3H−6

処理とデータ転送のオーバーラップのための自動並列化手法

古郷 誠†, 田中 嵩久†, 藤本 謙作†, 岡本 雅巳‡,
笠原 博徳†

†早稲田大学理工学部電気電子情報工学科,
‡(株) 東芝

1 はじめに

ローカルメモリあるいは分散共有メモリを持ったマルチプロセッサシステムでより高速な並列処理を実現するためには、データのローカルメモリあるいは分散共有メモリへの配置が必要不可欠である。ローカルメモリへのデータ分割・配置に関する研究は、現在活発に行われており、ユーザー指定によるデータ分割・配置のためのHigh Performance Fortran(HPF) [2]、ループ内の作業配列をローカル化するArray Privatization法[3]、自動データ分割・配置法[4, 5] などが提案されている。しかしながら、これらの手法によりデータの分割・配置をおこなってもデータ配置が処理の前後で異なる場合、ローカルメモリ上のデータを他のプロセッサエレメントに転送する必要があり、そのオーバーヘッドは高速化を阻む原因となる。

このため最近のシステムではデータ転送をCPUと並列に行なうデータ転送ユニットを各プロセッサエレメントに持たせ、CPUの処理と並行してデータ転送を行うことによりデータ転送オーバーヘッドを隠蔽しようとしている。しかしながら、ユーザが処理とデータ転送のオーバーラップを考慮したプログラミングを行なうことは困難である。

本稿では、OSCARマルチグレインコンパイラ [1, 6] を用いて、処理とデータ転送のオーバーラップ[7]を考慮してマクロタスク間のデータ転送オーバーヘッドを削減し、トータルの実行時間を最小化する自動並列化FORTRANコンパイラ(プリプロセッサ)のプロトタイプを開発すると共に、富士通のVPP500上で予備評価を行なった結果について述べる。

2 対象アーキテクチャ

今回対象とするマルチプロセッサシステムのモデルは、図1に示すようにCPU、ローカルメモリ(LM)及び分散共有メモリ(DSM)、データ転送ユニット(DTU)を持つプロセッサエレメント(PE)をクロスバ等のネットワークで接続したマルチプロセッサシステムである。

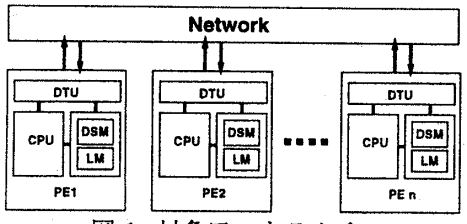

図 1: 対象アーキテクチャ

PE間のデータ転送は各PEの持つDTUによって、CPUのプログラム実行とオーバーラップして行なうことが可能である。また、各DTUはリモートリード・ライト双方向のデータ通信を同時に行うことが可能であるものとする。

*An Automaticic Parallelization Method for Overlaping of Macro Task Processing and Data Transfer

Makoto KOGOU†, Takahisa TANAKA†, Kensaku FUJIMOTO†,Masami OKAMOTO‡, Hironori KASAHARA†

† Waseda University. 3-4-1 Ohkubo Shinjuku-Ku, Tokyo 169, ‡Toshiba Corporation

3 マクロデータフロー処理

本節ではマクロデータフロー処理手法について述べる。OSCARマルチグレインコンパイラ [1, 6]ではソースとなるFortranプログラムを擬似代入文ブロック(BPA)、繰り返しブロック(RB)、サブルーチンブロック(SB)の3種類の粗粒度タスク(マクロタスク(MT))に分割する。ただし、Doallループは、ループインデックス範囲を分割することにより複数の部分Doallループに分割し、分割後の部分Doallループを新たにRBとして定義する。またサブルーチンはSBとして定義する。マクロタスク生成後、コンパイラは各階層においてBPA, RB, SB等のマクロタスク間のコントロールフローとデータ依存を解析し、それらを表したマクロフローグラフ(MFG)を生成する。さらに、MFGからMT間の並列性を最早実行可能条件解析により引き出し、その結果をマクロタスクグラフ(MTG)として表現する。

