4R-03　メモリ容量を考慮したプレロード・ポストストアスケジューリングアルゴリズムの評価

田中 崇久[†]，舟山 洋央[†]，飛田 高雄[†‡]，笠原 博徳[†‡]

[†] 早稲田大学理工学部電気電子情報工学科 [‡] アドバンスト並列化コンパイラ共同体

1　はじめに

マルチプロセッサシステム上で効率のよい並列処理を行うためには，各プロセッサ間やプロセッサと共有メモリ間のデータ転送オーバーヘッドを最小化する必要がある [1]．そのため，データの分割・配置をユーザが指定できる言語として High Performance Fortran [2] が提案されていると共に，ループ内の作業配列をローカルメモリ (Local Memory,LM) あるいは分散共有メモリ (Distributed Shared Memory,DSM) に配置する Array Privatization 法 [3]，自動的にデータ分割・配置を行うデータローカライゼーション手法 [4] 等の研究が行われている．しかし，これらの手法を利用してもすべてのデータ転送を除去するのは不可能であり，除去できなかったデータ転送のオーバヘッドを隠蔽するためには，CPU でのタスク処理とデータ転送をオーバーラップさせるプレロード・ポストストア手法 [5] が有効である．本稿では，そのデータ転送と処理のオーバーラップを可能とする LM の容量あるいは DSM の容量を考慮したプレロード・ポストストアスケジューリングアルゴリズムを提案すると共に，その性能を評価する．

2　対象アーキテクチャモデル

本論文で対象とするアーキテクチャは図 1に示すようなマルチプロセッサシステムで，各プロセッサエレメント (PE) は，CPU, DSM, データ転送ユニット (Data Transfer Unit,DTU) を持ち，それぞれの PE がクロスバネットワークで接続され，どの PE からもアクセス可能な集中共有メモリ (CSM) を持つものとする．PE 間のデータ転送は，各 PE の持つ DTU によって CPU のタスク処理とオーバーラップして行うことができるものとし，さらに各 DTU は他の PE の DSM や CSM へ直接データのストア，ロードをすることができる．

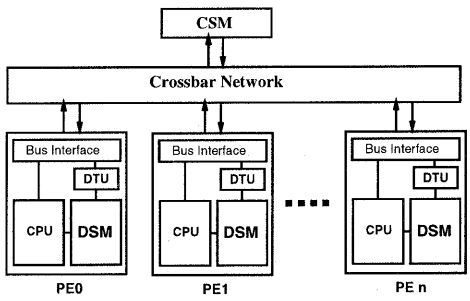

図 1: 対象アーキテクチャ

3　メモリ容量を考慮したプレロード・ポストストアスケジューリング

本章では，メモリ容量を考慮したスケジューリングアルゴリズムとデータプレロード・ポストストア手法について述べる．

3.1　対象タスクグラフ

スケジューリングアルゴリズムの公平な評価のために筆者等は，標準タスクグラフセット (Standard Task Gragh Set, STG) を提案している [6][7]．本論文における性能評価では，この STG にタスク間の転送データ名ならびにその転送コスト情報を付加したものを用いる．図 2にタスクグラフの例を示す．図 2中の各ノードはタスクを表し，ノード内の数字はタスク番号，ノードの左側の数字はタスクの実行時間を表している．また，ノードの右側の def，use はそのタスクで定義，使用される変数名である．ノード間のエッジはデータ依存関係を表しており，各エッジには先行タスクから後続タスクへのデータ転送コスト，転送される変数名情報が付加されている．

3.2　メモリ容量を考慮したスケジューリング

本稿では，ヒューリスティックマルチプロセッサスケジューリングアルゴリズム ETF/CP 法を基に，メモリ容量を考慮してプレロード・ポストストアを行うように拡張したスケジューリングアルゴリズム ETF/CP with PPM (ETF/CP with Preload Poststore considering Memory capacity) 法を用いる．ETF/CP with PPM 法の手順は以下の通りである．

1. 各タスクから出口ノードまでの最長パス (CP) 長を求める．

2. CP 長が長い順に，プライオリティリストを作成する．

3. 未割り当てタスクの集合から，全先行タスクの実行が終了しているもののリスト (レディタスクリスト) を作る．

4. レディタスクリストから 1 つを選択し，各 PE に割り当てた場合の最早実行開始時刻を，3.3 節で述べるメモリ容量を考慮したデータプレロード・ポストストアを使用して計算する．この時，データ転送時間の見積りは次のように行われる．

