0$ とすると

$$\phi_1 \cdot\cdot \phi_{i-1}\sum_{l=i+1}^{N}\phi_l \geqq 0. \tag{14}$$

式 (14) の不等号に関しては，式を満たすセッション i の重み ϕ_i を決定することは可能である．しかし，式 (14) の等号に関しては，重みが 0 となるセッションが存在することとなり，GPS の重みとして現実的ではない．しかしながら，以下の近似的な重みにより実現可能である．

$$\phi_i >> \phi_l, l = i+1, ..., N. \tag{15}$$

□

4.2 マルチノードにおけるスケジューラブル領域の比較

シングルノードでは，定理 2 より，ORC-GPS は EDF とほぼ同一のスケジューラブル領域を持つ．また，EDF の最大 End-to-End 遅延はシングルノードでの最大遅延の合計になるので，ORC-GPS の最大 End-to-End 遅延は，最悪でも EDF の最大 End-to-End 遅延と同一になる．

タイトな End-to-End 遅延を導出するためには，セッションの経路全体を考慮する必要がある．そこで，GPS [3] と同様に，セッション i に注目し，セッションの経路全体を考慮したモデルを用いる．ここで，A_j^m をノード m における時間 $(\tau, t]$ でのセッション j の流入量とし，S_j^m をノード m における時間 $(\tau, t]$ でのセッション j のサービス量とする．

（1） ノード m において，注目するセッション i 以外のセッション j は，$A_j^m \sim (\sigma_j^m, \rho_j)$ に従って，トラヒックが流入する．

（2） セッション j はその経路において以下の制約がある．

$$A_i^m = S_i^{m-1}. \tag{16}$$

上記の前提 (1) より，各々のセッションを独立に扱い，前提 (2) より，各々のノード間の依存性を考慮する．GPS では各々のノード m で，注目するセッション i とそれ以外のセッション

j のグリディになるタイミングによって，遅延が異なるので，その最大となる時点を計算する必要がある．これに対し，最大遅延を容易に導出する手法として，ユニバーサル曲線が用いられている．ユニバーサル曲線とは各ノードで全グリディシステムを仮定した場合に求められるサービス曲線のセグメントを全て抽出し，傾きの小さいセグメントから組み合わせて構築される曲線のことである．ORC-GPS に関しても同様の計算が必要なので，最大遅延の導出のためにユニバーサル曲線を用いる．

［定理 3］ マルチノードでは，ORC-GPS は EDF より大きいスケジューラブル領域を持つ．

([定理 3] の証明)

ノード k におけるセッション i の全グリディシステムでのサービス曲線は以下のペアの組み合わせで構成される．

$$(r_1^{i,k}, d_1^{i,k}), (r_2^{i,k}, d_2^{i,k}), ..., (r_{n_k}^{i,k}, d_{n_k}^{i,k}). \tag{17}$$

ここで，$d_j^{i,k}$ はノード k におけるセッション i の j 番目のセグメントの長さであり，$r_j^{i,k}$ はそのセグメントにおけるサービス量である．また，n_k はセグメント数である．さらに，以下が成り立つ．

$$r_1^{i,k} < r_2^{i,k} < ... < r_{n_k}^{i,k} \quad and \quad \sum_{j=1}^{n_k} d_j^{i,k} = d_i^k. \tag{18}$$

ここで，d_j^i はノード k におけるセッション i のセグメントの長さの合計である．ノード 1 から k までの $(r_j^{i,m}, d_j^{i,m})$ の集合 $E^{i,k}$ は以下である．

$$E^{i,k} = \bigcup_{m=1}^{k} \bigcup_{j=1}^{n_k} \{(r_j^{i,m}, d_j^{i,m})\}. \tag{19}$$

ユニバーサル曲線は集合 $E^{i,k}$ から傾きの小さいセグメントを順番に組み合わせて構成される．

各々のノード m でのサービス曲線を構成するペア $(r_{n_m}^{i,m}, d_{n_m}^{i,m})$ について以下が成り立つ．

$$\sum_{j=1}^{n_m} d_j^{i,m} = d_i^m \quad and \quad \sum_{j=1}^{n_m} r_j^{i,m} = \sigma_i^m. \tag{20}$$

また，セッション i の総ノード数が K_i の時，ユニバーサル曲線を構成するペア $(r_{n_m}^{i,m}, d_{n_m}^{i,m})$ について以下が成り立つ．

$$\sum_{m=1}^{K_i} \sum_{j=1}^{n_m} d_j^{i,m} = d_i^1 + d_i^2 + ... + d_i^{K_i}. \tag{21}$$

$$\sum_{m=1}^{K_i} \sum_{j=1}^{n_m} r_j^{i,m} = \sigma_i + \sigma_i^2 + ... + \sigma_i^{K_i}. \tag{22}$$

最大 End-to-End 遅延 d_i^* は，以下に定義される．

$$d_i^* = \max_{\tau \geqq 0} d_i(\tau). \tag{23}$$

ここで，$d_i(\tau)$ は時刻 τ におけるフロー i の遅延であり，ユニバーサル曲線 $U_i(0,t)$ を用いると以下で定義される．

$$d_i(\tau) = t - \tau, \quad U_i(0,t) = A_i(0,\tau). \tag{24}$$

ユニバーサル曲線におけるサービス量が σ_i を超えた場合，入力トラヒックの曲線の傾きは ρ_i となる．式 (21) と式 (22) より，明らかに時刻 $d_i^1 + ... + d_i^{K_i}$ の時点でサービス量が σ_i 以上となるので，最大 End-to-End 遅延 d_i^* は $d_i^1 + ... + d_i^{K_i}$ 以下となる．

□

5. まとめと今後の課題

本稿では，スケジューラブル領域の最大化を目的とするスケジューリング方式である ORC-GPS を提案した．ORC-GPS は，デッドラインベースではなくレートベースのサービスを行い，さらに，スケジューラブル領域を最大にするために，他のセッションの遅延を不当に増加させることを防ぐ手法である．

スケジューラブル領域の比較では，シングルノードでは，EDF と同等のスケジューラブル領域を持ち，さらに，マルチノードでも，EDF より大きなスケジューラブル領域をもつことを示した．これより，ORC-GPS は EDF や GPS と比較して，決定的遅延に関して有用な性質を持っていることが分かった．

今後の課題の 1 つ目として，本稿は流体モデルを前提としているので，パケットを前提とした手法と受付制御の提案がある．2 つ目として，平均レートが最低保証レートより大きい場合を考慮する必要がある．3 つ目として，ネットワーク資源の観点から決定的遅延保証は実用的でないので，統計的遅延保証を行う必要がある．

文　献

[1] L. Kleinrock, Queueing Systems Volumne II, John Wiley & Sons Inc., New York, 1975.

[2] A. Parekh, R. Gallager, “A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case,”IEEE/ATM Trans. Networking, 1(3), pp. 344-357, 1993.

[3] A. Parekh, R. Gallager, “A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Multiple Node Case,” IEEE/ATM Trans. Networking, 2(2), pp. 137-150, 1993.

[4] A. Demers, S. Keshav, and S. Shenkar, “Analysis and simulation of a fair queueing algorihtm,” Internet. Res. and Exper., vol.1, 1990.

[5] J. C. R. Bennect and H. Zhang, “WF^{2}Q:Worst-Case Fair Weighted Fair Queueing.” In Proceedings of INFOCOM’96 , pp. 120-128, San Francisco, CA, March 1996.

[6] C.L. Liu and J.W. Layland, “Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment,” J. Assoc. Comput. Mach., Vol.20, No.1, pp.41-46, Jan 1973.

[7] 花田 真樹，中里 秀則，“非割込み型 EDF スケジューリングの近似解析,”電子情報通信学会論文誌, VOL.J87-A No.12, pp.1518-1527, Dec 2004.

[8] D. Ferrari and D. Verma, “A Scheme for Real-Time Channel Establishment in Wide-Area Networks,” IEEE Journal on Selected Areas in Communications, Vol.8, Issue.3, pp.368-379, April 1990.

[9] D. Ferrari and D. Verma, “On the ability of establishing real-time channels in point-to-point packet-switched networks,” IEEE Transactions on Communications, Vol.42, Issue.234, pp.1096-1105, February-April 1994.

[10] L. Georgiadis, R. Guerin,V. Peris, K.N.Sivarajan, “Efficient network QoS provisioning based on per node traffic shaping,” IEEE/ACM Transactions on Networking, Vol.4, Issue.4, pp.482 - 501, Aug 1996

[11] L. Georgiadis, R. Guerin, A. Parekh, “Optimal multiplexing on a single link: delay and buffer requirements,” 13th Proceedings IEEE INFOCOM ’94, vol.2, pp.524 - 532, June 1994.

[12] J. Liebeherr, D.E.Wrege, D.Ferrari, “Exact admission control for networks with a bounded delay service,”Networking, IEEE/ACM Transactions on , Vol.4, Issue: 6, pp.885 - 901, Dec 1996.