4 処理とデータ転送のオーバーラップスケジューリング

本節では、階層型マクロデータフロー処理、及び処理とデータ転送のオーバーラップを用いてデータ転送のオーバーヘッドを隠蔽するスケジューリング手法について述べる。ただし本稿ではコンパイラへの組み込みの第一ステップとして、データ依存エッジのみから成るスタティックスケジューリングが可能なマクロタスクグラフを持つプログラムを対象とする。

4.1 PE間のデータ転送について

本手法では、異なるPE上のMT間でデータの授受が必要な場合、データ依存先行MT(プロデューサMT)は自LMからデータ依存後続MT(コンシューマMT)が実行されるPE上のDSMへ一括してデータ転送を行う。コンシューマMTは、プロデューサMTにより自PE上のDSMへ書かれた共有データをLMにコピーした後、タスクの実行を開始する。ただしDSMが十分大きければこのLMへのコピーは不要である。また実システムでのデータ転送では、プロデューサタスクT_jを実行したPE_jが、T_jの最後で、T_jが生成したデータD_jをLMからDTUをもちいてPE_iのDSMに書き込んでおき、コンシューマタスクT_iは実行開始前にPE_iのDSMからLMへデータを読込むリモートストア型のデータ転送を用いる。

4.2 処理とデータ転送のオーバーラップ

あるスケジューリング時点tにおいて、タスクT_iをプロセッサPE_iに割り当てる場合、そのデータ依存先行タスクT_jの実行終了時刻からT_iの時刻tまでの間で、T_jを実行したプロセッサPE_jが他のタスクT_kを実行中に、T_jが生成したデータD_jをPE_j上のDTUを使用して、PE_jのLMからPE_iのDSMへストアすることをポストストア[7]と定義する。スケジューラは、データ転送オーバーヘッドの隠蔽のためにこのポストストア手法を用いて処理とデータ転送のオーバーラップを積極的に利用する。

図2に横軸に時間軸をとった時の、各PEの処理状況、DTUのsend unit、recv unitの動作状況、すなわち生成されるスケジュールの例を示す。

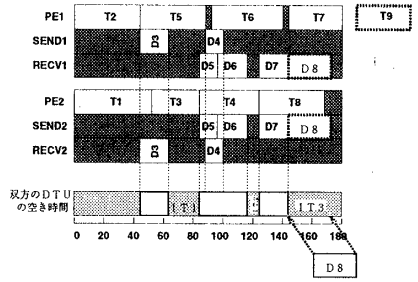

図 2: 処理とデータ転送のオーバーラップ

たとえば時刻146にPE_1に割り当てられるタスク7(T_7)を考える。この時、T_7のデータ依存先行タスクは、T_3とT_4であり、T_3からT_7へのデータ転送は時刻93から115のD_6、T_4からT_7へのデータ転送は時刻124から146のD_7によって行われている。ここでは、D_6のデータ転送はタスクの処理T_4 と、D_7はT_8とオーバーラップしていることが分かる。

4.3 スケジューリングアルゴリズム

処理すべきタスク集合を複数PEに割り当て、実行時間を最小化を目的としているマルチプロセッサスケジューリング問題は特殊な場合を除いてそのほとんどがNP困難な問題であることが知られている。そこで、本稿では評価に使用するマシンのデータ転送及び同期オーバーヘッドも考慮し、ヒューリスティックアルゴリズムETF/CP法を、ポストストアできるよう拡張したアルゴリズムETF/CP with PS法を用いる。ETF/CP with PS法は以下の手順で構成される。

1. 各タスクから出口ノードまでの最長パス長(CP)を求める。
2. 最長パス長が大きい順に、プライオリティリストを作成する。
3. 割り当ての終っていないタスク集合から全ての先行タスクが割り当てられているもの(レディタスク)を選び出す。
4. レディタスクを各PEに割り当てた場合のオーバーラップデータ転送を考慮した最早実行終了時刻を計算する。
   ここでデータ転送は、対象データ依存先行タスクの終了時刻以降で、現時間までの間で、DTU及びネットワークがデータ転送所要時間以上の間連続で空いている転送可能時間の中で最も早い時刻に挿入される。

