



バス結合マルチプロセッサシステムの解析モデルと解析*

古 谷 立 美**

Abstract

An analytic model has been developed to evaluate the performance of various bus-connected multiprocessor systems (BMPS). Some interesting points of their performance have been studied with the model. Of main features of the model and the present method of analysis are:

(1) They are expandable to analysis of various BMPS; (2) The load distribution of BMPS is obtainable in terms of probability distribution; (3) Dynamic performance of BMPS is obtainable.

The performance analysis has been made to examine effects of the buffer size, resource scheduling and memory interleave. Some criterions of BMPS design are given.

1. はじめに

近年マルチプロセッサシステム (MPS) に関して種々の提案がなされ、いくつかのシステムがインプリメントされた。一方半導体技術の進歩はめざましく、高性能なマイクロプロセッサが安価で入手できる傾向は、その MPS のエレメントとしての適任性から、MPS 開発の興味をさらに盛り上げている。MPS をリソース間結合方式の面から分類すると、バス結合方式、マトリクス・スイッチ方式、リング方式等がある。このうちバス結合方式は、バスを共有する性格上、バスにおけるデータの衝突問題はあるが、ハードウェア量の少なさ、システム構成上のフレキシビリティ等の利点から、広く用いられる傾向にある。しかし、バス結合方式を選ぶ場合、メモリに加えバスでの衝突問題があるため、システム設計に当っては、解析やシミュレーションによる十分な検討が必要である。過去バス結合方式の解析は、V.K. RAVIDRAN¹⁾ et al. や GEORGE REYLING, Jr²⁾ によるものがあるが、それらはシステムを暗箱として扱うアプローチのため、システム内部の負荷状態が明らかにならないこと、データ転送のモデル化が単純すること、確率モデルが静的で

ある等の問題点があった。

本論文は、先ず様々なバス結合マルチプロセッサシステム (BMPS) の性能解析を行うための解析モデルと解析法を示し、次にそれを用いた BMPS に関する 2, 3 の解析結果を示す。解析モデルは、システムをその構成単位であるリソース (PMS^{***}) レベルで解析するもので、システム内部の負荷状態が確率的に求まると共に、データ転送のモデル化も現実のものに近付けている。解析法は、計算機の演算能力を有効利用するもので、システムの動特性も求まる。BMPS の解析は、バッファサイズ、メモリ・インタリーブ、リソース・スケジューリングの効果に関して行った。

2. モデル化と解析法

この章では、解析の対象とする BMPS の概要を示しそのモデル化と解析法を示す。

2.1 バス結合マルチプロセッサシステムの概要

以下に示す解析モデルが広く現実の BMPS に適用されることも、本研究のねらいの一つであるため、解析の対象とする BMPS は、できるだけ一般性のあるものをめざし、その定義は、 b 本のバスに p 台のプロセッサ、 m 台のメモリ、 i 台の入出力装置等が結合され、それらの間でデータ転送が可能なものとする。

2.2 モデルの設定と基本モデル

モデル化は、様々なタイプの BMPS の性能解析を可能にするため、基本モデル (PM) と呼ぶ全体のモデ

* An Analytic Model for Bus-Connected Multiprocessor System and Some Results of Analysis by Tatsumi FURUYA (Computer Division, Electrotechnical Laboratory)