2.6. マルチパス環境における EDF スケジューリングの適用に関する検討

マルチパス環境における EDF スケジューリングの適用に関する検討

鎌村　星平†　　中里　秀則†　　富永　英義††,†

† 早稲田大学 大学院 国際情報通信研究科 〒 367-0032 埼玉県本庄市西富田大久保山 1101
†† 早稲田大学 理工学部 コンピュータ・ネットワーク工学科 〒 169-8555 東京都新宿区大久保 3-4-1
E-mail: †{sho,tominaga}@tom.comm.waseda.ac.jp, ††nakazato@waseda.jp

あらまし　本稿では，IP ネットワーク上での伝送時間保証を目的とした，経路制御手法を提案する．本提案では，複数の経路が存在する環境下において，背景トラヒックの影響によるネットワーク負荷を考慮した，フロー単位での測定ベースによる CAC (Connection Admission Control) を実行する．さらに，単一の経路上においては，パケットのデッドラインに基づいたスケジューリングを適用し，ユーザが要求する伝送時間の保証という観点での評価を行う．
キーワード　遅延保証，EDF (Earliest Deadline First)，複数経路,CAC (Connection Admission Control)

Applying Earliest Deadline First Scheduling to Multi-Path Network

Shohei KAMAMURA†, Hidenori NAKAZATO†, and Hideyoshi TOMINGAGA††,†

† Graduate School of Global Information and Telecommunication Studies，Waseda University
1101 Okuboyama Nishi-Tomida Honjo-Shi, Saitama, 367-0032 Japan
†† Department of Computer Science Engineering，Waseda University
3-4-1 Ohkubo，Bldg.55-N-06-02，Shinjuku-ku，Tokyo，169-8555 Japan
E-mail: †{sho,tominaga}@tom.comm.waseda.ac.jp, ††nakazato@waseda.jp

Abstract　In this paper, we propose a routing scheme for to guarantee delay over IP networks.In our proposed scheme,we perform a measurement-based Connection Admission Control(CAC) per flow considering network load with cross traffic in multi-path environment.In addition, packets are scheduled by Earliest Deadline First(EDF) scheduler on each path. We evaluate our proposed scheme in terms of guarantee in requested transmission time.
Key words　Guarantee of Delay，EDF (Earliest Deadline First)，Multi-Path，CAC (Conncetion Admission Control)

1. はじめに

近年の IP ネットワークの広帯域化に伴い，リアルタイム性を要求するアプリケーションが急増してきた．さらに，今後もアクセスネットワークが広帯域化していく事を考慮すると，End-to-End での QoS (Quality of Service) 保証技術の実現の為には，バックボーンネットワーク上での遅延保証技術が必要不可欠となることが予想される．そこで本研究では，遅延保証という観点から，特にユーザとの間の所定の契約を満たす伝送時間を保証するバックボーンアーキテクチャの実現を目的とする．

IP ネットワークにおいて End-to-End の遅延を保証する技術の一つとして，様々なスケジューリング方式が提案されている．その中でも代表的な手法としては， WFQ (Weighted Fair Queuing) のように，重み付けラウンドロビン方式で実行される， GPS (Generalized Processor Sharing) 方式，フロー毎の締切り時刻を優先度として扱う EDF (Earliest Deadline First) スケジューリング方式と大別できる [1]．前者は，遅延保証の観点では，必ずしも最適では無いスケジューリング方式であるのに対して，後者はノードにおける最大遅延を保証するという意味で，遅延保証の観点で最適なスケジューリング方式と言われている [2]．そこで，遅延保証を実現するには， EDF スケジューリングを利用した統計的保証サービスが有効であると考えられる．

一方で， EDF スケジューリングをマルチホップ環境で適用する事を考えると，背景トラヒックの影響により，各ノードにおけるホップ毎の遅延保証は，最大遅延の保証までを見積る必要があるため，シングルホップの場合のように，必ずしも最適なスケジューリング方式とはならない．従来の RIP (Routing Information Protocol) や， OSPF (Open Shortest Path First) のような最短経路選択型のルーチングアルゴリズムでは，フローは特定の経路に集約されてしまう可能性が高くなり，背景トラヒックの影響を大きく受ける EDF スケジューリングとの相性は良くない．一方で， MPLS (Multi Protocol Label Switching) トラヒックエンジニアリングで見られるように，エッジルータでフロー管理を行う手法では，複数の経路をパスという概念で扱うことにより，フローが特定の経路に集中することを防ぐことが可能である．

そこで，本研究では，背景トラヒックの影響を軽減させるた

めに，複数の経路を利用することを前提とする．この時，ユーザとの契約により，パケットに付加された要求伝送時間に応じて，対象とするフローに最適となる経路を選択する CAC (Connection Admission Control) 方式を検討する．この際の，CAC 制御方式の制御基準としては，

- フローを複数経路に分散させる
- 経路上で予想される伝送時間がフローの要求伝送時間に近い経路を選択する
- 新規フロー参加による遅延増加の影響を考慮する

とする．

本提案では，フローの経路への割り付けといった管理は，エッジルータで実行する一方，あるフローに注目すると，単一の経路上で EDF スケジューリングが実行される．こうすることで，上述したように， EDF スケジューリングを事実上，複数経路上で分散させて実行することで，契約伝送時間の違反率を低下させると同時に，マルチパスの制御・管理はエッジルータに集約することが可能となるため，遅延保証を実現すると共に，スケーラビリティの確保が容易となることが予想される．

以下，2. 章で本研究に関連する技術，及び研究について述べ，3. 章で提案方式について説明する．4. 章で，提案方式のシミュレーション評価を行い，5. 章で本研究のまとめと今後の課題を述べる．

2. 関 連 技 術

2.1 統計的保証サービス

従来の IntServ (Integrated Services)・RSVP (Resource ReSerVation Protocol) を用いた保証では，遅延保証の実現は可能であるが，スケーラビリティの確保という点では不十分である．一方で，相対的な保証を提供する DiffServ (Differentiated Services) では遅延保証の観点では十分な保証を提供出来ない．

これらのアーキテクチャに対して，本提案で用いる EDF スケジューリングでは，統計的保証という観点で，サービス保証を提供する．統計的保証では，セッション i に対し，遅延境界を T_i，違反率 δ_i，セッション i に属するパケットの総遅延を $Delay\ of\ Packet\ P$ とした時，

$$Pr[Delay\ of\ Packet\ P > T_i] \leqq \delta_i \qquad (1)$$

を実現することを目標とする．すなわち，決められた遅延を完全に保証するのでは無く，ある確率以内で保証する技術と言える．これは，先に述べたアーキテクチャと比較して，遅延保証の観点で有効であると考えられると同時に，ある程度のスケーラビリティの確保が可能となる．ただし，背景トラヒックが存在する実ネットワークを想定した場合，効率的な保証を実現する為には，次に述べる CAC 制御方式と組み合わせることが必要となる．

2.2 Connection Admission Control(CAC)

ネットワークリソースが有限である場合，ネットワークが無制限にフローの接続要求を受け付けると，新規接続フローの QoS 要求を満たせないと同時に，既存のフローの QoS 許容レベルも低下してしまう．そこで，このような場合には新規フローの接続要求を破棄し，既存フローの要求を保護するといった制御が必要となる．この制御を，コネクションアドミッションコントロール (CAC) 制御と呼ぶ．

CAC 制御方式においても，幾つかの手法が提案されている [4]．中でも，エンドポイントにネットワーク内部状態を推定するための機能を持たせ，CAC を実行するエンドポイント方式がスケーラビリティの観点で有効と考えられている．エンドポイント方式は，パケットの観測により，ネットワークの内部状態を推定するパッシブ方式 [5] と，新規参加フローと同程度のレートでプロービングを行うことで，ネットワーク内部状態を推定するアクティブ方式 [6] とに大別できる．

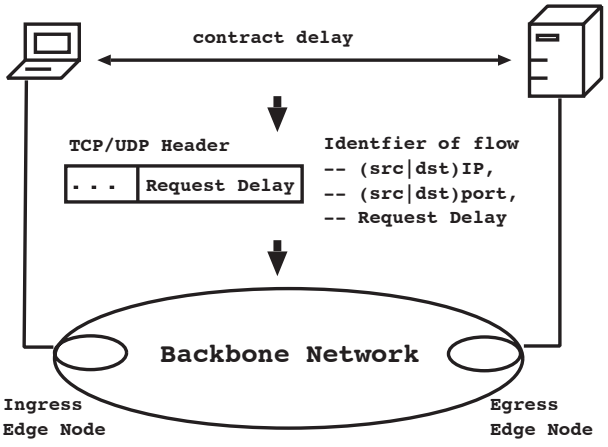

図 1 提案フレームワーク

前述したように，マルチホップ環境下での EDF スケジューリングは，背景トラヒックの影響を大きく受けしまう．そこで，こういった CAC 制御方式と組合せることで，フロー制御をマルチパス環境へと展開し， EDF スケジューリングのスケジューラビリティを向上させることで，統計的保証が，より効果的に行えると考えられる．

今回の提案では，複数ある CAC 制御方式の中でも，バックボーンネットワーク上でのスケーラビリティの確保ということを考慮して，エンドポイントにおける CAC 制御方式を想定した提案を行う．

2.3 関 連 研 究

シングルホップ/シングルパス環境下での EDF スケジューリングについては，幾つか提案がされているが [2]，前述した通り，実ネットワークにおいては背景トラヒックの影響を大きく受けてしまうため， End-to-End での保証という意味では，効果的な伝送時間保証が困難となってしまう．

一方で，マルチパス環境下での EDF スケジューリングの適用については, [3] がある．この検討方式では，ホップ毎にルーチングテーブル上に集約された遅延パラメータを用いることで，宛先ノードまでの遅延予測を行う．この際に，パケットのデッドラインを守れる範囲で迂回経路を柔軟に選択することで，負荷の分散を図っている．よって，シングルパス上での EDF スケジューリングと比較して，高負荷時までの遅延保証が可能となる．一方で，各ノードにおいて，複数経路を利用するための遅延予測ルーチングテーブルの構築が必要となり，コアルータでの制御が複雑となるという意味で，スケーラビリティの確保が困難であると考えられる．

本提案では，これらの手法と比較して，マルチパスを利用することでスケジューラビリティを向上させると共に，複数経路利用の制御をエッジノードに集約することによって，スケーラビリティを確保することが可能となることを目指す．

3. 提 案 方 式

3.1 提案フレームワーク

提案方式のフレームワークを図 1 に示す．まず，伝送時間の保証を必要とするエンドポイント間において，伝送時間に関する契約を行う．契約済みのフローに関しては， TCP/UDP ヘッダのオプション部を利用して，要求伝送時間値を付加し，バックボーンネットワーク上のエッジノードへと転送される．エッジノードでは，レイヤ 4 ヘッダ情報からフローを識別し，さら

