計算機アーキテクチャ　63－4
（1986・11・20）

並列計算機のソフトウェアと性能評価

Software and Performance Evaluation of Parallel Computers

中津山　恒[+]　堀口　進[+]　川添　良幸[++]　重井　芳治[+]
Hisashi Nakatsuyama　Susumu Horiguchi　Yoshiyuki Kawazoe　Yoshiharu Shigei

[+]東北大学工学部
Faculty of Engineering , Tohoku University

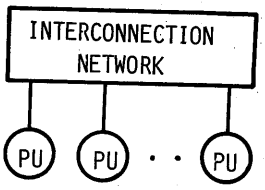
[++]東北大学情報処理教育センター
ECIP, Tohoku University

1．はじめに

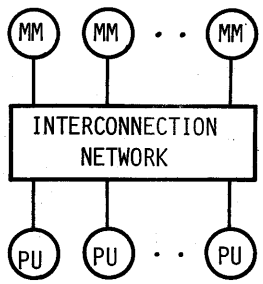
近年，集積回路技術の進歩により安価で高性能なマイクロプロセッサが市場に出ており，マルチマイクロプロセッサシステム(以後単にマルチプロセッサとする)の構築が可能となっている．高性能でかつ安価なコンピュータをという期待は益々高まっているが，マルチプロセッサはこれに応えるものとして注目を集めている．

1．1．　マルチプロセッサ

マルチプロセッサは多数のプロセッサとメモリからなる処理装置(ＰＵ)を相互結合網で結合したものであり，図１に示すような２つの形態に大別することができる[10][12]．図１(A)は疎結合とよばれ，ＰＵは相互結合網により相互に結合されておりＰＵ間インタラクションは明示的なＩ/Ｏ操作としてプログラム内に示されなければならない．一方，図１(B)は密結合とよばれ，やはり相互結合網が存在するがシステム内にはローカルメモリ以外に共有メモリが存在し，ＰＵ間インタラクションは単に共有メモリに対するリード/ライトとして行なうことができる．また相互結合網としてはクロスバ，単一共有バス(以後単に単一バスとする)，複数バス，オメガネットワーク，デルタネットワーク等さまざまなネットワークが提案されている．このうちオメガネットワーク，デルタネットワークは新しいプロセッサの付加に対応できないことと結合網が多段であるという２点からここでは除外し，以下単一バス，クロスバ，複数バスについて考える．またデータの転送を考えた場合，疎結合では基本的にメッセージ伝達で行う必要があるのに対し，密結合ではメイルボックス，ランデブ[11][13]等を実現できアルゴリズムに対して柔軟であるといえる．従って密結合を選択する．



(A)疎結合マルチプロセッサ



PU:PROCESSING UNIT
MM:MEMORY MODULE

(B)密結合マルチプロセッサ

図１．マルチプロセッサの２つの形態

2．マルチプロセッサの結合コストと性能

2．1．　結合網のコスト

一般にクロスポイントの数が大きくなると，システムの全コスト中で網の占めるコストはかなり大きな割合となる．ここでは相互結合網のコストを網内に存在するクロスポイント数に比例するものと考える[9]．即ち網のコストをＣ，クロスポイント数をｎとしたとき，

$$C \equiv O(n) \qquad \text{(式1)}$$

とする．システム中のバス，メモリ，プロセッサ数をそれぞれＢ，Ｍ，Ｎとするとき，単一バス，クロスバ，複数バスのスイッチの段数及びコストは表Ｉのようになる．但し，複数バスシステムＩはメモリをグループ分けしていないシステム

であり，複数バスシステムⅡは部分バス(partial bus)によりメモリをグループ分けしたシステムである．gはメモリグループ数である．

表１．各種結合形態のスイッチの段数とコスト

	スイッチの段数	コスト
単一バス	1	O(N)
複数バスⅠ	2	O(B(M+N))
複数バスⅡ	2	O(B(M/g+N))
クロスバ	1	O(M×N)

マルチプロセッサの性能はメモリ要求の際生じるバスの仲裁に大きく左右されるためスイッチの段数は重要である．またコストが高いということはシステムの構築が困難になることを意味する．各種マルチプロセッサの性能はその結合のコストと合わせて考慮すべきである．

