情報処理学会研究報告
IPSJ SIG Technical Report

Vol.2010-MPS-79 No.4
2010/7/12

メモリ資源のある計算機システムの待ち行列網による近似性能評価法の提案と評価

木下 俊之[†1] 高 秀 梅[†1]

計算機システムを性能評価する有効な方法のひとつに，待ち行列網による方法がある．本論文では，待ち行列網を用いてメモリ資源のある計算機システムを性能評価する近似法について述べる．ジョブは外部からシステムに到着するとメモリのひとつを占有し，メモリを占有しながら CPU 処理や入出力処理を実行する．ジョブが CPU 処理や入出力処理を完了すると，メモリを解放してシステムを出ていく．メモリは CPU や入出力装置に対して２次資源と考えられ，かつメモリ資源のある計算機システムの待ち行列網は開モデルであり，積形解を持たないため厳密解を求めることはできない．

近似の考え方は，対象とする計算機システムを CPU 処理や入出力処理を実行する“処理部”と，ジョブがメモリをどう使うかを評価する“メモリ部”に分けて解析する．待ち行列網を２つの部分に分割することにより，網の状態数が増大して数値計算を実行できなくなることを防ぐことができる．数値実験により，提案した近似法の有効性を評価した．

Queuing Network Approximation Technique for Evaluating Performance of Computer Systems with Memory Resources

Toshiyuki Kinoshita and Xiumei Gao

A queuing network technique is effective for evaluating the performance of computer systems. We discuss the queuing network technique of computer systems with memory resources. When a job arrives from outside the network, it occupies one of the memory resources, and executes CPU and I/O processing in the network occupying the memory. When the job completes CPU and I/O processing, it releases the memory and leaves the network. Since the memory resource is considered as a secondary resource for the CPU and I/O equipment, and the queuing network model of computer systems with memory resources is an open one, we cannot calculate its strict solutions.

In this paper, we propose an approximation technique for calculating the performance measures of computer systems with memory resources using the queuing network technique. In our proposed technique, we divide a network into two parts, one is ”processing part” in which a job executes CPU and I/O processing and the other is ”memory part” which indicates how the memory resources are used by jobs. By dividing the network into two parts, we can prevent increasing the number of states of the network and approximately calculate the performance measures of the network. We evaluate the proposed approximation technique using numerical experiments.

1. はじめに

計算機システムの性能を評価する手法のひとつに，待ち行列網を用いる方法がある．計算機の中には複数のジョブが存在し，ハードウェアやソフトウェアなどの資源を取り合うことにより待ちが発生している．この待ちを待ち行列網を用いて評価し，ハードウェアの利用率やジョブの応答時間などの性能指標を求める方法である．待ち行列網のうち，ある条件を満たすものは積形解と呼ばれる明示的な解を持つ [1)]．この積形解を用いれば，性能指標を容易に計算することができる．しかし資源を競合に排他制御を行う場合やメモリ資源を扱う場合などには，網は積形解を持たない．積形解を持たない待ち行列網の厳密解を求めるためには，待ち行列網の確率的な特徴を記述するマルコフ連鎖の平衡方程式を数値的に解く必要がある．この平衡方程式は，網の状態数と同数の未知数を持つ連立１次方程式になる．一般に計算機システムをモデル化した待ち行列網の状態数は，CPU や I/O 装置数，システム内のジョブ数が大きくなると指数関数的に増大する．このため連立１次方程式の未知数の個数が増大し，これ解くための数値計算が実行できなくなるという問題点を含んでいる．また，網が外部との出入りのある開モデルの場合の状態数は無数であり（システム内のジョブ数は無限大まで大きくなり得る），平衡方程式を数値的に解くことは事実上不可能である．