** 電子技術総合研究所電子計算機部計算機方式研究室

*** C.G. BELL の記法

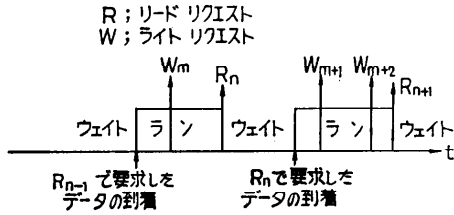


Fig. 1 Processor state model

ルを構成するための基本単位を設定し、複雑なシステムへの適用は、PM の重ね合わせで実現する。以下にモデル化のための仮定と、PM の構成を示す。

2.2.1 モデル化のための仮定

モデル化のための主な仮定は次のとおりである。

- (1) システムの同期動作: システムの状態変化は、等時間間隔 (T_s) の離散的時刻に発生する。
- (2) プロセッサの状態: プロセッサは、Fig. 1 に示すランとデータ待ちウェイトの2つの状態を持ち、以下に示す2種類のリクエストを発する。但し1リクエストによるデータ転送量は1単位データとする。
 - (i) リード・リクエスト: プロセッサがラン中に発するデータ要求で、これが発生するとプロセッサは要求したデータが到着するまでウェイト状態になる。Fig. 2(b) がリード・リクエストに対するデータの流れて、メモリの待ちに入るものがデータでなく、リクエストである点が他と異なる。
 - (ii) ライト・リクエスト: プロセッサがラン中に発するデータ送出要求で、これが発生してもプロセッサの状態は変わらない。但しプロセッサとメモリは、このリクエストに対するバッファを持ち、このリクエストが頻発してバッファサイズを越えた時は、オーバフローとして処理する。Fig. 2(a) は、ライトリクエストによるデータの流れてである。また、メモリアクセスの際アドレスがバスを経てメモリへ送られるタイプの BMPS では、アドレス転送をメモリへのライトリクエストと扱うことで処理可能である。
- (3) データ転送リクエストの発生: リクエストの発生は、式 (1) に示す二項分布に従う。

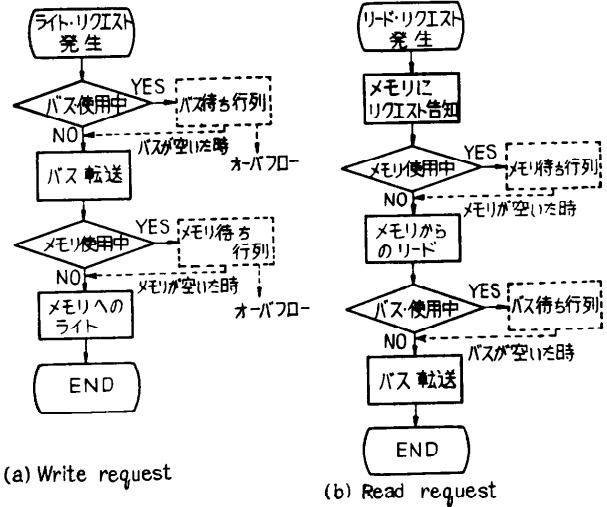


Fig. 2 Data flow by request

$$B(n, N) = {}_N C_n \lambda^n (1-\lambda)^{N-n} \quad (1)$$

ここで $B(n, N)$ は、ラン中のプロセッサが N 台のとき、単位時間間隔に n 台のプロセッサがリクエストを発生する確率で、 λ は各プロセッサが単位時間に1リクエストを発生する確率。

- (4) 各種データ転送の統一処理: 各種リソース間、バス間のデータ転送は、プロセッサとメモリ間の転送に置換えて処理する。

2.2.2 基本モデル (PM)

PM は、データ転送リクエスト頻度が等しく、かつ m 台の共有メモリそれぞれに等確率でアクセスするプロセッサ群が、一本のバスで共有メモリに結合された構成に対して一つ設定される。Fig. 3 は、PM の対象となるハードウェア構成を、Fig. 4(a), (b) (次頁参照) は、PM を示す。

一つの PM は、Fig. 2 に示す2つのリクエストに対応して、リード・リクエスト待ち行列系 (PM-R_k, k は k 番目の PM であることを示す。)とライト・リクエスト待ち行列系 (PM-W_k) の一対より成り、この待ち行列系のそれぞれに 2.3 節で述べる状態確率行列が作られる。Fig. 4 で λ_r , λ_w はそれぞれ各プロ

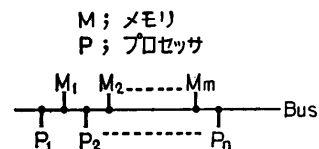
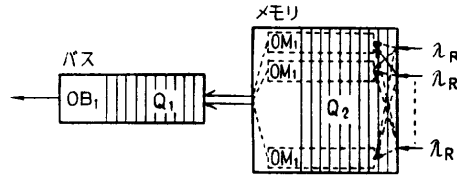
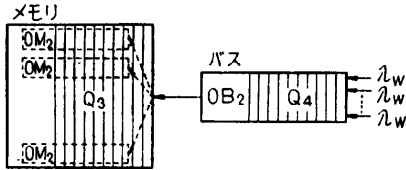
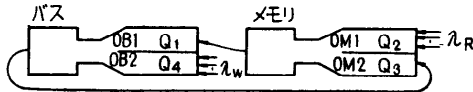


Fig. 3 Hardware structure for Primitive Model

(a) Read request queueing model. (PM- R_k)(b) Write request queueing model. (PM- W_k)

(c) Relation between hardware structure and PM.