尚システム中にバス，メモリ，プロセッサがM，N，B存在する複数バス，クロスバをそれぞれM×N×B，M×Nと表記することにする．

２．２．結合の価格性能比

結合の価格性能比ＣＰ(Cost Performance)を以下の様に定義する．

$$CP \equiv TH/C \qquad (式2)$$

ここでＴＨ，Ｃはそれぞれマルチプロセッサのスループット及びネットワークのコストである．各結合形態の価格性能比を求めるため離散型シミュレーション言語ＧＰＳＳを用いてそれぞれのスループットをシミュレーションにより求めた．シミュレーションモデルとして，初期状態では各プロセッサに１個ずつトークンが存在し，平均λの指数分布に従ってプロセッサに滞在してメモリアクセスを行ない，再び元のプロセッサに戻ると仮定している．クロスバのシミュレーションモデルを図２に示す．ここで，クロスバにおいて競合がない状態で１回メモリアクセスを行なうのに要する時間を単位時間と定義する．複数バスの場合には系内に冗長な経路が存在するため，その経路の選択に１単位時間を要するものと仮定した．従って，複数バスの場合には競合がない状態で１回のメモリアクセスを行なうのに２単位時間を要する．アクセス競合が生じた場合，任意に１台のＰＵが選択されるとした．アクセス競合の仲裁に要する時間は十分小さいとし，仲裁に起因する遅延は生じないものとした．系内に複数のメモリモジュールが存在するときには全メモリモジュールに均等にアクセスするものと仮定している．メモリアクセスはタスクを分割した結果生じるものと考え，単位時間当りプロセッサにトークンが存在する期待値の総和がスループットＴＨになるものとしている．即ち，

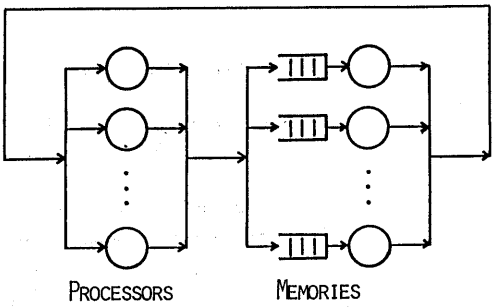


図２．クロスバのシミュレーションモデル

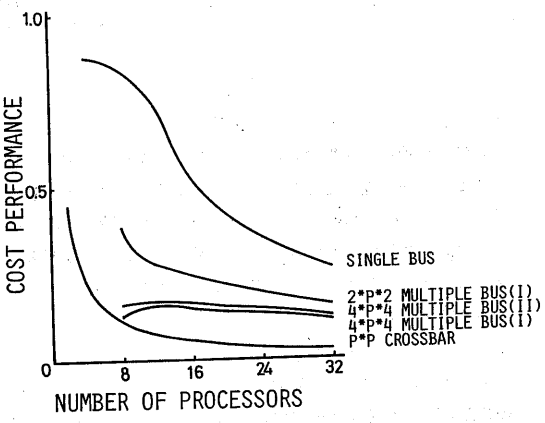


図３．各種アーキテクチャの価格性能比

$$TH \equiv \sum_{i=1}^{P} PA_i \qquad (式3)$$

とする．ここでＰ，PA_iはそれぞれ系内のプロセッサ数及びプロセッサｉの利用率である．λ＝0.1のシミュレーション値と表Ｉをもとに算出した価格性能比ＣＰを図３に示す．ここでは，コストの絶対値を求める訳ではないので，表Ｉのコスト$O(x)$をｘとして計算を行った．ＣＰはＢ，Ｍ，Ｎ，λの関数と考えられるが，クロスバはプロセッサ数とメモリモジュール数が等しいＮ×Ｎ型とし，また複数バスについては，システムⅠはｍ＝ｂ＝２及びｍ＝ｂ＝４について(2*P*2及び4*P*4，Ｐはプロセッサ数)，システムⅡについてはｍ＝ｂ＝４，ｇ＝２についてＣＰを求めた．

この結果，単一バスが非常によい価格性能比を得ていることが分る．クロスバは価格性能比の点からは使用を避けた方がよいと思われる．しかし，スループット自体について注目すれば，単一バスではプロセッサ数が大きくなるに従ってシ

ステムのスループットTHがメモリアクセス競合により1/λに飽和するため大規模なシステムを単一バスで構築することは得策ではない(図4)．これに対してクロスバはλが大きくなってもあまり性能が低下せず，N×Nの構成ではプロセッサの増加に対して線形にスループットが増加することが分かる(図5)．従って，単一バスとクロスバの中間的な特性をもち，メモリアクセス競合が小さいシステムを構築することが重要である．

