平成 16 年 1 月 16 日

知識情報工学専攻	学籍番号	003101
申請者氏名	岸　本　芳　典	

指導教官氏名	市 川 周 一

論　文　要　旨（修士）

論文題目	不均一クラスタ上での実行時間予測モデルとその評価

既存の科学技術計算応用の多くは計算負荷を PE に均等に割り当てるため，不均一クラスタ上で実行すると負荷不均衡による性能低下を生ずる．不均一クラスタ上で負荷を均衡化する手法の一つに，高速 PE 上に複数のプロセスを起動する手法(マルチプロセス法)がある．マルチプロセス法は実現が容易で，幅広い応用に適用可能である．

一般に，不均一クラスタ内の全てのプロセッサを使用しても，通信時間の発生により実行時間が最小になるとは限らない．マルチプロセス法で実行時間を最小化するには，(1) 最適な PE セットと，(2) 各 PE 上の最適なプロセス数を求める必要がある．本研究では，最適な実行方法(構成)を求める問題を組合せ最適化問題としてモデル化する．モデル化には，構成から実行時間を予測する近似式が必要である．

本研究では，HPL (High Performance Linpack)を応用例として，実行時間の実測値から予測モデルを構築し，(準)最適な PE 構成およびマルチプロセス数の予測が可能であることを示す．HPL のテストケースについて実行時間の測定を複数行ない，その測定結果から実行時間予測モデルを構築する．HPL のアルゴリズムから実行時間の近似式が求まるので，実測値を最小二乗法で処理して係数項を求める．このようなモデル化は実装や応用に依存しないため幅広い応用に適用可能である．

本研究では可能な構成の組み合わせ数を削減するため，(1) 等価な PE のマルチプロセス数を同一とし，(2) さらに通信時間は通信相手に依存しないと仮定した．2 種類の PE からなる不均一クラスタについて問題サイズ N=400～6400 の範囲でモデル構築を行ない， 予測により得られた最良構成を N=1600～9600 の実測における最良構成と比較した．その結果，N≧3200 で誤差 0～7.4%の(準)最適構成の予測に成功した．実測値の測定方法とモデルの精度の関係についても前述のクラスタ上で検討した．まず計算時間と通信時間について別々に構築したモデルについて評価した．その結果，通信時間の予測誤差が大きく実用的な精度は得られなかった．実装依存の手法を用いればこの誤差は削減できるが本研究の目的に反するため棚上げとした．次にテストケース数の削減とモデル精度への影響について検討した．サイズ N の測定点数を 9 点～5 点まで変化させたところ，N の測定範囲が十分に大きい場合には 5 点の測定で 9 点の場合と同様の精度を得られることが分かった．また 3 種類の PE からなる不均一クラスタについても評価を行い，クラスタの制約によりある PE のモデル構築に十分なテストケースを測定できない場合でも，精度の良い他種 PE モデルを性能比で定数倍して代用することにより N≧4800 で誤差 17.3%の構成を得られた．

今後はモデル精度の一層の向上，分枝限定法などによる探索空間の制限や近似解法の検討，HPL 以外の応用についても研究を進める必要がある．

不均一クラスタ上での
実行時間予測モデルとその評価

An Execution-Time Estimation Model for Heterogeneous Clusters

豊橋技術科学大学 大学院 工学研究科
知識情報工学専攻 市川研究室

003101 岸本 芳典

目次

1　はじめに

不均一クラスタとは，演算性能・通信速度・メモリ容量など構成や性能が異なる要素プロセッサ (PE) で構成されたクラスタを言う．近年，手近な計算機をよせ集めて一時的にクラスタとしたり，既存のクラスタに最新のプロセッサを追加して増強したいなどの要求が高まっている．

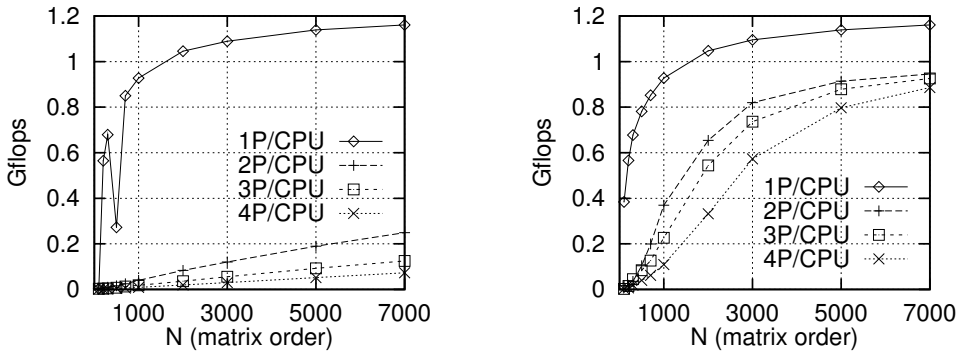
しかし，既存の多くの高性能計算 (HPC) 応用では MPP など PE 性能が均一な環境を想定しており，プロセッサ性能によらず負荷を各分散プロセスに均一に割り当てるため，プロセス間の実行時間不均衡によりピーク性能を発揮できない．応用を不均一環境向けに設計しなおせば不均一クラスタを効率よく利用できるが，蓄積された過去の膨大なソフトウェア資源を捨て去るのは得策ではない．

本研究の目的は，高速な PE 上にプロセスを複数起動することで，既存の応用を修正せずに不均一環境上で適切な負荷分散を行うことである．各プロセッサ上で起動するプロセス数は，クラスタの実行時間を最小化する組合せ最適化問題を解いて求める．

組合せ最適化問題としてモデル化するには，ある構成において各プロセスの実行時間を見積もるモデルが必要である．そこで，アルゴリズムで決定される通信量・計算量のオーダと，実行時間の実測値から，予測モデルを作成する．さらに実際の不均一クラスタに関して，実測された実行時間と予測された実行時間を比較し，モデル構築のための測定を十分に行った場合に本手法により最適〜準最適な構成が得られることを示す．

2　過去の研究

科学技術計算で多く用いられる行列行列積 (MMM) や LU 分解では，負荷分散手法としてブロックサイクリック分割が用いられることが多い．しかしこの方法は不均一環境では必ずしも最適な手法ではないため，いくつかの代替手法が提案されている．LU 分解では Kalinov ら[1] の Column-based 不均一ブロックサイクリック分割や，Beaumont ら[2] の ScaLAPACK 向け不均一 2 次元グリッド分割などがある．しかし，これらの不均一ブロックサイクリック分割ではソースの大規模な修正が必要となり，本研究の目的に合わない．また負荷分散はプロセッサのサイクルタイムだけに基づいており，通信時間を無視しているため



(a) MPICH–1.2.1　　(b) MPICH–1.2.2

図 1　MPICH のバージョンによる演算性能の差

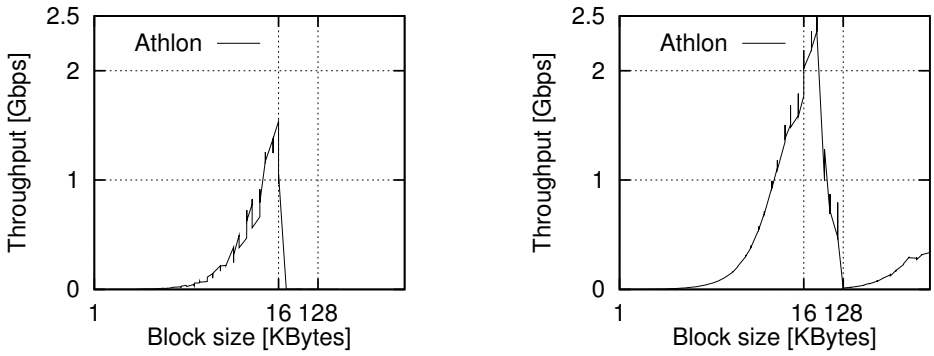
真に最適な分散となるか疑問が残る．

笹生ら[3] は HPL (High Performance Linpack)[4] について，ブロックサイクリック分割はそのままに複数の分割ブロックを 1 度に処理する修正を加えた．この修正には通信処理の大幅な変更が必要となるが，プロセス内通信を効率化することでオーバヘッドを軽減することができる．しかし，処理ブロック数の最適化は各 PE (1 CPU) の性能比に基づいており，並列実行時の実行時間最適化は今後の課題としている．

2.1　マルチプロセス法のオーバヘッド

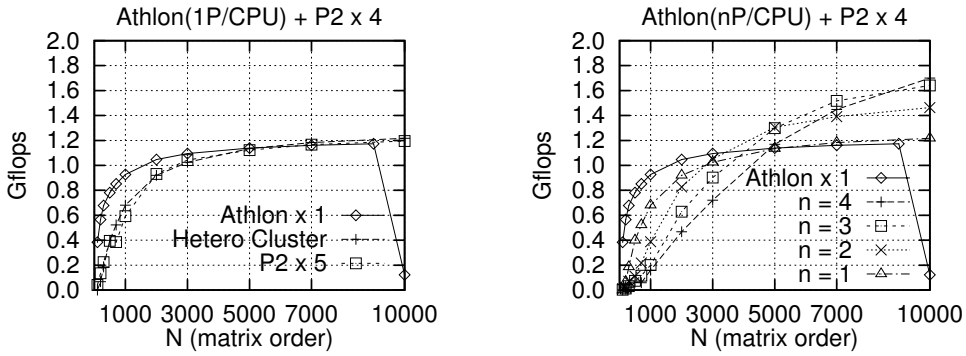
不均一クラスタ上で負荷を均衡化する直観的手法として，高速 PE 上に性能に応じた数のプロセスを起動する手法 (以下，**マルチプロセス法**) が考えられる．マルチプロセス法はソースの修正が不要で実現も容易であり，様々な応用に適用可能である．ただし，マルチプロセス法では複数プロセスの起動による実行時のオーバヘッドが問題になる．笹生ら[3] はマルチプロセス法について検討し，性能が低いとして棄却した．しかし，著者ら[5] が HPL に関して測定した結果では，問題サイズが大きい場合のオーバヘッドは実行時間の 2〜3 割程度であった．