OM_n : 待ち行列 $n(Q_n)$ のデータが T_n 間にメモリで処理される割合。
 OB_n : 待ち行列 $n(Q_n)$ のデータが T_n 間にバスで処理される割合。

Fig. 4 Primitive Model (PM)

セッサのリード及びライト・リクエストの発生頻度を示す。

解析されるシステムが、共有メモリを等頻度でアクセスする一つのプロセッサ群だけで構成されている最も単純な場合のモデル化は、一つのPMを作るだけで済む。Fig. 4 (c) は、その場合のPMとハードウェアの対応関係を示している。即ち、Fig. 4 (a) のメモリとバスは、それぞれ Fig. 4 (b) のメモリとバスと同一のハードウェアであり実際のシステムでは、これら2つの待ち行列の内容が混り合っ一つのハードウェアの待ち行列に入っている。そこでPMでは、(a), (b) それぞれの待ちの内容がどれだけ T_n 間に処理されるかを示す確率変数として OM , OB を導入している。この詳しい求め方は、式 (5), (6) に示す。

また、解析されるシステムが複雑で、等頻度でリクエストを発するプロセッサ群が多数ある場合等は、各プロセッサ群に一つずつPMを作り、それぞれの OB , OM を決定する時に、共有リソースを使用する全PMの待ち行列長を計算に入れることで処理できる。即ちFig. 4 (c) でいえば、バスとメモリへ入る待ち行列の数がPMが増える分だけ増え各待ちのデータが処理される割合が減ることになる。その他優先度のあるプ

ロセッサ群の処理も OM , OB 決定に重みを付けることで可能である。なお以下では、Fig. 4 のメモリとバスで処理中及び待ち行列中のデータとリクエストを、それぞれメモリ処理フェーズ、バス処理フェーズのデータとリクエストと呼ぶ。

2.3 状態確率行列と状態方程式

システム解析においては、システムの状態変数として、共有リソースの処理フェーズにあるデータ及びリクエスト数の確率を選び、状態方程式は、マルコフ過程の考え方で、差分方程式を立てる要領で行う。以下に状態方程式を立てる際必要な変数の定義と、PMに対する状態方程式を示す。但し以下では、 t を非負とし、変数名の添字 k は、 k 番目のPMを示す。

T_n : システムの状態変化の単位時間間隔。以下では、これをバスにおける一単位データ転送時間とする。

NP_k : PM $_k$ のプロセッサ数。

NM_k : PM $_k$ のメモリ数。

RM_k : 一単位データのメモリでの処理時間と、バスでの転送時間の比。

$$RM_k = \frac{\text{一単位データのメモリでの処理時間}}{\text{一単位データのバス転送時間}}$$

また以下では、 $RM_k \geq 1$ とする。

$IR_k(i, p, t)$: 時刻 t に p 台のプロセッサがランの時次の T_n 間に i 台がリードリクエストを発生する確率で、式 (1) に従う。但し $i > p$ または $i < 0$ または $p < 0$ のとき、

$$IR_k(i, p, t) = 0.$$

$$\sum_{i=0}^p IR_k(i, p, t) = 1 \quad (2)$$

$IW_k(i, t)$: $t \sim t+T_n$ 間に i 個のライト・リクエストが発生する確率。

$$IW_k(i, t) = \sum_{j=0}^{NP_k} RUN_k(j, t) \cdot IWT_k(i, j, t) \quad (3)$$

$$(i \geq 0)$$

ここで $RUN_k(j, t)$ は、時刻 t に j 台のプロセッサがラン状態の確率、 $IWT_k(i, j, t)$ は、時刻 t に j 台のプロセッサがランのとき、次の T_n 間に i 台のプロセッサがライト・リクエストを発生する確率で式 (1) に従う。

$$\sum_{i=0}^{NP_k} IW_k(i, t) = 1 \quad (4)$$

$OB(d, b, t)$: 時刻 t に、ある待ち行列系で b 個のデータがバス処理フェーズにあるとき、次の T_n 間に b 個中 d 個がバスを転送される確率。