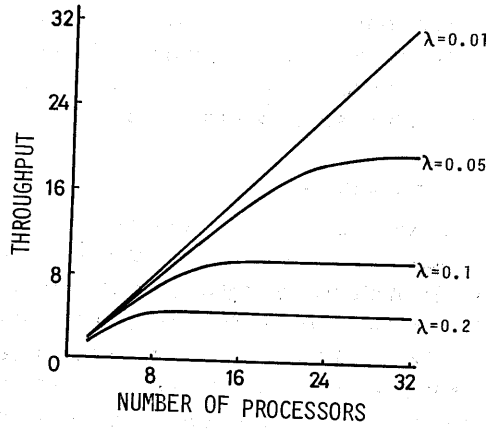


図4．GPSSによる単一バスのスループット

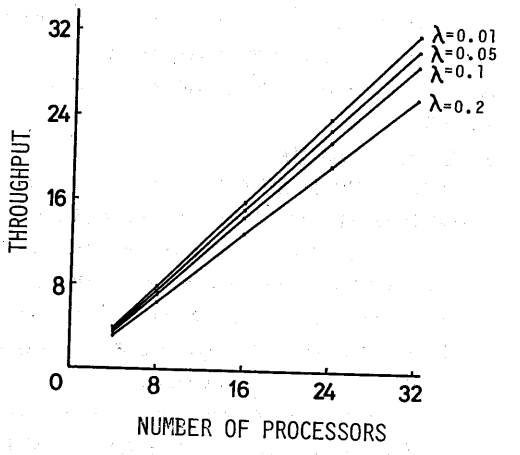


図5．GPSSによるクロスバのスループット

3．クラスタ式マルチプロセッサ

メモリアクセス競合を緩和するアプローチには，

(1)複数バスを用いる

(2)系内のプロセッサをクラスタ分けする

などがある．この内(2)のアプローチが価格性能比の点から好ましいと考えられる．(2)の手法によるマルチプロセッサの構成例を図6に示す．MP(Master Processor)は，プログラムの逐次的な部分の処理のほか，クラスタ間通信の仲介，I/O操作を行なう．

3．1．クラスタ分けによる効果

クラスタ分けした場合の結合コストは，クラスタ数をkとすれば，

$$C = O(n+k) \qquad (式4)$$

である．他のシステムと同様にシミュレーションによりTHを求め，CPを求めた．ただし，プロセッサは他クラスタのメモリモジュールにアクセスしないものとした．この場合のCPを単一バスのCPとともに図7に示す．同図からこの場合にはある程度プロセッサ台数が多くかつλが大きいとき，即ち競合が生じているとき，CPは単一バスに比較して非常に改善されていることがわかる．例えば，プロセッサ台数が32台の実用的なシステムでは，クラスタ数を4，λ=0.05としたとき価格性能比が約41% クラスタ式の方がよいことが分る．クラスタ数を大きくすればそれだけメモリ競合が緩和されるが，結合のコストも増加するため一概にクラスタ数が多いほど良いとは言えない．プロセッサ数が32から64台程度では4クラスタが良いと思われる．クラスタ分けした場合には他クラスタの共有メモリにアクセスする場合はMPを経由するため，クラスタ内の共有メモリへのアクセスよりも時間がかかる．そのため，処理をできるだけ局所化するよう注意してプログラムを作成しなければならない．

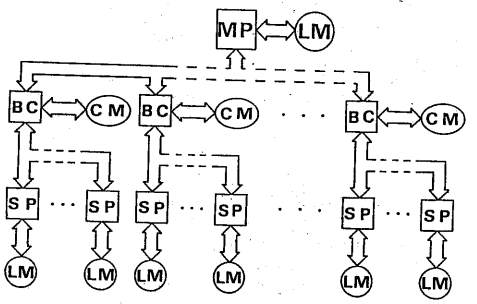


図6．クラスタ式マルチプロセッサ

3．2．試作クラスタ式マルチプロセッサMUGEN

3．1節で述べたクラスタ化の手法を用いて試作したマルチプロセッサシステムがMUGENである[7][8]．

MUGENは8台のSP(Slave Processor)，BC(Bus Controller)，CM(Common Memory)を1クラスタとし，4クラスタからなるシステムである．ハードウェアの概略を図8に示す．MP(Master Processeor)，SP，BCは Z80-A を用いている．CMへのアクセスはBCが制御している．Z80 CPUにはTS(Test & Set)命令に相当する命令がないが，BCによりバスの使用を認められたプロセッサは必要な処理が終了するまでバスを占有するので相互排除は自然に実現される．偏微分方程式や並列ソートの実行のためには隣接SPの通信

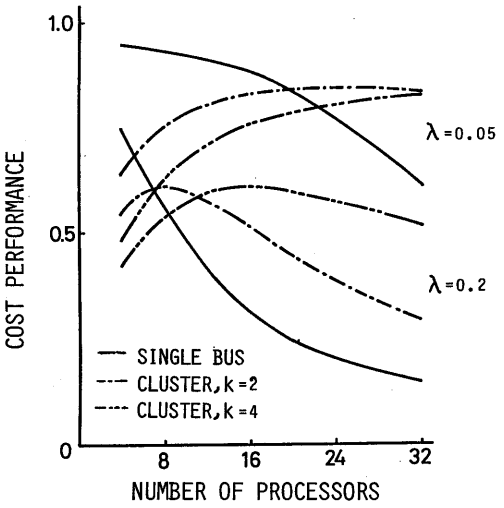
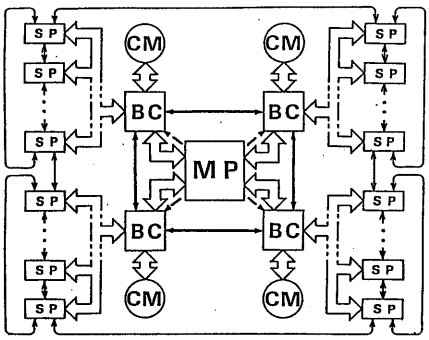


