

優先スケジュールを行なう多重プログラミング・ システムのボトルネック解析

西垣 通・山本 彰

(日立製作所システム開発研究所)

1. まえがき

計算機システムの設計、改造に際して、その性能予測が必要となる。特に重要なのは、システムの大局的挙動の簡便な予測である。シミュレーションは多大な工数と計算機時間を要するため、この目的には待行列網理論による解析が適している。しかし、優先度を含むシステムでは、一般に待行列網理論による非解は困難なことが知られている⁵⁾。一方、現実のシステムでは優先度にもとづく資源割当てを行なうことが多いので、その簡便な予測手法が要求されていた。

従来より上記問題の解決のため幾つかの近似手法が提案されている。Sevcik⁶⁾は shadow CPU という概念を用いて、CPU のみ2レベルの優先度、入出力装置は優先度なしのシステムを近似解析した。また最近 Neuse⁷⁾ は Norton の定理を応用し、CPU に限らず入出力装置を含む一般のプロセッサについて複数レベルの優先度のあるシステムの近似解析手法を提案した。Neuse⁷⁾ の手法は Sevcik の手法より適用範囲が広いが、必ずしも解が収束するとは限らないうえ複雑な優先度系では収束速度が低下し、簡便な手法とは言い難い。また両手法とも複数資源の同時割当てすなわち二次資源は対象外のため、メモリ割当てにおける優先度は扱えない。

筆者らは既に、メモリ、CPU、入出力装置の各資源において複数レベルの優先度をもつシステムの大局的挙動の簡便な予測を目的として、「漸近モデル (Asymptotic Model)」¹⁾ を提案した。その特徴は、平均応答時間、資源利用率などの厳密解を漸近解で近似する点にある。Kleinrock²⁾、Muntz³⁾、Denning⁴⁾ はシステムのボトルネックとなる資源について考察し、平均応答時間が呼源数の関数として漸近線を持つことを示した。これらの解析では、負荷のモデルは優先度のない単一クラスが仮定されており、メモリも扱われていない。漸近モデルはこのボトルネック解析の手法を拡張したものである。

本論文では、新たな漸近モデルとして「漸近モデル II (Asymptotic Model II)」を提案する。(なお以下、既に提案したモデルを AM I、本論文で述べるモデルを AM II と略称する。) AM I と AM II との相違は「等優先度」の扱いにある。前者では、ある資源の割当てに関して等優先度の呼には「単位時間あたり等量の資源」が割り当てられるフェア・シェア・サービスを仮定した。一方後者では、「資源1回使用あたりの平均待ち時間」が等しいと仮定する。現在多くのシステムでは等優先度の呼に対して FCFS (First Come First Served) で資源を割り当てるが、このようなシステムの挙動は AM II により予測できると考えられる。

2. ボトルネック解析の対象システム

解析の対象とするシステムを明示する。システムは図1に示すような「有限多重度の有限呼源モデル」で表わされる。図1で、各端末は思考時間が終わるごととトランザクションを発生する。資源の要求単位を「トランザクション」とよぶが、これは TSS コマンドやバッチ・ジョブに対応する。バッチ・ジョブについては、イニシエータを思考時間ゼロの端末とみなす(問題なのは高負荷状態なので、ジョブ・キューは常にジョブが存在すると仮定する。ジョブ・キューでの待ち時間は本論文の対象外とする)。また、トランザクションの working set は過度の

ページングを防ぐよう適切に選択されていると仮定する。メモリ容量を V 、平均 working set size を w とすると、最大多重度は次式で与えられる。

$$r = V/w \quad (2.1)$$

ここで V は w の倍数と仮定する。思考中以外の端末の数が r を越えると、越えた分だけのトランザクションはメモリ割当て待ちとなり、スワップ・インされるのを待つ。