$OB(1, b, t)$

$$= \begin{cases} \sum_{r=0}^{\max r} \frac{b}{b+r} \cdot OOB(r, t) & b \geq 1 \text{ のとき} \\ 0 & \text{その他} \end{cases} \quad (5)$$

$$OB(0, b, t) = \begin{cases} 1 - OB(1, b, t) & b \geq 0 \text{ のとき} \\ 0 & \text{その他} \end{cases} \quad (5')$$

ここで $OOB(r, t)$ は、時刻 t にバスを共有する他の待ち行列系のバス処理フェーズにあるデータの総和が r 個である確率。

$OM(d, m, t)$; 時刻 t に、ある待ち行列系でメモリ処理フェーズに m 個のデータ及びリクエストがあるとき、次の T_i 間に d 個が処理される確率。

($d \neq 0$ のとき)

 $OM(d, m, t)$

$$= \begin{cases} 0 & d < 0 \text{ または } m < 0 \text{ または} \\ & d > \min(NM_k, m) \text{ のとき} \\ \sum_{r=0}^{\max s} R(d, m, s) \cdot OOM(s, t) / RM_k & \text{その他} \end{cases} \quad (6)$$

$$OM(0, m, t) = \begin{cases} 1 & m = 0 \text{ のとき} \\ 1 - \sum_{d=1}^Y OM(d, m, t) & \text{その他} \end{cases} \quad (6')$$

但し、 $Y = \min(NM_k, m)$ 。

ここで $OOM(s, t)$ は、時刻 t に自分自身の待ち行列系以外のリクエスト及びデータがメモリ処理フェーズに s 個存在する確率。 $R(d, m, s)$ は、メモリ処理フェーズにある全データ及びリクエストのうち、 m 個が自分自身の待ち行列系のもので、 s 個が他の待ち行列系のもので、自分の系のうち d 個が処理される確率。

 $R(d, m, s)$

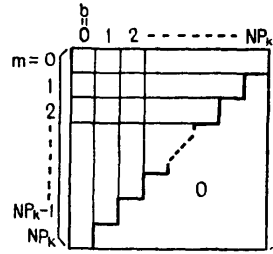
$$= \begin{cases} 0 & m \neq 0, s = 0, d = 0 \text{ のとき} \\ \sum_{w=d}^W OR(w, m+s) \cdot \frac{m C_d \cdot s C_{w-d}}{m+s C_w} & \text{その他} \end{cases} \quad (7)$$

但し、 $W = \min(NM_k, s+d)$ 。

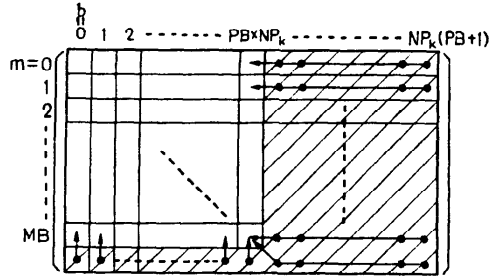
ここで $OR(w, l)$ は、 l 個のデータ及びリクエストが w 台のメモリに存在する確率。

PM に対する状態変数と状態方程式は、Fig. 4 (a), (b) の2つの待ち行列系のそれぞれに設定する。状態変数は、各時刻に、待ち行列系 (PM- R_k , PM- W_k) の各リソースの処理フェーズにあるデータ及びリクエスト数の確率とすることにより、次に示す状態確率行列 (SPM) が出来る。

(i) PM- R_k の状態確率行列 (PR_k)



(a) SPM----- PR_k



(b) SPM----- PW_k and PW'_k

Fig. 5 State probability matrix (SPM)

時刻 t に、PM- R_k のデータ及びリクエストがメモリ処理フェーズに m 個、バス処理フェーズに b 個存在する確率 $PR_k(m, b, t)$ をエレメントとする NP_k+1 ($m=0, 1, \dots, NP_k$) 行 $\times NP_k+1$ ($b=0, 1, \dots, NP_k$) 列の行列である。但し、リード・リクエストがプロセッサ数以上に発生することは無いので Fig. 5(a) に示す三角行列になる。

(ii) PM- W_k の状態確率行列 (PW_k)

