B_010

PCクラスタ上での階層統合型粗粒度タスク並列処理のMPI実装手法

MPI Implementation Scheme for Layer-Unified Coarse Grain Task Parallel Processing on a PC Cluster

吉田 明正†
Akimasa Yoshida

1 はじめに

マルチプロセッサシステム上での並列処理手法としては，従来よりループ並列化技術 [1] が広く用いられているが，最近ではループやサブルーチンレベルの粗粒度タスク並列処理 [2] が有効と考えられている．また，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を利用しつつ，対象プログラム中の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粗粒度タスク並列処理 [3] が提案されている．

本稿では，この階層統合型粗粒度タスク並列処理を，普及しているPCクラスタにおいてMPIにより実装する方法を提案する．また，データ転送オーバヘッドを軽減するためのデータローカライゼーション技術も取り入れている．PCクラスタ上で行った性能評価の結果，提案手法の有効性が確認されている．

2 階層統合型実行制御を伴う粗粒度タスク並列処理

粗粒度タスク並列処理 [2] では，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．また，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．

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 [2] 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 [2] 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．階層統合型実行制御を伴う粗粒度タスク並列処理では，各階層ごとに求めた最早実行可能条件に，階層開始マクロタスクを導入し各種条件を変換する [3]．

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，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PE（グルーピングなし）に割り当てられて実行される．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．この場合，第1階層から第3階層までの並列性（例えばMT511，MT512，MT52，MT6の間の並列性）が同時に利用されており，実行時間が大幅に短縮されることが分かる．

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

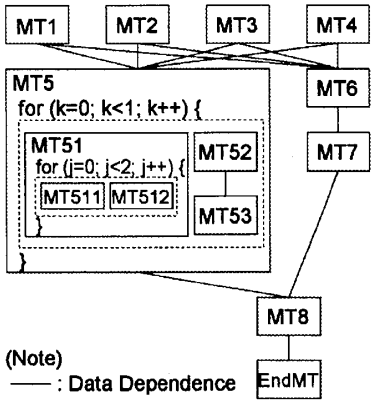

図1 階層型マクロタスクグラフ．

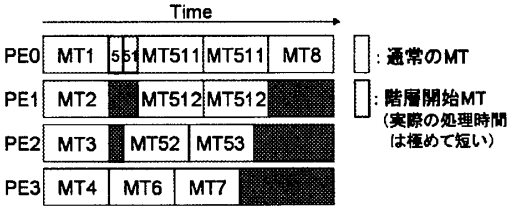

図2 階層統合型粗粒度タスク並列処理の実行イメージ．

3 階層統合型粗粒度タスク並列処理のMPI実装方法

本章では，PCクラスタ上で階層統合型粗粒度タスク並列処理をMPIにより実装する方法を述べる．本手法では集中型ダイナミックスケジューリング方式を採用しているため，PCクラスタ上の1PEをスケジューリング処理用とし，他PEをマクロタスク実行用とする．

スケジューリング処理用PEにおけるスケジューリングの基本手順を以下に示す．

(i) 階層統合型レディマクロタスクキューから絶対CP長 [3] の大きいMT_yを取り出し，アイドル状態のPE_jに対してマクロタスク番号yを送信（MPI_Send）する．

(ii) MT_yで参照される共有データを，スケジューリング処理用PEから，MT_yを実行するPE_jに送信（MPI_Send）する．

(iii) あるマクロタスクMT_zの実行がPE_kで終了した場合，そのマクロタスクを実行したPEの番号kを受信（MPI_Recv）する．

(iv) MT_zで定義された共有データをPE_kから受信（MPI_Recv）する．

(v) MT_zの終了ステートを設定する．

(vi) 新たに実行可能になるマクロタスクを階層統合型レディマクロタスクキューに投入する．

(vii) EndMTが終了してない間は(i)に戻る．EndMTが終了した際には，マクロタスク実行用PEに終了信号を送信（MPI_Send）する．

次に，マクロタスク実行用PE（PE_j）におけるマクロタスク実行の基本手順を以下に示す．

(i) スケジューリング処理用PEから送信されたマクロタスク番号yを受信（MPI_Recv）する．

(ii) MT_yで参照される共有データを，スケジューリング処理用PEから受信（MPI_Recv）する．

(iii) MT_yの実行コードを実行する．

(iv) MT_yの実行を終了したPEの番号jを，スケジューリング処理用PEに送信（MPI_Send）する．

(v) MT_yで定義された共有データを，スケジューリング用PEに送信（MPI_Send）する．

(vi) EndMTの終了信号を受信（MPI_Recv）していない間は(i)に戻る．

なお，関連研究としては，PCクラスタ上でマクロデータフロー処理を実現するために，データ到達条件[4]を用いる方法が提案されているが，階層統合型実行制御やデータローカライゼーションは対象とされていない．

