不均一クラスタ上での実行時間予測モデルとその改良

岸 本 芳 典[†]　市 川 周 一[†]

均一環境用に書かれた応用を不均一クラスタで実行すると，負荷不均衡により性能上の問題を生ずる．また，一部の PE には仕事を割り当てないほうが全体の実行時間が短縮できる場合がある．本研究では，高速な要素プロセッサ (PE) 上で複数のプロセスを起動することにより，全体の実行時間を短縮する方法を検討する．さらに，各 PE 上の実行時間を実測値からモデル化し，得られた予測モデルを用いて最適な PE 構成およびマルチプロセス数を予測する．モデルの合成によりパラメータ抽出時間を削減する方法も述べる．構築したモデルを使って，多くの場合に誤差 20%以下の準最適構成を求めることができた．

An Improved Execution-Time Estimation Model for Heterogeneous Clusters

Yoshinori Kishimoto[†] and Shuichi Ichikawa[†]

A heterogeneous cluster can incur the performance degradation caused by the load unbalance in executing the application for homogeneous cluster. The total execution time can be sometimes improved by neglecting some of the PEs because communication time is reduced. This study examines to invoke multiple processes on fast processing elements (PEs) to avoid load unbalance. In this study, the execution time of each PE is firstly modeled from measurement results. Then, the derived model is used to estimate the optimal PE configuration and process configuration. Model composition is also proposed to reduce parameter extraction time. The derived models yield sub-optimal configurations in most cases, where the errors are less than 20%.

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，手近な計算機を寄せ集めてクラスタを構築したり，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．このような場合，クラスタは必然的に不均一になる．

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各プロセスに負荷を均一に割り当てる．そのような応用を不均一クラスタで実行すると，プロセス間の実行時間不均衡により性能低下を生じる．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逐一書き直すのは容易でない．

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 (**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[1] は，HPL (High Performance Linpack)[2] にマルチプロセス法を適用して性能を測定し，性能低下が著しいと報告した．しかし著者ら[3] の実験では，性能低下は通信ライブラリ MPICH[4] の実装上の問題によるもので，MPICH の新しいバージョンを利用すれば大きな性能低下はなかった．性能改善後の測定では，HPL の問題サイズが大きい場合，オーバヘッドは実行時間の 2〜3 割程度である．この程度のオーバヘッドであれば，マルチプロセス法は充分に実用に耐えうると考えられる．

本研究は，マルチプロセス法を利用して，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を目的とする．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．

† 豊橋技術科学大学 知識情報工学系
Department of Knowledge-based Information Engineering, Toyohashi University of Technology

以下，本研究では次の順に説明を進める．2 章では，実行時間近似モデルの概要とパラメータ抽出方法について説明する．3 章では，実行時間の実測値からモデルパラメータを抽出し，得られたモデルを利用して実行時間を最小化する実行方法を予測する．さらに予測結果と実測値の比較結果を示す．最後に 4 章で，今後の課題についてまとめる．

2. 実行時間予測モデル

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．この問題を組合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近似式が必要である．

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．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自然に構築できる可能性がある．

2.1 モデルの概略

不均一クラスタ内のプロセスの実行時間は，ネットワーク構成や通信相手に依存する可能性がある．しかし通信相手毎にモデルを構築すると，不均一クラスタではモデルの種類が増えすぎて実用性が失われる．そこで本研究では，通信相手や通信トポロジーを無視して，モデルの単純化を行うこととする．また，不均一クラスタ内に複数の等価な PE が含まれている場合，等価な PE 上のマルチプロセス数は同一とすることで，モデルを単純化し，可能な構成の組合せを減らす．このような単純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 3 章で述べる．

等価な PE をまとめたグループを G_i で表すとする．各 G_i 内で使用するプロセッサ数を P_i $(0 \leq P_i \leq |G_i|)$，それらの PE 上のマルチプロセス数を M_i で表すと，システム内の全プロセス数 P は $P = \sum P_i M_i$ で表される．予測モデルの仕事は，N, P, M_i から PE_i 上の各プロセスの実行時間 T_i を予測することである．