時刻 t に、PM- W_k のライト・リクエストによるデータが、メモリ処理フェーズに m 個、バス処理フェーズに b 個存在する確率 $PW_k(m, b, t)$ をエレメントとする行列である。ライト・リクエストは、リード・リクエストと異なり、リクエスト数に制限が無いので、行列サイズは、無限大と成り得るが、プロセッサとメモリに有限長バッファを導入し、データがバッファサイズを越えた時はオーバーフローとして処理することにより、(メモリのバッファサイズ (MB)+1) 行 $\times (NP_k \times$ 各プロセッサのバッファサイズ (PB)+1) 列となり、解析可能にしている。Fig. 5 (b) の白地部分が PW_k で斜線部分がオーバーフローに相当する部分である。但しオーバーフローの考え方を導入することにより、オーバーフロー分の確率は、行列の $MB+1$ 行と $NP_k \times PB+1$ 列に集まるため、行列のエレメントの総和が1であることは保証される。またオーバーフローは、

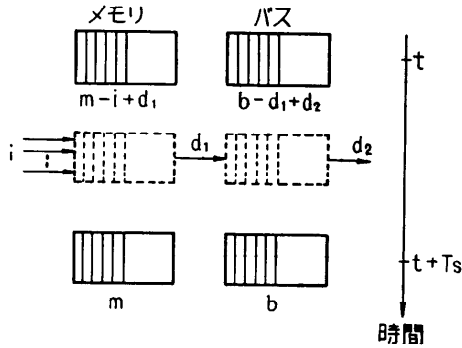


Fig. 6 Principle of state equation

バッファサイズを適当に選ぶことで、無視出来る程度に減少する。

次に各待ち行列系の状態方程式を示す。

(i) PM- R_k の状態方程式

$$PR_k(m, b, t+T_s) = \begin{cases} 0 & m < 0 \text{ または } b < 0 \text{ または } m+b > NP_k \text{ のとき} \\ \sum_{j=0}^{NM'} \sum_{i=0}^m (PR_k(m+j-i, b-j, t) \cdot OM(j, m+j-i, t) \cdot OB(0, b-j, t) \cdot IR_k(i, NP_k-(m-i+b), t) + PR_k(m+j-i, b+1-j, t) \cdot OM(j, m-i+j, t) \cdot OB(1, b+1-j, t) \cdot IR_k(i, NP_k-(m+b-i+1), t)) & \text{その他} \end{cases} \quad (8)$$

ここで $NM' = \min(NM_k, b)$ 。

Fig. 6 は、 $NM_k=1$, $RM_k=1$ の時の式(9)の原理図で、時刻 t に PM- R_k のデータ及びリクエストが、メモリ処理フェーズに $m-i-d_1$ 個、バス処理フェーズに $b-d_1+d_2$ 個有り、次の T_s 間に i 個のリード・リクエストが発生し、 d_1 個がメモリで処理されバスに送られ、 d_2 個がバスを通してプロセッサに送られた時、 $t+T_s$ 時のメモリ及びバス処理フェーズのデータ数は、それぞれ m , b となることを示している。

(ii) PM- W_k の状態方程式

ライト・リクエスト系に対する状態方程式も同様の方法によるが、データ・オーバフローの処理が必要である。ここでは、式が直観的に理解し易いこと、計算機での処理に都合が良いという理由から、オーバフローを許す確率行列 PW_k' を定義し、オーバフローに相当する確率を $MB+1$ 行と $NP_k \times PB+1$ 列に集める操作で PW_k を求める方法を示す。

$$PW_k'(m, b, t+T_s)$$

$$= \begin{cases} 0 & m < 0 \text{ または } b < 0 \text{ のとき} \\ \sum_{j=0}^{NM_k} \sum_{i=0}^b (PW_k'(m+j, b-i, t) \cdot OM(j, m+j, t) \cdot OB(0, b-i, t) \cdot IW_k(i, t) + PW_k'(m-1+j, b+1-i, t) \cdot OM(j, m-1+j, t) \cdot OB(1, b+1-i, t) \cdot IW_k(i, t)) & \text{その他} \end{cases} \quad (9)$$

PW_k' から PW_k を求める方法は、次のとおりで、Fig. 5 (b) は、その操作を示している。

$$PW_k(m, b, t)$$

$$= \begin{cases} PW_k'(m, b, t) & m < MB, b < NP_k \times PB \text{ のとき} \\ \sum_{g_1=MB}^{MB+1} PW_k'(g_1, b, t) & m=MB, b < NP_k \times PB \text{ のとき} \\ \sum_{g_2=NP_k \times PB}^{NP_k \times PB+NP_k} PW_k'(m, g_2, t) & m < MB, b=NP_k \times PB \text{ のとき} \\ \sum_{g_1=MB}^{MB+1} \sum_{g_2=NP_k \times PB}^{NP_k \times PB+NP_k} PW_k'(g_1, g_2, t) & m=MB, b=NP_k \times PB \text{ のとき} \end{cases} \quad (10)$$

