社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－HPC－111 (1)
2007／8／1

階層統合型粗粒度タスク並列処理における タスク階層決定手法

吉 田 明 正†

本稿では，マルチプロセッサを対象とした階層統合型粗粒度タスク並列処理において，実行時間最小化を目指したタスク階層の決定手法を提案する．階層統合型粗粒度タスク並列処理では，階層ごとにタスク間並列性を抽出した後，ダイナミックスケジューラが全階層のタスクを統一的にプロセッサに割り当てることにより，階層を越えたタスク間並列性を利用することが可能である．本手法では，さらなる性能向上を実現するために，各階層の並列度とスケジューリングオーバヘッドを考慮し，上位階層で十分な並列性を確保できるプログラムにおいては，下位階層のタスク集合を１回のダイナミックスケジューリングによりプロセッサに割り当てる方針をとる．ランダムマクロタスクグラフを用いた性能評価の結果，提案手法は並列実行時間を最小化しており，その有効性が確認された．

A Task-Layer Decision Scheme in Coarse Grain Task Parallel Processing

AKIMASA YOSHIDA†

This paper proposes a task-layer decision scheme to minimize the execution time in the layer-unified coarse grain task parallel processing on multiprocessors. In this parallel processing, the parallelism among tasks of each layer are exploited, and a dynamic scheduler assigns the ready tasks of all layers to processors simultaneously, so that the parallelism between different layers can be utilized. In this scheme, in order to improve the performance, considering the parallelism of each layer and the scheduling overheads, a set of lower-layer tasks may be assigned to a processor by one-time dynamic scheduling when the program has the enough upper-layer parallelism. The performance evaluations using the random macrotask graphs showed that the proposed scheme could minimize the parallel execution time.

1. はじめに

近年，マルチプロセッサやマルチコアプロセッサ上での並列処理手法として，ループ並列処理[1),2)]や，ループやサブルーチン等の粗粒度タスク間の並列性[3)~5)]を利用する粗粒度タスク並列処理が広く用いられている．

粗粒度タスク並列処理[5),8)]では，粗粒度タスク間の並列性を並列化コンパイラが抽出して階層型マクロタスクグラフを生成し，各階層の粗粒度タスクを，グルーピングしたプロセッサに階層的に割り当て並列処理を行っていた．この場合，対象プログラム中の各階層の粗粒度タスクは，その階層を処理すべきプロセッサグループに割り当てられて実行されるため，十分な台数のプロセッサを確保できない場合には，対象プログラムに内在する全階層の粗粒度タスク間並列性を利用できない可能性がある．

そこで，粗粒度タスク並列処理で用いられている階層型マクロタスクグラフ[5)]を利用しつつ，対象プログラム中の異なる階層の粗粒度タスクを統一的に取り扱い，異なる階層にまたがった粗粒度タスク間並列性を最大限に利用する階層統合型実行制御手法[6),7)]が提案されている．

本稿では，この階層統合型実行制御を伴う粗粒度タスク並列処理において，各階層の並列性とダイナミックスケジューリングオーバヘッドを考慮し，実行時間を最小化するタスク階層決定手法を提案する．本手法では，上位階層に並列性が多いプログラムの場合，上位階層タスクがプロセッサを多く確保するため，下位階層タスクが使用できるプロセッサ数は制限される．このような場合には，下位階層タスク集合を１タスクのように取り扱い，１回のスケジューリングで１プロセッサに割り当て，ダイナミックスケジューリングオーバヘッドを軽減する．

本稿の構成は以下の通りとする．第２章では階層統合型粗粒度タスク並列処理の概要を述べる．第３章では並列度を考慮したタスク階層決定手法について述べる．第４章では，ランダムマクロタスクグラフを用いたシミュレーション実行により，タスク階層決定手法の性能を評価する．第５章でまとめを述べる．

† 東邦大学理学部情報科学科
Department of Information Science, Toho University

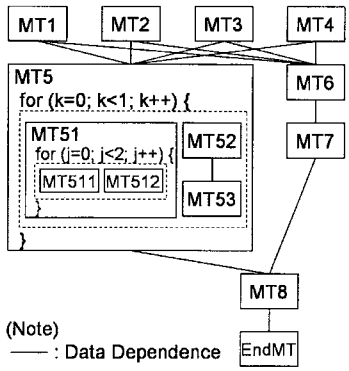

図 1　階層型マクロタスクグラフ（MTG）.