トランザクション・クラス i の平均応答/処理時間、平均メモリ占有時間、平均思考時間、最大多重度をそれぞれ T_i 、 R_i 、 z_i 、 c_i と表わす。 c_i は式(2.1)の w としてクラス i の平均 working set size w_i を用いることにより得られる。ここで R_i は T_i からクラス i の平均メモリ割当て待ち時間を差し引いた値に等しい。CPUと入出力装置は本モデルでは同様に扱われるので、これらをもとに「プロセッサ」とよぶ。クラス i のトランザクションによるプロセッサ j の平均使用時間を d_{ij} とすると、 R_i の下限値 R_{i0} は次式で与えられる(複数プロセッサの同時使用はないとする)。

$$R_{i0} = \sum_j d_{ij} \quad (2.2)$$

R_{i0} は全く資源割当て待ちが発生しない場合の平均応答時間に対応する。

プロセッサ j の利用率 u_j およびメモリ利用率 u_m は次式をみだす。

$$\begin{cases} u_j = \sum_i N_i d_{ij} / (T_i + z_i) \\ u_m = \sum_i N_i (R_i / c_i) / (T_i + z_i) \end{cases} \quad (j = 1, 2, \dots) \quad (2.3)$$

ここで N_i はクラス i の呼源数、すなわちクラス i のトランザクションを発生する端末数を示す。式(2.3)は各資源での到着時間間隔やサービス時間の分布形によらず成立する。

ボトルネック解析は、利用率の高いボトルネックとなる資源に着目する⁴⁾。このような資源の利用率を100%に等しいとみなすことにより、漸近解が得られる。すなわちボトルネック解析の仮定は次の通りである。

「仮定I: ある資源の割当て待ち時間は、当資源を100%使用するに足るだけの呼源が存在しない場合にはゼロとする。またこれ以外の場合には、当資源は常に100%使用されているとする。」

本仮定から平均応答時間 T_i の漸近解を求めることができる。ここで漸近解は、到着時間間隔やサービス時間の分布形によらず生ずる最小限の資源割当て待ち時間を含んでいる。実際には資源利用率が100%未満の場合でももちろみ割当て待ちは発生するので、漸近解は T_i を過小評価する傾向をもつ。