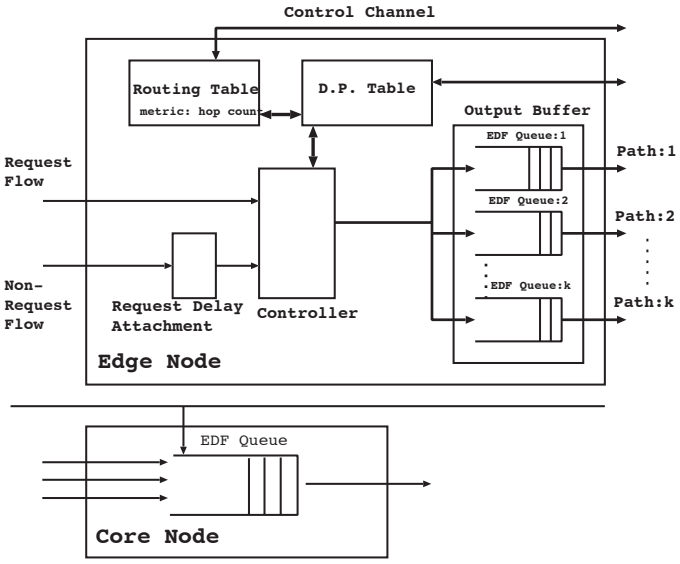

図 2　ノード構成

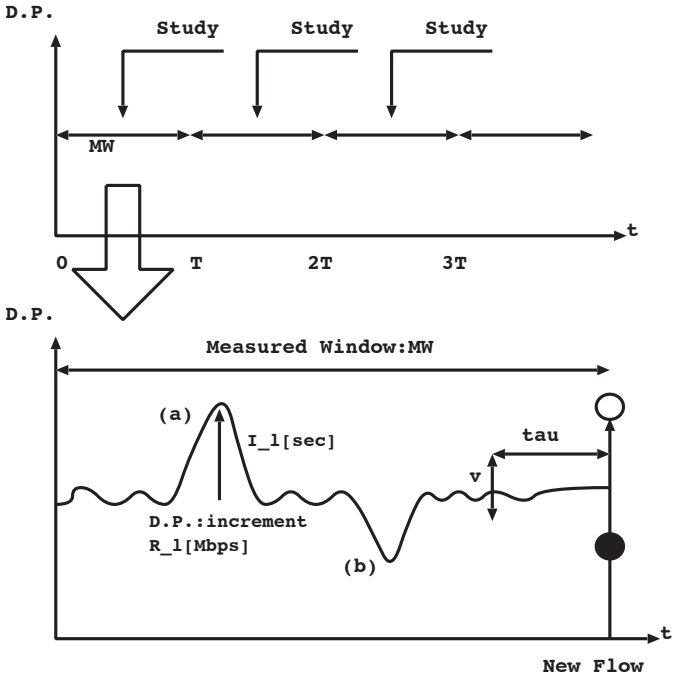

図 3　D.P.(Delay Prediction) 値

に，フローの要求伝送時間値を元に，各コアルータにおける転送デッドラインの計算，及びコアルータへのデッドライン通知を行う．

ここで，提案するフレームワークを実現するためには，以下の項目を検討する必要がある．

- エンドポイント間における伝送時間保証の契約に関して，要求/応答を実行するプロトコル
- バックボーンネットワーク上での伝送時間保証アーキテクチャ

今回の提案では，前者を簡略化し，エッジルータに要求伝送時間が付加されたフローが到着することを前提とし，後者の実現に焦点を当てて検討を行う．

3.2　ノード構成

図 2 にバックボーンネットワーク上でのノード構成を示す．エッジノードにおいて，入力フローは，要求伝送時間を持ったフローと，そうでないフローとが到着する．ここでは，要求伝送時間を持っていないフローに対して十分大きな要求伝送時間を付加することで， EDF スケジューラに対応させる．マルチパスの管理に関しては，ルーチングプロトコルにより，ホップ数をメトリックとして，ダイクストラ法を複数回適用することにより，k 個のパスを決定する．決定されたパス情報は，パスの D.P.(Delay Prediction) 値とフロー識別子と共に，D.P. テーブルに保存される．D.P. 値については，次節で解説する．k 個のパスが，それぞれ，k 個の EDF キューに対応しており，入力フローは，自身の要求伝送時間と，D.P. テーブル参照値を元に，コントローラによって，適当な EDF キューへと送信される．

一方コアノードにおいては，自身のルーチングテーブルを参照して，入力パケットを適切な次ノードへとホップする．この際，コアノードも， EDF スケジューラを装備しており，エッジノードにおいて，各フロー毎の要求伝送時間から算出された，デッドライン値を制御チャネルから受けとり，このデッドラインに応じた EDF スケジューリングを実行する．

なお，デッドライン値とは，フローの要求伝送時間を保証するために，各ノードにおいてパケットを受け取ってから送出するまでの相対締切時間の事であり，以下の通り算出する．フロー F の要求伝送時間を T'，パス k に対して，k 上の全ノード数を n，ノード i から送信されるリンク遅延を L_i とすると，フロー F の，ノード i におけるデッドライン値 d_i は，

$$d_i = \frac{T'}{n \times L_i} \tag{2}$$

によって算出する．

3.3　Delay Prediction Admission Control

本節では，マルチパス環境において EDF スケジューリングを実行するにあたって，背景トラヒックの影響を考慮した，パケットの測定ベースによるアドミッションコントロール制御方式を提案する．

フロー l が，時刻 t^{in} に入エッジノードに到着し，時刻 t^{out} に出エッジノードに到着したとする．この時，フロー l の D.P.(Delay Prediction) 値を，

$$D.P._l = t_l^{out} - t_l^{in} \tag{3}$$

と定義する．t^{out} が，出エッジノードから，入エッジノードへと通知され，入エッジノードでは，それを契機にフロー毎の D.P. 値を算出し，これを時系列に保存する．新規入力フローは，この D.P. 値を参照することで，自身の要求伝送時間に適当なパスを選択する．ここで適当なパスとは，フローの伝送時間要求に近い D.P. 値を持ったパスを言う．

次に，選択されたパス k において，入力フローが以下の基準を満たす場合，フローの参加を認め，そうでない場合は，拒否する．

（1） 要求伝送時間が D.P. 瞬時値を満たす

（2） 時間間隔 τ の間，安定度 α が一定値である

（3） フローの新規参加による遅延増加分を考慮しても，要求伝送時間を満たす

図 3 を元に，これら各手順の詳細について解説する．

3.3.1　D.P. 瞬時値の参照

今，時刻 t において，新規フロー l がパス k に参加する場合を考える．提案では，時間間隔 T の間，フローを観測し，D.P. 値を計測するものとする．このとき，$2T \leqq t \leqq 3T$ とすると，フロー l は，時刻 T から $2T$ の間の T 間の D.P. 値を参照する．これは，直前の T 間を観測することで，最新のフロー変動の影響を反映させる事を目的としている．また，要求伝送時間が

D.P. 瞬時値を満たすとは，時刻 t における D.P. 値より大きな値の要求伝送時間であることを意味する．

ここで，D.P. 瞬時値が要求伝送時間を満たすならば，次の工程である，安定度の参照に移行する．

3.3.2 時間間隔 τ の間の安定度 α の参照

上記の D.P. 瞬時値が要求伝送時間を満たす場合，次に，D.P. 値の安定度を検査する．安定度 α とは，時刻 t から，$t-k$ の間における，D.P. 値の変動を示す値で以下の手順により決定する．

```
for (k = 1; k <= τ; k++) {
        if (|DP(t) - DP(t - k)| < v) {
                variation_ok += 1;
        }
}
α = variation_ok / τ;
```

ここで，v は，D.P. 値の変動に対する閾値である．α の判定は，一時的に，D.P. 値が極端な値を示している状態を，新規参加フローが参照してしまう現象を防ぐためである．例えば，図 3 において，時刻 t における D.P. 値が，(b) の部分であった場合，一時的に下に変動している値を参照して，フローの参加を決定してしまい，参加の判断基準が甘いことになる．逆に，上に変動している部分 (a) を参照してしまうと，今度は，参加の判断基準が厳しくなってしまう．そこで，一定時間の間，安定した D.P. 値を出力していることを判断基準に加えることで，アドミッション制御の誤判断を防ぐことを目的とする．この α の値が，一定値以上であれば，次の工程である，参加フローの要求レートによる遅延増加の見積りへと，移行する．

3.3.3 参加フローの要求レートによる遅延増加の見積り

上記までの判定により，過去に発生したフローの伝送時間履歴情報である D.P. 値から，新規参加フローの要求伝送時間を満たす可能性の高いパスを決定する．ただし，この時に新規フロー自体が参加することで，対象としているパスの遅延が増加する可能性があるため，新規フロー自身の影響を考慮する必要がある．

そこで，参加フロー自身による遅延増加の影響を考慮した制御法を，以下の通りに定義する．まず，対象としているパス k において，流入フローの増加をトリガとして，D.P. 値が増加した場合 (図 3(a) のポイント) の，D.P. 値増分と，その時の入力フロー l のレートを記録する．このフロー l を参照フローとする．この時，パス k における，新規参加フローの要求レート R_{in}[Mbps]，参照フロー l によるパス k の D.P. 値の増分を I_l^k[ms]，参照フロー l の要求レートを R_l[Mbps] とした時，予測される D.P. 値の増分 ε は，

$$\varepsilon = \frac{I_l^k}{R_l} R_{in}[ms] \qquad (4)$$

となる．この予測される D.P. 値の増分 ε が，許容範囲に入っているならば，フロー l の参加を許可する．

以上，(1)〜(3) を満たしている場合，新規参加フローは，このパスへの参加を許可され，そうでない場合は参加を拒否される．参加を拒否されたフローは，次に D.P. 値が，自身の要求伝送時間に近いフローを持つパスに対して，同様に参加を試みる手順を繰り返す．