単独の Athlon (1.33GHz) 上に複数の MPI プロセスを起動した場合の，HPL の性能測定結果を図 1 に示す．通信ライブラリには MPICH[6] を使用した．図中，nP/CPU は，Athlon 上に n プロセスを起動して測定したことを示す．MPICH-1.2.1 を使うとピーク性能の 1〜2 割まで性能が低下するが，MPICH-1.2.2 ではピークの 7 割程度の性能が得られることがわかる．netpipe を用いて，同一 PE 上で MPICH-1.2.1 と 1.2.2 の通信性能を測定した結果が図 2 である．16KB ブロッ



(a) MPICH–1.2.1　　(b) MPICH–1.2.2

図 2　MPICH のバージョンによる通信性能の差



(a) 負荷不均衡時　　(b) 起動プロセス数

図 3　ヘテロクラスタの各パラメータにおける性能

ク以上の通信性能が大きく異なっていることがわかる．図 2 から，MPICH-1.2.1 での性能低下は同一 PE 上での通信不具合によるもので，この不具合は 1.2.2 では解決されていると考えられる．

実際に Athlon 1.33GHz × 1 ＋ Pentium-II 400MHz × 4 (1000base-SX 接続) の不均一クラスタで，HPL の性能を測定した (図 3)．Athlon も Pentium-II も各 1 プロセスで実行した場合は，Pentium-II × 5 の均一クラスタと同じ性能になってしまう (図 3(a))．この性能は Athlon × 1 と同程度である．一方，Athlon 上に 2〜4 プロセスの負荷をかけると性能が向上し，4 プロセス時 ($N = 10000$) には，ピーク性能 (約 2.2Gflops) の 77%の性能を発揮した (図 3(b))．

以上の結果から，マルチプロセス法のオーバヘッドは許容可能と考える．以

下, 本研究では，マルチプロセス法による最適化手法について，HPL を例として詳しく検討する．

3　実行時間予測モデル

3.1　マルチプロセス法

一般に，不均一クラスタ内の全てのプロセッサを使っても，実行時間が最小になるとは限らない．特に問題のサイズが小さい場合，必要以上に多くのプロセッサに負荷を分散すると，通信時間が増大して全体の実行時間を悪化させる場合がある．従って，マルチプロセス法を不均一クラスタに適用する際には，(1) 最適な PE 群を選択し，(2) 各 PE 上で起動する最適なプロセス数を決めなければならない．この問題を組み合わせ最適化問題としてモデル化するためには，与えられた PE 群とプロセス数に対して，その構成の実行時間を予測する式が必要である．

本研究では，クラスタ上で HPL のテストケース（複数セット）を実行して実行時間を測定し，その結果から実行時間予測モデルを構築する．アルゴリズムから実行時間の近似式を求めておき，実測値を最小二乗法で処理して近似式の定数項を求める．

実測値に基づくモデリング技術は多くの工学分野で利用されており，特に奇異なものではない．例えば回路シミュレーションのためのトランジスタモデル[7]では，デバイス物理に基づいた解析モデルに，測定結果に基づくパラメータを導入する場合が多い．本研究では，このようなモデル化を実行時間予測に用いて，組合せ最適化による実行時間最適化に利用する．

本研究で採用する技術 (マルチプロセス法とモデル化) は実装や応用に依存しないため，HPL 以外の幅広い応用に適用可能である．実測値に基づいたモデルであるため，通信バッファの影響やキャッシュ効率など，システム内の様々な未知のオーバヘッドを内包したモデルを構築できる可能性がある．

3.2　モデルの概略

HPL の問題サイズを N，PE_i 上のプロセス数を M_i とする．予測モデルは，N, 不均一クラスタ中で使用する PE 群, および M_i から PE_i 上の各プロセスの実行時間を予測する．本研究では，プロセス格子は横一列に限定して１次元ブロックサイクリック分割とした．本研究の手法はプロセス格子に依存しないので，もちろん他のプロセス格子についても同様にモデル構築は可能である．

HPL のアルゴリズムとソースコードを解析して，問題サイズ N に対する計算量・通信量のオーダを解析する．ここでは 2 次式，3 次式などの関数の概形が解ればよい．次に，得られた近似式の係数をパラメータフィッティングにより決定する．

フィッティングは計算時間 T_a と通信時間 T_c についてまとめて行う．この際，最終的な実行時間 T は各プロセス i の見積実行時間 T_{ai}, T_{ci} を用いて $T = \max_i(T_{ai} + T_{ci})$ であると仮定する．

また，不均一クラスタ内に複数の同性能 PE が含まれていた場合，同性能 PE 上のマルチプロセス数は同一とすることで，モデルを統一し組合せ数の減少を図る．以後，同性能 PE をまとめたグループをサブクラスタ G_i と表す．各 G_i 内で使用するプロセッサ数を P_i，それらのマルチプロセス数を M_i で表すと，全プロセス数 P は $P = \sum P_i M_i$ と表される．

本研究では，まず，不均一クラスタ上で可能な全ての構成 $[P_1, M_1, \cdots, P_n, M_n]$ について実行時間予測モデルを構築する (4 章)．このようなモデル化が妥当であるか否かは，最終的には実測値と比較して検証されなければならない．検証結果については 4.2 節で述べる．次に，全ての構成についてモデルを構築するとプロセッサ数の増加によりモデル数が爆発するという問題を解決するため，モデルの統合によるモデル数の削減について 5 章で述べる．モデル構築のために多大な測定時間を要する問題については，テストケース数の削減とモデル精度への影響を 7 章で検討する．

3.3 HPL の測定結果と実行時間

本研究ではまず，実行時間 T のみを予測するモデルを構築し評価する (4 章〜5 章)．次に，実行時間を計算時間 T_a と通信時間 T_c に分け，それぞれ別々に予測することでモデル精度の改善を行うことができるか検討する (6 章)．実行時間 T の実測値のみが必要な場合は，マスタープロセスの開始から終了までの時間を単純に計測するだけで済む．しかし，計算時間 T_a と通信時間 T_c を得るには実行時間 T の内訳を詳しく知る必要がある．本研究では HPL に既に用意されている実行時間測定機能を利用してこれらの実測値を算出する．

HPL 測定結果に含まれる実行時間の内訳とその集計処理を図 4 に示す．Total Time はそのプロセスの総実行時間である．rfact はパネル列再帰 LU 分解フェーズの実行時間を表し，パネル LU 分解 pfact と軸交換通信 mxswp を含む．update は更新フェーズを表し, 行方向ブロードキャスト通信 laswp を含む．uptrsv は後

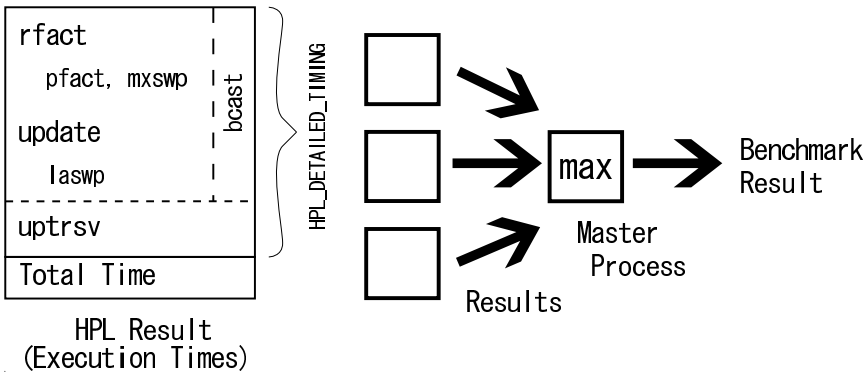


図 4　HPL 測定結果の内容と max による集計

退代入処理フェーズ,bcast は LU 分解全体を通じて行なわれるブロードキャスト転送を表す. 計算終了後，マスタープロセスは各プロセスの実行時間データから All Reduce 演算により各項目の最大値を求めベンチマーク結果として表示する.

これらの実行時間は通常のベンチマークでは計測されないが，HPL のコンパイル時に HPL_DETAILED_TIMING 定数を定義することにより計測処理が追加される．ただし bcast の測定処理はオリジナルには含まれていなかったため，ソースを修正することで独自に追加した.

以上の調査結果より，本研究では，以下の式で計算時間 T_a と通信時間 T_c を見積もり，実行時間 T は T_a と T_c の和であると仮定する.

$$T_a = (rfact - mxswp) + (update - laswp) + uptrsv$$
$$T_c = mxswp + laswp + bcast$$
$$T = T_a + T_c$$

3.4　N,P に対する実行時間近似式

HPL の実行時，問題サイズ N, プロセス格子サイズ $1 \times P$ に対して，通信量と計算量のオーダは以下のようになる.