図7．クラスタ式と単一バスの価格性能比の比較



M P : MASTER PROCESSOR
S P : SLAVE PROCESSOR
B C : BUS CONTROLLER
C M : COMMON MEMORY
⟺ DATA ADDRESS BUS
⟷ I/O PORT LINE (PARALLEL)
⟷ I/O PORT LINE (SERIAL)

図8．試作クラスタ式マルチプロセッサ(MUGEN)

が多いことから，バス競合を緩和するため隣接するＳＰ間は双方向シリアルポートで接続され互いに通信できる．またＢＣ間は並列ポートで完全結合され，クラスタ間の通信はＭＰを介在させずに行なうことが可能である．入出力はＭＰが行なっており，プログラムの開発はホストであるＰＣ9801 m3上で行ない，シリアルリンクによりプログラム，データの転送を行なっている．ＭＵＧＥＮの問題点としては，共有メモリがプロセッサの記憶領域の後半32ＫＢに割当てられていることによりプログラム領域が32ＫＢしかなく，特に入出力を行なうＭＰのプログラム領域が不足している．また，モニタ程度の制御プログラムしかもたないため，共有メモリ等の資源の管理をプログラムで記述しなければならないという問題が残る．現在，ＭＰを Z80 上位互換のＣＰＵ HD 64180 を用いて作り替えると共にＭＵＧＥＮをスタンドアロンのシステムとするためフロッピィディスクコントローラ，ＣＲＴコントローラなどを作成中である．

これまでＭＵＧＥＮにインプリメントされた応用プログラムとしては，並列ソート[8]，連立１次方程式の反復解法，積分計算等があり，並列ソートではプロセッサ24台の場合，１台のプロセッサでクイックソートを行なう時間に比して最大約8倍の性能向上が得られた．また積分計算は積分区間ごとに並列に計算可能であるので，32台のプロセッサを用いて sin x /x を区間[0,1]で積分した場合，１台のプロセッサで実行した場合の最大29.5倍の速度向上を得ている．

４．並列処理言語 para-C

試作機ＭＵＧＥＮのプログラムは当初アセンブリ言語を用いて記述してきた．しかし，アセンブリ言語はハードウェアの性能を最大限に引き出せる反面，コーディングには多大な時間を要する．そのためＣＰ/Ｍ上で稼働するＣコンパイラのライブラリを変更し，アセンブリ言語で記述した並列処理の起動及び通信用のサブルーチンとリンクして，ＭＰ，ＳＰの各プロセッサ用にプログラムを書くことでひとつの並列プログラムを記述してきた．しかし，ソースプログラムが複数になることと，アセンブリ言語での記述が以前として必要なことから記述性は高くはない．そのためＭＵＧＥＮの高級言語として para-C を既に提案しており[1]，コンパイラを作成した．以下に para-C の言語仕様及びコンパイラについて述べる．

４．２．para-Ｃ言語仕様

para-C ではＣ言語に並列性を陽に示すため，通常のＵＮＩＸのＣ言語から構造体，共用体，ポインタ等を除いたサブセット版のＣ言語にキーワードcobegin，process，body 及び forall を追加した。para-Cにおける並列処理のプリミティブはプロセスである．ここでは，プロセスはＳＰ上で純粋に逐次的に実行されるＣ言語の関数と定義する．cobegin はそれに続く process 文により並列処理が起動されることを示す．プロセスは純粋に逐次的に処理されるため，cobegin は入れ子にできない． process は，関数をＳＰに割付るとともに関数に番号付けを行う．また，起動するＳＰを指定する．body はその関数がプロセスであることを示すものである．プロセスの実行結果として得られるデータは共有メモリを介してもとのプログラムに引き渡されるため，プロセスとしてはデータ型を持たない．従ってこれは通常のＣ言語の void 型に相当する．forall 文は for 文と同様の制御構造であるが，これは process 文にのみ作用する．para-Cでは，ＳＰ間のデータの受渡しは関数 send，receive を用い

て記述する．これらの関数は，自分のＳＰ番号と相手のＳＰ番号からデータ受渡しの経路を実行時に判断して通信を行なう．これらの文，関数は抽象性が高く種々のマルチプロセッサにおける並列プログラムを記述できる．