本研究ではメモリ資源のある計算機システムを考える．ジョブは外部から網に到着するとメモリのひとつを占有し，これを占有しながら CPU 処理や I/O 処理を実行する．ジョブが CPU 処理や I/O 処理を完了すると占有していたメモリを解放し，網を出ていく．これより，メモリは CPU や入出力装置に対して 2 次資源と考えることができる．待ち行列網が 2 次資源を含むときは積形解を持たない．そしてメモリ資源のある待ち行列網は外部との出入りのある開モデルのため，積形解を持たない場合は厳密解を求めることはできない．そこ

†1 東京工科大学大学院コンピュータサイエンス専攻
Graduate School of Computer Science, Tokyo University of Technology

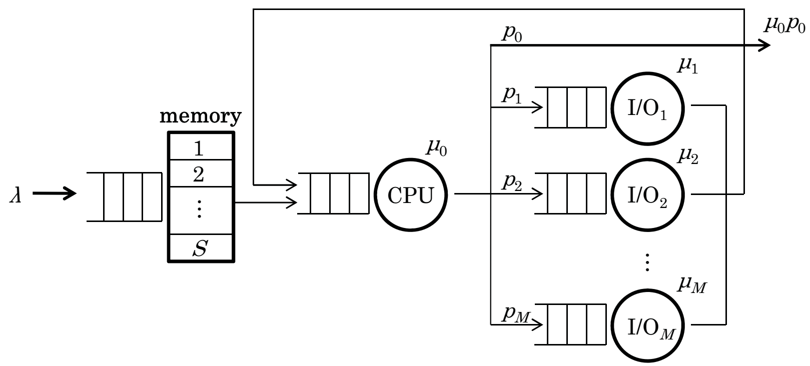

図 1 メモリ資源のあるセントラルサーバモデル
Fig. 1 Central server model with memory resources

で本研究では，メモリ資源を持つ待ち行列網の解を求める近似手法を提案する．近似の考え方は，網を記述するマルコフ連鎖の状態数が増大して数値計算を実行できなくなることを防ぐため，CPU 処理や I/O 処理を実行する "処理部" とメモリ資源の占有，解放を記述する "メモリ部" に分けて解析する．この 2 つの部分はともに積形解を持つので，容易に定常確率を求めることができ，後で両者を結合して全体の性能指標を計算するという方法である．このように対象のシステムを 1 次資源（CPU，I/O 装置など）と 2 次資源（メモリ資源）に分割してモデル化する方法は，階層型持ち行列の手法である [3)4)]．本研究は，階層型待ち行列をメモリ資源のある待ち行列網に応用したひとつの応用例と考えることができる．

2. モデルの記述

モデルを分割したうちの処理部（CPU，I/O 処理が実行される部分）は，従来のセントラルサーバモデルと同一で，1 個の CPU ノードと複数の I/O ノードからなる．I/O ノード数を M で表し，I/O ノードは 1 から M まで添え字 m で番号付けられる．$m=0$ のときは CPU ノードを表すものとする．CPU ノードでのサービス率を μ_0，I/O ノード m でのサービス率を μ_m とする．あるノードでのサービス時間は共通の指数分布に従う互いに独立な確率変数で，他のノードでのサービス時間とも独立である．すべてのノードでジョブは FCFS（First Come First Served）規律でスケジュールされる．ジョブは CPU ノードでの処理終了時に，次にいずれの I/O ノードに進むか，または処理を完了して網から離れるかを，確率的に選択する．I/O ノード m の選択確率を $p_m (m=1,2,\ldots,M)$ とし，処理の完了確率を p_0 とする（従って $\sum_{m=0}^{M} p_m = 1$ である）．

このセントラルサーバモデルにメモリ資源を付加する（図 1）．メモリ数を S で表す．ジョブは外部から到着率 λ でランダムに到着し，処理部に入る前にメモリ資源の一つを要求する．ジョブがメモリを要求したときにすべてのメモリが占有されていれば，ジョブはメモリ待ち行列に入ってメモリが解放されるのを待つ．ジョブはメモリを占有しながら処理部で CPU 処理や I/O 処理を実行し，CPU 処理や I/O 処理を完了するとメモリを解放して網から離れる．ジョブは処理部に入るときに必ずメモリを 1 つ獲得するので，メモリを獲得しているジョブ数は処理部のジョブ数と同数である．このことは処理部の最大ジョブ多重度がメモリ数 S に等しいことを意味する．処理部にいるジョブを $n\,(=0,1,2,\ldots,S)$ で表す．