単一クラスかつメモリ容量 V が無限大の場合、式(2.3)と仮定Iから、よく知られた次の漸近解がみちびかれる(図2参照)。

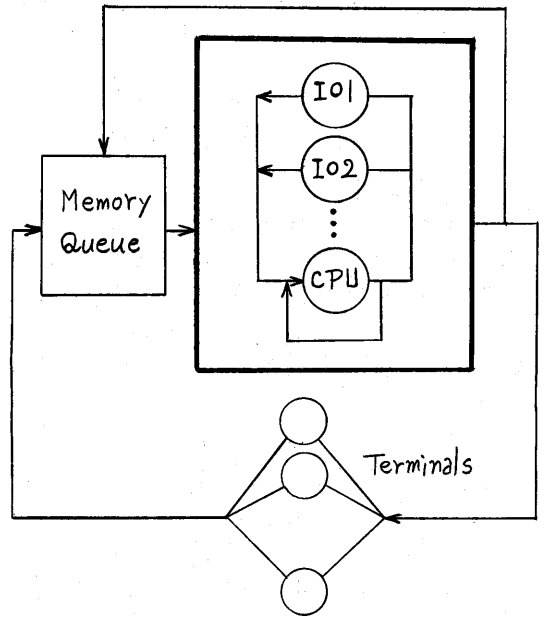


図1 システム・モデル
有限多重度の有限呼源モデル

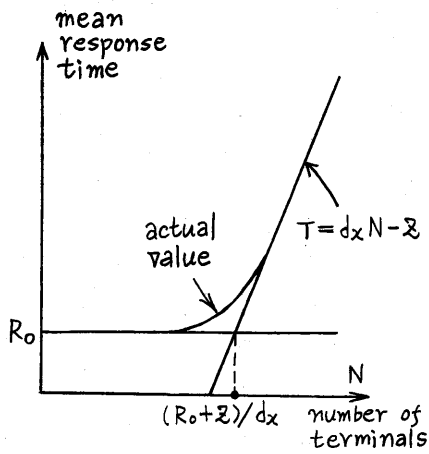


図2 無限多重度の場合の平均応答時間Tの漸近解

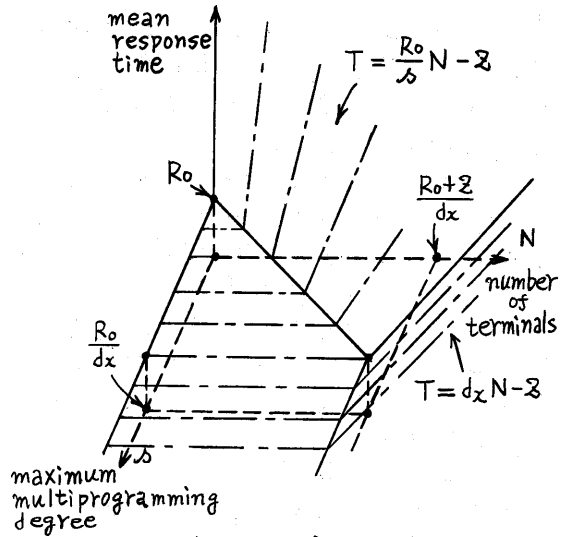


図3 有限多重度の場合の平均応答時間Tの漸近解

$$T = \begin{cases} dxN - z & : (R_0 + z)/dx \leq N \\ R_0 & : \text{上記以外の } N \end{cases} \quad (2.4)$$

ただし式(2.4)において、単一クラスなので添字iは省いてある。またxは $\max_j d_j$ ($\max_j u_j$)に対応し、ボトルネックになりうるプロセッサを示す。

メモリ容量V(最大多重度s)が有限の場合、メモリもボトルネックになりうる。この漸近解は式(2.5)で与えられる(図3参照)。証明は1)を参照されたい。

$$T = \begin{cases} dxN - z & : u_x = 1 \text{ をみたく } N \\ (R_0/\delta)N - z & : u_x < 1 \text{ かつ } u_m = 1 \text{ をみたく } N \\ R_0 & : \text{上記以外の } N \end{cases} \quad (2.5)$$

複数クラスの場合、次の仮定を設ける。

「仮定II: 優先スケジューリングは割り込み優先権出直し基準 (preemptive resume rule) にしたがうとする。したがって、より優先度の低いラングーションからの影響は無視する。優先度が低いラングーションには、優先度が高いラングーションの使用した残りの資源が割り当てられる。」

実際のシステムでは入出力装置は preemptive ではない。また入出力実行中のラングーションはその完了までスワップアウトできないので、メモリも場合によっては non-preemptive である。したがって仮定IIにより漸近解は、高優先度ラングーション、低優先度ラングーションの平均応答時間をそれぞれ過小、過大に評価する傾向をもつ。

いずれかの資源がその高優先度のラングーションで占有されると、低優先度のラングーションの平均応答時間は無限大となる。全てのクラスについて有限のT_iを得るため、解析の対象とするN (= Σ N_i)の範囲は限定される(Nを限定することにより議論の一般性は失われぬ)。T_iが無限大のとき、i以外のクラスの平均応答時間は、第3章で示す方程式で単に第iクラスの項を無視することにより得られる。

システム中の各資源割当て待行列において、そのサービス規律は任意でよいと

する。これは、いかなるサービス規律についても漸近解は同一のためである。ただしボトルネック資源の最低優先度クラスについては、「1回資源使用あたりの平均待ち時間が同一」となるサービスを仮定する。

3. 漸近モデルⅡ (AMⅡ)

漸近モデルAMⅡ (Asymptotic Model Ⅱ)の方程式を以下にみちびく。まず、メモリ容量 V が無限大すなわち全ての i について最大多重度 $n_i = \infty$ の場合についてのべる。このときメモリ割り当て待ちは無視できる。

プロセッサ j の最低優先度トランザクション・クラスの集合を γ_j とし、ボトルネック・プロセッサ(利用率が1なるプロセッサ)の集合を H とする。 H が空集合のとき、全ての i について $T_i = R_{i0}$ である。以下 H が空集合でないとして議論を進める。クラス i のトランザクションのプロセッサ j における平均待ち時間を τ_{ij} とすると、

$$T_i = R_{i0} + \sum_j \tau_{ij} \quad (3.1)$$

である。全てのクラスについて有限の T_i が存在するとき、仮定ⅠとⅡより、待ちが生ずる($\tau_{ij} > 0$)のはボトルネック資源の最低優先度クラスのトランザクションのみである。また $\tau_{ij} > 0$ のとき、「1回資源使用あたりの平均待ち時間が等しい」仮定から、 τ_{ij} は i による j の平均使用回数 θ_{ij} に比例する。すなわち次式がなりたつ。

$$\begin{cases} \tau_{ij} = \delta_{ij} \theta_{ij} \psi_j \\ \delta_{ij} = \begin{cases} 1 : i \in \gamma_j \text{ かつ } j \in H \\ 0 : \text{上記以外の場合} \end{cases} \end{cases} \quad (3.2)$$