通信のプリミティブな操作は本来ＯＳによって支援されるべきであるが，３．２節でも述べたようにＭＵＧＥＮはＯＳをもたないため全て関数で行わなければならない．そのため，共有メモリ上のある特定の領域をデータ受渡し用のチャネルとして使用することにより通信を行う．前出のメイルボックスやランデブの様な通信方式を支援するＯＳの設計が望まれる．

その他定義済の関数としては，ＭＰ－ＳＰ間の通信を行なう hsend 及び hreceive がある．また，para-C ではＳＰがアクセスする共有メモリの領域は基本的に固定であるが，大量のデータの受渡しのために共有メモリの領域を動的に獲得するための cmalloc，データの受渡し終了後，共有メモリの領域を開放する cmfree 等がある．

４．２．para-Cコンパイラ

para-C コンパイラは para-C のソースプログラムから自動的にＭＰ(Master Processor)およびＳＰ(Slave Processor)用の２種類のオブジェクトコード(アセンブリ言語)を出力する． 従ってプログラマはＭＵＧＥＮのハードウェアを意識せずに通常の高級言語プログラムと同様に一つのソースプログラムで並列処理を含む処理を容易に記述できる．

para-C で使用できるデータ型は，char(8bits)，int(16bits)，long(32bits)，float(32bits)の４種であり，全て符号付きである．図7にコンパイラのソフトウェアアーキテクチャを示す．コンパイラは， UNIX 上でＣ言語を用いて作成した１パスコンパイラであり，Ｃ言語のソースファイルで約3500行で記述されている．その後 MS-DOS 上に移植して，現在ＭＵＧＥＮのホスト上で稼働するクロスコンパイラとなっている．

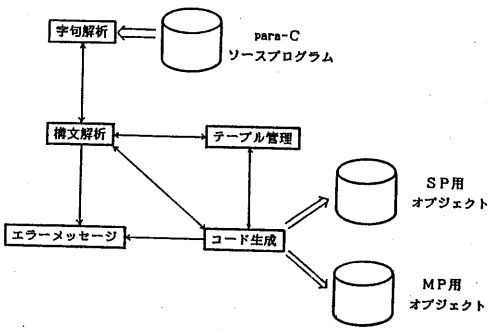


図９．para-Cコンパイラのソフトウェアアーキテクチャ

５．単一バスシステムとクラスタ式の性能評価

本章では，単一バスとクラスタ式に注目し，その効率及び性能の比較検討を行う．

５．１．処理時間一定の場合の効率の検討

各ＳＰでの処理時間が一定である場合について処理効率を検討する．ここでは，解析のため以下のような仮定をする。

A1．シリアルな処理はMPで，並列な処理はSPで実行され，通信は両者ともCMを介して行なう．

A2．１個のプロセッサで処理するとT_p[unit time]かかるタスクはN個のプロセッサで処理するとT_p/Nで終了する．

A3．MP-CM，SP-CM間で同一ののデータ転送に要する時間は等しい。

A4．タスクの実行中並列処理は１度だけ起動される．

また，以下のように変数を定義する．

T_p：タスクの処理量，

r ：１つのタスクのうち並列実行可能な処理量のT_pに対する比(並列実行可能率)，

t_T：１個のデータ転送に要する時間，

R'：t_TのT_pに対する比($R' \equiv t_T/T_p$)，

P ：プロセッサ数，

k ：クラスタ数，

E_S：単一バスによる実行時間のT_pに対する比，

E_C：MUGENによる実行時間のT_pに対する比，

とすれば，

$$E_S = \begin{cases} 1-r+r/P+(3P+1)R' & (rT_p/P \leqq (P-1)t_T) \\ 1-r+4PR' & (rT_p/P > (P-1)t_T) \end{cases} \quad (式5)$$

$$E_C = \begin{cases} 1-r+r/P+((1+2/k)P+1)R' & (rT_p/P \leqq (P/k-1)t_T) \\ 1-r+(1+3/k)PR' & (rT_p/P > (P/k-1)t_T) \end{cases} \quad (式6)$$

である．これをr＝0.98，k＝4のもとでグラフ化したものが図10，図11である．図10よりクラスタ分けによりR'＝0.05の場合には処理時間が単一バスの約1/2になり処理効率が向上している．図11からはｒが大きく，R'が小さいほど単一バスよりもクラスタ式の方が性能がよいことが分る．

５．２．待ち行列モデルによる性能評価

前節と同様にＳＰは他クラスタのＣＭへアクセスしないものと仮定し，ここではシステム内の客数が一定のM/M/1//M待ち行列を用いて性能評価を行なう[14]．客が到着しつつあるという状態のときには，その客が到着するまでに要する時間は平均1/λの指数分布の確率変数となる．また全ての客は互いに独立に行動する．また窓口の平均サービス率をμと