4 データローカライゼーションによるPE間データ転送最小化

本手法では，ダイナミックスケジューリングによりマクロタスクを割り当てており，マクロタスク間共有データは，スケジューリング処理用PEにおいて管理する方式をとる．このため，マクロタスク間データ転送は，スケジューリング処理用PEを経由して行われることになる．

本手法では，このようなマクロタスク間データ転送を最小化するために，パーシャルスタティック割当てを用いたデータローカライゼーション手法[5]を導入し，同一階層のマクロタスク間データ転送のみならず，異なる階層のマクロタスク間データ転送に対しても，できるだけPE上のローカルメモリを介して行う．

例えば，図3のプログラムにおけるデータローカライゼーション適用部分は，(MT1, MT711)，(MT2, MT712)，(MT3, MT713)，(MT4, MT72, MT75)，(MT5, MT73, MT76)，(MT6, MT74, MT77)となる．これらの各適用部分のマクロタスクは，パーシャルスタティック割当てを伴うダイナミックスケジューリングにより同一PEに割り当てられ，スケジューリング処理用PEを介したデータ転送は軽減される．

5 MPI実装による階層統合型粗粒度タスク並列処理の性能評価

性能評価に用いたPCクラスタは，7台のPCを1000BASE-Tのイーサネットで接続した構成となっている．各PCの仕様は，PentiumIII：933MHz，メモリ：256MB，OS：VineLinux3.2（Kernel-2.4.31），MPI：MPICH2（Version-1.03）となっている．

階層統合型粗粒度タスク並列処理を，図3の3階層の評価用プログラムに適用し，PCクラスタ上で性能評価を行う．このプログラムの各マクロタスクは，基本的に，`for (i=2; i<10000; i++) { b[i]=b[i-1]+b[i-2]+a[i]; }`のような形状をしており，各マクロタスク間のデータ転送数は10000個となっている．

本性能評価では，PCクラスタの1PEをスケジューリング処理用とし，他PEをマクロタスク実行用とした並列プログラムをMPIを用いて作成した．PCクラスタ上での実行結果は図4の通りであり，6PE実行において，データローカライゼーション無（集中データ配置）の場合2.00倍，データローカライゼーション有の場合8.33倍の速度向上が得られている．

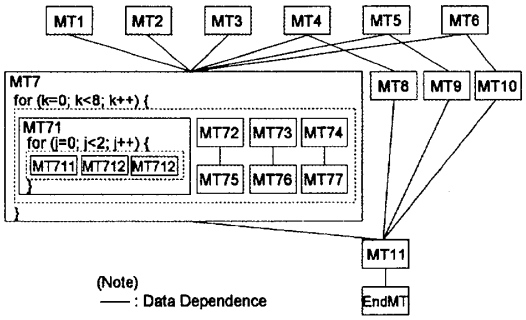

図3 評価用プログラムのマクロタスクグラフ．

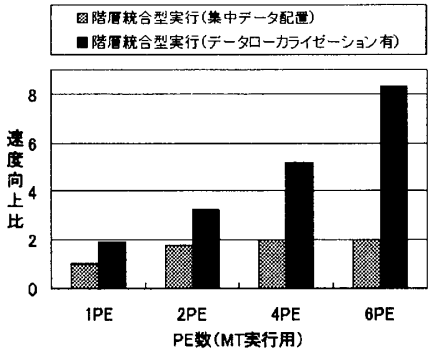

図4 PCクラスタ上での階層統合型実行．

6 おわりに

本稿では，PCクラスタ上で階層統合型粗粒度タスク並列処理を実現する方法を提案した．本手法により，ダイナミックスケジューリングを用いた階層統合型粗粒度タスク並列処理をMPI実装できるようになり，全階層にまたがった粗粒度タスク間並列性を利用することが可能となる．

また，データローカライゼーション技術を導入することにより，データ転送オーバヘッドの最小化も実現している．PCクラスタ上での性能評価の結果からも，提案するMPI実装手法の有効性が確認されている．今後の課題としては，ベンチマークプログラムを用いて性能評価を行うことがあげられる．

参考文献

[1] M. Wolfe. High performance compilers for parallel computing. *Addison-Wesley Publishing Company*, 1996.

[2] 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.

[3] 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.

[4] 本多弘樹, 上田哲平, 深川保, 弓場敏嗣. 分散メモリシステム上でのマクロデータフロー処理のためのデータ到達条件. 情報処理学会論文誌, Vol. 43, No. SIG 6(HPS 5), 2002.

[5] 吉田明正, 越塚健一, 岡本雅巳, 笠原博徳. 階層型粗粒度並列処理における同一階層内ループ間データローカライゼーション手法. 情報処理学会論文誌, Vol. 40, No. 5, 1999.