以上が PM $_k$ の状態方程式であるが、この様に PM を設定し方程式を立てることにより、若干の工夫、変形、組み合わせで、広く各種 BMPS の解析が可能となる。

2.5 状態方程式の解法

状態方程式の解法は、 T_s ごとに逐一状態確率行列を平衡状態に達するまで求めていく方法が便利であり、システムの動特性も同時に求まる利点がある。

Fig. 7 (次頁参照) がその流れ図である。

3. BMPS に関する解析

BMPS設計において興味深い3点に関する解析結果を示す。なお以下の解析では、特にことわらない限り、ライト・リクエスト用バッファとして、各プロセッサが3単位データ分、メモリ群が $3 \times NP$ 単位データ分を持つものとしている。なお、この章では、解析モデルが一つの PM で表される等の理由で PM の番号を指定する必要がないため、変数名としては、2. の変数名から添字を除いたものを用いる。

3.1 バッファ・サイズの検討

この節は、BMPSの設計でバッファ・サイズを決定する際の参考資料を示すと共に、この解析モデルの仮定の一つであるオーバフロー処理の妥当性の検証を示

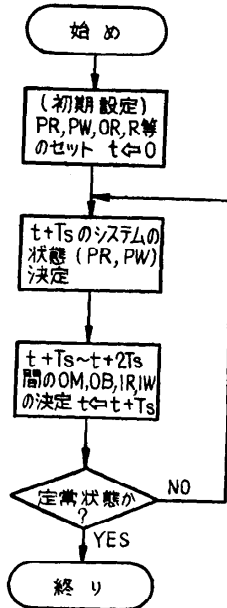


Fig. 7 Flow chart of analysis

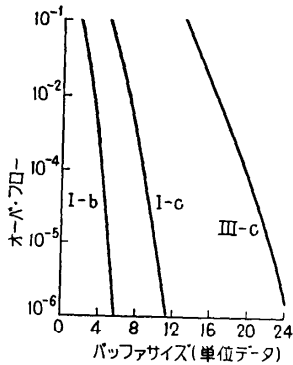


Fig. 9 Probability of data overflow

す。Fig. 8 は、 $NP=8$ の PM で、メモリとバス処理フェーズにあるデータ及びリクエスト数の分布を RM, λ_r, λ_w をパラメータとして求めたものである。この結果、ライト・リクエスト頻度がリード・リクエスト頻度と比べてあまり多くないとき、各プロセッサは、ライト・リクエスト用に数単位データ分のバッファを持つことで、オーバーフローはほとんど起こらないこと、また単にバッファ・サイズを増やしてもそれに比例した性能向上は望めないことが明らかになった。Fig. 9 は、Fig. 8 の代表的曲線について、バッファ・サイズとオーバーフローの関係求めたもので、これから数単位データ分のバッファ導入で、オーバ