ジョブの "CPU から外部への遷移" を "CPU から CPU への遷移" に置き換えることによって，処理部をジョブ数が一定の閉セントラルサーバモデルに変更することができる．この閉モデルでは，"CPU から CPU への遷移" が起きた場合は，そのジョブは終了して新しいジョブが生成したと考える．ゆえにジョブの平均応答時間は，引き続く 2 つの "CPU から CPU への遷移" 間の平均時間間隔になる．このジョブの平均応答時間は，ジョブの生存時間と考えることができる．

3. 近似の考え方

メモリ資源のある待ち行列網の厳密解を得るためには，網全体をひとつのマルコフ連鎖で記述する必要がる．しかしこの方法は網のノード数やジョブ数が増大した場合に，マルコフ連鎖の状態数の劇的な増加を引き起こす．そこで待ち行列網を処理部とメモリ部の 2 つの部分に分割し，それぞれをマルコフ連鎖で記述することにより，網の状態数の増加を防ぐことができる（図 2）．以下，次の記号を用いる．

τ_m : ジョブの生存中にノード m で受ける総サービス時間 $(m=0,1,\ldots,M)$
n_m: ノード m のジョブ数 $(m=0,1,\ldots,M)$
$\boldsymbol{n}=(n_0,n_1,\ldots,n_M)$: 処理部の状態ベクトル
$\boldsymbol{F}(n)=\{\boldsymbol{n}|\sum_{m=0}^{M} n_m = n, n_m \ge 0\,(m=0,1,\ldots,M)\} \quad (n=1,2,\ldots,S)$
: 処理部のジョブ数が n のときの処理部の可能なすべての状態の集合
$P_s(\boldsymbol{n})$: 状態 $\boldsymbol{n}$ の定常確率

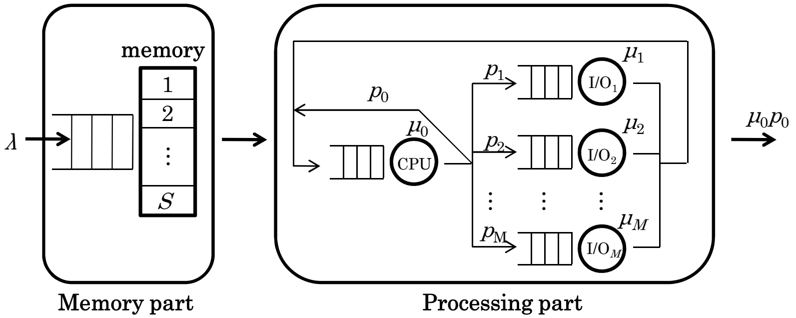

図 2 近似の考え方
Fig. 2 Concept of approximation

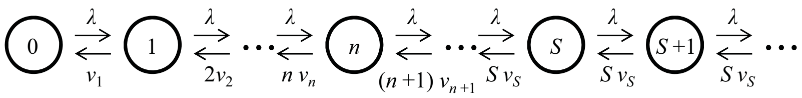

図 3 サービス率が可変の $M/M/S$ モデルの状態遷移
Fig. 3 Sate transition of modified $M/M/S$

この処理部は従来のセントラルサーバモデルと等価なので，処理部を記述するマルコフ連鎖は積形解を持つ．その処理部の定常確率 $P_s(\boldsymbol{n})$ は, 次式で求められる [2]．

$$P_s(\boldsymbol{n}) = \frac{\prod_{m=0}^{M} \tau_m^{n_m}}{\varphi(n, M)}$$

ここで $\varphi(n, M) = \sum_{\boldsymbol{n} \in \boldsymbol{F}(n)} \prod_{m=0}^{M} \tau_m^{n_m}$ は，処理部のジョブ数が n のときの定常確率の正規化定数である．この定常確率を用いて，処理部のジョブ数が n のときの処理部でのジョブの平均応答時間 T_n は，次の式で求められる．

$$T_n = \frac{n \cdot \varphi(n, M)}{\varphi(n-1, M)} \quad (n = 1, 2, \ldots, S) \qquad (1)$$