不均一クラスタ全体の実行時間 T は，$T = \max_i(T_i)$ で近似できる．明らかに T_i は HPL 実行時のプロセス格子に依存するが，本研究では横一列 (1 次元ブロックサイクリック分割) のプロセス格子に関して評価を行う．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．

2.2 過去のモデルの問題点

HPL の問題サイズ N, プロセス格子サイズ $1 \times P$ に対して，HPL 実行時の通信量は $O(N^2)$，計算量は $O(N^3)$ と見積もられる[1)5)]．著者らは以前の研究[5)] で，問題サイズ N に対する PE_i の計算時間 $T_{ai}(N)$，通信時間 $T_{ci}(N)$，PE_i の総実行時間 $T_i(N)$ を以下の式で見積もった．式中，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．

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

$$T_{ci}(N)|_{P,M_i} = k_4 N^2 + k_5 N + k_6 \quad (2)$$

$$T_i(N)|_{P,M_i} = T_{ai}(N)|_{P,M_i} + T_{ci}(N)|_{P,M_i} \quad (3)$$

このモデルでパラメータを抽出し，予測実行時間と実測実行時間を比較したところ，通信時間 $T_{ci}(N)$ に大きな誤差が残った．詳しい分析に関しては紙数の関係上省略するが，$T_{ci}(N)$ の誤差の原因の一部は，2.1 節に述べたような単純化にあることが推測された．従って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えられた．

しかしながら，2.1 節でも述べたように，そもそも自由度の多い不均一クラスタでモデルの数を増やすと，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目的 (既存の広範囲の応用に適用すること) から外れる．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，モデル構築時間も増大する．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．むしろ，実用的な誤差の範囲内でモデルを単純化し，より広い応用に適用可能な方法を探る．また，テストケース数の削減や，テストケース実行時間の削減，パラメータ数の削減に関して検討する．

2.3 N-T モデルと P-T モデル

本研究では，計算時間と通信時間を個別に見積もることをやめて，$T_i(N)$ の近似式を簡略化する．通信量は $O(N^2)$，計算量は $O(N^3)$ なので，$T_i(N)$ は以下の 3 次式として見積もる．

$$T_i(N)|_{P,M_i} = k_0 + k_1 N + k_2 N^2 + k_3 N^3 \quad (4)$$

式に含まれる係数 (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．このモデルを，過去の研究[5)] にならって **N–T モデル**と呼ぶ．

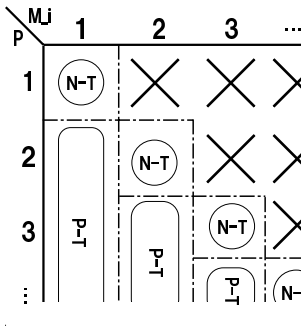

図 1 パラメータによるモデルの切り替え[5)]

N–T モデルは P と M_i の値ごとに必要である．可能な全ての $[P, M_i]$ について N–T モデルを作成すると，テストケース数とモデル数が非常に多くなるので，複数の N–T モデルからプロセス数 P をパラメータに含むモデル (**P–T モデル**) を構築する．過去の研究結果[5)] から，P–T モデル $T_i(N, P)$ は以下の式で表される．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．

$$T_i(N,P)|_{M_i} = k_4 P \cdot T_i(N)|_{P,M_i} + k_5 \frac{1}{P} \cdot T_i(N)|_{P,M_i} + k_6 \qquad (5)$$

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 間通信の有無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 1 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．図 1 の×の部分は，$P = \sum P_i M_i \geq \forall M_i$ であるから元々存在しない．マルチプロセス数 M_i を含めた計算時間・通信時間の定式化は，関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．

2.4 モデルの代用

式 (4) には係数が 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．実際には，測定点数が充分でないと正確なパラメータ抽出ができず，モデルが破綻する (3 章参照)．そこで充分離れた N に対して，充分な点数 (少なくとも 5〜6 点) は測定する必要がある．同様に，式 (5) には係数が 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，充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．

表 2 クラスタ構成パラメータ (2 種のプロセッサ)