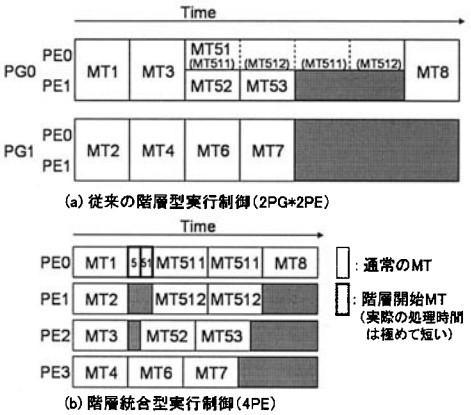

図 2　粗粒度タスク並列処理の実行イメージ.

2. 階層統合型粗粒度タスク並列処理

階層統合型粗粒度タスク並列処理では，粗粒度タスク並列処理手法[5] で用いられている並列性抽出技術を用いて，階層型マクロタスクグラフ（MTG）を生成し，その階層型 MTG に対して階層開始マクロタスク[6] を導入する．その後，全階層のマクロタスクを統一的に取り扱い，最早実行可能条件を満たした粗粒度タスク（マクロタスク）から順に，プロセッサに割り当てるダイナミックスケジューリングルーチンを生成する[5),6)].

ここで，従来の粗粒度タスク並列処理[5] (階層型実行制御方式と呼ぶ）では，各階層のマクロタスク（MT）は，プロセッサグループ（PG）に階層的に割り当てられて実行される．例えば，**図 1** のように，第 1 階層が MT1～MT8，MT5 内部の第 2 階層が MT51～MT53，MT51 内部の第 3 階層が MT511～MT512（ループでそれぞれ 2 回実行される）で構成されているプログラムを，2PG*2PE（各 PG は 2PE 構成で計 4PE）上で実行したイメージは，**図 2**(a) のようになる．

それに対して，階層統合型粗粒度タスク並列処理[6] では，図 2(b) に示すように，全階層のマクロタスクが統一的に取り扱われ，異なる階層のマクロタスク間並列性が最大限に利用される．

2.1 対象アーキテクチャ

階層統合型粗粒度タスク並列処理[6] は，現在までに，共有メモリ型マルチプロセッサ（SMP）やマルチコアプロセッサにおいて，Pthread や OpenMP を用いて実装されている．また，PC クラスタ上においても，MPI を用いて実装されている[7]．

2.2 粗粒度タスク並列処理のための並列性抽出

粗粒度タスク並列処理[5] は，階層的にループやサブルーチン等の粗粒度タスク間の並列性を抽出し，粗粒度タスク（マクロタスク）をプロセッサあるいはプロセッサグループに割り当て並列処理する方式である．

粗粒度タスク並列処理による実行では，まず，プログラム（全体を第 0 階層マクロタスクとする）を第 1 階層マクロタスク（MT）に分割する．マクロタスクは，擬似代入文ブロック（基本ブロック），繰り返しブロック（ループ），あるいは，サブルーチンブロックの 3 種類から構成される[5]．次に，第 1 階層マクロタスク内部に複数のサブマクロタスクを含んでいる場合には，それらのサブマクロタスクを第 2 階層マクロタスクとして定義する．同様に，第 L 階層マクロタスク内部において，第 $(L+1)$ 階層マクロタスクを定義する．ここで，繰り返しブロックが Doall ループ（あるいはリダクションループ）である場合には，粗粒度並列性を向上させるために，複数の部分 Doall ループに分割し，それらを別々のマクロタスクとして定義する．

マクロタスク生成後，各階層のマクロタスク間の制御フローとデータ依存を解析し，階層型マクロフローグラフ[5] を生成する．次に，制御依存とデータ依存を考慮したマクロタスク間並列性を最大限に引き出すために，各マクロタスクの最早実行可能条件[5] を解析する．最早実行可能条件は，制御依存とデータ依存を考慮したマクロタスク間の並列性を最大限に表しており，マクロタスクの実行制御に用いられる．

例えば，図 1 の MT5 の最早実行可能条件は，$1 \wedge 2 \wedge 3 \wedge 4$ と求められ，MT5 は MT1～MT4 の実行終了後に実行可能となることを表している．各階層のマクロタスクの最早実行可能条件は，図 1 のような階層型マクロタスクグラフ（MTG）[5] によって表すことが可能である．

