資源割当て優先度のある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††

メモリ，CPU，入出力装置など諸資源の割当てに優先度のある多重プログラミング・システムにおいて，その大局的性能を簡便に予測する漸近モデル AMII (Asymptotic Model II) を提案する．AMII の特徴は，利用率が高くボトルネックとなる資源での待ちに着目し，性能値をその漸近解で近似する点にある．漸近解は資源サービス時間の平均値より定まり分布形にはよらないことから，従来困難であった優先スケジュールおよび二次資源であるメモリの扱いが可能である．AMII の方程式を示し，その求解手続き，計算量，性能予測能力などにつき，すでに筆者が提案した漸近モデル AMI との比較において論ずる．AMII の予測データを実測データにより検証し，その有効性ならびに適用条件について述べる．

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．とくに重要なのは，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，待行列網理論での求解は一般に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うことが多いので，この場合の簡便な性能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いくつかの近似手法が提案されている．Sevcik 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[6]．また最近 Neuse らは 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[7]．Neuse らの手法は 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)」を提案した[1]．その特徴は，システムのボトルネックとなる資源に着目し，平均応答時間，資源利用率などの性能値をその漸近解で近似する点にある．漸近解は資源サービス時間の平均値のみから定まり分布形にはよらないので，分布形が複雑で厳密解が求まらないシステムをも解析することができる．性能予測の実際的目的は設計，改造する計算機システムが処理できる負荷量の見積りなので，正確な性能値は不明でもその大局的，構造的性質が求まれば十分な場合も多い．漸近モデルは性能の構造的性質を分析する手がかりを与える．なお，平均応答時間が呼源数の関数として漸近解をもつことは Kleinrock[2], Muntz[3], Denning[4] らにより示されたが，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，優先度のある複数クラスおよび二次資源のメモリを扱い可能としたものであり，性能評価ツール ISCP[10] (Interactive tool for System Configuration Planning) の一環をなす．

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，すでに提案したモデルを AMI，本論文で述べるモデルを AMII と略称する.) AMI と AMII 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

† Bottleneck Analysis of a Multiprogrammed Computer System with Priority Scheduling Disciplines by TOHRU NISHIGAKI and AKIRA YAMAMOTO (Systems Development Laboratory, Hitachi, Ltd.).

†† (株)日立製作所システム開発研究所

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，AM II はこのようなシステムの大局的挙動の予測を目的としたものである．

まず第2章で解析するシステムのモデルを定義し，第3章で AM II の方程式を示す．第4章では AM II の求解の条件につき考察し，シミュレーション結果をもとにその性能予測能力を検討する．さらに第5章では AM II の性能予測能力を実測データにより検証する．

2. ボトルネック解析のシステム・モデル

2.1 前提条件

用語および前提条件を整理する．資源の要求単位を「トランザクション」とよぶ．トランザクションは TSS コマンドやバッチ・ジョブに対応する．資源使用特性と優先度が共通のトランザクションの集合を「クラス」とよぶ．また，資源のうちで CPU と入出力装置をともに「プロセッサ」とよぶ（本モデルでは資源サービス時間の平均値のみに着目するので，両者を区別する必要はない）．

次に AM II における仮定を列挙する．

「仮定 1：ジョブ・キューでの待ちは本論文の対象外とし，ジョブ・キューには常にトランザクション（バッチ・ジョブ）が存在するとする．」

「仮定 2：ひとつのトランザクションが複数のプロセッサを同時に使用することはないとする．」

「仮定 3：トランザクションからある資源へのサービス要求が出されたとき，システム全体として当資源を 100% 使用するに足るだけのサービス要求（呼源）がない場合には当資源使用のための待ち時間はゼロとする．またこれ以外の場合すなわちサービス要求の総量（呼源数）が当資源の能力を越える場合には，当資源は常に 100% 使用されるとする．」

「仮定 4：優先スケジュールは割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，より優先度の低いトランザクションからの影響は無視する．優先度が低いトランザクションには，優先度が高いトランザクションの使用した残りの資源が割り当てられる．」

「仮定 5：等優先度のトランザクションについては，『資源1回使用あたりの平均待ち時間が同一』となるサービスが行われるとする．」

2.2 システム・モデル

システムのモデルを提示し，さらに前節で述べた仮定について述べる．システムは**図1**に示す「有限多重度の有限呼源モデル」で表されるとし，その定常状態について論ずる．各資源についてそのサービス時間分布は任意である．図1で，各端末は思考時間が終わるごとにトランザクションを発生する．仮定1より，バッチ・ジョブについてはジョブ・スケジューラ（イニシエータ）を思考時間ゼロの端末とみなすことができる．