HPL の各計算処理の計算量は次式で表される[3].

$$rfact = \frac{3}{2} \cdot N^2 + O(N)$$
$$update = \frac{2N^3}{3P} + \frac{P+1}{P} \cdot O(N^2) + O(N)$$
$$uptrsv = \frac{1}{P} \cdot O(N^2)$$

上記より計算量は問題サイズ N に対して $O(N^3)$ で抑えられるので，予測モデ

表 1　HPL 実行環境

Node 1	AMD Athlon 1.33 GHz, Main memory 768 MB
Node 2–3	Intel Pentium-III 866 MHz (dual processor), Main memory 768 MB
Node 4–7	Intel Pentium-II 400 MHz (dual processor), Main memory 768 MB
Network	100base-SX (NetGear GA-620), 100base-TX (Intel Pro100+)
OS	RedHat Linux7.0J (kernel 2.4.2)
コンパイラ	gcc 2.96, -DHPL_DETAILED_TIMING -fomit-frame-pointer -O3 -funroll-loops -W -Wall
ライブラリ	MPICH–1.2.5, ATLAS 3.2.1

ルは 3 次式となる．ただし実測によれば $update$ の実行時間が支配的で，$rfact$ と $uptrsv$ の実行時間は $update$ の 1/100 程度である ($N = 9600$)．そのため，計算量は $update$ の式に基づいて見積もることとする．

各プロセスの通信量は，HPL ソースの通信量に関するコメントから, 次のように概算した．

$$
\begin{aligned}
mxswp &= O(1) \\
laswp &= \frac{1}{P} \cdot O(N^2) \\
bcast &= (P-1) \cdot O(N^2)
\end{aligned}
$$

上記より通信量は問題サイズ N に対して $O(N^2)$ で抑えられるので，予測モデルは 2 次式となる．プロセス数 P に対しては P と $\frac{1}{P}$ が 2 次の項にかかっているため，予測モデルは両項を含んだ形とする．

問題サイズ N に対する PE_i の実行時間 $T_i(N)$ は，計算量が $O(N^3)$，通信量が $O(N^2)$ と見積もられるため，式 (1) のように 3 次式として予測する．添字の P, M_i は特定の構成 $[P, M_i]$ に対するモデルであることを表す．

$$T_i(N)|_{P,M_i} = k_0 N^3 + k_1 N^2 + k_2 N + k_3 \qquad (1)$$

式に含まれる定数 (k_0〜k_3) は，後ほどテストケースの実測値から最小二乗法で決定する．式 (1) はこれらの定数の線形関数であり，GSL(GNU Scientific Library)[8] の gsl_multifit_linear() 関数によりパラメータ抽出が可能である．この式には未知パラメータが 4 つ含まれるため，各構成ごとに実行時間 $t_i(N)$ を 4 点以上実測する必要がある．以下，この予測式を N–T 予測モデルと呼ぶ．

4　N–T 実行時間予測モデルとその評価

本研究では，表 1 に示す不均一クラスタを用いて HPL を実行し，実行時間予測モデルを構築する．以下の測定では，ネットワーク接続に 100base-TX だけ

表 2　HPL 測定時のクラスタ構成パラメータ

	サイズ	Athlon		Pentium-II		構成数
	N	P_1	M_1	P_2	M_2	
モデル構築時	400〜6400	0〜1	1〜6	0〜8	1	62
評価用実測時	1600〜9600	0〜1	1〜6	0〜8	1	62

を用いている．

本研究では，不均一クラスタ内の同性能 PE をサブクラスタ G_i として扱い，不均一クラスタ上で可能な全ての構成 $[P_1, M_1, \cdots, P_n, M_n]$ について，問題サイズ N を変えながら測定を行う．マルチプロセス数 M_i の測定範囲は他種 PE との性能比を参考にして決定する．たとえば，プロセッサ A がプロセッサ B より 4 倍高速であれば，A のマルチプロセス数 M_A は M_B の 1〜5 倍程度の範囲で測定する．

4.1　２種類のプロセッサからなるクラスタ

この節では，表 1 に示す不均一クラスタのうち，Athlon (Node 1) と Pentium-II (Node 4-7) だけを用いて実行時間予測モデルの評価を行う．モデル構築時と評価時の測定パラメータを表 2 に示す．ここで，Athlon の PE 数とプロセス数を P_1, M_1，Pentium-II の PE 数と P_2, M_2 とする．

表 2 (構築時) の組み合わせで，テストケースの実行時間を測定した．サイズ $N = 400,\ 600, 800, 1200, 1600, 2400, 3200, 4800, 6400$ のそれぞれに対して計 62 構成のテストケースを測定する．N の測定範囲に対するテストケース実行時間は表 3 の通りである．ここでは，N= 400〜6400 の間の 9 点を測定し N–T 予測モデル中の未知パラメータの数 4 に対して十分な数の実行時間を測定する．このときの測定時間は表 3 より 41043.7 秒となる．

実行時間の測定結果から，モデルパラメータを抽出した．パラメータ抽出には GSL の gsl_multifit_linear() 関数を用いる．GSL によるパラメータ抽出時間は 1ms 以下で，無視できる程度である．

こうして作成したモデルを用いて，N =1600,3200,4800,6400,8000,9600 の 6 つのサイズについて，それぞれ表 2 (評価時) の全てのクラスタ構成に対して実行時間を予測し，実行時間が最小となる構成（予測最良構成）を求めた．また，今回の評価では高速な Athlon 側のみマルチプロセス実行を行うこととした．Athlon と Pentium-II のピーク性能比はおよそ 1:4 であるので，Athlon のプロセス数 M_1 は 1〜6 の範囲とした．Pentium-II ではマルチプロセス実行を行わ

表 3　測定所要時間 (N–T 単体モデル)

N の範囲	HPL 測定数　所要時間 [sec.]	
400〜2400	$62 \times 6 = 372$	4100.0
400〜3200	$62 \times 7 = 434$	8100.6
400〜4800	$62 \times 8 = 496$	18836.8
400〜6400	$62 \times 9 = 558$	41043.7

表 4　N に対する予測値・実測値の最良値と誤差 (N–T 単体モデル)

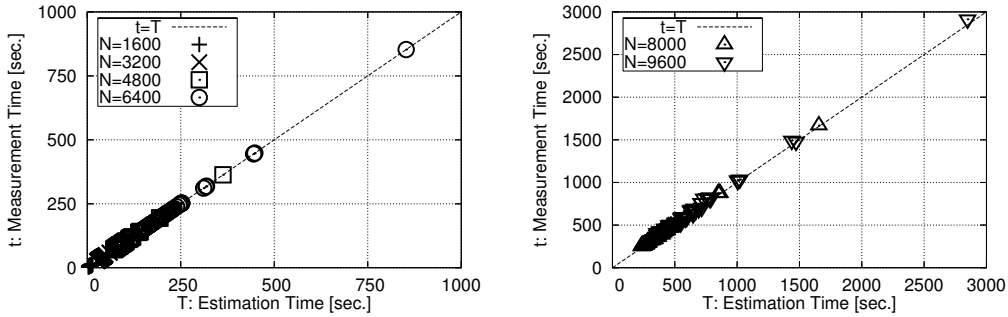
サイズ	予測による最良構成			実測による最良構成		誤差	
N	P_1, M_1, P_2, M_2	τ	$\hat{\tau}$	P_1, M_1, P_2, M_2	$\hat{T}$	$(\tau - \hat{T})/\hat{T}$	$(\hat{\tau} - \hat{T})/\hat{T}$
1600	1,1,0,0	3.40	3.48	1,1,0,0	3.48	-0.022	0.000
3200	1,1,0,0	26.13	25.93	1,1,0,0	25.93	0.008	0.000
4800	1,2,8,1	71.15	71.29	1,1,8,1	71.27	-0.002	0.0003
6400	1,4,8,1	138.70	138.72	1,4,8,1	138.72	-0.0002	0.000
8000	1,4,8,1	231.90	249.06	1,4,8,1	249.06	-0.069	0.000
9600	1,5,7,1	356.38	408.91	1,4,8,1	386.55	-0.078	0.058

ないため，プロセス数 M_2 は 1 となっている．

4.2　モデル評価

表 4 に，モデルの評価結果を示す．予測最良構成の予測実行時間を τ，予測最良構成の実測実行時間を $\hat{\tau}$ とあらわしている．さらに表 2 (評価時) の全ての組合せについて実際に実行時間を測定し，実行時間が最小となる構成（実測最良構成）を調べて表 4 に示した．$\hat{T}$ は，実測最良構成の実測実行時間である．

実測実行時間の誤差 $(\hat{\tau}-\hat{T})/\hat{T}$ は，$N \leq 8000$ では 0.3% 以下であり，$N = 9600$ でも 5.8%と実用的な誤差である．予測実行時間の誤差 $(\tau - \hat{T})/\hat{T}$ は $N = 1600$ で 2.2%と若干誤差が大きくなっているが，実行時間は 3 秒程度と短いので，時間差は 0.08 秒と小さい．$3200 \leq N \leq 6400$ では誤差は 0.8%以下でありモデル構築範囲内の予測値は非常に正確である．一方，$N \geq 8000$ における誤差は 6.9〜7.8%とやや大きくなる．図 5 に不均一クラスタで可能な全ての構成について予測実行時間 T と実測実行時間 t の相関を調べた結果を示す．対角線 $t = T$ に近いほど予測値と実測値がよく一致していることを表す．モデルの内挿範囲である $1600 \leq N \leq 6400$ の範囲では全ての点がほぼ対角線上にある．一方，$8000 \leq N \leq 9600$ では予測実行時間が若干小さく見積もられている構成があることが分かる．これはモデルを外挿して使用するため実測値に対して予測値が若干ずれてしまうためである．しかし，その誤差は 1 割以下であり十分実用的



(a) $N = 1600$〜6400 の相関　　(b) $N = 6400$〜9600 の相関

図 5　N–T 単体モデル実測・予測相関図

であると言える．

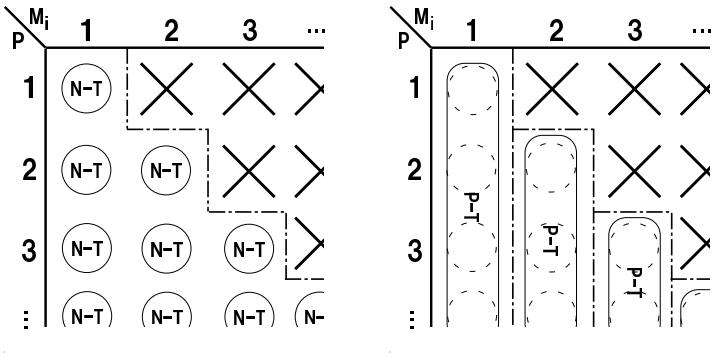
このように N–T 予測モデルは実用的な精度を持つが，モデル数が多くテストケースの測定に多くの時間を要する点が問題である．また，プロセッサの数や種類が増加すれば測定すべきケース数は爆発的に増加するため，モデル構築は実用上不可能となる．そこで本研究ではさらなるモデルの簡略化を行い，この問題の解決を図る．