   例えば図2においてPE_1にT_9を割り当て、そのデータ依存先行タスクT_1からのデータ転送D_8を挿入する場合を考える。スケジューラはT_1が終了する時刻50以降でPE1のDTUのrecv unit、PE2のDTUのsend unitの両方が空いているIT_1、IT_2、IT_3の順でデータ転送が挿入可能能かどうか調べていく。この場合データ転送時間は15[u.t]なので、IT_1、IT_2には挿入されず15[u.t]以上が連続で空いているIT_3に挿入される。同様の手法で全てのデータ依存先行タスクからのデータ転送を挿入し、最早実行終了時間を計算する。

   ただし考察中のマクロタスがRB内部で生成されている場合にはイタレーション間でデータ転送が必要となる。この場合、同一イタレーションでのデータ転送を優先し、イタレーション間でのデータ転送は同一イタレーションでのデータ転送を遅らさない範囲でデータポストストアを用いてオーバーラップ転送される。その際、転送されるデータは受信側PEのDSMのバッファに転送され、そのデータを使用する直前でLMに転送されるため誤ってデータが上書きされることはない。
5. 最早実行終了時刻の最も小さいタスクとPEの組合せを一個だけ選び出しその組み合わせでタスクをPEに割り当てる(ETF)。複数ある場合は、2で作成したプライオリティリストでの優先順位の高いものを割り当てる。
6. 3から5を割り当て対象タスクがなくなるまで繰り返す。

5 VPP500上での評価

今回提案した手法を用いた自動並列化プリプロセッサをOSCAR Fortran マルチグレインコンパイラ上に実装し、富士通のVPP500上で評価を行った。プリプロセッサはFortran 77プログラムをソースとして読込み、並列化されたVPP Fortranをオブジェクトとして出力する。

データ転送コードの挿入の際、スケジューリング結果を実行時に忠実に実現するためにはMT実行中に転送命令を発行する必要がある。しかしながらRBなどの中には実行途中で転送命令を発行させることが難しいものが存在するため、データ転送命令はMTの終了時あるいは開始時に発行することとした。評価に使用したプログラムは、航空流体解析に使用されているIAF (Implicit Approximate Factorization)解法で、本評価に当たってはプログラム中より、特徴的な部分を取り出したカーネルプログラムを使用した。プロセッサ1、2、3、4台において処理とデータ転送のオーバーラップを用いた場合と用いなかった場合において比較した結果を表1に示す。表1よりプロセッサ4台の場合には処理とデータ転送のオーバーラップにより約28%処理速度が向上していることが分かる。

表 1: 処理とデータ転送のオーバーラップの効果 (μs)

	PE数			
	1	2	3	4
プレロードなし	217914	134183	111522	94830
プレロードあり	217914	122012	92966	74383

6 まとめ

本稿では、マクロデータフロー処理における処理とデータ転送のオーバーラップ手法を提案し、これを可能とする自動並列化プリプロセッサをインプリメントし実機(VPP500) 上で評価を行った。今後の課題としては今回のデータ依存のみをもつマクロタスクグラフのスタティックスケジューリングだけでなく、一般のマクロタスクグラフにおけるダイナミックスケジューリング時の、処理とデータ転送のオーバーラップスケジューリングがあげられる。

本研究の一部は通産省次世代情報処理基礎技術開発事業マルチプロセッサコンピューティング領域研究により行なわれた。

参考文献

[1] 笠原: “並列処理技術”, コロナ社 (1991-06).
[2] Forum,H.P.F.:High Performance Fortran Language Specification DRAFT Ver.1.0,High Performance Fortran Forum(1993)
[3] Tu,P.and Padua,D.:Automatic ArrayPrivatization,6th Annual Workshop on Languages and Compilers for Parallel Comuting(1993)
[4] Anderson,J.and Lam,M.:Global Optimizations for Parallelism and Locality on Scalable Parallel Machines,Proc. of the SIGPLAN'93 Conference on Programming Language Design and Implementation,pp.112-125(1993)
[5] A.Yoshida, K.Koshizuka, H.Kasahara: “Data-Localization for Fortran Macro-Dataflow Computation Using Partial Static Task Assignment”, ICS(1996)
[6] H.Kasahara, H.Honda, S.Narita: “A Multi-Grain Parallelizing Compilation Scheme for OSCAR”, Proc. 4th Workshop on Languages and Compilers for Parallel Computing (Aug. 1991).
[7] 藤原、白鳥、鈴木、笠原: “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”, 電子情報通信学会論文誌(D-I),J75-D-I,8 pp.495-503 (1998).