漸近解を得るための準備として基礎的な関係式を以下にみちびく．メモリ容量を V，トランザクションの平均ワーキングセット・サイズ(平均メモリ占有量)を w とすると，最大多重度 s は平均的には次式で与えられる．

$$s = V/w \tag{2.1}$$

思考中以外の端末の数が s を越えると，越えた分だけのトランザクションはメモリに入りきれずメモリ割当て待ちとなる．

トランザクション・クラス i の平均応答時間（バッチの場合は平均処理時間），平均メモリ占有時間，平均思考時間，最大多量度をそれぞれ T_i, R_i, z_i, s_i と表す．s_i は式(2.1)の w としてクラス i トランザク

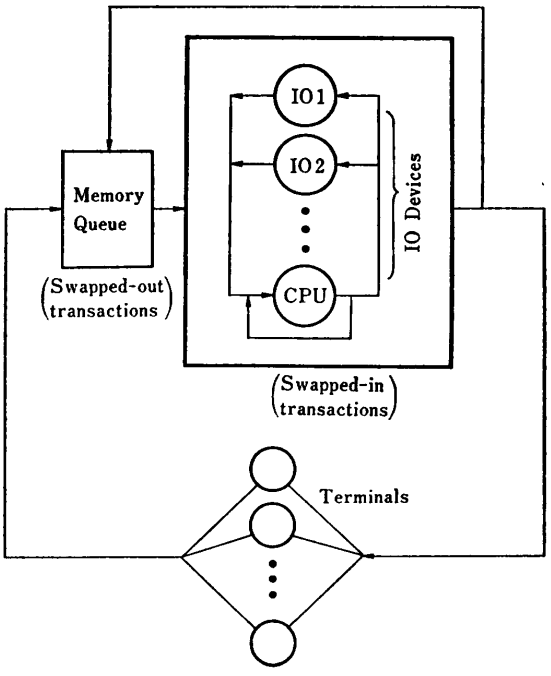


図 1 システム・モデル—有限多重度の有限呼源モデル
Fig. 1 The system model—finite population model with limited degree of multiprogramming.

ションの平均ワーキングセット・サイズ w_i を用いることにより得られる．ここで T_i は R_i にクラス i トランザクションの平均メモリ割当て待ち時間を加えた値に等しい．クラス i 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，トランザクション間の競合が無く待ちが生じない場合の平均応答時間 R_{i0} は仮定2より次式で与えられる．

$$R_{i0}=\sum_j d_{ij} \tag{2.2}$$

R_{i0} は R_i の下限値を表す．

クラス i トランザクションは，平均として (T_i+z_i) の間にプロセッサ j を d_{ij} 時間，メモリを R_i 時間，それぞれ使用する．したがってプロセッサ j の利用率 u_j およびメモリ利用率 u_m は次式で与えられる[1]．

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

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)は各資源の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% とみなすことにより，漸近解が得られる．これを表現したのが仮定3である．具体的には，資源割当て待ちが無いと仮定して計算した資源利用率が1より小のとき「待ち時間=0」の方程式，これが1以上のとき「資源利用率=1」の方程式から，それぞれ漸近解が得られる．漸近解は性能の構造的性質を表しており，資源サービス時間の平均値のみに依存し分布形にはよらない．実際には資源利用率が1未満の場合でも分布形に応じて確率的な資源割当て待ちは発生するが，これは近似誤差となって現れる．

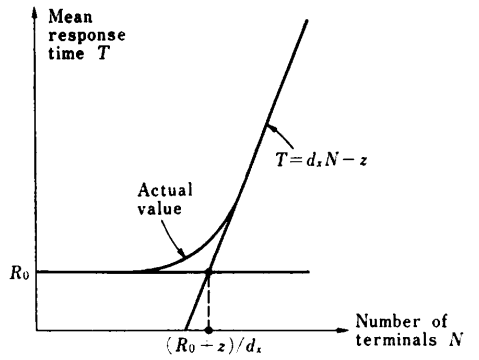
単一クラスの漸近解について以下述べる．まずメモリ容量 V が無限大の場合には，仮定3と式(2.3)とから，次のよく知られた平均応答時間の漸近解がみちびかれる[2)~4)]（**図2**参照）．

$$T=\begin{cases} d_x N-z & : (R_0+z)/d_x \le N \\ R_0 & : \text{上記以外の } N \end{cases} \tag{2.4}$$