5　P–T 実行時間予測モデルとその評価

不均一クラスタ上で可能な全ての構成 $[P_1, M_1, \cdots, P_n, M_n]$ について実行時間予測モデルを構築する手法では，プロセッサ数の増加によるモデル数の爆発が避けられない．そこで本研究では通信相手や通信トポロジーを無視することにより，モデルのさらなる簡略化を行う．

不均一構成時の各サブクラスタ G_i の実行時間は通信相手によらず全プロセス数 $P = \sum P_i M_i$ のみに依存すると仮定する．この簡略化によりモデル構築に必要なテストケースは各サブクラスタ G_i の均一クラスタ動作時の構成 $[P_i, M_i]$ のみとなり，複数のサブクラスタで構成されたテストケースを省略することができる．図 6(a) に G_i の構成 $[P, M_i]$ と N–T モデルの配置を示す．$P = M_i$ に対応する N–T モデルは単独 PE 上で 1 つまたは複数のプロセスを起動した場合の実行時間を予測する．$P > M_i$ に対応する N–T モデルはクラスタ動作時の実行時間を予測する．×の部分は $P = \sum P_i M_i$ であるからもともと存在しない．

各サブクラスタ G_i の実行時間 T_i は全プロセス数 P により予測モデルを外挿



(a) N–T モデル　　(b) P–T モデル

図 6　サブクラスタの構成パラメータとモデルの対応

して得ることにし，クラスタ全体の実行時間 T は T_i の max であると仮定する．このため，全プロセス数 P をパラメータに含む予測モデルが新たに必要となる．

5.1　P-T 実行時間予測モデル

3.4 節における検討と N–T モデルより，全プロセス数 P に対する G_i の実行時間予測モデルは式 (2) のようになる．添え字の M_i は特定のマルチプロセス数に対するモデルであることを表す．

$$T_i(N,P)|_{M_i} = k_4 P \cdot T_i(N)|_{P,M_i} + k_5 \frac{1}{P} \cdot T_i(N)|_{P,M_i} + k_6 \qquad (2)$$

式 (2) は定数 k_4〜k_6 の線形関数であり，GSL の gsl_multifit_linear() 関数によりパラメータ抽出が可能である．この式には未知パラメータが 3 つ含まれるため，M_i ごとに 3 つ以上の N–T モデル出力が必要である．以下，この予測式を P–T 予測モデルと呼ぶ．P–T モデルは図 6(b) のように M_i ごとに複数の N–T モデルを統合して構築する．

5.2　モデルの切り替え

単独 PE_i での実行時 ($P = M_i$) とクラスタ実行時 ($P > M_i$) では，PE 間通信の生むなど実行過程が大きく異なる．そのため P–T 予測モデルを $P = M_i$ まで含めて構築すると，モデルの精度が低下する可能性がある．そこで $P = M_i$ では N–T モデルをそのまま使用し，P–T モデルは $P > M_i$ の N–T モデルから構築する．

図 7 は，パラメータに応じて N–T モデルと P–T モデルを切り替える様子を示している．マルチプロセス数 M_i を含めて計算時間・通信時間の定式化は，関

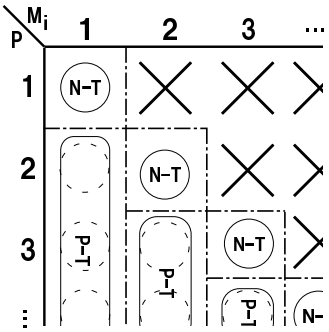


図 7　ビニング

与する要素が多いため困難である．そこでマルチプロセス数 M_i ごとに異なる P–T 予測モデルを構築して用いる．

このように，条件によって使用するモデルを切り替える手法は，回路シミュレーションのトランジスタモデリングでは “ビニング” として広く知られている[7]．トランジスタモデルのビニングは，支配的な物理過程が異なる領域に対して異なるモデルを適用するという思想で行われるが，本研究の実行時間モデルでは実行過程 (通信など) が異なる場合に，異なるモデルを適用する．

5.3　モデルの代用

式 (1) には未知パラメータが 4 つあるので，N–T モデル $T_i(N)|_{P,M_i}$ を決定するには，少なくとも 4 つの異なる N に関して実行時間を測定する必要がある．同様に，式 (2) には未知パラメータが 3 つあるので，P の異なる最低 3 つ (できれば 4 つ以上) の N–T モデルが必要である．

このように，モデル構築には一定数以上の測定が必須だが，テストケース実行の時間的制約や，不均一クラスタの構成の都合で，十分な測定点数が確保できない場合がある．そのような場合，他の PE の予測モデルを借用する方法が考えられる．

例えば，ある PE の P–T モデルが，都合により作成できないとする．しかし少なくとも $P = 1, M = 1$ の N–T モデルを作成することはできるので，既に P–T モデルの存在する PE との対象 PE の間で N–T モデルの出力値を比較して，二乗誤差が最小となる実行時間比を求めることができる．この比をもとに代用元 PE の P–T 予測モデルを定数倍し，対象 PE の代用 P–T モデルを作成することができる．また，代用元および対象 PE が 2 つ以上ある場合にはクラスタ構成時 $P = 2, M = 1$ の N–T モデルを比較することにより，クラスタ動作時の通

表 5　HPL 測定時のクラスタ構成パラメータ

	サイズ	Athlon		Pentium-II		構成数
	N	P_1	M_1	P_2	M_2	
モデル構築時	400〜6400	1	1〜6	1〜8	1〜6	54
評価用実測時	1600〜9600	0〜1	1〜6	0〜8	1	62

信性能を含めたより正確な性能比を得ることができる．

このような “モデルの代用” を活用すると，不均一クラスタにおけるモデル作成労力と作成時間を大きく削減することができる．もちろん，作成されたモデルが十分な精度を持つかどうかは，実測値との比較によって検証されなければならない．本研究でも，本章以降では P–T モデル構築の際に PE が不足している場合にはモデルの代用を行うことで対応する．また，代用の積極的活用については 7.3 節で検討する．

5.4　２種類のプロセッサからなるクラスタ

4.1 節と同じ不均一クラスタを用いて実行時間予測モデルの評価を行う．同様に，Athlon の PE 数とプロセス数を P_1, M_1，Pentium-II の PE 数と P_2, M_2 とする．モデル構築時と評価時の測定パラメータを表 5 に示す．

表 5 (構築時) の組み合わせで，テストケースの実行時間を測定する．サイズ $N = 400, 600, 800, 1200, 1600, 2400, 3200, 4800, 6400$ のそれぞれに対して Athlon は 6 通り，Pentium-II は 48 通りテストケースを測定する．

Athlon は 1 台しかないため P_1 の測定点数が 1 であり，P–T モデルの抽出が不可能である．このため 5.3 節で述べたモデルの代用を行う．Pentium-II についてもマルチプロセス数 $1 \leq M_2 \leq 6$ について測定し，得られた P–T モデルに定数 0.307 を乗じて，Athlon の P–T モデルとして代用する．

ただし Pentium-II ノードの測定時にプロセスを複数起動する場合，指定した実行方法と実際の実行方法が異なる問題がある．これは，SMP ノードに含まれる PE のうち片方だけに複数のプロセスを起動することができず，2 つの PE に均等にプロセスが割り当てられてしまうためである．そこで，問題の発生する構成に対して，PE を 1 つしか使用しないノードはモデル構築時のみシングルプロセッサのカーネルに切り替えて測定することにする．各 N におけるテストケース実行時間は表 6 の通りである．

表 6　測定所要時間 (P–T モデル)

サイズ N	Athlon 計 [sec.]	Pentium-II 計 [sec.]
400	3.9	96.7
600	7.4	130.1
800	10.8	178.8
1200	20.5	305.2
1600	37.4	508.5
2400	97.5	1117.3
3200	197.2	2042.2
4800	566.0	5360.0
6400	1239.5	10950.3
総計	2180.2	20689.1

表 7　N に対する予測値・実測値の最良値と誤差 (P–T モデル)

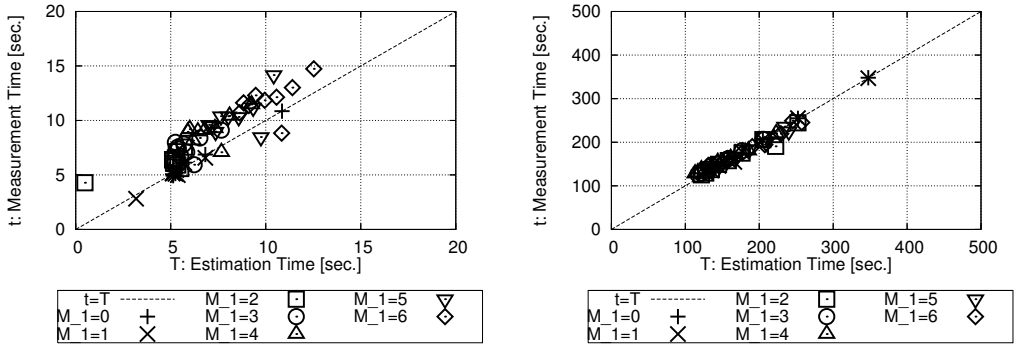
サイズ	予測による最良構成			実測による最良構成		誤差	
N	P_1, M_1, P_2, M_2	τ	$\hat{\tau}$	P_1, M_1, P_2, M_2	$\hat{T}$	$(\tau - \hat{T})/\hat{T}$	$(\hat{\tau} - \hat{T})/\hat{T}$
1600	1,2,0,0	0.48	4.28	1,1,0,0	2.82	-0.828	0.518
3200	1,1,0,0	20.04	20.42	1,1,0,0	20.42	-0.018	0.000
4800	1,4,8,1	57.67	68.73	1,1,8,1	64.00	-0.099	0.074
6400	1,4,8,1	113.19	128.04	1,2,8,1	125.24	-0.096	0.022
8000	1,4,8,1	195.33	226.25	1,3,8,1	222.86	-0.124	0.015
9600	1,4,8,1	309.25	340.86	1,4,8,1	340.86	-0.093	0.000