2.3 階層開始マクロタスク

粗粒度タスク間並列性を抽出した後，階層統合型実行制御[6] を適用する場合，全階層のマクロタスクを統一的に取り扱うため，階層開始マクロタスクを導入する．第 L 階層マクロタスクを内部に持つ上位の第 $(L-1)$ 階層マクロタスクを，第 L 階層用の階層開始マクロタスクとして取り扱う．この階層開始マクロタ

スクは，内部の第 L 階層マクロタスクの実行を開始するために使用される．この階層開始マクロタスクの導入により，当該階層のマクロタスクの実行が可能になったことが保証され，全階層のマクロタスクを同時に取り扱うことが可能となる．

例えば，図 1 の MT5 の場合，内部に第 2 階層 MT（MT51，MT52，MT53）を含んでおり，MT5 は第 2 階層用の階層開始マクロタスクとして扱われる．同様に，MT51 の場合，内部に第 3 階層 MT（MT511，MT512）を含んでおり，MT51 は第 3 階層用の階層開始マクロタスクとして扱われる．

2.4 最早実行可能条件の階層統合型変換

階層開始マクロタスクの導入により，従来の階層ごとに求めた最早実行可能条件[5]を階層統合型実行制御用に変換する．具体的には，第 L 階層マクロタスクの最早実行可能条件が「true」（即ちその階層が実行可能になればすぐに実行可能）である場合，その条件を「第 L 階層用の階層開始マクロタスク MT_i の終了」に置き換える．

図 1 の場合，第 2 階層の MT51 と MT52 の最早実行可能条件は，従来の階層型実行制御用では「true」であるため，階層統合型実行制御用では「その階層の階層開始マクロタスク MT5 の終了」と置き換える．なお，本方式では，階層開始マクロタスクとしての MT_i の実行終了を表す終了ステートは，階層開始マクロタスク自身に発行させ，MT_i 内部の第 L 階層の実行終了を表す終了ステートは，第 L 階層の ExitMT に発行させている．

2.5 階層統合型実行制御によるマクロタスクスケジューリング

階層統合型実行制御によるマクロタスクスケジューリングでは，各マクロタスクはその最早実行可能条件[5]が満たされた後，レディマクロタスクキューに投入され，プライオリティの高い（Critical-Path（CP）長の大きい）マクロタスクから順にレディマクロタスクキューから取り出されてプロセッサに割り当てられる．

階層統合型実行制御では，全ての階層のマクロタスクが統一的に取り扱われ，それぞれのプロセッサ（グルーピングなし）に割り当てられ実行される．例えば，図 1 の階層型マクロタスクグラフの場合，図 2(b) に示すように，MT1〜MT8 の第 1 階層マクロタスク，MT5 内部の MT51〜MT53 の第 2 階層マクロタスク，MT51 内部の MT511〜MT512 の第 3 階層マクロタスクを統一的に取り扱い，それらを各プロセッサに割り当てて実行することが可能となる．

3. 並列度を考慮したタスク階層決定手法

階層統合型粗粒度タスク並列処理では，通常，全ての階層のマクロタスクをスケジューリングの対象とすることにより，プログラム中の並列性を最大限に利用することが可能となる．しかしながら，対象プログラムの上位階層のマクロタスク間並列性が，全プロセッサ（PE）を有効利用するのに十分な場合，下位階層のマクロタスク間並列性を利用する必要はない．そこで，本手法では，上位階層のマクロタスクグラフ（MTG）から順に利用できる並列性を解析し，十分な並列性を確保できた段階で，その階層のマクロタスクを最下位階層マクロタスク（その内部のマクロタスクに対してはダイナミックスケジューリングを適用しない）と定義し，そのマクロタスクを 1 回のダイナミックスケジューリングで PE に割り当てる．この結果，並列性を最大限に利用した上で，ダイナミックスケジューリングオーバヘッドを軽減することが可能になる．