次にメモリ部は，窓口数が S 個の $M/M/S$ 待ち行列モデルであると考える．ただし通常の $M/M/S$ 待ち行列モデルでは、サービス中の客数に関わらず窓口でのサービス率は不変であるが、メモリ部の $M/M/S$ モデルはサービス中の客数によってサービス率が変化するモデルを考える。処理部のジョブ数が n のときの処理部でのジョブの平均応答時間 T_n は, メモリの平均占有時間に等しい．ゆえに処理部の完了率 ν_n は $\nu_n = \frac{1}{T_n} (\ldots (2))$ と表されるので，ν_n は占有されているメモリ数 n に依存して変化する．図 3 に，メモリを占有しているジョブ数に応じてサービス率が変化する $M/M/S$ 待ち行列モデルの状態遷移を示す．これは拡張された生成死滅過程である．メモリを占有しているジョブ数が n のときの定常確率 $Q_S(n)$ は，次のように表される．

$$\begin{aligned}
\lambda Q_S(0) &= \nu_1 Q_S(1) \\
(\lambda + n\,\nu_n)\, Q_S(n) &= \lambda Q_S(n-1) + (n+1)\,\nu_{n+1}\, Q_S(n+1) \quad (n = 1, 2, \ldots, S-1) \\
(\lambda + S\,\nu_S)\, Q_S(n) &= \lambda Q_S(n-1) + S\,\nu_S\, Q_S(n+1) \quad (n = S, S+1, \ldots)
\end{aligned}$$

これらの方程式を解いて，次のような積形解が得られる．

$$Q_S(n) = \begin{cases} Q_S(0)\, \dfrac{1}{n!} \displaystyle\prod_{i=1}^{n} \left(\frac{\lambda}{\nu_i}\right) & (n = 1, 2, \ldots, S) \\ Q_S(0)\, \dfrac{1}{S!\, S^{n-S}} \displaystyle\prod_{i=1}^{S} \left(\frac{\lambda}{\nu_i}\right) \cdot \left(\frac{\lambda}{\nu_S}\right)^{n-S} & (n = S+1, S+2, \ldots) \end{cases} \qquad (3)$$

式 (1) と (2) を (3) に代入すると，次の結果が得られる．

$$\begin{aligned}
Q_S(n) &= Q_S(0)\, \frac{1}{n!} \prod_{i=1}^{n} (\lambda T_i) \\
&= Q_S(0)\, \frac{1}{n!}\, \lambda^n \prod_{i=1}^{n} \frac{i\,\varphi(i, M)}{\varphi(i-1, M)} \\
&= Q_S(0)\, \frac{1}{n!}\, \lambda^n\, \frac{n!\,\varphi(n, M)}{\varphi(0, M)} \\
&= Q_S(0)\, \lambda^n\, \varphi(n, M) \qquad (n = 1, 2, \ldots, S) \qquad (4) \\
Q_S(n) &= Q_S(0)\, \frac{1}{S!\, S^{n-S}} \prod_{i=1}^{S} (\lambda T_i) \cdot (\lambda T_S)^{n-S} \\
&= Q_S(0)\, \frac{1}{S!\, S^{n-S}}\, \lambda^S\, \frac{S!\,\varphi(S, M)}{\varphi(0, M)} \times \left\{ \frac{\lambda \cdot S\,\varphi(S, M)}{\varphi(S-1, M)} \right\}^{n-S} \\
&= Q_S(0)\, \lambda^S\, \varphi(S, M) \left\{ \frac{\lambda\,\varphi(S, M)}{\varphi(S-1, M)} \right\}^{n-S} \qquad (n = S+1, S+2, \ldots)
\end{aligned}$$