5.5　モデル評価

作成したモデルを用いて，N =1600,3200,4800,6400,8000,9600 の 6 つのサイズについて，それぞれ表 5 (評価時) の全てのクラスタ構成 (62 構成) に対して実行時間を予測し，予測最良構成を求めた．また同構成に対して実際に実行時間を測定し，実測最良構成を調べた．結果を表 7 に示す．

実測実行時間の誤差 $(\hat{\tau} - \hat{T})/\hat{T}$ は，$N = 1600$ では 51.8 %と若干大きいが，$N \geq 3200$ では 0〜7.4%と最適〜準最適解を選択している．予測実行時間の誤差 $(\tau - \hat{T})/\hat{T}$ は $N = 1600$ で 82.8%とかなり大きくなっているが，実行時間は 3 秒程度と短いので，時間差は 2 秒程度に収まっている．$N \geq 3200$ では誤差は 12.4%以下に収まっており，実用上十分に正確であるといえる．

図 8 に不均一クラスタで可能な全ての構成について予測実行時間 T と実測実行時間 t の相関を調べた結果を示す．$N = 1600$ では不均一構成時の予測実行時間が小さめに見積もられている．この誤差の原因は後に述べる 6.1 節の検討からモデルの簡略化による通信時間の見積もり誤差であると判断できる．一方，$N = 9600$ では全ての構成がほぼ対角線上にあり予測値と実測値がよく一致している．これは，計算時間が支配的になるにつれ通信時間の誤差の影響が相対



(a) $N = 1600$ の相関　　(b) $N = 6400$ の相関

図 8　P–T モデル実測・予測相関図

的に小さくなるためであろう.

6　計算・通信時間分離モデル

本章では実行時間予測モデルを計算時間および通信時間に分離してモデル化することで，モデル精度を改良できるか検討する．3.4 節における検討より，問題サイズ N に対する G_i の計算時間 $T_{ai}(N)$ および通信時間 $T_{ci}(N)$ を式 (3)〜(4) のように見積もる．添字の P, M_i は特定の構成 $[P, M_i]$ に対するモデルであることを表す．この予測式をこれまでのモデルにならい，以下，N–T 予測モデルと呼ぶことにする.

$$T_{ai}(N)|_{P,M_i} = k_0N^3 + k_1N^2 + k_2N + k_3 \tag{3}$$

$$T_{ci}(N)|_{P,M_i} = k_4N^2 + k_5N + k_6 \tag{4}$$

同様に，3.4 節および N–T 予測モデルより，全プロセス数 $P = \sum P_iM_i$ から G_i の計算時間 $T_{ai}(N, P)$ および通信時間 $T_{ci}(N, P)$ を式 (5)〜(6) のように見積もる．添え字の M_i は特定のマルチプロセス数に対するモデルであることを表す.

$$T_{ai}(N, P)|_{M_i} = k_7\frac{T_a(N)|_{P,M_i}}{P} + k_8 \tag{5}$$

$$T_{ci}(N, P)|_{M_i} = k_9P \cdot T_c(N)|_{P,M_i} + k_{10}\frac{1}{P} \cdot T_c(N)|_{P,M_i} + k_{11} \tag{6}$$

6.1　モデル評価

テストケースの測定およびモデルの構築は 5.4 節と同様に表 5 (構築時) のパラメータで行う．作成したモデルを用いて，不均一クラスタで可能な全ての構

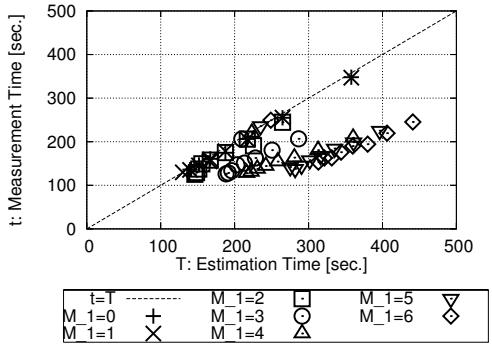
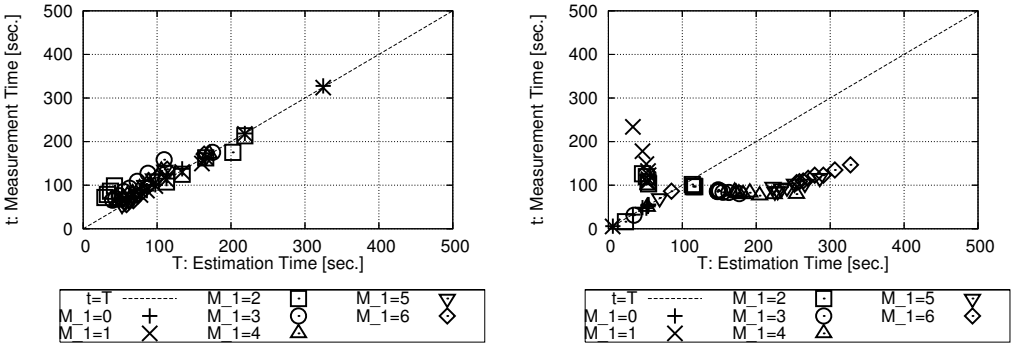


図 9　分離モデル実測・予測相関図 ($N = 6400$)



(a) 計算時間の相関　　(b) 通信時間の相関

図 10　予測誤差の原因

成についてサイズ $N = 6400$ における予測実行時間 T と実測実行時間 t の相関を調べた結果を図 9 に示す．この結果，マルチプロセス数 M_1 が 3 以上の場合に誤差が大きくなることがわかった．

誤差の原因を調べるため，予測実行時間と実測実行時間の相関について，さらに計算時間および通信時間に分けて調べた (図 10)．この結果，計算時間の実測値と予測値はよく一致していた．一方，通信時間は均一構成時の予測は正しいが，不均一な構成の予測については誤差が大きいことが分かった．通信時間に大きな誤差が発生する原因は不均一クラスタモデルの簡略化で通信相手やトポロジーを無視したためと考えられる．

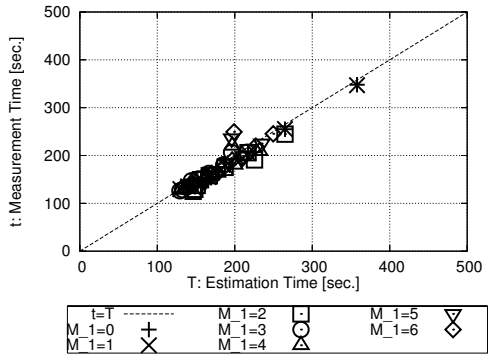


図 11　補正後の分離モデル実測・予測相関図 ($N = 6400$)

6.2　モデルの補正と評価

この誤差を補正するため，本研究ではモデル予測値と実測値の逆比をかけることにする．図 9 では各マルチプロセス数 M_1 ごとに強い線形性が見られるため，線上の 1 構成の実測値のみを取得すれば M_1 上の予測値全体の補正が可能である．

今回の評価例では各 M_1 に対して $N = 6400, P_1 = 1, M_1 = 3$〜$6, P_2 = 8, M_2 = 1$ の不均一構成時の実測値をもとに補正を行なった．$M_1 \leq 2$ の範囲については予測値と実測値が良く一致しているため補正は行なわない．また $P_2 = 8$ で補正したのは，Pentium-II を全て投入した場合の精度を良くするためであるが，妥当性については検討の余地がある．

上記手法による補正後の相関関係を図 11 に示す．補正前に比べてほとんどの構成が対角線上にあり精度が改善されている．

改良した予測モデルを評価するため，N =1600,3200,4800,6400,8000,9600 の 6 つのサイズについて，それぞれ表 5 (評価時) の全てのクラスタ構成 (62 構成) に対して実行時間を予測し，予測最良構成を求めた．また同構成に対して実際に実行時間を測定し，実測最良構成を調べた．結果を表 8 に示す．

実測実行時間の誤差 $(\hat{\tau} - \hat{T})/\hat{T}$ は，$N = 1600$ では 51.8% とやや大きいが，実行時間は 3 秒程度と短いので，時間差は 1.5 秒程度である．$N \geq 3200$ では 0.8%以下の最適〜準最適解な構成が選択されている．予測実行時間の誤差 $(\tau - \hat{T})/\hat{T}$ は $N = 1600$ で 25.6%と若干誤差が大きくなっているが，これも時間差は 0.72 秒と小さい．$3200 \leq N \leq 9600$ では誤差は 3.2%以下と実用上十分に高い精度が得られている．

表 8　N に対する予測値・実測値の最良値と誤差 (補正後の分離モデル)

サイズ	予測による最良構成			実測による最良構成		誤差	
N	P_1, M_1, P_2, M_2	τ	$\hat{\tau}$	P_1, M_1, P_2, M_2	$\hat{T}$	$(\tau - \hat{T})/\hat{T}$	$(\hat{\tau} - \hat{T})/\hat{T}$
1600	1,2,0,0	2.10	4.28	1,1,0,0	2.82	-0.256	0.518
3200	1,1,0,0	20.04	20.42	1,1,0,0	20.42	-0.019	0.000
4800	1,1,8,1	65.21	64.00	1,1,8,1	64.00	0.019	0.000
6400	1,3,8,1	129.23	126.24	1,2,8,1	125.24	0.032	0.008
8000	1,3,8,1	215.64	222.86	1,3,8,1	222.86	-0.032	0.000
9600	1,3,8,1	331.58	341.11	1,4,8,1	340.86	-0.027	0.001