4-1. データ依存のある先行タスクからのデータがどこにあるか調べる．割り当て予定の PE の DSM にデータがある場合，データ転送は必要ない．割り当て予定の PE と異なる PE の DSM にある場合や，CSM にデータがある場合は，転送されるデータが割り当て予定の PE の DSM に収まるか計算し，DSM 容量が不足し収まらない場合は，次の手順で DSM を確保するのに必要な時間を計算する．

4-1-1. DSM 内の各データの生死解析を行い，これ以後使用されないデータを削除し，メモリを解放する．

4-1-2. 生死解析の結果，不要となるデータを削除しても DSM 容量が不足する場合は，DSM 内にデータが収まる分だけの空き領域ができるまで，最も最近使用されていないデータ (LRU) を CSM へストアし，メモリを解放する．

4-2. データのある場所からのデータ転送に必要な時間以上連続して DTU が空いている時刻を求め，データ転送を行う．

4-3. 割り当て予定タスクの定義データ量を調べ，DSM 容量が不足する場合は手順 4-1-1,4-1-2 のようにメモリを確保する．

5. レディタスクがなくなるまで手順 4 を繰り返し，実行開始時刻の最も早いタスクと PE の組合せを一つ選択し割り当てる (ETF)．同じ条件の候補が複数ある場合は，手順 2 で作成したプライオリティ(CP) リストでの優先順位の高いものを割り当てる．

6. 割り当て対象タスクがなくなるまで手順 3 から 5 を繰り返す．

---

* Performance Evaluation of Preload-Poststore Scheduling Algorithm Considering Memory Capacity

Takahisa Tanaka[†], Hirochika Funayama[†], Takao Tobita[†‡], Hironori Kasahara[†‡]

[†] Department of Electrical, Electronics and Computer Engineering, Waseda University

[‡] Advanced Parallelizing Compiler Project

3.3 メモリ容量を考慮したプレロード・ポストストア

図2のタスクグラフを，メモリ容量を考慮して，プレロード・ポストストア無しと有りでスケジューリングしたガントチャートを図3，図4に示す．ここで，変数A1，A2，A3，A4，A5は配列で各データサイズが10MB，V1，V2，V3，V4はスカラで各データサイズが5MB，各PEのDSM容量を40MBと仮定する．タスクを割り当てるとき，タスクで定義・使用されるデータ量を計算し，DSMに収まりきらないデータがある場合は，割り当てるタスクの実行に直接関係のないデータを生死解析の結果により削除するかCSMへ退避する必要がある．図3において，PE1でタスク3が終了した時点でPE1のDSM内のデータ(A1,A2,A3,V2)はすでに35MBあり，タスク4で定義されるデータ(V3)とタスク2から転送されるデータ(A4)がPE1のDSMに収まりきらない．そこで，3.2節で述べた手順に従い，生死解析の結果，DSM上のデータ(V2)を削除すると共に，データ(A3)をCSMへストアしてDSM上から削除し，PE0からのデータ転送(図3における2→4)を行う．

プレロード・ポストストア有りの場合は，図4のようにPE1でデータ(A3)をCSMへストアする際，タスク3の実行とデータ転送をオーバーラップさせて行う．同様にタスク4で使用されるデータ(A4)のPE0からの転送もタスク3の実行とオーバーラップさせて行う．この手法により，データ転送オーバーヘッドを隠蔽することができる．