処理部のジョブ数が n のときの処理部のジョブの平均応答時間 T_n を定常確率 $Q_S(n)(n=0,1,2,\ldots)$ で重み付けて平均することにより，処理部でのジョブの平均応答時間が求められる．また $Q_S(n)$ により直接に平均メモリ待ち時間が求められるので，この 2 つ加えることによりシステム全体のジョブの平均応答時間を求めることができる．提案する近似法のアルゴリズムは，次のようにまとめられる．

Step 1 処理部の閉セントラルサーバモデルの積形解を用いて，処理部のジョブ数が n のときの処理部でのジョブの平均応答時間 T_n を求める．

Step 2 (4) 式により、処理部のジョブ数が n のときのメモリ部の定常確率 $Q_S(n)$ を求める．

Step 3 $Q_S(n)$ により平均メモリ待ち時間を求め，これと処理部でのジョブの平均応答時間を加えて，システム全体のジョブの平均応答時間を求める．

4. 数値実験

提案した近似法を評価するために、数値実験を行った。使用したパラメータは次の通りである。

1. メモリ資源数　$S=2,4,6,8$
2. 到着率　$\lambda=0.01,0.02,\ldots,0.08,0.09$
3. I/O ノード数　$M=2$
4. ジョブの各ノードでのサービス率
   $\mu_0=1.0,\quad \mu_1=\mu_2=0.5$
5. CPU サービス終了後の次ノードの選択確率
   $p_0=1/9,\ p_1=p_2=4/9$
   従って各ノードでの総サービス時間は，次のようになる．

$$\tau_0=\frac{1}{\mu_0}\cdot\frac{1}{p_0}=9.0$$
$$\tau_1=\frac{1}{\mu_1}\cdot\frac{p_1}{p_0}=8.0$$
$$\tau_2=\frac{1}{\mu_2}\cdot\frac{p_2}{p_0}=8.0$$

図 4，5 に，メモリ資源数 S が 4 のときの処理部の平均応答時間とシステム全体の平均応答時間をそれぞれ示す．比較のために，処理部のジョブ数が最小の $S=1$ と仮定した場合と，最大の $S=4$ とした場合の処理部の平均応答時間を併記した．処理部の平均応答時間は，到着率 λ が大きくなるにつれて，最小の $S=1$ の場合の値から最大の $S=4$ の場合の値まで近似的に直線的に増加する．一方，システム全体の平均応答時間も，λ が大きくなるにつれて最小の $S=1$ の場合の値から最大の $S=4$ の場合の値まで増加するが，λ が小さい範囲では最小の場合と漸近的に同様の振る舞いを示し，λ が大きくなって上限に近い範囲では最大の場合と漸近的に同様の振る舞いを示すことが分かる．図 6 に，メモリ資源数 S が 2, 4, 6, 8 のときのシステム全体の平均応答時間を示す．S が増加するにつれてシステム全体の平均応答時間は短くなり，システムの許容量は大きくなっている．

5. ま と め

メモリ資源を持つ計算機システムを待ち行列網で性能評価するための近似法を提案し，数値実験によりその評価の特性を解析した．近似の考え方は，CPU 処理や I/O 処理を解析する処理部とメモリの使われ方を解析するメモリ部に分割するという方法である．処理部は従来の閉のセントラルサーバモデルの積形解を用いて解析し，メモリ部はメモリを占有しているジョブ数によって完了率が変化する $M/M/S$ 待ち行列モデルを用いて解析する．