4. シミュレーション評価

提案手法の有効性を検討するために，計算機シミュレーションによる評価を行った．

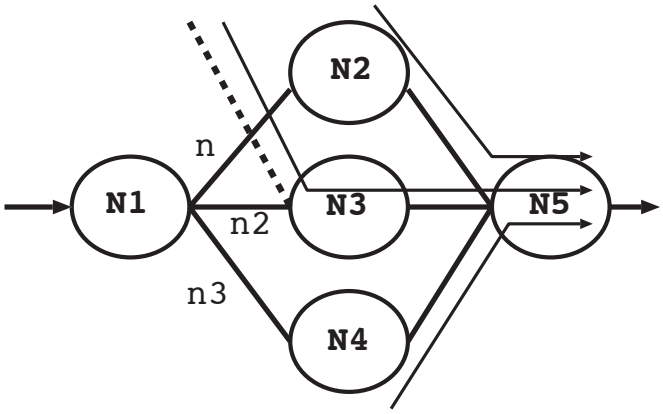

図 4 シミュレーショントポロジ

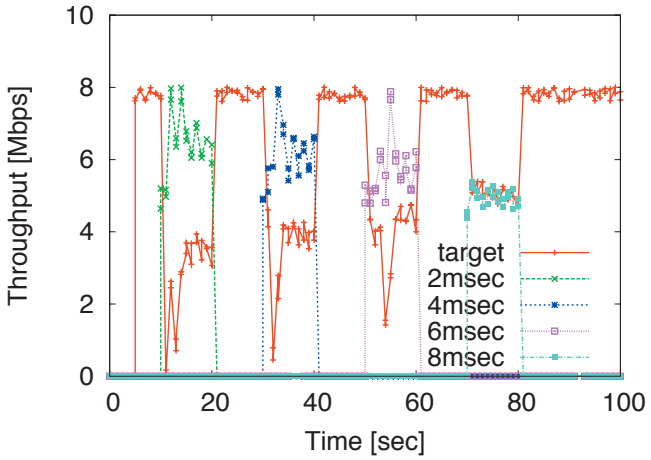

図 5 EDF スケジューラのスループット特性

4.1 前提条件

複数のユーザが，1 台のストリーミングサーバに接続する状況を想定する．提案に基づいて，伝送時間の保証を要求するユーザが，複数の拠点からフローを発生させる状況を考える．ここで，バックボーンネットワーク上では，1 つのフローが 1 つのリンク帯域を圧迫することは無いと想定し，複数のフローの発生によって，ボトルネックリンクが発生するものとする．また各フローは，明示が無い場合は，ポアソントラヒックとして発生させている．

シミュレーショントポロジを図 4 に示す．対象とするフローにおいて，N1 が入エッジノード，N5 が出エッジノードとする．N1-N2，N1-N3，N1-N4 間は，それぞれ，n, n_2, n_3 のホップ数を持つリンクを意味し，各シミュレーションケースにおいて，適宜設定する．さらに，N2-N5，N3-N5，N4-N5 間においては，別ユーザによるクロストラヒックが発生するものとする．ここで，N1-N2-N5, N1-N3-N5, N1-N4-N5 のパスをそれぞれ，ID=1, ID=2, ID=3 のパスと呼ぶ．

また，リンク容量はすべて，10Mbps，リンク遅延 L_i は 1ms とする．提案方式で用いるパラメータに関しては，$T = 100s$, $\tau = 5.0s$, $v = 1.0ms$, $k = 3$,$\varepsilon \leqq 1.0$ とする．

4.2 シングルパス上での評価

シングルパス上において，アドミッション制御を用いないで，EDF スケジューラの特性評価を行った．

図 4 において，$n_1 = 1$ とし，ID=1 のパスを利用する．ノード N1 から，8Mbps の要求レート，8ms の要求伝送時間を持つ

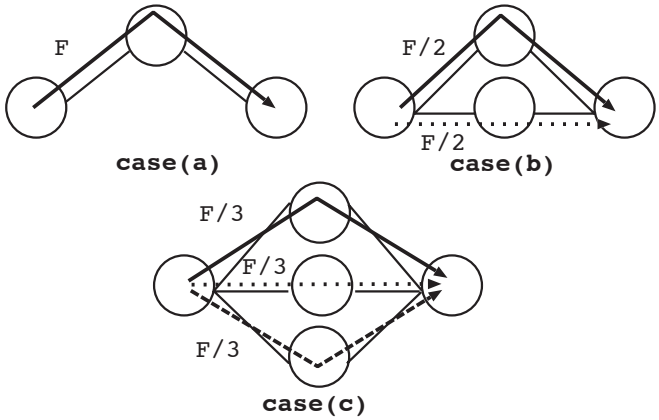

図 6　フロー分散の例

ターゲットフロー (CBR) を発生させつつ，N2-N5 間に，要求伝送時間の異なるクロストラヒックを発生させた．図 5 に，その際のスループット特性を示す．

図から分かるように，要求伝送時間に厳しいフローほど，EDF スケジューラにおいて，優先的にスケジューリングされるため，スループットが向上する．さらに，クロストラヒックの要求伝送時間が，ターゲットフローの要求伝送時間 8ms に近くなるにつれて，同程度のスループットになっていく事が分かる．70 秒から 80 秒の間で発生させたクロストラヒックにおいては，ターゲットフローと同じ要求伝送時間を持っているため，同じデッドライン値が付加される事により，均等にスケジューリングされている事が分かる．

4.3　マルチパス上での評価

次に，マルチパス環境において，アドミッション制御を用いない場合のパフォーマンスを評価する．

$n_1 = n_2 = n_3 = 1$ とした場合の，ネットワーク負荷と違反率の特性を図 7 に示す．ここで，ネットワーク負荷とは，1 つのパスのみ使用した場合の帯域利用率を意味し，クロストラヒックの発生量を変更することで，負荷を変動させる．一方，ターゲットフローは，ノード N1 から 3Mbps の要求レートで発生させ，パスの数に応じて均等に分散させる．フロー分散の例を，図 6 に示す．case(a) はシングルパスのみを利用し，フローを ID1 のパスに流す．対して，このフローを，ID1,ID2 のパスに均等に分散させたのが case(b)，ID1,ID2,ID3 のパスに均等に分散させたのが case(c) である．

また，要求伝送時間は 2-10ms の乱数を与えることで，各要求伝送時間を持つフローが混在することを想定した．

図より，シングルパス上では，負荷が 0.5 の時点で違反が発生していた (3%) のに対し，case(b), case(c) の場合では発生していない．また，負荷が 0.75 付近での違反率は，case(a), case(b), case(c) の場合でそれぞれ，12%, 6%, 4%と，低下していく事が分かる．よって，統計保証における違反率の許容レベル δ を設定した際に，マルチパスの利用によって，より高負荷時まで保証が可能となる事が分かる．

次に，$n_1 = 1, n_2 = 2, n_3 = 3$ とすることで，ホップ数によるコストを，$ID1 < ID2 < ID3$ として評価を行う．これは，第二候補，第三候補として選ばれるパスが，最短経路と同一のホップ数では無い，より現実のネットワークに近い状況を想定した．シングルパスのみを利用する場合と比較して，マルチパスの利用により利用可能帯域が 3 倍となる点では，先の評価と同じであるが，ホップ数の増加によって遅延コストが増加する点で異なる．また，フローの発生条件は先の評価と同じである．この結果を，図 8 に示す．

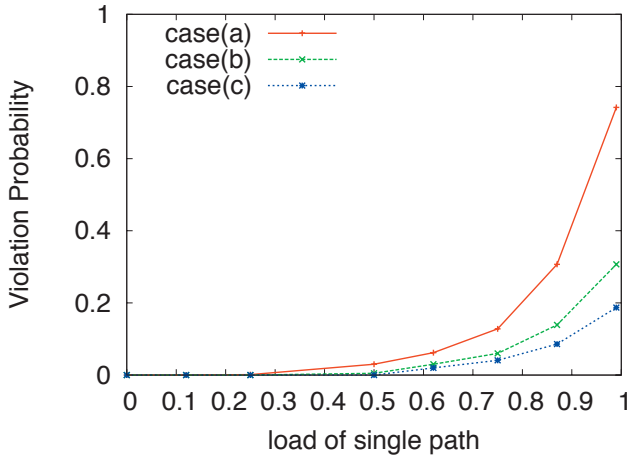

図 7　ネットワーク負荷 - 違反率特性:$n_1 = n_2 = n_3 = 1$

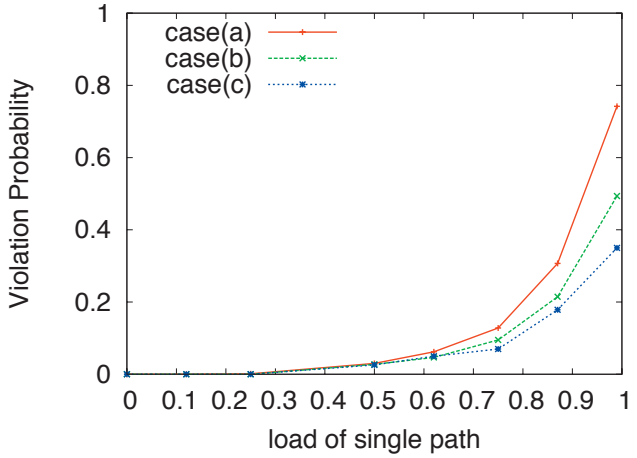

図 8　ネットワーク負荷 - 違反率特性:$n_1 = 1, n_2 = 2, n_3 = 3$

結果より，図 7 の場合と比較して，複数経路化による違反率の改善効果が小さくなっている事が分かる．この原因として，case(c) の場合を考える．case(c) では，ID1 のパスを通るフローにおいては，ネットワーク負荷が減少することで違反率が低下するが，ID2,ID3 のパスを通るフローにおいては，ホップ数の増加による遅延コストが増加してしまうため，逆に違反率が増加する．これらの合計値を算出した結果として，先の結果と比べて違反率が上昇してしまったと予想される．

以上より，複数の経路がすべて同じホップ数であれば，マルチパスの利用により違反率の低下が見込まれる．一方で，より現実的なネットワークを想定した各経路のホップ数が異なる場合では，均等にフローを分散させても違反率の改善効果はそれほど期待できない事が分かった．

4.4　Delay Prediction Admission Control の評価

前節で述べたように，ホップ数の異なるパスに均等にフローを分散させるだけでは，マルチパスの利用による違反率の改善効果が小さくなる．そこで，提案手法である D.P. 値を利用したアドミッション制御手法を用いることで，この問題に対する改善効果の評価を行った．なお，トポロジは図 4 を用い，$n_1 = 1, n_2 = 2, n_3 = 3$ とした．

図 9，10 に，T=100s の間フローを発生させ，その際に測定した D.P. 値の例を示す．図 9 では，ID1 のパスを利用して，10s から，50s までクロストラヒックの数を 10s 間隔で増やし，50s から 90s までクロストラヒックの数を 10s 間隔で減らした

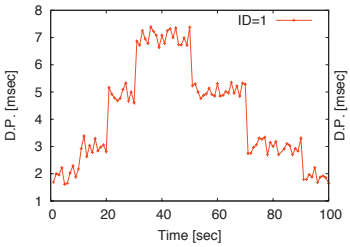

図 9　D.P. 値 (a)