ここで ψ_j は1回 j を使用するための平均待ち時間である。式(3.1),(3.2)を式(2.3)第1式に用いて次式を得る。

$$1 = \sum_i d_{ij} N_i / (R_{i0} + \sum_k \delta_{ik} \theta_{ik} \psi_k + \sum_j \theta_{ij} \psi_j) \quad (j \in H) \quad (3.3)$$

式(3.3)より $\psi_k (> 0)$ を求め、式(3.1),(3.2)より T_i を求めることができる。ただし得られた T_i を式(2.3)に代入したとき

$$0 \leq u_j \leq 1 \quad (j = 1, 2, \dots) \quad (3.4)$$

が成立することが必要である。いいかえると、所与の N に対して H は式(2.3),(3.1)-(3.4)より定まる。

次にメモリ容量 V が有限すなわち $n_i \neq \infty$ の場合について述べる。いま、メモリ利用率 u_m が1より小さすなわちメモリがボトルネックでなければ、 T_i は式(2.3),(3.1)-(3.4)より求まる。したがって以下メモリ・ネックすなわち $u_m = 1$ と仮定し、メモリ割り当ての最低優先度クラスの集合を γ_m と表わす。クラス i のトランザクションの平均メモリ待ち時間を τ_{im} とすると次式がなりたつ。

$$\begin{cases} R_i = R_{i0} + \sum_j \tau_{ij} \\ T_i = R_i + \tau_{im} \end{cases} \quad (3.5)$$

$\tau_{im} > 0$ のとき τ_{im} は i のメモリ平均使用回数 θ_{im} に比例するという仮定から式(3.6)が得られる(短小なトランザクションでは $\theta_{im} = 1$ だが、長大なものはしばしばスワップ・アウトされるので $\theta_{im} \geq 2$ となりうる)。 ψ_m はメモリ1回使用、すなわちスワップ・インされるための平均待ち時間である。

$$\begin{cases} \tau_{im} = \delta_{im} \theta_{im} \psi_m \\ \delta_{im} = \begin{cases} 1 : i \in \gamma_m \text{ かつ } u_m = 1 \\ 0 : \text{上記以外の場合} \end{cases} \end{cases} \quad (3.6)$$

結局式(3.3)の拡張として、式(2.3), (3.2), (3.5), (3.6)より次式が得られる。

$$\begin{cases} 1 = \sum_{i \in H} d_{ij} N_i / (R_{i0} + \sum_k \delta_{ik} \theta_{ik} \gamma_k + \delta_{im} \theta_{im} \gamma_m + \varepsilon_i) & (j \in H) \\ 1 = \sum_{i \in H} \{ (R_{i0} + \sum_k \delta_{ik} \theta_{ik} \gamma_k) / \omega_i \} N_i \\ \quad / (R_{i0} + \sum_k \delta_{ik} \theta_{ik} \gamma_k + \delta_{im} \theta_{im} \gamma_m + \varepsilon_i) \end{cases} \quad (3.7)$$

式(3.7)から γ_k, γ_m を求めることにより π_i が得られる。 $u_m < 1$ のときは式(3.7)の第1式(このとき式(3.3)と一致)のみを用いればよいから、式(3.7)はAMIIの方程式の一般形を与える。もちろん解は式(3.4)をみたすことが必要である。

4. AMIIの予測能力

漸近モデルAMIIの予測能力、求解手続きなどについて、AMIとの比較において述べる(AMIの詳細は1)を参照されたい)。AMIとAMIIとの相違は等優先度の扱いにある。実システムで等優先度のトランザクションにフェーズ・シェア・サービスを行なう場合はAMI, FCFSを行なう場合はAMIIの予測精度がそれぞれ優れると考えられる。なお、求解手続きはAMIIがAMIより複雑である。これはAMIの方程式が線形系に帰着でき容易に解析解が得られるのに対し、式(3.7)は非線形のためである。

式(3.7)により解が求まるための条件につき考察する。いま、ボトルネック・プロセッサの集合Hに属するプロセッサの個数を n ($|H|=n$)とし、プロセッサ j ($j=1, 2, \dots, n$)がボトルネックとする。トランザクション・クラス数を l とし、クラス i のトランザクションの各プロセッサにおける平均待ち時間の和 $\sum_j \tau_{ij}$ を b_i とかく。式(3.2)より、

$$b_i = \sum_{j=1}^n \delta_{ij} \theta_{ij} \gamma_j \quad (i=1, 2, \dots, l) \quad (4.1)$$