数値実験により，到着率が小さい範囲では処理部の負荷が最小の場合と漸近的に同等の傾向を示し，到着率が上限に近い範囲ではジョブ数が最大の場合と同等の傾向を示すことが確かめられた．

今後は，厳密解またはシミュレーションによる値と比較することにより，提案手法の近似精度を調べることが必要である．

参 考 文 献

1) F. Baskett, K. M. Chandy, R. R. Muntz and F. G. Palacious, “Open, Closed, and Mixed Networks of Queues with Different Classes of Customers,” J. ACM, Vol. 22, No. 2, pp. 248–260, April 1975.
2) H. Kobayashi, “Modeling and Analysis, ” Addison-Wesley Publishing Company, Inc. 1978.
3) T. Kurasugi and I. Kino, “Approximation Method for Two-layer Queueing Models,” Performance Evaluation 36–37, pp. 55–70, 1999.
4) J. A. Rolia and K. C. Sevcik, “The Method of Layers,” IEEE Trans. on Software Engineering, Vol. 21, No. 8, pp. 689–700, Aug. 1995.
5) T. Kinoshita and Y. Takahashi, “A Queuing Network Modeling and Performance Evaluation Method for Computer Systems with Resource Requirement,” IEICE D-I, Vol. J 82-D-I, No. 6, pp. 701–710, Jun. 1999.

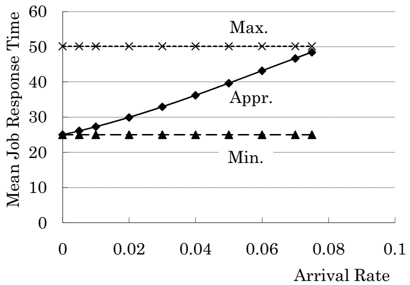

図 4　処理部の平均応答時間（$S = 4$）
Fig. 4　Mean job response time in processing part (S=4)

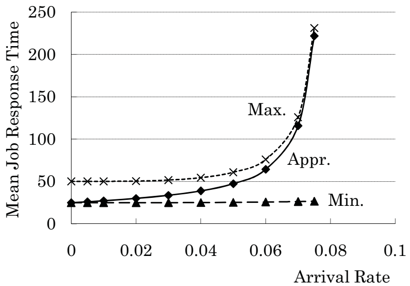

図 5　メモリ待ちを含むシステム全体の平均応答時間（$S = 4$）
Fig. 5　Mean job response time in entire system (S=4)

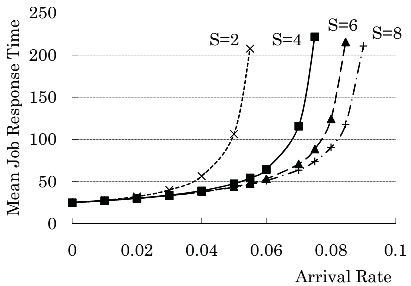

図 6　メモリ待ちを含むシステム全体の平均応答時間（$S = 2, 4, 6, 8$）
Fig. 6　Mean job response time in entire system (S=2, 4, 6, 8)

6) T. Kinoshita and Y. Takahashi, "An Approximate Method by Queuing Network Modeling for Performance Evaluation of Computer Systems with Exclusively-used Resources," IPSJ, Vol. 42, No. SIG14 (TOM5), pp. 1–13, Dec. 2001.
7) T. Kinoshita, "Queuing Network Model for Analyzing Influence of Exclusively Used Resource on Performance of Computer Systems," Proceedings of PDPTA 2004, pp. 291–297, Jun. 2004.
8) T. Kinoshita, "Approximate Technique for Queuing Network for Evaluating Performance of Computer Systems with Small Rate of Resource Requirements," Proceedings of PDPTA 2005, pp. 440–446, Jun. 2005.
9) T. Hasumi, X. Gao and T. Kinoshita, "An Approximation Queuing Network Technique for Evaluating Performance of Computer Systems with Resource Requirements," Proceedings of PDPTA 2009, pp. 690–696, July 2009.