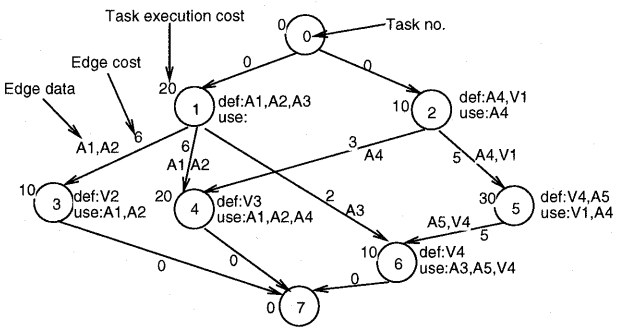

図 2: タスクグラフ

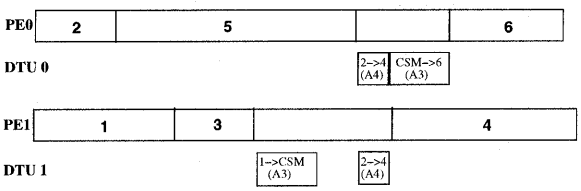

図 3: プレロード・ポストストアなしスケジュール

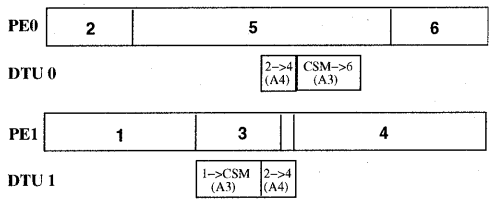

図 4: プレロード・ポストストアありスケジュール

4 性能評価

本章では，メモリ容量を考慮してプレロード・ポストストアを使うETF/CP with PPMアルゴリズムとメモリ容量は考慮するがプレロード・ポストストアを使用しないアルゴリズムETF/CP with M(ETF/CP with Memory capacity)の性能評価について述べる．本論文では3.1節で述べた条件で平均タスク数100のタスクグラフを200例用意した．なお，PE-PE間のアクセスコストと，PE-CSM間のアクセスコストの比は1:4，各タスクグラフの総タスク実行時間と総データ転送時間の比は1:0.3，DSM容量を各タスクグラフ中で最大のデータを必要とするタスクのデータ量の4倍とし，PE数1，2，4，8台で実行する場合のスケジュールをETF/CP with PPM及びETF/CP with Mそれぞれによって求めた．ETF/CP with MのPE1台でのスケジュール長を1としたときの各アルゴリズムの各PE数での速度向上率を図5に示す．図5のように，1PE，2PE，4PE，8PEで，ETF/CP with PPMではそれぞれ1.06，2.09，3.74，4.94倍，ETF/CP with Mではそれぞれ1.00，1.89，3.17，4.34倍の速度向上率が得られた．また，PE数1，2，4，8台のそれぞれで，ETF/CP with PPMはETF/CP with Mより1.06，1.11，1.18，1.14倍の速度向上率が得られている．以上の結果より，メモリ容量を考慮した場合でも，プレロード・ポストストア手法の有効性が確認された．

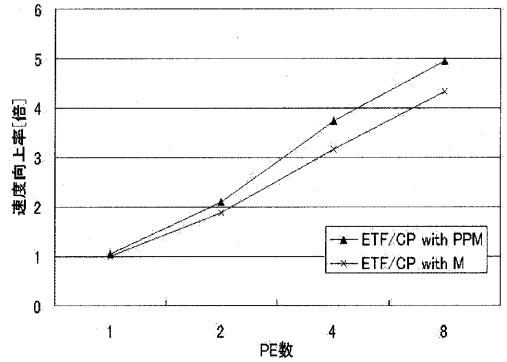

図 5: 各アルゴリズムの速度向上率

5 まとめ

本論文ではメモリ容量を考慮したプレロード・ポストストアスケジューリング手法を提案し，標準タスクグラフセット(STG)をもとにランダムに生成した転送データ名つきタスクグラフ200例で評価を行った．この結果，DSM容量を考慮した場合でも，プレロード・ポストストアを行った場合は，行わない場合より18 %の速度向上が得られ，プレロード・ポストストア手法の有効性が確認された．今後の課題として，スケジューリングアルゴリズムの改善を行うと共に性能評価の充実を行っていく予定である．

本研究の一部はNEDOアドバンスト並列化コンパイラプロジェクト及び学振未来開拓事業の一環として行われた．

参考文献

[1] 笠原 博徳，並列処理技術，コロナ社，June，1991
[2] HPF Forum, High Performance Fortran Language Specification
[3] P.Tu and D.Padua, Automatic Array Privatization, 6th Annual Workshop on Languages and Compiler for Parallel Computing, 1993
[4] H.Kasahara, A.Yoshida, A Data-Localization Compilation Scheme Using Partial Static Task Assignment for Fortran Coarse Grain Parallel Processing, J.Parallel and Distributed Computing, May, 1998
[5] 藤原 和典，白鳥 健介，鈴木 真，笠原 博徳，データプレロードおよびポストストアを考慮したマルチプロセッサスケジューリングアルゴリズム，信学論，Vol.J75-D-I, No.8, pp.495–503, August, 1992
[6] T.Tobita, H.Kasahara, A Standard Task Graph Set for Fair Evaluation of Multiprocessor Scheduling Algorithms, In Proc. ICS99 Workshop, pp.71-77, June, 1999
[7] 飛田高雄，笠原博徳，標準タスクグラフセットを用いたマルチプロセッサスケジューリングアルゴリズムの性能評価，並列処理シンポジウム JSPP2000，pp.131-138，June，2000