である。これをベクトル形式で表わすと次のようになる。

$$b = D \cdot \gamma \quad (4.2)$$

ただし、

$$b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_l \end{pmatrix}, \quad D = \begin{pmatrix} \delta_{11} \theta_{11} & \cdots & \vdots \\ \cdots & \delta_{ij} \theta_{ij} & \cdots \\ \vdots & \cdots & \delta_{ln} \theta_{ln} \end{pmatrix}, \quad \gamma = \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_n \end{pmatrix} \quad (4.3)$$

とする。式(3.7)の解が求まるためには、次式が成立することが必要であり、ほとんどの場合十分でもある(証明は付録)。

$$\text{rank}(D) = n \quad (4.4)$$

(ただし、式(4.4)が成立せず、 $\text{rank}(D) = n' < n$ の場合も、 $(n-n'+1)$ 個のボトルネック・プロセッサにおける各 γ の線形和を改めて γ_j とおき、ボトルネック・プロセッサの数を n' とみなすことにより求解が可能となる。)

ニュートン・ラフソン法を用いて式(3.7)の数値解を求めることができる。ここで式(3.7)は複数の解をもつ。したがって妥当な解を得るためには適切な初期値を設定する必要がある。筆者らは、端末数(呼源数)を1から順に増加していき、端末数 $(N-1)$ のときの解を初期値として端末数 N のときの解を求めるという方法をとった。

以上のようにして求めたAMIIの平均応答時間の予測結果を、AMIの予測結果およびGPS/Sシミュレーションの結果と比較したのが図4である。この例では N 個の端末は2等分され、それぞれクラスA, クラスBのトランザクションを発生する($N_A = N_B = N/2$)。なおトランザクションの資源使用特性はA, Bとも共通とするので、 $R_{i0}, d_{ij}, \theta_{ij}, \omega_i, \varepsilon_i$ などの諸変数における添字 i ($i=A, B$)

は以下省略する。使用する入出力装置は1個とし、これをI/Oと表わす。図4は、 $\theta_{CPU} = \theta_{IO} = 8$, $d_{CPU} = 160\text{ms}$, $d_{IO} = 340\text{ms}$ (したがって $R_0 = 500\text{ms}$), $\alpha = 20\text{sec}$, $\beta = 10$ の場合である。すなわちトランザクションはCPUを平均20ms使用するごとに入出力装置を平均42.5ms使用し、これを8回繰り返した後完了する。

GPPSシミュレーションにおいては、入出力装置のサービス時間としてディスクのランダム・アクセス時のシーク、サーチ時間分布を用いた。またCPUサービス時間および思考時間については、その分布が一般に不明なので、指数分布と一斉分布の2つの場合についてシミュレートした。実際の分布がアーラン分布であれば、求める値は得られた2つの値の間に位置する筈である。優先度はCPUおよびメモリについてクラスAがBより高く、入出力装置は等優先度でFCFSでサービスすると仮定した。これは実際のシステムで多くみられる形態である。

端末数 N を増していくと、 $N \geq 60$ で入出力ネックとなり、 $N \geq 69$ で入出力ネックかつメモリ・ネックとなる。入出力ネック ($60 \leq N < 69$) では、AMIとAMIIの予測結果は同一であり $T_A = T_B$ だが、これはクラスAとBでトランザクションの資源使用特性が同一のためである。入出力装置に加えてメモリがネックになると ($N \geq 69$)、AMIとAMIIの予測結果は異なる。シミュレーション結果では、 N を増加しても多重度は既に上限に達しているので入出力装置の待ち時間は一定であり、また T_B についてはこれにメモリ待ち時間が加わる。この傾向はAMIIで正しく予測されている。一方AMIでは、単位時間あたりの入出力装置の使用量がクラスA、Bトランザクションで等しいと仮定するので、入出力装置をFCFSで割り当てる場合の T_A と T_B を正しく予測することができない。