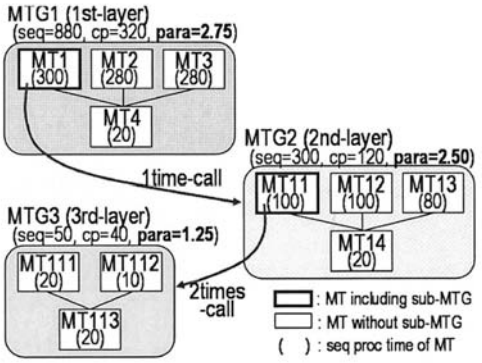

図 3 階層型マクロタスクグラフの並列度．

3.1 並列度による最下位階層 MTG 候補の検出

本手法では，各階層の $MTGi$ に対して，$MTGi$ のクリティカルパス長（入口ノードから出口ノードまでの最長パス長）CP_{MTGi} と，$MTGi$ の逐次処理時間 Seq_{MTGi} を算出し，$MTGi$ の並列度（$MTGi$ の平均的な粗粒度タスク間並列性）$Para_{MTGi}$ を，Seq_{MTGi}/CP_{MTGi} として求める．ここで，CP_{MTGi} は PE 数が無制限の場合の $MTGi$ の最小処理時間であり，Seq_{MTGi} は 1PE で実行した処理時間となる．各マクロタスクの処理時間は逐次処理時間を用いる．

次に，上位階層の $MTGi$ から順に，その並列度 $Para_{MTGi}$ を，$MTGi$ の処理に必要な PE 数として確保する．下位階層の $MTGj$ を並列実行するときには，その MTG を呼び出す上位階層マクロタスクの 1PE も使用することができるため，$MTGj$ のために別途必要な PE 数は，$Para_{MTGj}-1$ となる．このとき，PE に余りがなくなった，あるいは，$MTGj$ が他 MTG の呼出しをもたない場合には，その $MTGj$ は最下位階層 MTG 候補として定義され，$MTGj$ 内部のマクロタスクは最下位階層マクロタスク候補となる．

例えば，**図 3** の階層型マクロタスクグラフは，3 階層構成となっており，MTG1 の MT1 が MTG2 を 1 回呼び出し，MTG2 の MT11 が MTG3 を 2 回転分

呼び出している．仮に 3PE 実行とすると，MTG1 がその並列度 $para = 2.75$ を実現するために 2.75PE を確保し，次に MTG2 が並列度 $para = 2.50$ に相当する PE 数の確保を試みる．このとき，剰余 PE 数は $3.00 - 2.75 = 0.25$ であり，MTG2 を呼び出す MT1 用の 1PE を加えても，1.25PE しか確保できないことになる．それゆえ，MTG2 は最下位階層 MTG 候補となる．

3.2 並列処理時間を考慮した最下位階層 MTG の決定

前節で求めた最下位階層候補の $MTGi$ は，全 PE 数分の並列度を得るために，最上位階層からどの階層までの並列性を利用すればよいかを求めたものである．しかしながら，最下位階層 MTG 候補を最下位階層 MTG と定義するのが望ましいかどうかを，スケジューリング時間を含めた並列処理時間と逐次処理時間を比較することにより判定する．

ここで，最下位階層候補 $MTGi$ の逐次処理時間を Seq_{MTGi}，1 つの MT を割り当てるのに要するダイナミックスケジューリング時間を $ScheCost$，$MTGi$ 用として確保できる PE 数（上位階層の 1PE を含む）を $GivenPE_{MTGi}$，$MTGi$ を構成するマクロタスク数を $MTnum_{MTGi}$ と表現する．このとき，スケジューリングオーバヘッドを考慮した並列処理時間は，$MAX(CP_{MTGi}, Seq_{MTGi}/GivenPE_{MTGi}) + ScheCost * MTnum_{MTGi}/GivenPE_{MTGi}$ として算出される．

$MTGi$ の逐次処理時間が並列処理時間より短い場合には，$MTGi$ を 1PE で実行した方が処理時間が短縮可能であるため，$MTGi$ の並列性は利用されることなく，$MTGi$ を呼び出した上位マクロタスクが最下位階層マクロタスクとして定義され，この $MTGi$ は 1PE で実行されることになる．

但し，($MTGi$ の逐次処理時間)*(上位マクロタスクからの呼出し回数) が，全処理時間に対して大きな割合を占める場合には，PE 間の負荷アンバランスを招く可能性がある．そこで，本インプリメントでは，($MTGi$ の逐次処理時間)* (上位マクロタスクからの呼出し回数) が (全処理時間/(2*PE)) より大きい場合には，負荷バランスを配慮して，$MTGi$ を呼び出した上位マクロタスクを最下位階層マクロタスクと定義しない．即ち，$MTGi$ は，通常通り並列処理の対象となる．