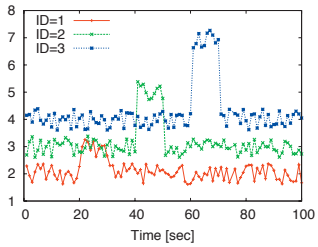

図 10　D.P. 値 (b)

場合の例を示した．これから，ネットワーク負荷の増減に応じて，D.P. 値も増減する事が分かる．また，図 10 では，ID1, ID2，ID3 の各パスに対して，同時にフローを発生させた場合の例を示した．さらに，各パスに対して，一時的にクロストラヒックを発生させる事で D.P. 値の増加も計測した．図 10 から，ホップ数の影響が D.P. 値に反映され，各ホップ数に応じた D.P. 値が測定されている事が分かる．また，40s から 50s の間において，ID2 のパス上でクロストラヒックの影響により負荷が増加している状況では，ID3 のパスよりも大きな D.P. 値となる事が分かる．

以上より，D.P. 値は，

- ネットワーク負荷の変動による遅延変動の影響
- ホップ数による遅延の影響

の二点が反映された値として与えられる事が分かる．

続いて，図 10 の状況でフローが発生していたとして，この後に 3 つのフローが新規に参加する状況を考える．ここで，参加するフローの要求伝送時間はそれぞれ，3ms，6ms，9ms とする．この時に，提案するアドミッション制御に従って D.P. 値に基づいたフロー配置を行った場合，マルチパスを利用しつつも提案とは異なる配置を行った場合，最短経路にすべて配置した場合とを測定した結果を，図 11 に示す．ここで，D.P. 値に従った場合では，要求伝送時間が 3ms，6ms，9ms のフローはそれぞれ，ID1,ID2，ID3 のパスに配置される．一方で，提案と異なる配置では，要求伝送時間が 3ms，6ms，9ms のフローはそれぞれ，ID3,ID2,ID1 のパスへと配置させた．

図 11 より，アドミッション制御を行う場合は，違反率はゼロとなる．一方で，提案と異なる配置では，3ms と，9ms のフローの配置を逆にしただけにもかかわらず，3ms のフローに対して違反が発生する．この場合では，要求伝送時間が 9ms のフローは最短経路 (ID1) を通るため，十分余裕をもってデッドラインを守れる一方，その反動で要求伝送時間が 3ms のフローは遅延コストの最も大きいパス (ID3) を通るため，最短経路のみを利用する場合よりも大きな違反率となる．また，最短経路のみを利用した場合では，ホップ数によるコストは最も小さいが，ネットワーク負荷による影響のため，すべてのフローが違反する結果となった．

結果として，同じネットワーク資源に対しても，フローの配置方法によって違反率に差が出ることが確認できた．提案する D.P. 値を利用したアドミッション制御を用いることで，ネットワーク負荷の変動とホップ数の影響による，パスに対する伝送時間の予測が可能となり，マルチパスを利用した遅延保証の観点で，有効な手法として利用できる事が確認できた．

一方，評価に対する課題として，今回のアドミッション制御に利用した図 10 の D.P. 観測値では，ホップ数の影響しか反映されていないため，図 9 に示したように，ネットワーク負荷が変動する状況においても，最適なパスを選択することを確認する必要がある．また，要求伝送時間が異なるフローがより複雑に混在する状況での評価や，各フローの違反時間の分布の評価

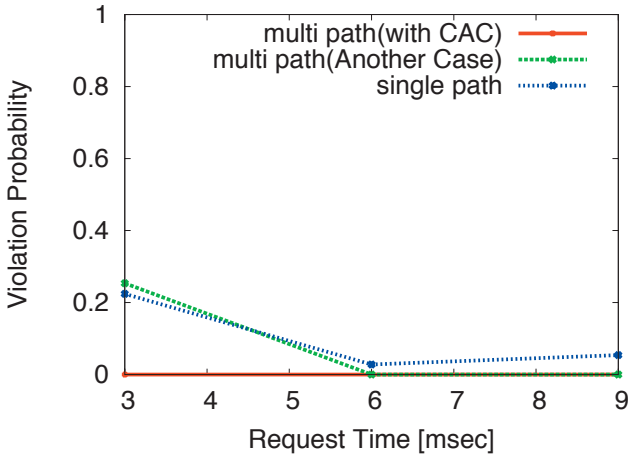

図 11　マルチパス上での要求伝送時間-違反率特性 (2)

なども今後行う必要がある．

上記を踏まえて，フロー変動の予測を行う本提案では，実際に発生したトラヒックパターンにおいても，正常に動作をする事を評価する必要があるため，ストリーミングログのトラヒック発生パターンなどを利用することで，実際のネットワークにも適用できる事を今後確認していきたい．

5. まとめと今後の課題

本稿では，IP ネットワークにおいて，End-to-End 間での契約伝送時間を保証するための，バックボーンアーキテクチャについて提案した．提案では，単一の経路においては，各ノードにおいてパケット単位での EDF スケジューリングを実行し，フロー単位ではマルチパスを利用することで，ある程度の高負荷時においても，統計保証に基づいたソフトな遅延保証を提供する．さらに，対象とするパス上での遅延予測値である，D.P. 値に基づいたアドミッション制御を行うことで，各パスに対して効率的にフローの分散が行われることをシミュレーション評価により示した．

今後の課題としては，事前にフローが発生していない場合でのアドミッション制御方式の検討や，バーストに対する効果的なシェーピング方式の検討，また，考察で述べた課題に関して，特に実際のトラヒックパターンに対する提案方式の有効性について検討を行う必要がある．

文　献

[1] V.Sivaraman, F.M. Chussi, M.Gerla, “End-to-End statistical delay service under GPS and EDF scheduling: A comparison study,” proc IEEE INFOCOM, pp.1113-1122 2001 4.

[2] M.Andrews, “Probablistic end-to-end delay bounds for earliest deadline first scheduling,” Proc IEEE INFOCOM, 2000.

[3] 辻岡，飯田，杉山，村田，“IP ネットワークにおけるデッドラインに基づく優先制御・経路制御法,” 電子情報通信学会論文誌 B Vol.J86-B No.12 pp.2501-2510，2003 12.

[4] 間瀬，“インターネットにおけるスケーラブルなアドミッションコントロール方式,” 電子情報通信学会誌，vol.85 no.9，pp 655-661，2002 11.

[5] C.Cetinkaya and E.W.Knightly, “Egress Admission Control,” Proc. IEEE INFOCOM, 2000.

[6] J.Qiu and E.Knightly, “Measurement-based admission control with aggregate traffic envelopes,” Proc. IEEE ITWDC, 1998 Sep.

2.7. 非割込み型 EDF スケジューリングの近似解析

非割込み型EDFスケジューリングの近似解析

花田　真樹[†]　　中里　秀則[†]

† 早稲田大学大学院国際情報通信研究科
〒 169–0051 東京都新宿区西早稲田 1–3–10 早大 29–7 号館
E-mail: †hanada@fuji.waseda.jp

あらまし　ソフトリアルシステムでは，デッドラインミスが即座に致命的な問題とはならない．しかし，システム性能の平均値，最悪値を把握し，十分な検討しておくことは重要である．リアルタイムスケジューリングの代表的な手法として，$EDF(EarliestDeadlineFirst)$ スケジューリングがある．EDF スケジューリングはデッドラインの早い順に優先順位を付ける非常にシンプルな手法であり，事前に起動時間がわからない非周期タスクに適用可能である．本研究では，タスクの各属性が確率分布に従うものと仮定し，EDF スケジューリングを適用した場合の性能を数学的に解析する．その解析結果は，シミュレーションにより評価する．システムとしては，最も単純な単一サーバモデルを対象とする．仮定するタスクは，発生はポアソン，相対デッドライン時間は指数分布，実行時間は相対デッドライン時間 $\times 1/n$ とする．また，タスクの実行中に割込みを行わないノンプリエンプション方式とする．

キーワード　EDF スケジューリング, 非周期タスク, リアルタイム

Approximate analysis for non-preemption EDF scheduling

Masaki HANADA[†] and Hidenori NAKAZATO[†]

† Graduate School of Global Information and Telecommunication Studies, Waseda University
Nishiwaseda 1–3–10, Shinjiku-ku, Tokyo, 169–0051 Japan
E-mail: †hanada@fuji.waseda.jp

Abstract　In soft real-time system, missing deadline doesn't cause failure directly, but it is important to calculate and investigate the average and worst system performance in advance. The well-known algorithm for real-time scheduling is EDF(Earliest Deadline First) scheduling algorithm. EDF scheduling algorithm assigns priorities according to deadlines, and is able to handle not only periodic tasks but also aperiodic tasks. We analyze system performance(e.g. mean waiting time) on the assumption that the arrival, the exection time and the relative deadline of the task are described in terms of probability distribution functions, and evaluate it through the simulation. Specifically, we assume the system with Poisson arrival, exponential relative deadline, exponential execution time, uniproccessor, infinite buffer size and non-peemption.