しかし，このようなマルチプロセス数ごとの線形性が常に存在するとは限らず，応用に依存しない手法であるとは言いがたい．一方，このような補正なしにモデルの誤差を減らすためには，(1) モデルの種類を増やす，(2) モデルを複雑にする，(3) より多くのパラメータを導入するなどの対処が必要になると考られる．

しかし，モデルの数を増やせば組合せが増えて収拾がつかなくなる．モデルを複雑にして精度を上げることは可能だが，モデルを応用に特化させると本研究の手法を既存の広範囲の応用に適用できなくなる．モデルパラメータを増やせば，パラメータ抽出のためのテストケース数が増え，モデル構築時間が増大するなど，いずれも望ましくない．

以上の理由から，本研究では，モデルの精度を上げて誤差を減らすことを棚上げとする．実用的な誤差の範囲内であればモデル数やパラメータ数の少ない，より単純なモデルを用いたほうが，より広い応用に手法を適用できる．

7　測定時間の削減とモデル精度

本研究では，実際の不均一クラスタ上で実行した多数のテストケースの測定結果より実行時間予測モデルを構築する．しかしクラスタの測定には単純な構成の場合でも長時間を要する．例えば，5.4 節の 2 種類のプロセッサからなる単純なクラスタでも，その測定時間は表 6 に示すように 6.4 時間程度を必要とする．この章ではテストケースを減らすことでクラスタ測定時間を削減し，モデル精度がどの程度悪化するか調べる．実用的なモデル精度を保ちながらどの程度測定時間を短縮することができるか明らかにする．

7.1　モデル測定点数の削減

測定時間を削減するにはサイズ N が大きいテストケースを減らせば効果的である．5.4 節では表 5 (構築時) の測定パラメータで，サイズ $N = 400, 600, 800, 1200,$

表 9　測定所要時間 (測定数削減時)

	N–T モデル構築時に測定する N	測定点数	測定時間 [sec.]
N9	400, 600, 800, 1200, 1600, 2400, 3200, 4800, 6400	9	22868.8
N8	400, 600, 800, 1200, 1600, 2400, 3200, 4800	8	10679.5
N7	400, 600, 800, 1200, 1600, 2400, 3200	7	4753.5
N6	400, 600, 800, 1200, 1600, 2400	6	2514.1
N5S	400, 600, 800, 1200, 1600	5	1299.3

表 10　N に対する予測値・実測値の最良値と誤差 (N8 モデル)

サイズ	予測による最良構成			実測による最良構成		誤差	
N	P_1, M_1, P_2, M_2	τ	$\hat{\tau}$	P_1, M_1, P_2, M_2	$\hat{T}$	$(\tau - \hat{T})/\hat{T}$	$(\hat{\tau} - \hat{T})/\hat{T}$
1600	1,1,0,0	2.83	2.82	1,1,0,0	2.82	0.002	0.000
3200	1,1,0,0	20.77	20.42	1,1,0,0	20.42	0.017	0.000
4800	1,4,8,1	57.31	68.73	1,1,8,1	64.00	-0.105	0.074
6400	1,4,8,1	110.17	128.04	1,2,8,1	125.24	-0.120	0.022
8000	1,3,8,1	187.68	222.86	1,3,8,1	222.86	-0.158	0.000
9600	1,3,8,1	287.74	341.11	1,4,8,1	340.86	-0.156	0.001

$1600, 2400, 3200, 4800, 6400$ のそれぞれに対して測定を行っている．この節では，同じパラメータで測定上限を $N = 6400$ から 1600 まで順にテストケースを削減し，モデルの予測精度がどの程度悪化するか調べる．N–T 予測モデルは実測値が 4 点あれば構築できるが，未知パラメータ数と実測値の数が等しいと過剰なフィッティングによりモデル精度が実用的でないほど悪化する可能性があるため，ここでは余裕をもたせ下限を 5 点とした．

表 9 にモデル名と N の測定範囲および測定所要時間をまとめた．N を 9 点測定するモデルを N9 モデルとし，以下順に N を 5 点まで削減する．N の測定数が 5 点のモデルについては後ほど N=400〜2400 以外の測定範囲の場合についても検討するため S を末尾に付加して区別している．

N9〜N5S のそれぞれについてモデルを作成し，予測最良構成と予測実行時間を見積もった．N8〜N5S モデルの評価結果を表 10〜表 13 に示す．N9 モデルの評価結果は表 7 と同じである．

図 12 に実測実行時間の誤差 $(\hat{\tau} - \hat{T})/\hat{T}$ および予測実行時間の誤差 $(\tau - \hat{T})/\hat{T}$ をまとめた．図 12(a) の実測実行時間誤差では，N9 および N8 モデルの誤差は $N \geq 3200$ では最大 7.4%とほぼ変わらない．$N = 1600$ では N9 モデルの誤差は 51.8%とやや悪いが，時間差は 1.5 秒程度である．N6 モデルの誤差は最大 1.2%と N9,N8 モデルよりも良く予想以上の結果となったが，後で述べるように予測実行時間誤差は N9,N8 モデルよりも大きく，偶然の結果である可能性が高い．N7

表 11　N に対する予測値・実測値の最良値と誤差 (N7 モデル)

サイズ	予測による最良構成			実測による最良構成		誤差	
N	P_1, M_1, P_2, M_2	τ	$\hat{\tau}$	P_1, M_1, P_2, M_2	$\hat{T}$	$(\tau - \hat{T})/\hat{T}$	$(\hat{\tau} - \hat{T})/\hat{T}$
1600	1,1,0,0	2.82	2.82	1,1,0,0	2.82	0.001	0.000
3200	1,1,0,0	20.80	20.42	1,1,0,0	20.42	0.018	0.000
4800	1,4,8,1	56.29	68.73	1,1,8,1	64.00	-0.120	0.074
6400	1,4,8,1	100.63	128.04	1,2,8,1	125.24	-0.197	0.022
8000	1,6,8,1	148.40	267.81	1,3,8,1	222.86	-0.334	0.202
9600	1,6,8,1	180.22	421.43	1,4,8,1	340.86	-0.471	0.236

表 12　N に対する予測値・実測値の最良値と誤差 (N6 モデル)

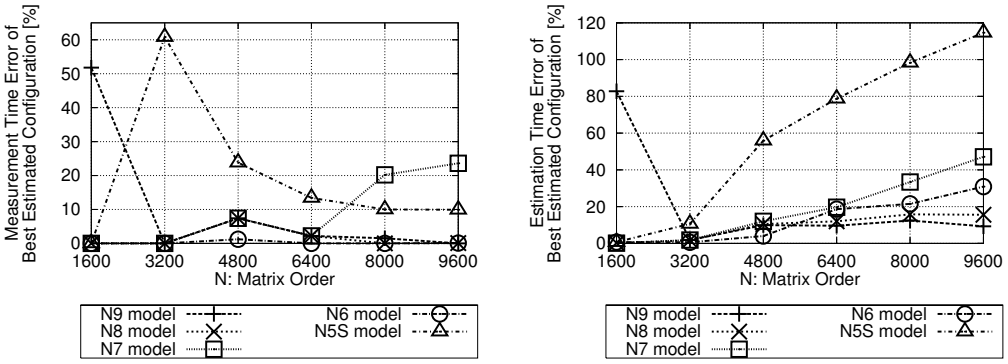
サイズ	予測による最良構成			実測による最良構成		誤差	
N	P_1, M_1, P_2, M_2	τ	$\hat{\tau}$	P_1, M_1, P_2, M_2	$\hat{T}$	$(\tau - \hat{T})/\hat{T}$	$(\hat{\tau} - \hat{T})/\hat{T}$
1600	1,1,0,0	2.84	2.82	1,1,0,0	2.82	0.007	0.000
3200	1,1,0,0	20.54	20.42	1,1,0,0	20.42	0.006	0.000
4800	1,1,0,0	66.55	64.75	1,1,8,1	64.00	0.040	0.012
6400	1,2,8,1	148.52	125.24	1,2,8,1	125.24	0.186	0.000
8000	1,3,8,1	270.78	222.86	1,3,8,1	222.86	0.215	0.000
9600	1,3,8,1	446.28	341.11	1,4,8,1	340.86	0.309	0.001

表 13　N に対する予測値・実測値の最良値と誤差 (N5S モデル)

サイズ	予測による最良構成			実測による最良構成		誤差	
N	P_1, M_1, P_2, M_2	τ	$\hat{\tau}$	P_1, M_1, P_2, M_2	$\hat{T}$	$(\tau - \hat{T})/\hat{T}$	$(\hat{\tau} - \hat{T})/\hat{T}$
1600	1,1,0,0	2.84	2.82	1,1,0,0	2.82	0.007	0.000
3200	1,4,8,1	18.25	32.83	1,1,0,0	20.42	-0.106	0.608
4800	1,5,8,1	28.24	79.24	1,1,8,1	64.00	-0.559	0.238
6400	1,5,8,1	26.64	142.05	1,2,8,1	125.24	-0.787	0.134
8000	1,5,8,1	3.82	245.21	1,3,8,1	222.86	-0.983	0.100
9600	1,5,8,1	-49.66	374.49	1,4,8,1	340.86	-1.146	0.099