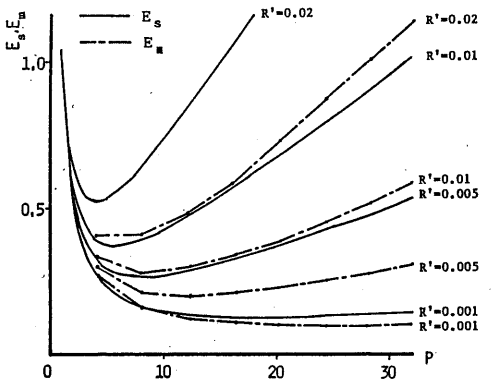


図10. 実行時間比とプロセッサ数の関係(r=0.98, k=4)

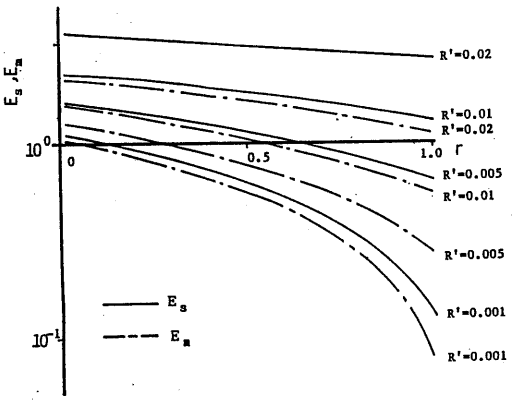


図11. 実行時間比とｒの関係(P=32, k=4)

する．窓口にｉ人の客が存在する確率をp_iとする．この場合 にはp_iは容易に計算でき，

$$p_i = p_0(\lambda/\mu)^i M!/(M-K)! \qquad (式7)$$

を得る．ただし，$0 \leqq i \leqq M$とする．ここで，

$$p_0 = \left[\sum_{i=0}^{M} (\lambda/\mu)^i M!/(M-K)! \right]^{-1} \qquad (式8)$$

である．ここで，Mはプロセッサ数と考えれば，スループットは到着しつつある客の平均人数となり，

$$TH = M - \sum_{i=0}^{M} i\, p_i \qquad (式9)$$

として得られる．MUGENの場合，これが各クラスタに適用されるから，クラスタ数をｋとすれば，p_0，p_iの式でそれぞれM←M/ｋとおいて得られるp_iを用いて，

$$TH = M - k \sum_{i=0}^{M/k} i\, p_i \qquad (式10)$$

として計算できる．これらの式でλ＝0.1，μ＝1として得られるスループットはシミュレーション言語ＧＰＳＳを用いて得られたスループットに良く一致する(図12)．シミュレーションではサービス率μ＝1(一定)としているが，両者のスループットは殆ど一致し，シミュレーションモデルをM/M/1//Mで近似できることが分る．

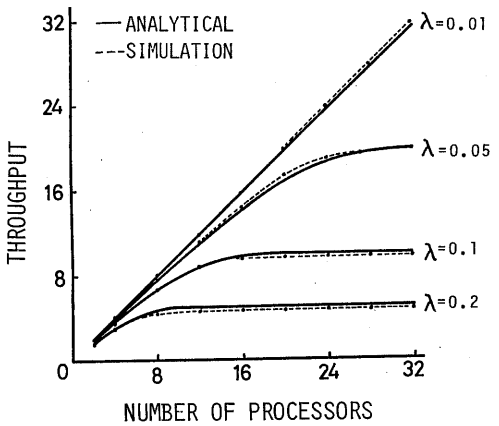


図12. 待ち行列を用いた単一バスのスループット

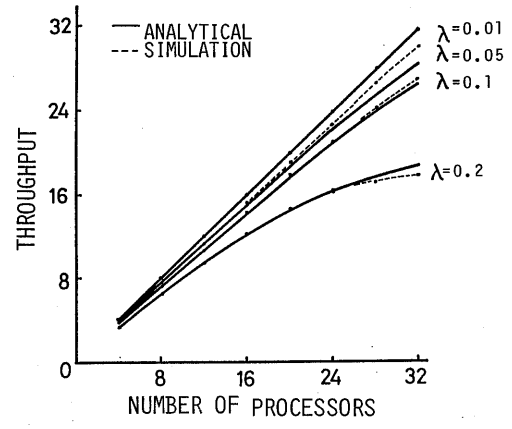


図13. 待ち行列を用いたクラスタのスループット

５．３．期待値モデルによる性能評価

Bhandarakar[4]，Hoogedoorn[5]らはマルコフモデルを用いてマルチプロセッサの性能評価を行ない厳密解を得ている．しかし，このアプローチでは構成要素の数が増すに従って状態方程式の解を求めるのが困難になる．従ってより簡便な解法が望まれる．ここではプロセッサの状態を処理，待ち