Key words　EDF Scheduling, aperiodic task, real-time

1. はじめに

現在，マルチメディア等の発達により，制御系のみではなく情報系のデータを扱うシステムのリアルタイム性が重要となっている．これまでに，事前に起動時間が分からない非周期タスクに適用可能なスケジューリング (キューイング) 手法が多く提案されている．非リアルタイム系として，$FCFS$，$SJF(SPT)$ [1]，$SRPT$ [7]，RR [2] [3] 等があり，リアルタイム系として，EDF [4]，LLF 等があげられる．非リアルタイム系に関しては，多くの解析が行われ，待ち行列理論 [5] [6] [7] [8] [9] [10] として構築されている．リアルタイム系に関しては，起動時間等を含めた十分なタスクの情報を必要とするハードリアルタイムが中心であったため，非周期タスクに適用した場合の解析があまり進んでいない．しかしながら，情報系のデータを扱うソフトなリアルシステムが増加している状況を考慮すると，予め，リアルタイムスケジューリングを適用した場合の効果や性能を数学的に解析しておくことは重要である．

本研究では，最も単純な単一サーバモデルに対し，タスクの各属性が確率分布に従うものと仮定し，EDF スケジューリン

グを適用した場合の解析を行う．

EDF スケジューリングはデッドラインの早い順に優先順位を付ける非常にシンプルな手法である．非周期タスクに対して，プリエンプションが可能な場合，最大遅れを最小にすることに関して最適であり，周期タスクに対しては，プロセッサ使用率が 100%までデッドラインが保証される [4]．

第 2 章ではシステムモデルとタスクの発生と各属性の定義について述べる．第 3 章では待ち時間分布と平均待ち時間の導出のために理論的に解析し，第 4 章では，第 3 章で提案した理論を用いて近似的に解析する．第 5 章では，第 4 章の解析結果をシミュレーションにより評価する．

2. システムモデルとタスクの発生と各属性

システムモデルとして最も単純な単一サーバモデル (図 1) を対象とする．システムを全体として 1 つの処理装置と見なし，内部構造は考慮しない．

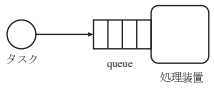

図 1　単一サーバモデル

タスクは発生すると，即座に実行可能となり，デッドライン順にキューに格納される．タスクの発生はポアソン，相対デッドライン時間は指数分布とする．タスクの実行時間は，相対デッドライン時間が短いタスクは，実行時間も同様の割合で短いという仮定を用い，相対デッドライン時間 $\times 1/n(n$ は定数) とする．デッドラインは，発生時刻に相対デッドライン時間を足した値とする．

仮定するタスクの発生と各属性の詳細について，以下に説明する．

- 全体としては到着率 λ のポアソン到着とする．相対デッドライン時間が $(x, x+dx)$ であるタスクの到着率は $\lambda(x)dx$ であり，$\lambda = \int_0^\infty \lambda(x)dx$ となる．
- 全体としての相対デッドライン時間の確率分布関数は $F(x)$ であり，到着するタスクの相対デッドライン時間が $(x, x+dx)$ である確率は $dF(x)(= f(x)dx)$ となる．全体としての到着率は λ であるので，$\lambda(x)$ は以下となる．

$$\lambda(x) = \lambda \frac{dF(x)}{dx} = \lambda f(x) \tag{1}$$

- 各タスクの相対デッドライン時間はその実行時間を n 倍したものとする．これを実行時間比率と呼ぶ．
- 実行時間だけに注目した場合，それは指数分布である．実行時間の確率分布関数 $F_{exec}(x)$，確率密度関数 $f_{exec}(x)$，平均実行時間 $E_{exec}[x]$ は以下である．

$$F_{exec}(x) = 1 - e^{-(n\mu)x} \tag{2}$$

$$f_{exec}(x) = (n\mu)e^{-(n\mu)x} \tag{3}$$

$$E_{exec}[x] = \int_0^\infty x(n\mu)e^{-(n\mu)x}dx = \frac{1}{n\mu} \tag{4}$$

- 実行時間比率 n の時，相対デッドライン時間 x のタスクの待ち時間の分布について以下に定義する．

実行時間比率 n の時，相対デッドライン時間 x のタスクの待ち時間を τ とし，待ち時間 τ が t 以下となる確率を $P(\tau \leqq t)$ とした時，それを分布関数を $W_n(x,t)$ として定義する．タスクの待ち時間の密度関数 $w_n(x,t)$ は，分布関数 $W_n(x,t)$ を用いると以下に定義される．

$$w_n(x,t) = \frac{dW_n(x,t)}{dt} \tag{5}$$

平均待ち時間 $WQ_n(x)$ は密度関数 $w_n(x,t)$，あるいは分布関数 $W_n(x,t)$ を用いると以下に定義される．

$$WQ_n(x) = \int_0^\infty t w_n(x,t)dt \tag{6}$$

$$WQ_n(x) = \int_0^\infty (1 - W_n(x,t))dt \tag{7}$$

3. 待ち時間分布，平均待ち時間の導出

これまでに，優先権スケジューリングの平均待ち時間の導出について多くの議論が行われてきた [6] [8]．従来の枠組 (HOL 優先権，サービス時間依存) [6] では，優先権となる属性 (例えば，相対デッドライン時間，実行時間) の確率分布が与えられれば，平均待ち時間に対する再帰の方程式を容易に定義でき，それを解くことにより求められた．しかし，EDF スケジューリングでは，優先権となる属性がデッドラインなので，発生と相対デッドライン時間の 2 つを考慮しなければならず，従来の枠組の適用は困難である．当然，EDF スケジューリングでは，相対デッドライン時間が優先権でないので，キュー内で相対デッドライン時間の長いタスクが相対デッドライン時間の短いタスクより先に実行される可能性がある．このように，優先権を相対デッドライン時間とした場合との差分のタスク数を正確に求められれば，平均待ち時間を求めることが可能となる．本研究では，平均待ち時間に対する再帰の方程式を定義するために，待ち時間の密度関数を用いる．

相対デッドライン時間 x のタスク (以下，注目タスクと呼ぶ) の待ち時間は以下の要素の合計となる．また，注目タスクの比較対象となる相対デッドライン時間 t のタスクを対象タスクと呼ぶ．

（1） 注目タスク (相対デッドライン時間 x) の到着時に実行中のタスクによる待ち時間 W_o

（2） 注目タスク (相対デッドライン時間 x) がキューに到着した時点で，注目タスクより先に実行されるタスクによる待ち時間 $W1_n(x)$

（3） 注目タスク (相対デッドライン時間 x) がキューにいる間に到着したタスクの中で，注目タスクより先に実行されるタスクによる待ち時間 $W2_n(x)$

3.1 各要素の待ち時間

各要素 (W_o，$W1_n(x)$，$W2_n(x)$) の待ち時間について求める．

（1） 注目タスクの到着時に実行中のタスクによる待ち時間 W_o

W_o は相対デッドライン時間に依存しない．実行時間だけに

注目すればよい．実行時間が $\frac{t}{n}$ であるタスクに，注目タスクが到着した場合，残余寿命時間の期待値は $\frac{t}{2n}$ である．ここで，相対デッドライン時間 t のタスクの実行中である時間割合 (使用率) は $\rho(t) = \lambda(t)\frac{t}{n}$ である．W_o は以下となる．

$$W_o = \int_0^\infty \rho(t)(\frac{t}{2n})dt = \frac{\lambda}{(n\mu)^2} \tag{8}$$

（2） 注目タスク (相対デッドライン時間 x) がキューに到着した時点で，注目タスクより先に実行されるタスクによる待ち時間 $W1_n(x)$

待ち時間 $W1_n(x)$ は以下の 2 つの要素に分けられる．

- 注目タスク (相対デッドライン時間 x) より短い相対デッドライン時間をもつタスクによる待ち時間

キュー内に存在している対象タスク (相対デッドライン時間 t) は，注目タスク (相対デッドライン時間 x) より先に到着しているので，注目タスクより短い相対デッドライン時間をもつ対象タスクは，全て先に実行される．対象タスクの到着率は $\lambda(t)$，待ち時間は $\int_0^\infty zw_n(t,z)dz$ であるので，リトルの公式より，対象タスクのタスク数は $\lambda(t)\int_0^\infty zw_n(t,z)dz$ である．また，相対デッドライン時間が t であるタスクの実行時間は $\frac{t}{n}$ である．したがって，以下となる．

$$\int_0^x \frac{t}{n}\lambda(t)\left(\int_0^\infty zw_n(t,z)dz\right)dt \tag{9}$$

- 注目タスク (相対デッドライン時間 x) より長い相対デッドライン時間をもつタスクによる待ち時間

キュー内に存在している対象タスク (相対デッドライン時間 t) は，注目タスク (相対デッドライン時間 x) より先に到着しているので，注目タスクより長い相対デッドライン時間をもつ対象タスクでも，そのいくつかは先に実行される．これを求めるために，対象タスクの待ち時間の密度関数を用いる．対象タスクがキューに存在し，注目タスクが到着する場合を考える．これを図 2 に記す．

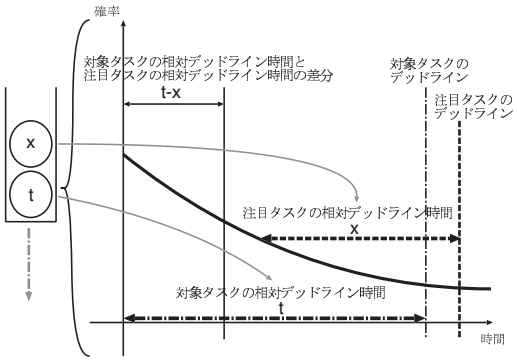

図 2　対象タスクが先に実行されるパターン 1

対象タスクが先に実行されるのは，対象タスクのデッドラインが注目タスクのデッドラインより早い場合である．したがって，注目タスクの到着時刻と対象タスクの到着時刻の差が両タスクの相対デッドライン時間の差分 $(t-x)$ より大きい場合である．対象タスクの待ち時間が z である場合に，z が $(t-x)$ 以内である場合は対象タスクは注目タスクに影響を与えない．z が $(t-x)$ 以上であれば，$(t-x)$ から z の間が注目タスクの到着範囲となる．この条件を満たす対象タスクの待ち時間は $\int_{t-x}^\infty (z-(t-x))w_n(t,z)dz$ となる．対象タスクの到着率は $\lambda(t)$ であるので，リトルの公式より，対象タスクのタスク数は $\lambda(t)\int_{t-x}^\infty (z-(t-x))w_n(t,z)dz$ となる．また，相対デッドライン時間が t であるタスクの実行時間は $\frac{t}{n}$ である．したがって，以下となる．

$$\int_x^\infty \frac{t}{n}\lambda(t)\left(\int_{t-x}^\infty (z-(t-x))w_n(t,z)dz\right)dt \tag{10}$$

式 (9)，(10) より，

$$W1_n(x) = \int_0^x \frac{t}{n}\lambda(t)\left(\int_0^\infty zw_n(t,z)dz\right)dt + \int_x^\infty \frac{t}{n}\lambda(t)\left(\int_{t-x}^\infty (z-(t-x))w_n(t,z)dz\right)dt \tag{11}$$

（3） 注目タスク (相対デッドライン時間 x) がキューにいる間に到着したタスクの中で，自分より先に実行されるタスクによる待ち時間 $W2_n(x)$

注目タスク (相対デッドライン時間 x) より長い相対デッドライン時間をもつタスクは注目タスクに影響を与えない．対象タスク (相対デッドライン時間 t) は注目タスク (相対デッドライン時間 x) より後に到着するので，注目タスクより短い相対デッドライン時間をもつタスクの全てが，先に実行されるとは限らない．上記と同様，これを求めるために，注目タスクの待ち時間の密度関数を用いる．注目タスクがキューに存在し，対象タスクが到着する場合を考える．これを図 3 に記す．