前節で求めた図 3 の最下位階層候補 MTG2 の場合，$GivenPE_{MTG2}$=1.25，$ScheCost$=10[u.t] とすると，その並列処理時間は逐次処理時間より短くなるので，並列処理適用の最下位階層となる．一方，MTG2 の MT11 から呼び出される MTG3 には逐次処理が適用される．

4. タスク階層決定手法を伴う階層統合型粗粒度タスク並列処理の性能評価

4.1 評価用マクロタスクグラフによる性能評価

本節では，図 4 のコールグラフで表される 3 種類の 6 階層マクロタスクグラフ Type-I（4width-6layer），Type-II'（Single-4descendant-6layer），Type-III（Perfect-4descendant-6layer）と，3 種類の 4 階層マクロタスクグラフ Type-I'（8width-4layer），Type-II'（Single-8descendant-4layer），Type-III'（Perfect-8descendant-4layer）を用いて性能評価を行う．Type-I'～Type-III' の形状は，Type-I～Type-III の形状にそれぞれ対応しており，descendant 数（1 つの MTG から呼び出す下位階層 MTG 数）を 4 から 8 に変更し，layer 数（最大階層数）を 6 から 4 に変更した形となっている．

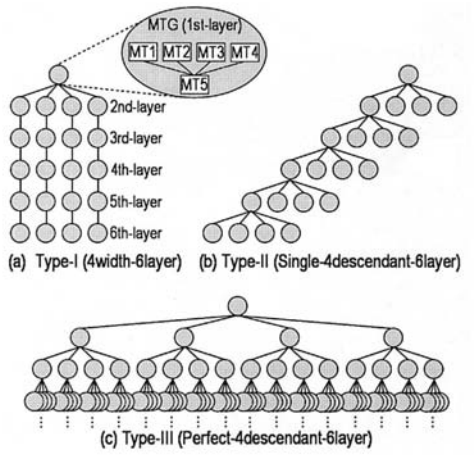

図 4 評価用マクロタスクグラフのコールグラフ．

図 4(a) の Type-I の場合，円ノードは MTG を表しており，21 個の MTG（それぞれは MT1～MT5）から構成されている．各マクロタスクの処理時間は，下位階層 MTG の呼び出しをもたない場合 100[u.t] とし，一方，下位階層 MTG の呼び出しをもつ場合には，下位階層 MTG の処理時間*繰返し回数（2 回転）を MT 処理時間としている．第 1 階層（最上位階層）の MTG では，その構成要素となる MT1～MT4 までが，直接 descendant の MTG（for 文のボディに相当）を 2 回転分呼び出している．第 2 階層～第 5 階層の MTG は，それぞれ MT1～MT5 から構成されており，MT1 のみが直接 descendant の MTG を 2 回転分呼出すものとする．

図 4(b) の Type-II の場合にも，円ノードは MTG を表しており，21 個の MTG から構成されている．各 MTG は MT1～MT5 より構成されており，各階層

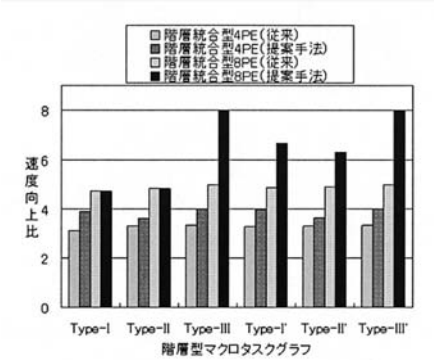

図 5 評価用マクロタスクグラフの階層統合型実行.