の2つに限定し，期待値を用いてスループットの解析を行なうことにする[2][3]．

まず単一バスを以下のようにモデル化してスループットを求める．競合がない状態でプロセッサがメモリ要求を出してから1回のメモリアクセスを完了するのに要する時間を単位時間とし，1個のプロセッサ上で単位時間で完了する仕事を単位仕事と定義する．また，プロセッサが取り得る状態を処理(P-state)，共有メモリへのアクセス(T-state)の2つに限定する．メモリ競合による待ち状態はT-stateに含まれるものとする．この様にプロセッサの状態を2つに限定することにより，各プロセッサが単位時間当りに共有メモリへアクセスする確率をmとすれば，プロセッサ数がPであるとき単位時間当りn台のプロセッサがメモリアクセス要求を出す確率$P_s(P, n)$は，次の二項分布に従うとみなせる．

$$P_s(P, n) = {}_pC_n m^n (1-m)^{P-n} \qquad (式11)$$

上式において，$1 < i \leqq P$の$P_s(P, i)$はi個のプロセッサによりバスが要求される確率であるので，単位時間当りこの内1つのプロセッサがバスを占有してメモリアクセスを行ない，残る$(i-1)$プロセッサはバス空き待ちの状態を通過して，順次メモリアクセスを行なう．従って，単位時間当りにバス空き待ちの状態に遷移するプロセッサ数の期待値は，

$$h_s = \sum_{i=2}^{P} (i-1)\,{}_pC_i m^i (1-m)^{P-i} \qquad (式12)$$

であるので，h_s個のプロセッサが順次バス占有待ち状態に入り，これを完了するまでの時間はh_s単位時間である．即ち，バス競合により単位仕事当りh_s単位時間だけ余分に処理時間がかかることになる．つまりP単位仕事が単一バスでは$(1+h_s)$単位時間で終了すると考えられる．従って，単一バスのスループットTH_sは，

$$TH_s = P/(1+h_s) \qquad (式13)$$

となる．

次に，クラスタ式について検討する．単一バスと同様にして，1個のプロセッサが単位時間当りにメモリアクセス要求を行なう確率をmとし，メモリアクセス要求を行なったときにそれがクラスタ外の共有メモリへのアクセスである確率をm'とする．MPのサービス時間は単位時間に等しいとする．また，他クラスタからの要求はクラスタ内の要求より優先順位が高いものとする．このとき単位時間当りにバス占有待ちに遷移するプロセッサの期待値は1クラスタに付き，

$$C_c = \sum_{i=2}^{P/k} (i-1)\,{}^{P/k}C_i m^i (1-m)^{P-i} \qquad (式14)$$

となる．またMPの占有待ちに遷移するプロセッサの期待値は，

$$C_M = \sum_{i=2}^{k} \sum_{j=2}^{i} (j-1)\,{}_kC_i m_a{}^i (1-m_a)^{k-i} \times {}_iC_j m'^j (1-m)^{i-j} \qquad (式15)$$

となる．ここでm_aはクラスタ内で少なくとも1個のプロセッサが共有メモリを要求する確率である．即ち，

$$m_a = 1-(1-m)^{P/k} \qquad (式16)$$

この場合，バス占有待ちのプロセッサが1回のメモリアクセスに要する時間の期待値は，

$$h_c = (1+C_c)(1+2m'(1+C_M)) \times (1+C_c p/k) \qquad (式17)$$

である．ただし，pはMPに少なくとも1個のプロセッサがサービス要求を行なっている確率である．即ち，

$$p = 1 - \sum_{i=0}^{k} (1-m')^i\,{}_kC_i m_a{}^i (1-m_a)^{k-i} \qquad (式18)$$

である．以上によりクラスタ式のスループットは，

$$TH_c = P/h_c \qquad (式19)$$

式13及び式19をグラフ化すればそれぞれ図14，図15の様になる．

単一バスに対する期待値モデルでは，バス要求を行なっているプロセッサ数の期待値が1の付近を除けばシミュレーション値との誤差が 10 % 程度の良い近似が行なえることが分かった．シミュレーションとの誤差はシステムが常に平均的な状態にあると仮定していることに起因する．より良い近似を行うためには各状態にあるプロセッサ数の変動を考慮する必要がある．このモデルではクラスタ間のインタラクションが存在するときも容易にスループットを求めることができ，近似解法として優れていると思われる．クラスタ式の場合は，単一バスよりも多くのプロセッサが存在しないと平均的な状態にならないため，ある程度プロセッサが多いときシミュレーション値にほぼ一致する．