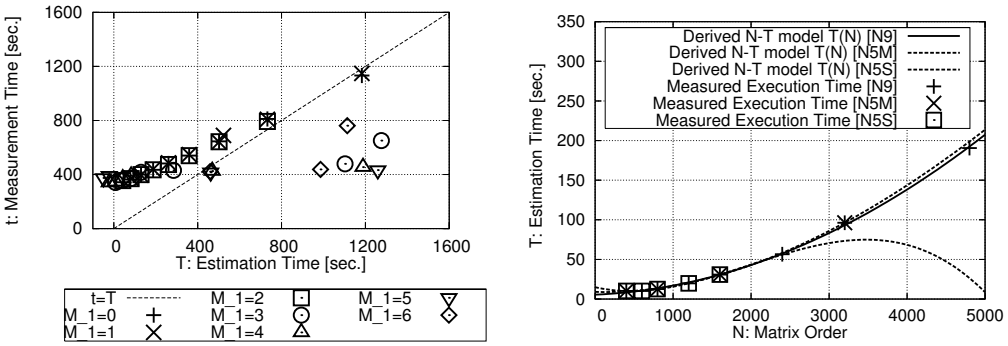
モデルの誤差は N=6400 以降収束せず増大してしまい，サイズ N が大きい場合には N5S モデルよりも精度が悪く実用的でない．測定時間が 20 分程度と最も小さい N5S モデルの誤差は最大 60.8%とやや大きいが、サイズ N が大きい場合には 10%程度に収まっており実用的な精度と言える．

図 12(b) の予測実行時間誤差では，N9, N8 モデルの誤差はそれぞれ最大 12.4%, 15.8%と 2 割以下に収まり実用的な値となっている．一方，N5S〜N7 モデルの誤差は N とともに増大している．N7 モデルの誤差は最大 47.1%と、N6 モデルの最大誤差 30.9%より大きい．測定点数の少ない N6 モデルが N7 モデルより優れている理由として，(1)N6 モデルの測定値の精度が偶然良かったか，(2)N7 モデルの構築に使用した測定値の何らかの異常 (測定中に何らかの負荷がかかった等) が考えられる．N7 モデルの実行時間誤差が収束せずに増え続けてしまう



(a) 実測実行時間誤差　　(b) 予測実行時間誤差図

図 12　N9〜N5S の予測最良構成の誤差



(a) 実測・予測相関図 ($N = 9600$)　　(b) N–T モデルの破綻

図 13　N5S モデルの検討

こと，予測実行時間誤差が $N \geq 3200$ 以降，他のモデルよりも速いペースで増え続けることから考えると，後者の測定値の異常の可能性が高いと考えられる．一方，N5S モデルの誤差は全体的に大きく，$N = 9600$ では 100%を超える異常な挙動を示している．しかし，実測実行時間の誤差は $N = 9600$ でも 10%程度と少ない．

この原因を調べるため N5S モデルの予測値と実測値の相関をとった (図 13(a))．予測値は実測値から大きく外れ，一部の構成では負の予測実行時間を返している．しかし予測値と実測値の間には正の相関があり，予測値の大小関係は実測

表 14　測定所要時間 (5 点測定時)

	N–T モデル構築時に測定する N	測定点数	測定時間 [sec.]
N9	400, 600, 800, 1200, 1600, 2400, 3200, 4800, 6400	9	22868.8
N5S	400, 600, 800, 1200, 1600	5	1299.3
N5M	400, 800, 1600, 3200, 6400	5	15264.6
N5L	1600, 2400, 3200, 4800, 6400	5	22115.2

表 15　N に対する予測値・実測値の最良値と誤差 (N5M モデル)

サイズ	予測による最良構成			実測による最良構成		誤差	
N	P_1, M_1, P_2, M_2	τ	$\hat{\tau}$	P_1, M_1, P_2, M_2	$\hat{T}$	$(\tau - \hat{T})/\hat{T}$	$(\hat{\tau} - \hat{T})/\hat{T}$
1600	1,1,0,0	2.82	2.82	1,1,0,0	2.82	-0.001	0.000
3200	1,1,0,0	20.81	20.42	1,1,0,0	20.42	0.019	0.000
4800	1,4,8,1	58.89	68.73	1,1,8,1	64.00	-0.080	0.074
6400	1,4,8,1	113.29	128.04	1,2,8,1	125.24	-0.095	0.022
8000	1,4,8,1	190.43	226.25	1,3,8,1	222.86	-0.146	0.015
9600	1,4,8,1	293.52	340.86	1,4,8,1	340.86	-0.139	0.000

値の大小関係を比較的正しく再現しているため，予測最良構成に大きな誤りが現れなかったものと考えられる．

7.2　5 点測定時のモデル精度

図 13(b) に N5S モデルの $P_2 = 8, M_2 = 5$ の N–T 予測モデルを示す．予測実行時間の異常の理由は，この図 13(b) のように N が大きい場合に $T_i < 0$ となるような 3 次式が抽出されているためである．

このようなパラメータが抽出された原因は，(1)N の測定範囲が小さすぎるたか，(2)N の測定点数が少なすぎたかのどちらかであろうと考えられる．そこで，N の測定点数は 5 点に固定し，測定範囲を変化させたモデルを評価することでどちらが原因なのかを特定することにした．N の測定範囲を広げることで精度が改善されれば測定点数の少なさが原因ではないことになる．

表 14 にモデル名と N の測定範囲および測定所要時間をまとめた．N5S は N=400〜1600 の小さい範囲の 5 点，N5L は N=1600〜6400 の 5 点，N5M はそれらの中間で N=400〜6400 の 5 点とした．N5S モデルの測定時間は最も短く 20 分程度，N5M の測定時間は 4.2 時間程度，N5L モデルは N9 モデルとほぼ同じ 6 時間程度である．

N9，N5S〜N5L のそれぞれについてモデルを作成し，予測最良構成と予測実行時間を見積もった．N5M〜N5L モデルの評価結果を表 15〜表 16 に示す．N9 モデルの評価結果は表 7，N5S モデルの評価結果は表 13 と同じである．

図 14 に実測実行時間の誤差 $(\hat{\tau} - \hat{T})/\hat{T}$ および予測実行時間の誤差 $(\tau - \hat{T})/\hat{T}$

表 16　N に対する予測値・実測値の最良値と誤差 (N5L モデル)

サイズ	予測による最良構成			実測による最良構成		誤差	
N	P_1, M_1, P_2, M_2	τ	$\hat{\tau}$	P_1, M_1, P_2, M_2	$\hat{T}$	$(\tau - \hat{T})/\hat{T}$	$(\hat{\tau} - \hat{T})/\hat{T}$
1600	1,1,0,0	2.74	2.82	1,1,0,0	2.82	-0.029	0.000
3200	1,1,0,0	20.49	20.42	1,1,0,0	20.42	0.004	0.000
4800	1,4,8,1	57.33	68.73	1,1,8,1	64.00	-0.104	0.074
6400	1,4,8,1	113.28	128.04	1,2,8,1	125.24	-0.095	0.022
8000	1,4,8,1	199.55	226.25	1,3,8,1	222.86	-0.105	0.015
9600	1,4,8,1	323.88	340.86	1,4,8,1	340.86	-0.050	0.000

をまとめた．図 14(a) の実測実行時間誤差では，N5M および N5L モデルの誤差は N9 モデルとほぼ同一である．N=1600 の誤差は 51.8%から 0%に減少しており精度が向上している．5 点しか測定しないモデルでも N 大の範囲まで測定すれば十分に実用的であることが分かる．

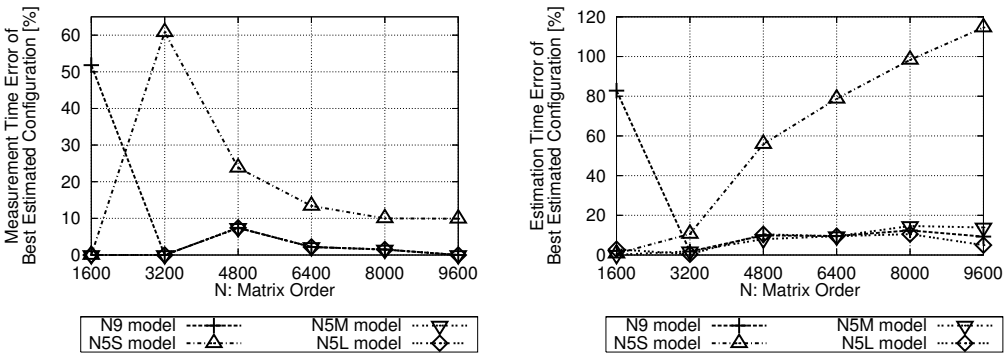
図 14(b) の予測実行時間誤差では，N9 モデルの最大誤差が 12.4%であるのに対し，N5M モデルの最大誤差は 14.6%と若干精度が悪化する．これは測定点数を削ったことによる誤差の増大であると考えられる．一方，N5L モデルの最大誤差は 10.5%で，N9 モデルよりも少ない測定点数で良い精度が得られている．これは HPL では計算時間が支配的であるため，予測誤差の大きい通信時間が支配的な N 小の範囲を測定しない方がモデル精度が向上するためと考えられる．

この節では，(1) 可能な限り N が大きい範囲で 5 点程度を測定することで実用的な精度をもったモデルが得られること，(2) さらに測定時間を削減する場合は N の測定範囲を小さくするしかないが，その分，予測誤差は増大すること，(3) 過度に N の測定範囲を小さくした場合には N–T 予測モデルが破綻し，実行時間が負となるパラメータが抽出されてしまうことを示した．

7.3　代用モデルの使用と測定時間の削減

本節では，表 1 に示す 3 種類のプロセッサからなる不均一クラスタ全体を用いて評価を行う．本節で用いる測定パラメータを表 17 に示す．ここで，Athlon の PE 数とプロセス数を P_1, M_1，Pentium-III の PE 数とプロセス数を P_2, M_2，Pentium-II の PE 数とプロセス数を P_3, M_3 とする．N については，N9 モデルと同じ N=400〜6400 の 9 点で測定する．Athlon の P–T モデルは 5.4 節と同様に Pentium-II の P–T モデルに定数 0.307 を乗じて代用する．