	Athlon		Pentium-II	
	P_1	M_1	P_2	M_2
構築時	1	1〜6	1〜8	1〜6
評価時	0〜1	1〜6	0〜8	1

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．

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充分な精度を持つかどうかは，測定によって検証されなければならない．本研究でも，3 章で述べるとおり，モデルの代用によってモデル作成労力を削減する．評価結果については 3 章で述べる．

3. 実行時間予測モデルの評価

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 をグループ化して扱い (2.1 節)，各グループ G_i 内で問題サイズ N，投入 PE 数 P_i，マルチプロセス数 M_i 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．例えば，プロセッサ A が 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．これらの実測値から，G_i に含まれる PE 用のモデルを作成する．G_i の PE 数が少なくてモデルが構築できない場合は，2.4 節で述べたように，他のグループのモデルを代用する．

3.1 2 種のプロセッサからなる不均一クラスタ

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 数とプロセス数を P_2, M_2 とする．

表 2 (構築時) の組合せで，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，Athlon は 6 通り，Pentium-II は 48 通りのテストケースを測定する．各 N におけるテストケース実行時間は**表 3**

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	1000base-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

表 3 測定所要時間 (sec.)

サイズ N	Athlon	Pentium-II
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
Total	2180.3	20689.0

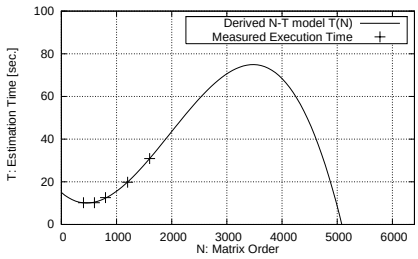

図 2 パラメータ抽出の破綻例

の通りである．

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 (GNU Scientific Library)[6] の `gsl_multifit_linear()` 関数を用いる．GSL によるパラメータ抽出時間は 1 ms 以下で，無視できる程度である．表 2 から明らかな通り，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．そこで Pentium-II の P–T モデルに定数 0.307 を乗じて，Athlon の P–T モデルとして用いる (モデルの代用)．

こうして作成したモデルを用いて，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．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なうこととした．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なわないため，評価時の M_2 は 1 となっている．ただし Athlon の P–T モデルは Pentium-II の P–T モデルから作成するため，モデル構築時には $M_2 = 1$〜6 とする．

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 $\hat{\tau}$ と表している．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．$N = 1600$ では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，時間差としては数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，実用上充分に正確であるといえる．

このモデルを構築するためのテストケース実行時間は，表 3 に示したとおり，のべ 6 時間余りである．Athlon と Pentium-II の測定は並行して実施できるが，それでも Pentium-II の測定に 5 時間半以上必要になる．プロセッサの種類が増えると，更にテストケース実行時間は増加する．

そこで次に，N の測定点数を減らした場合の予測精度に関して調べた．測定する N を，N = 400, 800, 1600, 3200, 6400 に間引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．結果は**表 5** に示す通りである．予測最良構成は実測最良構成と大きく変わることはなく，誤差も全体で 15%程度に収まっている．このときのテストケース実行時間は，のべ 4 時間程度まで削減される (表 3 参照)．

2.4 節でも述べたように，N の測定点数は 5 点は必要である．さらにテストケース実行時間を削減するには，N の小さい範囲で測定するしかない．**表 6** に，N = 400, 600, 800, 1200, 1600 の測定結果から構築したモデルの評価結果を示す．このときテストケース実行時間は 20 分程度である．しかしこのモデルでは，予測実行時間 τ が異常な挙動を示すため，予測実行時間の誤差が非常に大きくなっている．

予測実行時間の異常の理由は，N の範囲が狭いために不適切なモデルが構築されることである．**図 2** は $P_2 = 8, M_2 = 5$ の場合のモデルであるが，$N > 5000$ で $T_i < 0$ となるような 3 次式が抽出されている．図 2 のような不適切なモデルを避けるためには，N の範囲を広くとり，N の測定点数も少なくとも 5 点はとる必要がある．