なお、AMIIで図4の予測結果を得るための計算量は400Kステップ程度であった。AMIでは解析解が得られるので計算量はほとんど問題とならない。

5. 実測による検証

実システムの測定データとAMIIによる予測結果を比較することにより、AMIIの予測精度を検証した。実システムはHITAC M200H、メモリ容量8-12M Bytes、入出力装置としてH-8589-1ディスク4台とH-8595-1ディスク11台、端末35-38台からなる。アプリケーションは端末から起動されるトランザクションとジョブ・キューからイニシエータにより起動されるバッチ・トランザクションからなる。端末起動トランザクションはI (Interactive) 型すなわちエディット系の短小なコマンド処理と、E (Executive) 型すなわちラン系のプログラム翻訳実行コマンド処理が約30対1の比率で混在する。またバッチ・トランザクションは50種類のFortranプログラムの翻訳実行処理であり、イニシエータ数は4-5本である。した

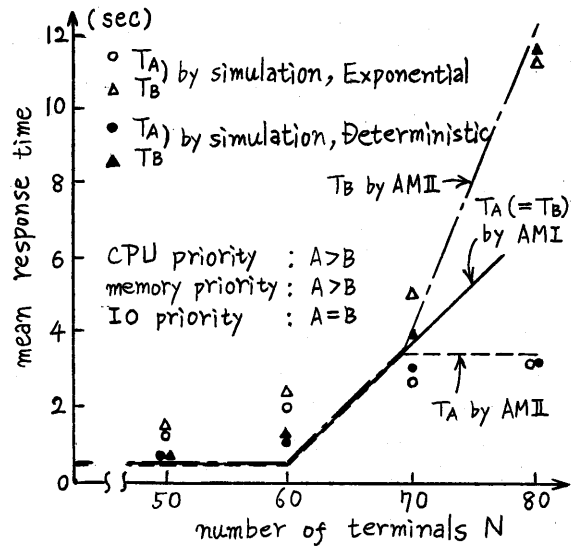


図4 AMIとAMIIの予測能力の比較

がって予測モデルでは, I型端末起動, E型端末起動, バッチの3つのトランザクション・クラスを設定する。ボトルネック資源はCPUとスプール用ディスクであった。実システムではCPUに関しI型端末起動トランザクションの優先度が最も高く, E型端末起動トランザクションに対する単位時間あたりのCPU割当て量がバッチ・トランザクション

表1 AMIIによる平均応答/処理時間予測の検証
(上段:予測値, 下段:実測値)

case no.	number of terminals	number of batch initiators	memory capacity (MB)	I-type terminal initiated	E-type terminal initiated	Batch
1	33	4	8	{ 0.21 0.36	{ 4.70 3.36	{ 21.6 22.9
2	52	4	8	{ 0.21 0.62	{ 5.05 4.53	{ 23.6 25.5
3	58	5	12	{ 0.21 0.43	{ 6.40 4.99	{ 30.3 30.2

(sec)

に対するそれをやや上回るようスケジュールが行なわれる⁸⁾。特殊な場合を除き, CPU割当て量にもとづくスケジュールの効果は予測が難しいので, 予測モデルでは2レベルのCPU割当て優先度を設定し, I型端末起動トランザクションを高, E型端末起動およびバッチ・トランザクションを低とした。またスプール用ディスクについては, 端末起動トランザクションはI型E型ともにこれをほとんど使用しないので優先度は問題とならない。

なお実測データによればメモリはボトルネックでなく, トランザクションごとのページングの挙動も明確でなかったので, ページングとスワッピングは予測モデルから除外した(実システムでは, メモリに余裕がある限り思考中端末のworking setもメモリ内に保つというデマンド・スワッピング方式を採用しているため, トランザクション到着ごとにスワッピングが発生するとは限らない)。

端末数, バッチ・イニシエータ数, メモリ容量をパラメータとして, ケース1-3の3つの場合について平均応答/処理時間の実測データとAMIIの予測結果を比較したのが表1である。I型端末起動トランザクションについては, 負荷を増やしたかによって誤差の増大がみられる。これは, 高優先度トランザクションどうしの待ちを評価できないというボトルネック解析の限界を示している。さらにケース2と3の実測データを比べると, 負荷増大にもかかわらず後の方が応答性がよいが, これはメモリ増加にともないデマンド・スワッピング方式によるスワッピング省略の確率が増したためと考えられる。ページング, スワッピングを考慮すればこれに関連した誤差短縮は可能であろう。なお高優先度トランザクションの平均応答時間をより高い精度で予測する方法として, 既存の待行列網理論をI型端末起動クラスのみについて用いることが考えられる。これは, 低優先度クラスによる影響が小さいためである。既存手法とAMIIとの組合せは今後の課題である。