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 はボトルネックになる可能性のあるプロセッサ $(d_x=\max_j d_j)$ を表す．

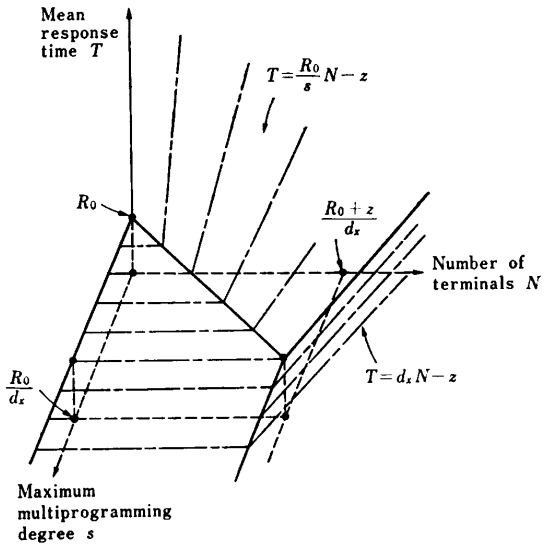
メモリ容量 V が有限の場合，メモリもボトルネックになりうる．この漸近解は式(2.5)で与えられる（**図3**参照）．証明は 1) を参照されたい．

$$T=\begin{cases} d_x N-z & : u_x=1 \text{ をみたす } N \\ (R_0/s)N-z & : u_x<1 \text{ かつ } u_m=1 \text{ をみたす } N \\ R_0 & : \text{上記以外の } N \end{cases} \tag{2.5}$$



d_x: max (the mean use time of processor j by a transaction)
R_0: the response time without queueing delays
z: mean think time

図2 無限多重度の場合の平均応答時間 T の漸近解
Fig. 2 The asymptotes of the mean response time T with unlimited degree of multiprogramming.



d_x: max (the mean use time of processor j by a transaction)
R_0: the response time without queueing delays
z: mean think time

図3 有限多重度の場合の平均応答時間 T の漸近解
Fig. 3 The asymptotes of the mean response time T with limited degree of multiprogramming.

複数クラスの場合の優先度の扱いは仮定4に示したものである．preemptive な優先度を仮定したが，実システムでは入出力装置はいったんサービスが開始されると処理完了まで割当て変更はされないので pre-emptive ではない．また入出力実行中のトランザク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．したがって漸近解は，高優先度トランザクション，低優先度トランザク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．

最後の仮定5は，仮定 1～4 と異なり，漸近モデルに共通の仮定ではなく AMII 特有の仮定である．等優先度トランザクションに対するサービスをいかに仮定するかにより，優先度のある漸近モデルの方程式の形は一般に異なる[1]．

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 が無限大の場合について述べる．このときメモリ割当て待ちは無視できる．

プロセッサ j を使用するトランザクション・クラスのうち，j の割当て優先度が最低のクラスの集合を y_j とする．(j が高優先度のトランザクションで占有されると，低優先度トランザクションの j での待ちは無限大となり，その平均応答時間は無限大となって以下に述べる方程式に影響を与えない．したがって実際には，y_j は j で有限の待ちが生ずるトランザクションの集合にほかならない．) また，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

$$T_i=R_{i0}+\sum_j \tau_{ij} \tag{3.1}$$

である．全てのクラスについて T_i は有限とすれば，仮定3と仮定4よりボトルネック資源の最低優先度トランザクションの待ちのみを評価すれば十分である．また $\tau_{ij}\neq 0$ のとき，仮定5より τ_{ij} は i による j の平均使用回数 θ_{ij} に比例する．以上より，1回 j を使用するための平均待ち時間を ψ_j とすると次式がなりたつ．