表 6 において，τ の誤差は非常に大きいのに，予測最良構成は実測最良構成から大きく外れていない．そこで $N = 9600$ において，表 6 のモデルの予測値と実測値の関係を調べてみた (**図 3**)．予測は実測と大き

表 4 予測最良構成と実測最良構成 (N = 400, 600, 800, 1200, 1600, 2400, 3200, 4800, 6400)

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

表 5 予測最良構成と実測最良構成 (N = 400, 800, 1600, 3200, 6400)

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

表 6 予測最良構成と実測最良構成 (N = 400, 600, 800, 1200, 1600)

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

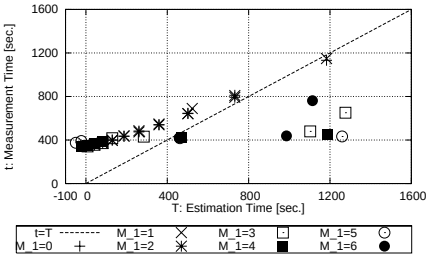

図 3 表 6 のモデルにおける実行時間の予測値と実測値 (N = 9600)

く外れ，一部の構成では負の値を返している．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．

3.2 3 種のプロセッサからなる不均一クラスタ

ここまで本研究では，2 種類のプロセッサからなる非常に単純な不均一クラスタを扱ってきた．本節では，表 1 に示す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 7** に示す．ここで，Athlon の PE 数とマルチプロセス数を P_1, M_1，Pentium-III を P_2, M_2，Pentium-II を P_3, M_3 で表すことにする．N については，表 4 と同じ 9 点で測定する．

表 7 クラスタ構成パラメータ (3 種のプロセッサ)

	Athlon		Pentium-III		Pentium-II	
	P_1	M_1	P_2	M_2	P_3	M_3
構築時	1	1〜6	1〜4	1〜3	1〜8	1〜6
評価時	0〜1	1〜6	0〜4	1〜3	0〜8	1

Penitum-III は 4 プロセッサあるので，$P_2 = 2, 3, 4$ の実測値から P-T モデルを構築することができる．その場合の評価結果を**表 8** に示す．このときのテストケース実行時間は，Athlon が 2180 秒，Pentium-III が 5200 秒，Pentium-II が 20689 秒で，合計 8 時間ほどである．表 8 から明らかな通り，誤差は少々大きめである．

次に，Penitum-III の P-T モデルを Pentium-II の P-T モデルで代用する (係数 0.637 を乗じて作成する) 場合の評価結果を，**表 9** に示す．このときのテストケース実行時間は 7 時間弱である．明らかに，このモデルの方が表 8 よりも精度が良く，$N \geq 4800$ で誤差は 17%程度である．表 8 では Pentium-III の P–T モデルを最小限 ($P_2 = 2, 3, 4$) の実測値から作成したため，誤差が大きくなったと考えられる．充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

表 8 プロセッサ 3 種からなる不均一クラスタの最良構成予測 (Pentium-III の P–T モデルを実測から作成)

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 9 プロセッサ 3 種からなる不均一クラスタの最良構成予測 (Pentium-III の P–T モデルを代用で作成)

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

結果になる場合がある．

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，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．また，本節では N を 9 点測定してモデルを構築したが，3.1 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．

4. 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，実測値から実行時間予測モデルを構築して，実際にモデルを使って (準) 最適構成を予測することに成功した．

予測モデルの一層の精度向上に関しては，今後の課題とする．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ゆきたい．

謝辞 本研究の一部は，堀情報科学振興財団・第 11 回研究助成「不均一な分散処理環境のための行列計算高速化手法」，科学研究費補助金・基盤研究 (C)(2)13680410，文部科学省 21 世紀 COE プログラム「インテリジェントヒューマンセンシング」の援助により行われた．

参考文献

1) 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).
2) 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/.
3) 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).
4) Gropp, W. and Lusk, E.: MPICH – A Portable Implementation of MPI. http://www-unix.mcs.anl.gov/mpi/mpich/.
5) 岸本芳典, 市川周一: 不均一クラスタ上での実行時間予測モデルとその評価, 情処研報 2003–HPC–95, pp. 161–166 (2003).
6) 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).