E型端末起動トランザクションについては, 予測誤差は11-40%である。実測データの方が全般に値が小さいが, この理由はCPU割当てにおけるE型端末起動トランザクションの優遇が予測モデルに反映されていないためである。一方, 長大なバッチ・トランザクションについては, 予測誤差は0-7%にとどまった。これはAMIIによる, 優先度をもつシステムの大局的挙動, 特に低優先度トランザクションの挙動の予測可能性を示している。

6. あすび

ボトルネックとなる資源での待ちに着目することにより、メモリ、CPU、入出力装置の割当てに優先度のある多重プログラミング・システムの性能をその漸近解で簡便に予測する「漸近モデルⅡ (AMⅡ)」を提案した。既に提案したAMⅠとは異なり、等優先度の呼に対して「資源1回使用あたりの平均待ち時間が同一」と仮定した。この結果、等優先度の呼にFCFSで資源を割り当てるシステムにおいては予測精度が向上することをシミュレーションにより確かめた。さらにAMⅡの予測能力を実測データにより検証した。従来理論的解析が困難であった優先スケジュールを行なうシステムの大局的挙動の予測、特に低優先度の呼の挙動の予測にAMⅡが有効であることを確認した。

今後の課題として、既存の待行列網理論とAMⅡとの併用により高優先度の呼の挙動予測精度を向上させることなどがあげられる。

終りに本研究について御指導いただいた東京大学大須賀節雄助教授、ならびに本研究の機会を与えて下さった当社システム開発研究所三浦武雄所長、同ソフトウェア工場服部陽一部長、有益な助言を与えて下さった同ソフトウェア工場野口健一郎主任技師、同システム開発研究所大町一彦主任研究員、の諸氏に深く感謝いたします。

(参考文献)

- 1) Nishigaki, T. et al.: An Approach to the GRM Performance Analysis by Asymptotic Approximation, JIP, Vol.3, No.2, pp.59-67 (1980).
- 2) Kleinrock, L.: Certain analytic results for time-shared processors, Proc. IFIP, pp.838-845 (1968).
- 3) Muntz, R.R. et al.: Asymptotic properties of closed queueing network models, Proc. 8th Annu. Princeton Conf. on Inf. Sci. Syst., pp.348-353 (1974).
- 4) Demming, P.J. et al.: The Operational Analysis of Queueing Network Models, Comp. Surv., Vol.10, No.3, pp.225-261 (1978).
- 5) Baskett, F. et al.: Open, closed, and mixed networks of queues with different classes of customers, J. ACM, Vol.22, No.2, pp.248-260 (1975).
- 6) Sevick, K.C.: Priority scheduling disciplines in queueing network models of computer systems, Proc. IFIP, pp.565-570 (1977).
- 7) Neuse, D. et al.: A Method for Approximate Analysis of General Queueing Networks, University of Texas at Austin Technical Report TR-153 (1980).
- 8) Nishigaki, T. et al.: An Experiment on the General Resources Manager in Multi-programmed Computer Systems, JIP, Vol.1, No.4, pp.187-192 (1979).
- 9) 日立製作所: H-8549-2 ディスク制御装置 H-8589 ディスク駆動装置, ハードウェア・マニュアル 8080-2-007 (1974).

(付録)

式(4.1)の b_i を用いて式(3.7)を次のように書き直す。

$$\begin{cases} 1 = \sum_{i=1}^n d_{ij} N_i / (R_{i0} + b_i + \delta_{im} \theta_{im} \psi_m + \varepsilon_i) & (j=1, 2, \dots, n) \\ 1 = \sum_{i=1}^n \{ (R_{i0} + b_i) / \alpha_i \} N_i / (R_{i0} + b_i + \delta_{im} \theta_{im} \psi_m + \varepsilon_i) \end{cases}$$

まず $\text{rank}(D) \leq n$ は明らか。 $\text{rank}(D) \leq n-1$ のとき、独立変数 b_i の数は $(n-1)$ 個以下となり、上式の解は一般に存在しない。したがって式(4.4)は必要。一方、式(4.4)が成立するとき、上式は $(n+1)$ 個の独立変数 $b_{i1}, b_{i2}, \dots, b_{in}, \psi_m$ により表わせる。係数が特殊な場合を除き、この方程式は解をもち、結局式(4.1)より $\psi_1, \psi_2, \dots, \psi_n, \psi_m$ が求まる。