2D−5　マルチグレイン並列処理における サブルーチンを含むデータローカライゼーション手法

宇治川 泰史†, 成清 暁博†, 小幡 元樹†, 吉田 明正‡, 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,
‡東邦大学理学部情報科学科, ††(株) 東芝

1　はじめに

ローカルメモリへのデータ分割・配置に関する研究としてHigh Performance Fortran(HPF) [1]、Array Privatization法 [2]、自動データ分割・配置法 [3] などが提案されているが、これらの方式では、ユーザーに高度な知識が要求される、適用対象が単一のループに限られる、スタティックな割当が行われる場合にしか適用できない等という問題があった。これに対し筆者等は、ダイナミックスケジューリングを用いた粗粒度並列処理において、複数の粗粒度タスク (マクロタスク) 間での共有データの授受をローカルメモリを介してを行なうデータローカライゼーション手法 [4] を提案し、粗粒度並列処理、中粒度並列処理、近細粒度並列処理を階層的に組み合わせたマルチグレイン並列処理 [5, 6] を行う OSCAR Fortran コンパイラに組み込んでいる。しかし従来のデータローカライゼーション手法はマクロタスクがループである場合のみを対象としていた。そこで本稿では、サブルーチン、ループ間でのデータローカライゼーション手法を提案する。

2　マルチグレイン並列処理

OSCAR マルチグレインコンパイラではプログラムを以下に示す 3 種類のマクロタスク (MT) に階層的に分割する。

- BPA(Block of Pseudo Assignment statements)
  基本ブロック及び基本ブロックを融合したブロック
- RB(Repetition Block)
  最外側ナチュラルループ
- SB(Subroutine Block)
  サブルーチン

各階層の MT は最早実行開始条件解析によりマクロタスクグラフ (MTG) で記述された後、階層的に定義されたプロセッサクラスタ (PC) 間で並列処理される。また各階層で PC に割り当てられた MT は、さらに PC 内のプロセッサエレメントにより階層的に並列処理される。例えば MT が RB ならば、ステートメントレベルの近細粒度並列処理、あるいは Do-all や Do-across などの中粒度並列処理、また RB が大規模であり内部でサブ MT が階層的に定義できる場合には階層的にマクロデータフロー処理を適用する。

3　データローカライゼーション手法

本章では、OSCAR マルチグレインコンパイラにおける SB を含めたデータローカライゼーション手法について提案する。データローカライゼーションとは、ローカルメモリを介してデータ授受を行ないデータ転送オーバーヘッドを最小化するために、適切にデータを分割し同一プロセッサに割り当てる手法である。本手法は、(1) 中間コードレベルでのサブルーチンのインライン展開、(2) ループ整合分割と再サブルーチン化、(3) データ転送解析、(4) 性能予測とコード生成、(5) パーシャルスタティックタスク割当を伴うダイナミックスケジューリングルーチンの生成により実現される。

3.1　ループ整合分割と再サブルーチン化

本節では、データ依存関係のある MT を配列データの使用範囲が等しくなるように MT を分割するループ整合分割 [4] について述べる。ループ整合分割の対象となるターゲットループグループに SB が存在するとインタープロシージャ解析 [7] を伴ったループイタレーション間データ依存解析が必要である。しかし現状のインタープロシージャ解析では複数ループ間で厳密なイタレーションレベルのデータ依存解析ができないため、本稿では一度 SB をインライン展開した中間コードを用いてデータ依存解析及びループ分割を行なった後、再度サブルーチンに戻すループ整合分割手法を用いる。

3.1.1　ターゲットループグループ (TLG) の生成

SB に対しインライン展開を適用し、生成された MTG においてデータ依存エッジで結ばれた RB 集合からループ整合分割の対象となる TLG を生成する。TLG に属することのできる RB の条件を以下に示す。