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

式(3.1)および(3.2)を式(2.3)第1式に用いて次式を得る．

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

式(3.3)より $\psi_k(>0)$ を求め，式(3.1)，(3.2)より T_i $(i=1,2,\cdots)$ 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

$$0\leq u_j\leq 1 \quad (j=1,2,\cdots) \tag{3.4}$$

が成立することが必要である．いいかえると，所与の N_i 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．メモリ・ネックすなわち $u_m=1$ のときの解は次のようにして求まる．まず，メモリの割当て優先度が最低のクラスの集合を y_m と表す．クラス i 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．

$$\begin{cases}R_i=R_{i0}+\sum_i \tau_{ij} \\ T_i=R_i+\tau_{im}\end{cases} \tag{3.5}$$

$\tau_{im}\neq 0$ のとき，仮定5より τ_{im} は i のメモリ平均使用回数 θ_{im} に比例するから式(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．

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

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．

$$\begin{cases}1=\sum_i d_{ij}N_i/(R_{i0}+\sum_k \delta_{ik}\theta_{ik}\psi_k+\delta_{im}\theta_{im}\psi_m+z_i) \\ \qquad (j\in H) \\ 1=\sum_i\{(R_{i0}+\sum_k \delta_{ik}\theta_{ik}\psi_k)/s_i\}N_i \\ \quad /(R_{i0}+\sum_k \delta_{ik}\theta_{ik}\psi_k+\delta_{im}\theta_{im}\psi_m+z_i)\end{cases} \tag{3.7}$$

式(3.7)から ψ_k, ψ_m を求めることにより T_i $(i=1,2,\cdots)$ 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 の性能予測能力

漸近モデル AMII の性能予測能力，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ア・シェア・サービスを行う場合は AMI，FCFS の場合は AMII がそれぞれ対応する．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る[1]のに対し，式(3.7)は非線形のためである．

式(3.7)の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 $(j=1,2,\cdots,n)$ 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

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

である．これは次のようなベクトル形式で表せる．

$$\boldsymbol{b}=\boldsymbol{D}\cdot\boldsymbol{\phi} \tag{4.2}$$

$$\boldsymbol{b}=\begin{pmatrix}b_1\\b_2\\\vdots\\b_l\end{pmatrix},\quad \boldsymbol{D}=\begin{pmatrix}\delta_{11}\theta_{11} & & & \\ & \ddots & & \\ \cdots\cdots & & \delta_{ij}\theta_{ij} & \cdots\cdots \\ & & & \ddots \\ & & & \delta_{ln}\theta_{ln}\end{pmatrix},\quad \boldsymbol{\phi}=\begin{pmatrix}\phi_1\\\phi_2\\\vdots\\\phi_n\end{pmatrix} \tag{4.3}$$

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十分である（証明は付録)．

$$\mathrm{rank}(\boldsymbol{D})=n \tag{4.4}$$

(ただし，式(4.4)が成立せず $\mathrm{rank}(\boldsymbol{D})=n'<n$ 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 ϕ の線形和を改めて ϕ_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．

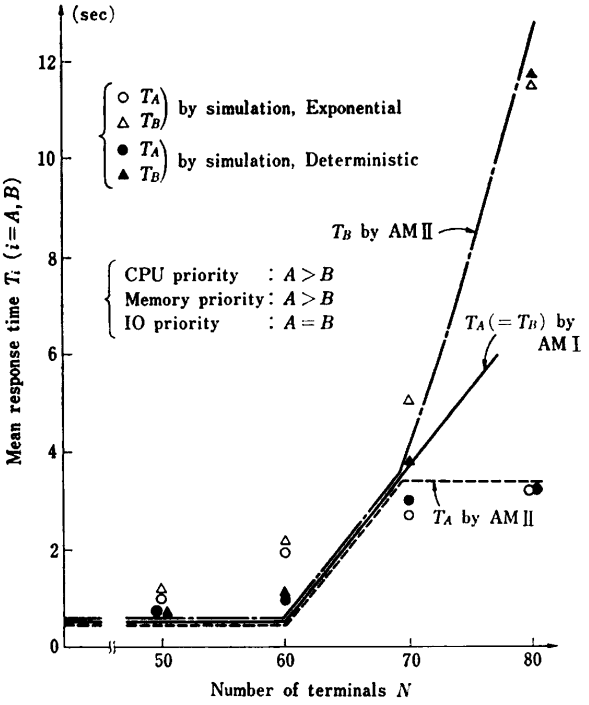


図 4 AMI と AMII の性能予測能力の比較
Fig. 4 Performance predictability of AMII in comparison with AMI.