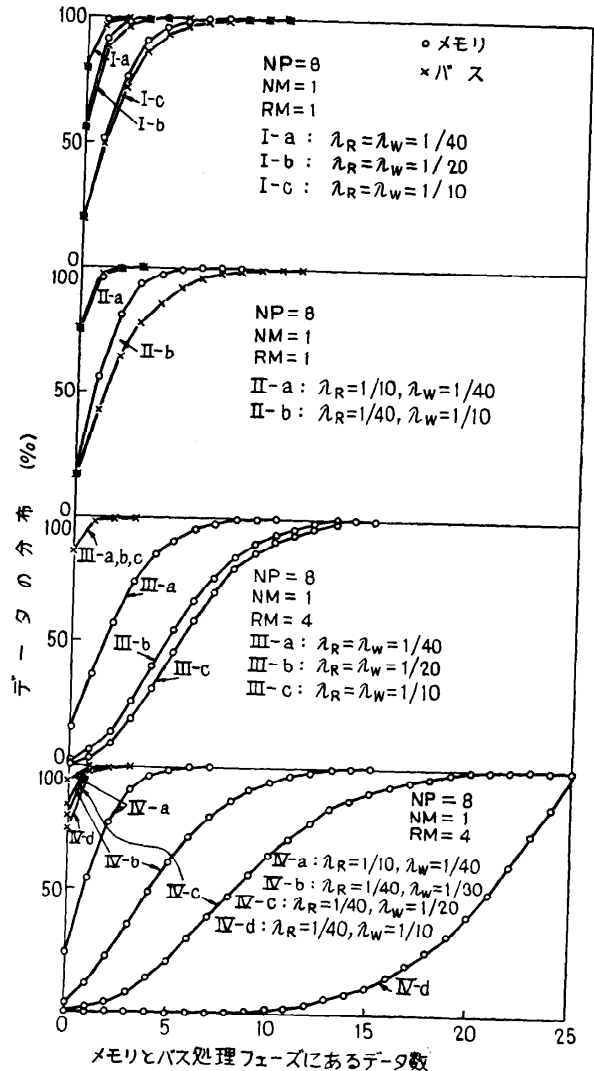


Fig. 8 Data distribution on resource

ローは 10^{-6} 程度に減少し、解析への影響は無視出来ると言える。

3.2 メモリ・インタリーブの効果

ここでいうインタリーブとは、 NM が 2 以上で各プロセッサが各メモリに等確率でアクセスするものである。この方法は、メモリ処理時間がプロセッサ速度に比べて大きい時等に有効なものである。Fig. 10 (次頁参照) は、プロセッサ数とプロセッサ稼働率の関係を種々のケースについて求めたものであり、Fig. 11 (次頁参照) は、その時のリソース利用率と待ちの長さの

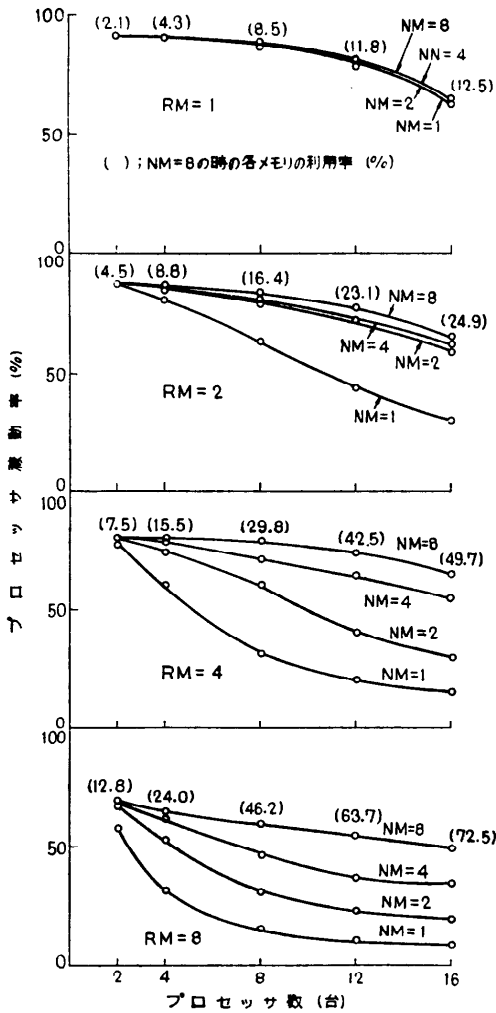


Fig. 10 Effects of memory interleaving

関係を調べたものである。この結果、BMPS のインタリーブ数としては、RM 程度に分けることが合理的であることがわかる。しかし、リソースの利用率がかなり高い (70% 程度) 場合でも、システムのパフォーマンスに与える影響は、一般の待ち行列 (無限長待ちを許す行列, Fig. 11 の破線) の感覚に比べ、非常に少ないことも明らかになった。

3.3 リソース・スケジューリング

複数のプロセッサに共有されるリソースに、簡単なスケジューリング機能を設け、プロセッサに優先度を付けて処理した時のシステムへの影響を調べてみた。ここでは、1 台のメモリ (RM=2) を共有するリクエスト頻度の異なる 2 群 (各群 4 台) のプロセッサ群の一