- Doall、Reduction、Sequential ループ等の RB である。
- 各 RB は配列変数に関するデータ依存エッジのみで結ばれている。

3.1.2　TLG 内データ依存解析

上述のように定義された TLG 内の MT は全て RB であるので、従来のループ間データ依存解析が可能である。そこで複数 RB 間においてある RB のイタレーションと他の RB のイタレーション間のループイタレーション間データ依存解析 [4] を行う。その解析結果より、ループ分割に必要なパラメータであるグループ変換インデックス範囲を各 TLG に対し求め、さらに標準ループイタレーション変換係数・標準ループイタレーションデータ依存範囲の下限値・上限値を TLG 内の各 RB ごとに求める。

例えば図 1(a) の 4 つの MT からなる TLG において、イタレーション間データ依存解析を行うと、図 1(b) のような結果が得られる。図 1(b) は、RB4 の t 番目のイタレーションが、RB3(SB3)・RB2・RB1 にデータ依存するイタレーション範囲を表している。

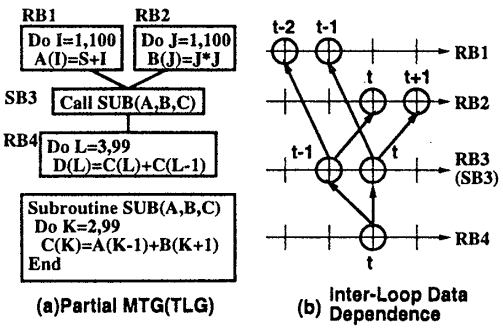

図 1: ターゲットループグループ (TLG)

---

* A Data-Localization Scheme for a Program with Subroutine in Multi-Grain Parallel Processing

Yasushi UJIGAWA†, Akihiro NARIKIYO†, Motoki OBATA†, Akimasa YOSHIDA‡, Masami OKAMOTO††, Hironori KASAHARA†

† Waseda University. 3-4-1 Ohkubo Shinjuku-Ku, Tokyo 169-8555, ‡Toho University, ††Toshiba Corporation

3.1.3 TLG 内のループ分割とデータローカライゼーショングループ (DLG) の生成

3.1.2節で求めたパラメータを用いてループ分割を行い、さらにもともと SB であったループに対しては、この分割後 SB 化を行う。これはループ分割の際、コードのコピーを伴う MT の生成を行うと、分割数に比例してコードサイズが増大する可能性があるためである。そこで分割対象となるループに対しサブルーチン化を行ない、生成された SB 内のループの初期値・終値・ステップを新たな変数で置き換え、それらを仮引数として登録する。次に計算された分割後のループの初期値・終値・ステップを実引数としてコール文に登録する。その結果、SB 自体のコード量は元のままで増加するのはコール文のみとなり、プログラムサイズの抑制を考慮した分割が可能となる。最後に、整合分割により生成された MT に対し、共有配列データにアクセスする MT 集合から DLG を生成する。

この分割方法にもとづき図 1(a) の TLG を 2 分割したとすると、分割後の TLG は図 2のようになる。図 2において、各網掛けの部分が MT 間で配列データをローカルメモリ経由で授受できる DLG を表している。

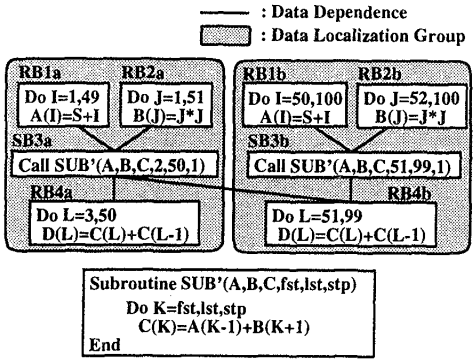

図 2: ループ整合分割の例

3.2 データ転送の解析

データローカライゼーションを適用する配列変数は、その配列変数が使用される前に必要な範囲のデータを集中共有メモリからローカルメモリへデータ転送する必要がある。その際、すでにローカルメモリに配置されているデータに対するデータ転送は無用であるため、入力依存も考慮して必要なデータ転送命令のみを生成するようなデータ転送解析を行う。また SB では、コール文に含まれる配列変数の使用範囲を解析するために、内側階層で行ったデータ転送解析の結果を利用して SB における配列変数の必要なデータ範囲を求め、他の MT とのデータ転送解析を行うことにする。