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 GPSS 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，$R_{i0}, d_{ij}, \theta_{ij}, s_i, z_i$ 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 IO と表す．図4は，$\theta_{CPU}=\theta_{IO}=8$，$d_{CPU}=160$ ms，$d_{IO}=340$ ms (したがって $R_0=500$ ms)，$z=20$ sec，$s=10$ の場合である．すなわちトランザクションは CPU を平均 20 ms 使用するごとに入出力装置を平均 42.5 ms 使用し，これを8回繰り返した後完了する．

GPSS シミュレーションにおいては，入出力装置のサービスをシミュレートするために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[9]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，指数分布と一点分布の二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二つの値の間に位置するはずである．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 で割り当てると仮定した．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によれば，N を増加しても多重度はすで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．このような性能の構造的性質は AMⅡ で正しく予測することができる．一方 AMⅠ では，単位時間あたりの入出力装置使用時間がクラス A，B いずれのトランザクションでも等しいと仮定するので，本例のように入出力装置を FCFS で割り当てる場合には T_A と T_B を正しく予測することができない．

本例に限らず一般に，等優先度トランザクションに対し FCFS でサービスを行うシステムに対しては AMⅠ でなく AMⅡ を用いるべきである．ただし漸近解は確率的待ちを含まないため，実際の性能値と予測値の間には常に誤差が生ずる．誤差の大小は資源サービス時間の分布形に依存し，一般にその分散値とともに増大する[1]．たとえば図4で入出力ネック状態 $(60 \leq N < 69)$ に着目すると，CPU サービス時間と思考時間が指数分布にしたがうとき，一点分布の場合に比べて顕著な誤差がみられる．この原因は，確率的待ちによって平均応答時間が増大し，入出力装置の利用率を示す式(3.7)の右辺が1より小さくなるためである．しかし性能の構造的性質，たとえば負荷を増したとき性能値が急激に変化する限界条件などは，漸近解より定まり資源サービス時間の分布形にはよらない．AMⅡ によって解析できるのは性能の構造的性質であり，性能値自体の予測は必ずしも可能とはいえない．

なお，AMⅡ で図4の予測結果を得るための計算量は 400 K ステップ程度であった．AMⅠ では解析的に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 の予測能力を検証した．実システムは HITAC M200H，メモリ容量 8～12 M 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 33～58 台からなる．アプリケーションは，端末から起動される TSS のトランザクションと，ジョブ・キューからジョブ・スケジューラにより起動されるバッチ・トランザクションとからなる．端末起動トランザクションはⅠ(Interactive) 型すなわち入力編集などの短小なコマンド処理と，E (Executive) 型すなわちプログラム翻訳実行などのやや長大なコマンド処理とが約 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，バッチ多重度（イニシエータ数）は4～5である．したがって AMⅡ 予測モデルでは，Ⅰ型端末起動，E型端末起動，バッチの三つのトランザクション・クラスを設定した．

ボトルネック資源は CPU とスプール用ディスクであった．CPU に関し，実システムでは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・トランザクションに対するそれをやや上回るようスケジュールが行われる[8]．CPU 割当て量にもとづくスケジュールを行った場合の予測は一般に困難なので[1]，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，優先度は無く，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．

なお実システムにおいて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らかではなかったので，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ワーキングセット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，バッチ多重度(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 の三つの場合について平均応答時間（バッチの場合は平均処理時間）の

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

Table 1 Validation of AMII prediction of mean response time/mean processing time. (upper row : predicted, lower row : measured)

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	

(I : Interactive, E : Executive) (sec)

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，負荷を増すにしたがって誤差の増大がみられる．これは，低優先度トランザクションが有限の平均応答時間をもつとき高優先度トランザクションどうしの待ちを評価しないという，ボトルネック解析の限界を示している．一般に，高優先度トランザクションに生ずる確率的待ちを評価する目的には，AMII は有効ではない．表1で実測値は予測値の2倍程度の値となっているが，前述のように誤差は資源サービス時間の分布形に依存し，場合に応じてこれより増加ないし減少する．平均応答時間が急速に変化する負荷条件など性能の構造的性質は AMII により求まるが，性能値自体の評価にはむしろ既存の待行列網理論が有効な場合もある．すなわち，高優先度トランザクションについては低優先度トランザクションによる影響は小さいとしてこれを無視すると，優先度の無い単一クラス・モデルによる扱いが可能である．本例でも，既存の待行列網理論[5]をI型端末起動トランザクションのみに対し適用すれば，表1より優れた近似精度が得られると考えられる．なお，ケース2とケース3の実測データを比較すると，負荷増大にもかかわらず後者の方が応答性が向上しているが，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推定される．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 割当てにおいてE型端末起動トランザクションをバッチ・トランザクションより優遇するスケジュールが予測モデルに反映されていないためである．