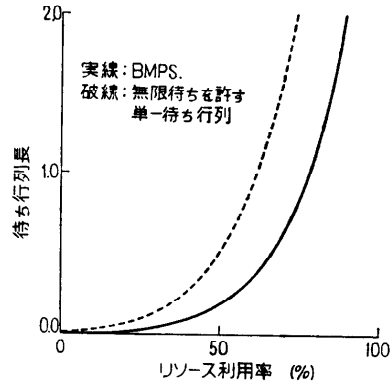


Fig. 11 System characteristics

方に確率的優先度を与えた時のシステムの性能を調べた。ここで確率的優先度とは、一般に優先度といったとき、高優先度のものを全ての低優先度のものに優先させるのに対し、高優先度のものをある確率で優先させるというものである。Fig. 12 (次頁参照) は、メモリとバスにスケジューリング機構を持たせた時の効果を、 λ_r, λ_w をパラメータとして求めたものであり、図中の破線は、スケジューリングを行わないときのものである。この結果、待ちが長いときには、スケジューリングが有効といえるが、3.2 の結果と総合して考えると、リソースをリソースバウンドにならない程度に利用する限りは、あまり性能に影響を与えないこと、またリソースバウンドのときは、そのリソースを増やす等の策が有効であることがわかった。

4. むすび

BMPS の解析モデルと数点の解析結果を示した。このモデルと解析法は、システム構成があまり複雑でない場合特に有効で、次の様な利点がある。

- (i) システム内部の負荷状態が、一般のシミュレーションでは得られない確率分布の形で求まる。
- (ii) 各種システムへの適用が可能で、バスバウンド、メモリバウンドいずれのシステムへも適用できる。
- (iii) システムの動特性が求まる。

この解析法は、バッファ・サイズの設定が解析時間に大きな影響を与えることが明らかである。今回の解析では、常にバッファ・サイズを固定したため、無駄があった。そこでバッファサイズは、解析段階にオーバーフローをチェックし自動的に決定するアルゴリズム

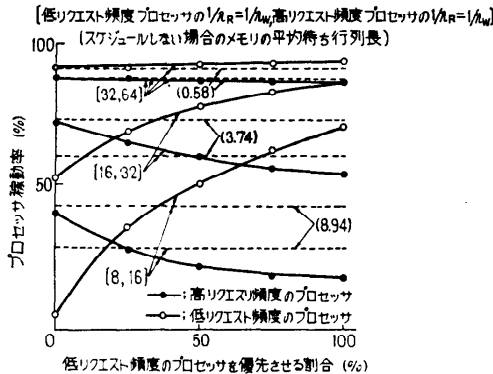


Fig. 12 Performance evaluation of resource shedding

を導入することにより解析時間は大幅に改良されると思われる。

解析は、システム設計の立場で最も興味深い三点を選んだ、これらの結果は、従来設計者が勘で行っていた決定に一つの参考資料を与えることができたと思う。またこれらの解析のいくつかについてシミュレーションシステム⁷⁾を用いた場合との比較も行ったが、結果は非常に良く一致したことを付け加えておく。

最後に、有益な御助言をいただいた飯塚肇室長、藤井狷介の両氏ならびに御検討いただいた研究室の皆様、また本研究の機会を与えられた黒川一夫電子計算機部長に謝意を表します。

参考文献

- 1) V.K. RAVIDRAN and THAMPY THOMAS: Characterization of Multiple Microprocessor Networks. Digest 1973 IEEE Computer Society International Conference, pp. 133~137.
- 2) George Reyling, Jr: Performance and Control of Multiple Microprocessor Systems. Computer Design, pp. 81~86 (March, 1974).
- 3) K.V. Sastry, et al: Markovian Models for Performance Analysis of Multiprocessor system, Minnesota Univ. (January, 1974).
- 4) K.M. Chandy: The Analysis and Solutions for General Queuing Networks, Proc. Sixth Annual Princeton Conference on Information Sciences and Systems. pp. 224~228 (March, 1972).
- 5) Harry Rudin, Jr: Performance of Simple Multiplexer Concentrators for Data Communication, IEEE Trans. communication. Vol. COM-19, No. 2, pp. 178~187 (APRIL 1971).
- 6) V. L. Wallance, et al: Markovian Models and Numerical Analysis of Computer System Behavior, Proc. AFIPS SJCC, Vol. 28, pp. 141~148 (1966).
- 7) 古谷立美他: ポリプロセッサシミュレーションシステム (PPSS1), 情報処理学会大会予稿. (1974).
- 8) 古谷立美: バス結合マルチプロセッサシステムの解析モデルと解析結果, 電子通信学会電子計算機研究会資料, Vol. 75, No. 20, p. 1~10.

(昭和50年6月3日受付)

(昭和50年9月3日再受付)