3.3 性能予測とコード生成

プログラム中の全ての配列変数に対してデータローカライゼーション手法を適用することは、データ転送オーバーヘッドが大きくなり実行時間の遅延が生じることがあるため、必ずしも有効ではない。そこで各配列変数ごとにデータローカライゼーションを適用した場合の処理コスト (lm_cost) と適用しない場合の処理コスト (csm_cost) を求める。SB に関しては、下階層から上階層へボトムアップなコスト伝搬を行ない、それぞれのコストを求める。そのようにして求めた各配列変数のコストにおいて、lm_cost < csm_cost となる配列変数のみデータローカライゼーションを適用する。データローカライゼーションを適用することが決定した配列変数に対しては、3.2節で解析したデータ転送コード及びローカルメモリへアクセスするコードを生成する。

3.4 パーシャルスタティックタスク割当を伴うダイナミックスケジューリングルーチンの生成

3.1.3で生成された DLG 内の複数 MT でデータローカライゼーションを実現するためには、これらの MT が実行時に同一の PC にスケジューリングされなければならない。そこで、コンパイル時に DLG に属する入口 MT の内の 1 つの MT がある PC に割り当てられたら、その DLG に属するその他の全ての MT はその PC に割り当てを行うようなルーチンを生成する。

4 OSCAR 上での性能評価

本章では、提案手法をアプリケーションプログラムに適用し、OSCAR シミュレータ上で評価した結果について述べる。性能評価プログラムとして、Specfp95 の 1 つである SU2COR.f のサブルーチン INT4V を用いた。INT4V は 3 つの RB と 2 つの SB から構成されており、これらの RB と SB におけるデータローカライゼーション手法の評価を行った。

この部分プログラムを 1 プロセッサでシーケンシャル実行すると処理時間は 490[ms] であった。次に 4 プロセッサを使用してマルチグレイン並列処理を行うと処理時間は 178[ms] となり、シーケンシャル処理と比べて 2.76 倍の速度向上が得られた。さらに提案されたデータローカライゼーションを適用すると、処理時間は 164[ms] となり、シーケンシャル処理と比べて 2.99 倍、マルチグレイン並列処理のみを適用した場合と比べて 8.4%の速度向上が得られた。

5 まとめ

本稿ではマルチグレイン並列処理において、サブルーチンとループを含む部分マクロタスクグラフをデータローカライゼーショングループとしてループ整合分割後に同一の PC に割り当て、それらの MT 間のデータ授受をローカルメモリを介して行うデータローカライゼーション手法を提案した。また、OSCAR シミュレータ上での部分プログラムにおける性能評価の結果より、SB においてもデータローカライゼーション手法が有効に適用できることが確認できた。

今後の課題としては、条件分岐を含んだ TLG の抽出とその依存解析、インライン展開を伴わない SB を含む MT 集合におけるデータローカライゼーション、及びループ分割に伴うコードサイズ増大の抑制などがあげられる。

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並列処理分野マルチプロセッサコンピューティング領域研究の一環として行われた。

参考文献

[1] Forum, H.P.F.: "High Performance Fortran Language Specification DRAFT Ver.1.0", High Performance Fortran Forum (1993).

[2] Tu,P. and Padua, D.: "Automatic Array Privatization", 6th Annual Wrorkshop on Lnaguages and Compilers for Parallel Computing (1993).

[3] Anderson, J. and Lam, M.: "Global Optimization for Parallelism and Locality on Scalable Parallel Machines", Proc. of the SIGPLAN '93 Conference on Programming Language Design and Implementation, pp.112-125 (1993)

[4] H. Kasahara, A. Yoshida: "Data-Localization Comlilation Scheme Using Partial Static Task Assignment for Fortran Coarse Grain Paralle Processing", Journal of Parallel Computing, Special Issue on Languages and Compilers for parallel Computers, (1998-5)

[5] 笠原: "並列処理技術", コロナ社 (1991-6).

[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] Zyiyuan Li, Pen-Chung Yew: "Interprocedural Analysis for Parallel Computing", CSRD, (1988).