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．一般に優先度のあるシステムの性能予測で問題となるのは，低優先度トランザクションがより高い優先度（同一優先度を含む）のトランザクションから受ける影響の評価である．利用率が100% に近いボトルネック資源が存在し，高優先度トランザクションに生ずる確率的待ちがボトルネック資源の利用率に与える影響が小さいとき，低優先度トランザクションの性能値を AMII で予測することができる．理由は式(3.7)より明らかである．高優先度トランザクションの確率的待ちの大小は資源サービス時間の分布形に依存するが，本例のようにボトルネック資源の利用率の中で高優先度トランザクションによる部分の占める割合が小さいとき，この影響をほぼ無視できる．表1においてバッチ・トランザクションの予測誤差が比較的小さいのは，以上の理由による．一方，各資源の利用率がいずれも中程度以下のときなど上記条件が成立しない場合は，低優先度トランザクションの性能値自体を AMII で予測することは有効ではない．

6. むすび

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．漸近解は，システム性能の構造的性質，たとえば負荷を増していったとき性能値が急激に変化する限界条件などを分析する手がかりを与える．すでに提案した AMI[1] とは異なり，等優先度のトランザクション(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，AMI では扱えなかった，等優先度トランザクションに FCFS で資源を割り当てるシステムの解析が可能となった．さらに AMII の予測能力をシミュレーションおよび実測により検証した．従来理論的解析が困難であった優先スケジュールを行うシステムの大局的挙動の予測に AMII が有効であることを確認した．ただし，AMII によって正確な性能値を予測することは，必ずしも可能ではない．AMII は確率的待ちを評価できないため，常に予測誤差が生ずる．たとえば，各資源の利用率が中程度の場合，性能値自体の予測には AMII は有効ではない．比較的小さい予測誤差を得るための条

件についても考察を加えた.

今後の課題として，既存の待行列網理論と AMⅡとの併用により性能値の予測精度を向上させることなどがあげられる.

謝辞 終りに，本研究についてご指導いただいた東京大学大須賀節雄教授，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 Oueueing Network Models, *Proc. 8-th Annu. Princeton Conf. on Inf. Sci. Syst*, pp. 348-353 (1974).
4) Denning, 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) Sevcik, 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 Austine Technical Report TR-153 (1980).
8) Nishigaki, T. et al.: An Experiment on the General Resources Manager in Multiprogrammed 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).
10) 大町他：計算機システムの性能評価技法“ISCP”の開発，日立評論，Vol. 61, No. 12, pp. 65-68 (1979).

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.

$$\begin{cases} 1=\sum_{i=1}^{l} d_{ij}N_i/(R_{i0}+b_i+\delta_{im}\theta_{im}\psi_m+z_i) \\ \qquad\qquad (j=1,2,\cdots,n) \\ 1=\sum_{i=1}^{l}\{(R_{i0}+b_i)/s_i\}N_i \\ \quad /(R_{i0}+b_i+\delta_{im}\theta_{im}\psi_m+z_i) \end{cases}$$

まず $\mathrm{rank}(\boldsymbol{D})\leq n$ は明らか. $\mathrm{rank}(\boldsymbol{D})\leq n-1$ 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 $b_{i_1}, b_{i_2}, \cdots, b_{i_n}, \psi_m$ により表せる. この方程式は解をもち，結局式(4.1)より $\psi_1, \psi_2, \cdots, \psi_n, \psi_m$ が求まる.

(昭和 56 年 11 月 18 日受付)
(昭和 57 年 4 月 19 日採録)

訂　　正

第23巻第5号 pp. 562-569 (1982) に掲載しました西垣氏他の論文「資源割当て優先度のある多重プログラミング・システムのボトルネック解析」に次の訂正があります.

p. 565 右段の (3.3) 式

誤：$1=\sum_{j} d_{ij} N_i/\cdots$　　正：$1=\sum_{i} d_{ij} N_i/\cdots$