このモデルではプロセッサの状態を2つに限定したことにより，隣接プロセッサ間に通信リンクが存在するときでも，共通メモリと通信リンクの使用確率を考えることにより，こ

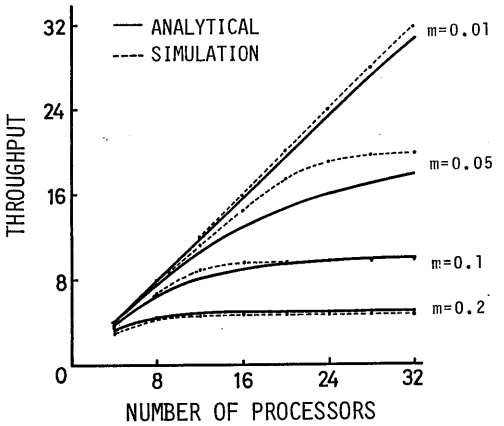


図14. 期待値を用いた単一バスのスループット

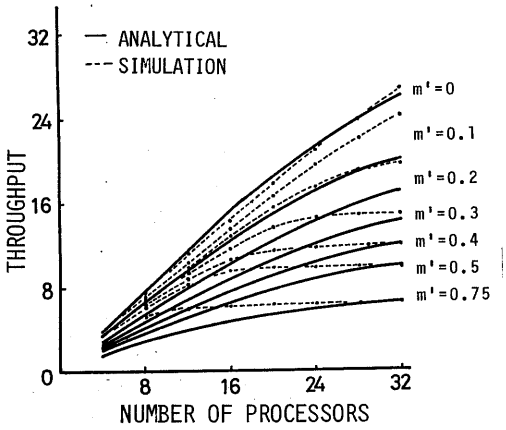


図15. 期待値を用いたクラスタのスループット(m=0.1)

の手法を容易に適用することができる.

6. まとめ

バス結合マルチプロセッサについて結合のコスト及び価格性能比を定義し，クラスタ分けにより価格性能比の良いマルチプロセッサシステムが構築できることを示した．クラスタ分けの手法を用いた試作機ＭＵＧＥＮについて概説し，その高級言語 para-C について説明した．さらに，単一バスとクラスタの理論的な性能解析を行ない，シミュレーション値との比較検討を行なった．その結果，M/M/1//M待ち行列による解析は非常に有効であることが分かった．ただし，クラスタ式に適用した場合，待ち行列網が複雑になり，解析が困難となる．期待値を用いた性能解析はより簡便であるが，期待値に重み付けを行うなどの改良の余地がある．今後は試作機の改良やＯＳの設計を含むシステム全般のより詳細な研究が望まれる．

〈謝辞〉 日頃，御指導戴く東北大学奈良久教授並びに中村維男助教授に深く感謝致します．

〈参考文献〉

[1]中津山他："並列計算機MUGENにおける並列処理言語,"昭和61年度信学総合全大,1481.

[2]鈴木他："バス結合マルチプロセッサのスループットの近似解法,"信学論，J69-D，No.7，pp.1127-1130(昭61-7).

[3]増山："バス結合マルチプロセッサシステムとそれらのスループット比,"信学論，J69-D，pp.264-267(昭61-2).

[4]D.P.Bhandarcar："Analysis of memory interference in multiprocessors," IEEE Trans. on Comput., vol.C-24, pp. 897-908, Sep. 1975.

[5]C.H.Hoogendoorn："A general model for memory interference in multiprocessors," IEEE Trans. on Comput., vol.C-26, pp. 998-1005, Oct. 1977.

[6]堀口他："マルチプロセッサによる並列ソーティング,"東北大通研シンポジウム，pp63-68(昭60-8).

[7]中田他："クラスタ方式を用いた共有メモリ型並列処理システムの試作,"情処第30回全大，5B-3.

[8]高木他："クラスタ式マルチプロセッサのシステムソフトウェア," 情処第33回全大，4C-6.

[9]Chita R.Das："Bandwidth Availability of Multiple-Bus Multiprocessors," IEEE Trans. on Comput.,Vol.C-34, pp. 918-926, Oct. 1985.

[10]黒川他："結合問題," 情報処理，Vol.27，pp. 1005-1021(昭61-9).

[11]情報処理振興事業協会(編)：最新Ａｄａ基準文法書，bit別冊，共立出版(1984).

[12]高橋："並列処理のためのプロセッサ結合方式,"情報処理 Vol.23，pp.201-209.

[13]Peter Wegner AND Scott A. Smolka："Processes, Tasks, and Monitors : A Comparative Study of Concurrernt Programming Primitives," IEEE trans. on Software Eng. , Vol. SE-9, No.4, July 1983.

[14]L.Kleinrock："Queueing Systems Volume I：Theory," John Wiley & Sons, Inc. (1975).