で 1 つの MTG（その MTG の MT1）が下位階層の MTG を 2 回転分呼び出している．図 4(c) の Type-III の場合，1365 個の MTG（それぞれ MT1〜MT5 より構成）からなっており，全ての MTG（その MTG の MT1）が，下位階層の MTG を 2 回転分呼び出している形となる．

本性能評価では，6 種類の上記マクロタスクグラフに対して，提案手法を伴う階層統合型粗粒度タスク並列処理を適用し，シミュレーションにより評価する．本シミュレーションでは，共有メモリ型マルチプロセッサシステムでの分散型ダイナミックスケジューリング実装をモデル化しており，各 PE で実行されるダイナミックスケジューラは，共通のレディマクロタスクキューをアクセスするため，排他的に動作するものとしている．ダイナミックスケジューリング時間は，対象となる階層型マクロタスクグラフの最下位マクロタスクの平均処理時間の 20%をスケジューリング処理時間としている．

図 5 はシミュレーションによる実行結果を表しており，提案するタスク階層決定手法を適用すると，4PE の場合は全種類のマクロタスクグラフにおいて，4 倍近い速度向上（1PE の逐次処理時間比）が得られており，低オーバヘッドで並列性が十分に引き出せていることが分かる．また，8PE 実行の場合，Type-I や Type-II のマクロタスクグラフでは，8PE を利用するだけの十分な並列性がないため，プロセッサが余剰となった状況にあり，提案するタスク階層決定手法を適用しても処理時間は適用前と同じになっている．一方，Type-III，Type-I'〜Type-III' のように並列性が大きいプログラムでは，タスク階層決定手法の効果が非常に大きく，プロセッサ数にスケーラブルな性能が達成されていることが分かる．

次に，ダイナミックスケジューリング処理時間を，最下位階層マクロタスクの平均処理時間の 0%から 30%まで変動させて評価を行う．**図 6** は，Type-II'(Single-8Width-4Layer) のマクロタスクグラフを 8PE 実行したものであり，1 回のダイナミックスケジューリング時間を，Type-II' プログラムの最下位階層マクロタスクの処理時間（100[u.t.]）の 0%，10%，20%，30%と可変にしている．スケジューリングオーバヘッドが増すと，従来手法では速度向上比が極端に低下しているが，提案手法では最下位階層マクロタスクの定義を適切に行うことにより，ダイナミックスケジューリングオーバヘッドが軽減されており十分な速度向上が達成されている．

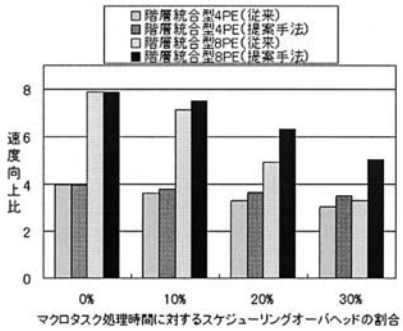

図 6 Type-II'(Single-8Width-4Layer) グラフの 8PE 実行.

以上の結果から，提案手法は，プロセッサ数が増加した場合やスケジューリング時間が増加した場合にも，並列性を有効利用しつつダイナミックスケジューリングオーバヘッドを軽減することが可能であり，顕著に処理時間が短縮されることが確かめられた．

表 1 ランダムマクロタスクグラフ生成パラメータ.

パラメータ	値
最大階層数	6
MTG 内 MT の下位階層 MTG 生成率	0–40[%]
下位階層 MTG の繰返し回数	1–2
各 MTG の幅	1–16
各 MTG の高さ	1–4
先行データ依存エッジ数	1–4
最下位階層マクロタスクの処理時間	1-100[u.t]

4.2 ランダムマクロタスクグラフによる性能評価

本節では，**表 1** のパラメータを用いて，6 階層ランダム MTG を 20 個生成し，シミュレーションにより性能評価を行った．各パラメータは可変の値をとっており，ランダムにその値を選んで MTG を生成している[8]．各 MTG から下位階層の MTG を呼び出す割合は 0〜40%としており，その呼出し回数は 1〜2 としている．呼出し回数が 1 の場合にはサブルーチン呼出しに相当しており，呼出し回数が 2 の場合は 2 回転の収束ループのイメージになる．各階層の MTG におけるマクロタスクの配置は，幅が 1〜16，高さが 1〜4 となっており形状的には並列性が大きく見えるが，マクロタスク処理時間のばらつきが大きいため，各階層

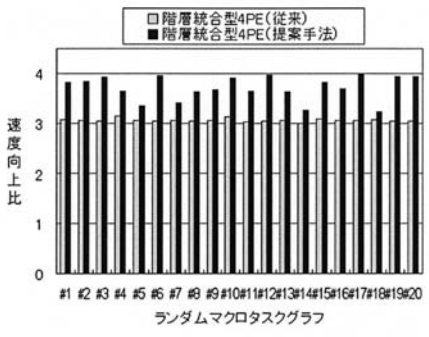

図 7 ランダムマクロタスクグラフの 4PE 上での階層統合型実行.