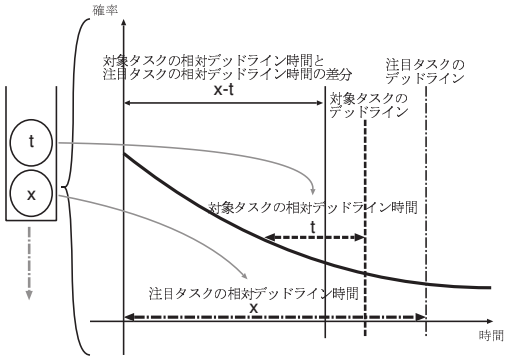

図 3　対象タスクが先に実行されるパターン 2

対象タスクが先に実行されるのは，対象タスクのデッドラインが注目タスクのデッドラインより早い場合である．したがって，対象タスクの到着時刻と注目タスクの到着時刻の差が両タスクの相対デッドライン時間の差分 (x-t) より小さい場合である．注目タスクの待ち時間が z である場合に，z が $(x-t)$ 以内であれば，0 から z が対象タスクの到着範囲となる．z が $(x-t)$ 以上であれば，0 から $(x-t)$ が対象タスクの到着範囲となる．この条件を満たす注目タスクの待ち時間は $\int_0^{x-t} zw_n(x,z)dz + \int_{x-t}^\infty (x-t)w_n(x,z)dz$ となる．対象タスクの到着率は $\lambda(t)$ であるので，リトルの公式より，対象タスクのタスク数は $\lambda(t)\int_0^{x-t} zw_n(x,z)dz + \lambda(t)\int_{x-t}^\infty (x-t)w_n(x,z)dz$ となる．また，相対デッドライン時間が t であるタスクの実行

時間は $\frac{t}{n}$ である．したがって，$W2_n(x)$ は以下となる．

$$W2_n(x) = \int_0^x \frac{t}{n}\lambda(t)\left(\int_0^{x-t} z w_n(x,z)dz\right)dt + \int_0^x \frac{t}{n}\lambda(t)\left(\int_{x-t}^{\infty}(x-t)w_n(x,z)\right)dzdt \quad (12)$$

3.2 待ち時間分布と平均待ち時間

前章までに，平均待ち時間 $WQ_n(x)$ の各要素 W_o，$W1_n(x)$，$W2_n(x)$ を求めた．また，平均待ち時間 $WQ_n(x)$ は式 (6) より，待ち時間の密度関数を用いて定義できる．よって，式 (8)，(11)，(12) より，待ち時間の密度関数のみを用いた再帰の方程式となる．

$$\begin{aligned}\int_0^{\infty} z w_n(x,z)dz &= \frac{\lambda}{(n\mu)^2} \\ &+ \int_0^x \frac{t}{n}\lambda(t)\left(\int_0^{\infty} z w_n(t,z)dz\right)dt \\ &+ \int_x^{\infty} \frac{t}{n}\lambda(t)\left(\int_{t-x}^{\infty}(z-(t-x))w_n(t,z)dz\right)dt \\ &+ \int_0^x \frac{t}{n}\lambda(t)\left(\int_0^{x-t} z w_n(x,z)dz\right)dt \\ &+ \int_0^x \frac{t}{n}\lambda(t)\left(\int_{x-t}^{\infty}(x-t)w_n(x,z)dz\right)dt \quad (13)\end{aligned}$$

また，式 (5)，(7) より，待ち時間の分布関数を用いると式 (13) は以下の再帰の方程式となる．

$$\begin{aligned}\int_0^{\infty} 1-W_n(x,z)dz &= \frac{\lambda}{(n\mu)^2} \\ &+ \int_0^x \frac{t}{n}\lambda(t)\left(\int_0^{\infty} 1-W_n(t,z)dz\right)dt \\ &+ \int_x^{\infty} \frac{t}{n}\lambda(t)\left(\int_{t-x}^{\infty} 1-W_n(t,z)dz\right)dt \\ &+ \int_0^x \frac{t}{n}\lambda(t)\left(\int_0^{x-t} 1-W_n(x,z)dz\right)dt \quad (14)\end{aligned}$$

4. 待ち時間分布の近似

式 (14) を直接的に解くことは困難なので，EDF スケジューリングとタスクの発生，各属性のいくつかの特徴を利用して近似的に解く手法を用いる．以下の 2 つのアプローチを用いる．

- 待ち時間分布を近似的に定義する．
- 境界条件の特定のために式 (14) の 3 番目の項を近似的に定義する．

4.1 待ち時間分布の近似的な定義

EDF スケジューリングの待ち時間分布の特徴を以下に示す．

（1） ポアソン到着，指数分布の実行時間を想定した場合，待ちが生じる確率である待ち合わせ率 Π はスケジューリング手法に依存しない．したがって，EDF スケジューリングを適用した場合でも待ち合わせ率 Π は $FCFS$ スケジューリング と同じである．$FCFS$ スケジューリング の場合の待ち合わせ率 Π は以下である．

$$\Pi = \frac{\lambda}{n\mu} \quad (15)$$

（2） 優先権はデッドライン順であるので相対デッドライン時間に依存した待ち時間の分布となる．$FCFS$ スケジューリングの場合は，到着順であるのでどの相対デッドライン時間でも常に同じ待ち時間の分布となるが，EDF スケジューリングの場合は，相対デッドライン時間が長いタスクは，待たされる確率が高くなり，待ち時間の分布の傾きはなだらかになる．逆に，相対デッドライン時間が短いタスクは，待たされる確率が低くなり，待ち時間の分布の傾きは急な分布になる．

特徴 (1)，(2) より，相対デッドライン時間 x のタスクの待ち時間の分布関数を以下に定義する．

$$W_n(x,t) = 1 - \Pi e^{-(n\mu-\lambda)K(x)t} \quad (16)$$

ここで，$K(x)$ は相対デッドライン時間に依存した係数である．これは，$FCFS$ スケジューリングの場合を $1(=K(x))$ とした場合の傾きの割合を表している．当然，$FCFS$ スケジューリングの場合では，相対デッドライン時間に依存しないので $K(x)=1$ の固定となる．EDF スケジューリングの場合は，相対デッドライン時間 x によって $K(x)$ が変動する．x が大きくなるにつれて $K(x)$ は小さくなる．両スケジューリングの待ち時間を τ とし，τ が t より大きくなる確率 $P(\tau > t) = 1 - W_n(x,t)$ を比較すると，図 4 となる．$FCFS$ スケジューリングの場合は，どの相対デッドライン時間でも常に同じ分布となり，EDF スケジューリングの場合は，相対デッドライン時間が長いタスク $(K(x) < 1)$ は，待たされる確率が高くなり，相対デッドライン時間が短いタスク $(K(x) > 1)$ は，待たされる確率が低くなることを示している．

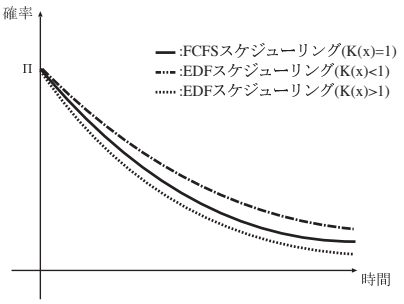

図 4 $FCFS$ スケジューリングと EDF スケジューリングの待ち時間の分布 $(1 - W_n(x,t))$

ここでは，待ち時間の分布関数と相対デッドライン時間がどのような依存関係にあるかが重要である．具体的な $K(x)$ の値は次章の再帰の方程式を解くことにより求まる．

式 (16) を用いると式 (14) は以下となる．

$$\begin{aligned}\int_0^{\infty} \frac{\lambda}{n\mu} e^{-(n\mu-\lambda)K(x)z}dz &= \frac{\lambda}{(n\mu)^2} \\ &+ \int_0^x \frac{t}{n}\lambda(t)\left(\int_0^{\infty} \frac{\lambda}{n\mu} e^{-(n\mu-\lambda)K(t)z}dz\right)dt \\ &+ \int_x^{\infty} \frac{t}{n}\lambda(t)\left(\int_{t-x}^{\infty} \frac{\lambda}{n\mu} e^{-(n\mu-\lambda)K(t)z}dz\right)dt\end{aligned}$$

$$+\int_0^x \frac{t}{n}\lambda(t)\left(\int_0^{x-t}\frac{\lambda}{n\mu}e^{-(n\mu-\lambda)K(x)z}dz\right)dt \qquad (17)$$

4.2 境界条件の特定のための近似的な定義

式 (17) は，3 番目の項の性質上，解くのは困難であり，境界条件を与える必要がある．境界条件を $x=0$ とした場合，式 (17) の 2 番目と 4 番目の項は 0 になる．しかし，3 番目の項は特定できない．ここでは境界条件の特定のために 3 番目の項の近似を行う．3 番目の項は，注目タスク (相対デッドライン時間 x) がキューに到着した時点で，自分より相対デッドライン時間が長いタスクによる待ち時間である．式 (1)，式 (15)，式 (17) より，3 番目の項は以下である．

$$\lambda\int_x^\infty \frac{t}{n}\left(\int_{t-x}^\infty \frac{\lambda}{n\mu}e^{-(n\mu-\lambda)K(t)z}dz\right)\mu e^{-\mu t}dt$$
$$=\lambda\int_x^\infty \frac{t}{n}\left(\frac{\lambda/(n\mu)}{K(t)(n\mu-\lambda)}\right)$$
$$\times\left(e^{-(n\mu-\lambda)K(t)(t-x)}\right)\mu e^{-\mu t}dt \qquad (18)$$

式 (18) において，x と t に依存した項を $R(x,t)$ として，以下に示す．

$$R(x,t)=\left(\frac{1}{K(t)}\right)\left(e^{-(n\mu-\lambda)K(t)(t-x)}\right)\left(\mu e^{-\mu t}\right) \qquad (19)$$