Pentium-III は 4 プロセッサあるので，P_2=2,3,4 に対応する N–T モデルから P–T モデルを構築することができる．この場合のモデルを SUBST1 と呼ぶことにする．このときのテストケースの実行時間は，Athlon が 2180 秒，Pentium-III



(a) 実測実行時間誤差　　(b) 予測実行時間誤差図

図 14　N9,N5S〜N5L の予測最良構成の誤差

表 17　HPL 測定時のクラスタ構成パラメータ

	サイズ	Athlon		Pentium-III		Pentium-II		構成数	測定時間
	N	P_1	M_1	P_2	M_2	P_3	M_3		
SUBST1	400〜6400	1	1〜6	1〜4	1〜3	1〜8	1〜6	66	7.8 時間
SUBST2	400〜6400	1	1〜6	1	1〜3	1〜8	1〜6	57	6.9 時間
評価用実測時	1600〜9600	0〜1	1〜6	0〜4	1〜3	0〜8	1	818	−−

が 5200 秒，Pentium-II が 20689 秒で，合計 7.8 時間である．

しかし，4 プロセッサの場合，P–T 予測モデルの未知パラメータ数 3 に対し，N–T モデルを 3 つしか用意できないため，モデルの精度は非常に悪いと考えられる．そこで，Pentium-III の P–T モデルも Pentium-II の P–T モデルに定数 0.637 を乗じて代用する．この場合のモデルを SUBST2 と呼ぶことにする．このときのテストケースの実行時間は，Athlon が 2180 秒，Pentium-III が 1993 秒，Pentium-II が 20689 秒で，合計 6.9 時間である．

SUBST1，SUBST2 のそれぞれについてモデルを作成し，N =1600,3200,4800, 6400,8000,9600 の 6 つのサイズについて，それぞれ表 17 (評価時) の全てのクラスタ構成 (818 構成) に対して実行時間を予測し，予測最良構成と予測実行時間を見積もった．評価結果を表 18〜表 19 に示す．

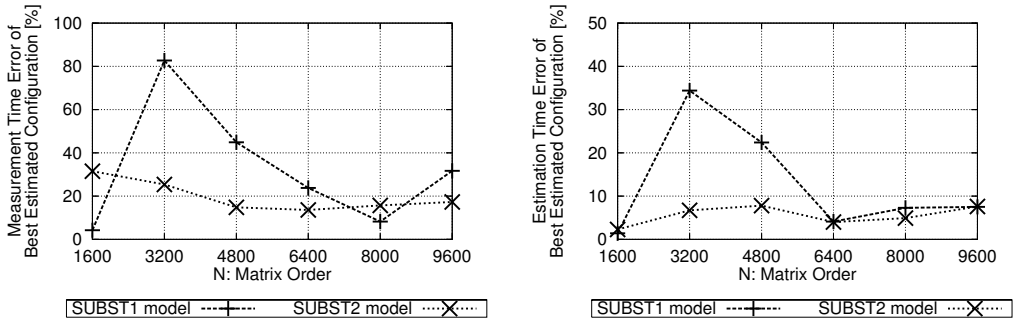
図 15 に実測実行時間の誤差 $(\hat{\tau} - \hat{T})/\hat{T}$ および予測実行時間の誤差 $(\tau - \hat{T})/\hat{T}$ をまとめた．図 15(a) の実測実行時間誤差では，SUBST1 の最大誤差 82.7% に対し SUBST2 は最大誤差 31.5%であり，$N \geq 4800$ では誤差 17.3%に収まる．モデル構築範囲において N にともなって誤差が減少しているのは，予測誤差の小

表 18　N に対する予測値・実測値の最良値と誤差 (SUBST1 モデル)

サイズ	予測による最良構成			実測による最良構成		誤差	
N	$P_1, M_1, P_2, M_2, P_3, M_3$	τ	$\hat{\tau}$	$P_1, M_1, P_2, M_2, P_3, M_3$	$\hat{T}$	$(\tau - \hat{T})/\hat{T}$	$(\hat{\tau} - \hat{T})/\hat{T}$
1600	1,1,0,0,0,0	3.38	3.47	1,1,2,2,0,0	3.33	0.014	0.042
3200	1,4,0,0,8,1	24.64	33.49	1,1,2,2,0,0	18.33	0.344	0.827
4800	1,4,0,0,8,1	63.15	74.79	1,2,2,2,0,0	51.61	0.224	0.449
6400	1,5,4,3,8,1	108.98	129.54	1,2,2,2,0,0	104.65	0.041	0.238
8000	1,4,4,3,8,1	202.36	204.01	1,2,4,2,8,1	188.60	0.073	0.082
9600	1,5,4,3,0,0	314.81	385.58	1,3,4,2,8,1	292.77	0.075	0.317

表 19　N に対する予測値・実測値の最良値と誤差 (SUBST2 モデル)

サイズ	予測による最良構成			実測による最良構成		誤差	
N	$P_1, M_1, P_2, M_2, P_3, M_3$	τ	$\hat{\tau}$	$P_1, M_1, P_2, M_2, P_3, M_3$	$\hat{T}$	$(\tau - \hat{T})/\hat{T}$	$(\hat{\tau} - \hat{T})/\hat{T}$
1600	1,1,4,1,0,0	3.25	4.38	1,1,2,2,0,0	3.33	-0.023	0.315
3200	1,2,4,1,0,0	17.11	22.98	1,1,2,2,0,0	18.33	-0.067	0.254
4800	1,2,4,1,0,0	47.58	59.23	1,2,2,2,0,0	51.61	-0.078	0.148
6400	1,2,4,1,0,0	100.48	118.90	1,2,2,2,0,0	104.65	-0.040	0.136
8000	1,4,4,1,8,1	179.30	218.27	1,2,4,2,8,1	188.60	-0.049	0.157
9600	1,4,4,1,8,1	270.37	343.51	1,3,4,2,8,1	292.77	-0.076	0.173



(a) 実測実行時間誤差　　(b) 予測実行時間誤差図

図 15　SUBST1〜2 の予測最良構成の誤差

さい計算時間が支配的になってくるためである．一方，モデルの構築範囲外で N にともなって誤差が増大しているのは，外挿により予測誤差が増大することによる影響であると考えられる．図 15(b) の予測実行時間誤差では，SUBST1 の最大誤差 34.4%に対し，SUBST2 は最大誤差 7.8% である．

SUBST1 の誤差は実測・予測実行時間誤差ともに SUBST2 よりも大きい．従って，少ないプロセッサ数で構築された P–T モデル (SUBST1) を使用するより、多くのプロセッサ数で構築された P–T モデル (SUBST2) を使用するほうが精度が向上すると言える．

つまり十分な測定点数が取れない場合，無理に実測値からモデルを構築する

よりも，精度の高いモデルから代用するほうが良い結果になる場合がある．

またプロセッサの種類が増えるとテストケース実行時間が増加するが，精度の良いモデルの代用を積極的に利用することにより，テストケース実行時間を抑制できる．本節では N を 9 点測定してモデルを構築したが，7.2 節で示した通り，精度を保ったまま N を 5 点程度まで削減してテストケース実行時間を削減することができる．

8 おわりに

本研究では，不均一クラスタ上で既存の HPC 応用を負荷分散するため，マルチプロセス法について検討した．最適な PE 群およびマルチプロセス数を選択するため，実測値から実行時関予測モデルを構築し，実際にモデルを使って最適〜準最適構成を予測することに成功した．

今回の研究では，最適構成を探すために，不均一クラスタ上で可能な全構成について総当りで実行時間を予測した．しかし，このような素朴な手法では，プロセッサの種類が増加すれば組み合わせ数が爆発する．今後は分枝限定などの手法を取り入れて，探索空間を制限したい．また，準最適構成を求めるための近似解法 (ヒューリスティックなど) についても検討し，その求解時間や精度について評価を進めてゆきたい．

予測モデルの一層の精度向上も，今後の課題である．また，HPL 以外の応用に関しても検討し，本手法が有効であるか評価を進めて行きたい．

参 考 文 献

1) Kalinov, A. and Lastovetsky, A.: Heterogeneous Distribution of Computations while Solving Linear Algebra Problems on Networks of Heterogeneous Computers, *Proc. HPCN Europe 1999* (Sloot, P., Bubak, M., Hoekstra, A. and Hertzberger, B.(eds.)), IEEE Computer Society Press, pp. 191–200 (1999).

2) Beaumont, O., Boudet, V., Petitet, A., Rastello, F. and Robert, Y.: A Proposal for a Heterogeneous Cluster ScaLAPACK (Dense Linear Solvers), *IEEE Transaction on Computers*, Vol. 50, No. 10, pp. 1052–1070 (2001).

3) 笹生健, 松岡聡, 建部修見: ヘテロなクラスタ環境における並列 LINPACK の最適化, 情処研報 2001–HPC–86, pp. 49–54 (2001).

4) Petitet, A., Whaley, R. C., Dongarra, J. and Cleary, A.: HPL – A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers. http://www.netlib.org/benchmark/hpl/.
5) 岸本芳典, 市川周一: 不均一クラスタ上での並列 Linpack の性能に関する検討, 並列処理シンポジウム JSPP2002, pp. 177–178 (2002).
6) Gropp, W. and Lusk, E.: MPICH – A Portable Implementation of MPI. http://www-unix.mcs.anl.gov/mpi/mpich/.
7) Cheng, Y. and Hu, C.: MOSFET のモデリングと BSIM3 ユーザーズガイド, 丸善 (2002).
8) Galassi, M., Davies, J., Theiler, J., Gough, B., Jungman, G., Booth, M. and Rossi, F.: GNU Scientific Library Reference Manual (2003). Edition 1.4, for GSL Version 1.4.