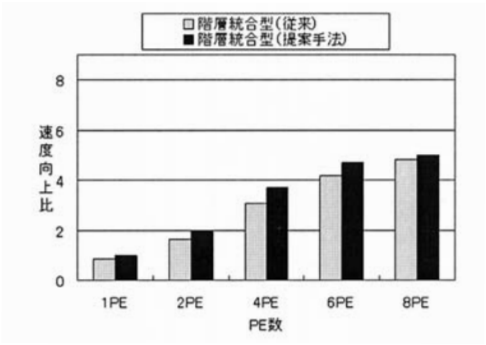

図 8 ランダムマクロタスクグラフの階層統合型実行 (20 グラフ平均値).

の並列度は低い.

本評価では，これらのランダム MTG を，階層統合型粗粒度タスク並列処理（提案手法の適用有無）によりシミュレーション実行する．なお，ダイナミックスケジューリングオーバヘッドは，最下位階層マクロタスクの平均処理時間の 20%としている.

図 7 は，6 階層のランダムマクロタスク（20 例）を 4PE 上でシミュレーション実行した結果であり，提案手法の適用により，ダイナミックスケジューリングオーバヘッドが軽減され，9 例で 20%以上の速度向上が得られている.

次に，プロセッサ数を 1〜8 まで変動させ，前述の 20 例のランダムマクロタスクグラフの平均値を取ったものを**図 8** に示す．本評価で生成した 20 例のランダムマクロタスクグラフは，各階層の並列度が小さいにもかかわらず，4PE で 17%，6PE で 10%，8PE で 3%の性能向上が得られており，並列性を利用しつつダイナミックスケジューリングオーバヘッドが大幅に軽減されていることが分かる.

5. おわりに

本稿では，階層統合型粗粒度タスク並列処理において，マクロタスク並列度を用いたタスク階層決定手法を提案した．階層統合型粗粒度タスク並列処理では，従来，全ての階層のマクロタスクがダイナミックスケジューリングの対象となっていたが，本手法では上位階層で十分な並列性が得られた場合に，下位階層のマクロタスク集合は 1 回のダイナミックスケジューリングでプロセッサに割り当てることにより，ダイナミックスケジューリングオーバヘッドを大幅に減らすことが可能である.

6 階層ランダムマクロタスクグラフを用いたシミュレーションによる性能評価の結果，提案するタスク階層決定手法の適用により，並列性を利用しつつ，ダイナミックスケジューリングオーバヘッドを軽減することが可能であり，階層統合型粗粒度タスク並列処理の有効性が確かめられた.

今後の課題としては，ベンチマークプログラムを用いて有効性を評価することがあげられる.

参考文献

1) M. Wolfe. High performance compilers for parallel computing. *Addison-Wesley Publishing Company*, 1996.
2) R. Eigenmann, J. Hoeflinger, and D. Padua. On the automatic parallelization of the Perfect benchmarks. *IEEE Trans. on Parallel and Distributed System*, Vol. 9, No. 1, Jan. 1998.
3) C. J. Brownhill, A. Nicolau, S. Novack, and C. D. Polychronopoulos. Achieving multi-level parallellization. *Proc. of ISHPC'97*, 1997.
4) X. Martorell, E. Ayguade, N. Navarro, et. al. Thread Fork/Join techniques for multi-level parallelism exploitation in NUMA multi-processors. *Proc. of International Conference on Supercomputing*, 1999.
5) 笠原博徳, 小幡元樹, 石坂一久. 共有メモリマルチプロセッサシステム上での粗粒度タスク並列処理. 情報処理学会論文誌, Vol. 42, No. 4, 2001.
6) 吉田明正. 粗粒度タスク並列処理のための階層統合型実行制御手法. 情報処理学会論文誌, Vol. 45, No. 12, 2004.
7) 吉田明正. PC クラスタ上での階層統合型粗粒度タスク並列処理. 情報処理学会研究報告, 2006-HPC-107-22, 2006.
8) A. Yoshida. An Overlapping Task Assignment Scheme for Hierarchical Coarse Grain Task Parallel Processing. Journal Concurrency and Computation: Practice and Experience, Wiley, Vol. 18, Issue 11, 2006.