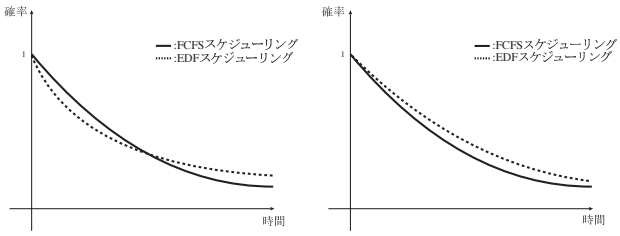

図 5　x が小さい場合の 2 番目の項　図 6　x が大きい場合の 2 番目の項

式 (19) の 2 番目の項 ($e^{-(n\mu-\lambda)K(t)(t-x)}$) の性質について考えると，注目しているのは，相対デッドライン時間が x のタスクであり，対象としているのは，相対デッドライン時間が t のタスクなので，積分範囲を考慮すると t が x から ∞ まで変動する．x が小さい場合は，$K(x)$ が 1 より大きい値となり，t は x から変動するので，$K(t)$ も 1 より大きい値から変動する．x が大きい場合は，$K(x)$ が 1 より小さい値となり，$K(t)$ も 1 より小さい値から変動する．すなわち，注目タスクの相対デッドライン時間が x が小さい場合は，図 5 のように $FCFS$ スケジューリングの場合と交差する分布となる．逆に，注目タスクの相対デッドライン時間 x が大きい場合は，図 5 のように $FCFS$ スケジューリングの場合と交差せずに常に大きい分布となる．交差する場合は，$FCFS$ スケジューリングの場合との誤差が打ち消される．交差しない場合は，誤差が大きくなる．しかし，注目タスクの相対デッドライン時間 x が大きい場合，式 (19) の 3 番目の項 ($\mu e^{-\mu t}$) より，誤差が非常に大きくなる部分では，式 (19) の値自体が非常に低くなる．つまり，$FCFS$ スケジューリングを仮定しても，ある区間で一時的に多少の誤差が表れるが，再びその誤差は減少していくと考えられる．式 (18) の 3 番目の項に関しては，$FCFS$ スケジューリングの待ち時間の分布関数に置換えてもその影響は非常に少ないと考えられる．

置換えを行うと以下の式となる．

$$\int_0^\infty \frac{\lambda}{n\mu}e^{-(n\mu-\lambda)K(x)z}dz=\frac{\lambda}{(n\mu)^2}$$
$$+\int_0^x \frac{t}{n}\lambda(t)\left(\int_0^\infty \frac{\lambda}{n\mu}e^{-(n\mu-\lambda)K(t)z}dz\right)dt$$
$$+\int_x^\infty \frac{t}{n}\lambda(t)\left(\int_{t-x}^\infty \frac{\lambda}{n\mu}e^{-(n\mu-\lambda)z}dz\right)dt$$
$$+\int_0^x \frac{t}{n}\lambda(t)\left(\int_0^{x-t}\frac{\lambda}{n\mu}e^{-(n\mu-\lambda)K(x)z}dz\right)dt \qquad (20)$$

上記の方程式を解くと $K(x)$ が求まる．式 (16) より，待ち時間の分布関数が求まり，式 (7) より，平均待ち時間が求まる．

5. シミュレーションによる評価

本研究では，提案した近似解析で求めた待ち時間分布，平均待ち時間とシミュレーションした結果を比較し，評価する．また，近似解析による影響を明確にし，適応できる範囲を記す．シミュレーションでは，下記のパラメータを用いて，タスクを 1000 万個発生させる．

5.1 パラメータと結果

- 平均待ち時間 $WQ_n(x)$

負荷率に対する評価，実行時間比率 n に対する評価を行う．平均到着率 λ，平均相対デッドライン時間 $1/\mu$，実行時間比率 n を変化させる．パラメータを表 1 に示す．

表 1　平均待ち時間 $WQ_n(x)$

λ	0.001	0.005	0.009	0.006	0.006
μ	0.01	0.01	0.01	0.01	0.01
n	1	1	1	3	6
$WQ_n(x)$	図 7	図 8	図 9	図 10	図 11

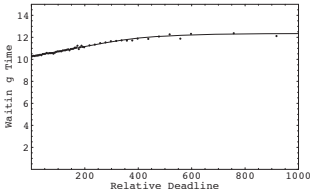

図 7　$(\lambda=0.001,\mu=0.01,n=1)$

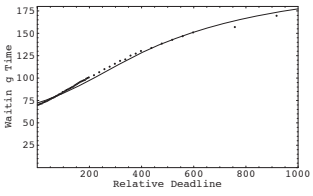

図 8　$(\lambda=0.005,\mu=0.01,n=1)$

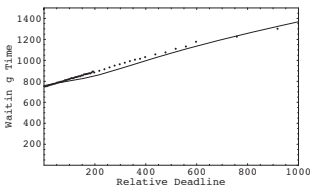

図 9　$(\lambda=0.009,\mu=0.01,n=1)$

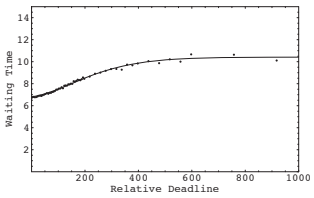

図 10 $(\lambda = 0.006, \mu = 0.01, n = 3)$

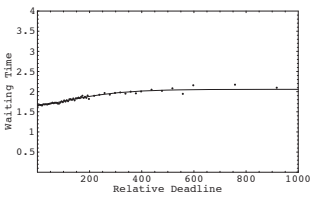

図 11 $(\lambda = 0.006, \mu = 0.01, n = 6)$

- 待ち時間の分布 $1 - W_n(x,t)$

相対デッドライン時間 x に対する評価を行う．パラメータを表 2 に示す．ここで，$x = 100$ については相対デッドライン時間が 95 から 105 の間にあるタスクの待ち時間，$x = 300$ については相対デッドライン時間が 290 から 310 の間にあるタスクの待ち時間の分布を示す．

表 2 待ち時間分布 $1 - W_n(x,t)$

λ	0.005	0.005
μ	0.01	0.01
n	1	1
x	100	300
$1 - W_n(x,t)$	図 12	図 13

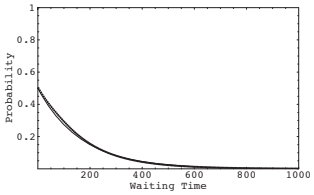

図 12 $(x = 100)$

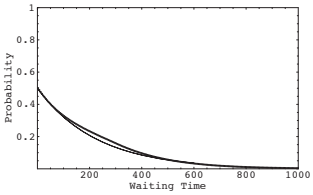

図 13 $(x = 300)$

5.2 評　　価

負荷率に注目し，図 7，図 8，図 9 について評価する．低負荷 (図 7) と中負荷 (図 8) の状態では，よく一致している．高負荷 (図 9) の状態では，平均実行時間の 1.5 倍あたりから 4 倍あたりまで，多少のずれが生じている．これは，式 (13) の 3 番目の項を $FCFS$ スケジューリングを仮定した場合と置換えた影響である．高負荷 (図 9) では，最大で 5%ほどの誤差が生じている．図 10，図 11 は実行時間比率を変化させた場合である．実行時間比率を変化させても一致している．しかし，高負荷になると前述と同様に多少の誤差が生じると考えられる．非常に高負荷な場合を除いては一致し，非常に厳密な精度を必要としない場合は提案した手法は有用である．

相対デッドライン時間 100 と 300 の両待ち時間分布もよく一致している．相対デッドライン時間 300 の待ち時間分布において，多少の誤差が表れているのは，対象とする相対デッドライン時間の範囲 (290,310) を広めにとっているためと考えられる．また，厳密な待ち時間分布を表しておらず，近似による影響が考えられる．しかし，前述と同様，非常に厳密な精度を必要としない場合は提案した手法は有用である．

6. おわりに

リアルタイムスケジューリングの代表的な手法である $EDF(EarliestDeadlineFirst)$ スケジューリング に対しての近似解析を行った．本研究では，最も単純な単一サーバモデルを対象とし，タスクの各属性が確率分布に従うものと仮定し，EDF スケジューリングを適用した場合の効果や性能を数学的に解析した．解析結果は，シミュレーションにより評価し，十分に一致することを確認した．

EDF スケジューリング は，一般的な優先権スケジューリングの平均待ち時間導出の枠組みを適用できないために，新たな枠組みを提案した．本文では示していないが，待ち時間分布より，リアルタイムシステムの性能評価指標である平均遅延時間，デッドラインミス率の算出も可能である．解析の対象としているモデルは最も単純ではあるが，システム設計時，あるいはシステム変更時などに，どのような効果があるかを把握するための十分な指標となり得る．現実では様々なタスクの発生，相対デッドライン時間，実行時間も考えられる．特に，すべてのタスクが確率的な発生，相対デッドライン時間 (あるいは実行時間) であると仮定できないリアルタイムシステムも多く存在し，ある程度，決定的な (周期的等) な発生，相対デッドライン時間，実行時間などの解析も今後，必要である．

文　　献

[1] Phipps, T.E.,"Machine Repair as a Priority Waiting Line Problem," Operations Research, Vol. 4, pp.76-85, 1956.
[2] Sakata, M., S. Noguchi and J. Oizumi, "An Analysis of the M/G/1 Queue Under Round-Robin Discipline," Operations Research, Vol. 19, pp.371-385, 1971.
[3] E. G. Coffman, Jr. , R. R. Muntz , H. Trotter, "Waiting Time Distributions for Processor-Sharing Systems," Journal of the ACM (JACM), v.17 n.1, p.123-130, Jan. 1970.
[4] J.R. Jackson, "Scheduling a production line to minimize maximum tardiness," Research Report 43, Management Science Research Project, University of California, Los Angeles, 1955.
[5] L. Kleinrock, Queueing Systems, Volumne I, John Wiley & Sons Inc., New York, 1975.
[6] L. Kleinrock, Queueing Systems, Volumne II, John Wiley & Sons Inc., New York, 1975.
[7] 牧野都治, 待ち行列の応用, 森北出版, 東京, 1969.
[8] 宮脇一男，長岡崇雄，毛利悦造, 待合わせ理論とその応用, 日刊工業新聞, 東京, 1961.
[9] 西田俊夫, 待ち行列の理論と応用, 朝倉書店, 東京, 1971.
[10] 森村英典，大前義次, 応用待ち行列理論，日科技連出版社，東京, 1975.
[11] 亀田尚夫, 紀一誠, 李頡, 性能評価の基礎と応用，共立出版, 東京, 1998.