階層型マクロデータフロー処理のための ダイナミック／スタティック併用スケジューリング手法*

1L－1

桐原 正樹†, 岡本 雅巳†, 赤鹿 秀樹‡, 笠原 博徳†

早稲田大学理工学部電気電子情報工学科†, 日本電信電話‡

1 はじめに

マルチプロセッサシステム用 Fortran 自動並列化コンパイラにおいて従来自動並列化が不可能であったループ並列性以外の並列性を抽出するために、筆者等は粗粒度の並列性を利用した並列処理 (マクロデータフロー処理) 手法 [2, 3, 4]、および基本ブロック内部の近細粒度並列処理手法 [1] を提案している。また、筆者等は粗粒度・中粒度・近細粒度並列処理を階層的に適用する並列処理手法であるマルチグレイン並列処理手法 [2]、さらにループやサブルーチン等の粗粒度タスク内で階層的にマクロデータフロー処理を行なう階層型マクロデータフロー処理も提案している。

この階層型マクロデータフロー処理では、粗粒度タスク間のスケジューリング方法として、実行時不確定性に対応するためにダイナミックスケジューリング手法を用いてきた。しかし、ダイナミックスケジューリング手法は実行時にスケジューリングを行なうためにスケジューリングオーバーヘッドが大きくなってしまうという問題が生じる。そこで、スケジューリングオーバーヘッドを軽減し、さらにマクロタスク間のデータ転送オーバーヘッドを最小化するために、本稿では、ダイナミックスケジューリング手法とスタティックスケジューリング手法との併用法を提案する。

2 階層型マクロデータフロー処理

OSCAR マルチグレインコンパイラ [2, 4, 5, 7] では Fortran プログラムを各階層毎に以下に示す三種類のマクロタスク (MT) [2] に分割する。

- BPA(Block of Pseudo Assignment statements)
  基本ブロックおよび複数の小基本ブロックを融合したブロック、または 1 つの基本ブロックを分割することによって得られるブロック
- RB(Repetition Block)
  Do ループや IF 文による後方分岐等によって生成される最外側ナチュラルループ
- SB(Subroutine Block)
  インライン展開が有効に適用できないと判断されたサブルーチン

このように分割生成された MT はそれぞれプロセッサクラスタ (PC) に割り当てられ、粗粒度並列処理される。さらに PC 内部のプロセッサエレメント (PE) によりマルチグレイン並列処理が実行される。MT が BPA の時、内部では各ステートメントをタスクとして定義し近細粒度並列処理を適用する。RB の時、内部では Doall 処理が可能なら Doall 処理による中粒度並列処理（ループ並列化）が適用される。ループ並列化が適用できないループに関してはループボディの近細粒度並列処理を行なう。また、RB が大規模であり内部でサブ MT が定義できるような場合や、MT が SB の時は内部のサブ MT に対してマクロデータフロー処理を適用する。この時、RB や SB 内部で生成されるサブ MT も BPA、RB、SB の三種として定義でき、階層的な MT の定義が可能である。

このように階層的に定義された MT は各階層で MT 間のコントロールフロー解析やデータフロー解析を行ない、解析によって求められる依存関係は各階層毎のマクロフローグラフ（MFG）で表現される。さらに MT 間の並列性を抽出するために、各階層のマクロフローグラフに対して最早実行可能条件解析 [3] を行ない、解析によって得られた結果を各階層毎のマクロタスクグラフ（MTG）で表現する。

3 階層型マクロデータフロー処理におけるスケジューリング手法

各階層の MT を PC に割り当てるスケジューリングとして用いられるのはコンパイル時にスケジューリングを行なうスタティックスケジューリング [1] と、実行時にスケジューリングを行なうダイナミックスケジューリング [4, 6] である。この 2 つのスケジューリングには以下のような特徴がある。

- ダイナミックスケジューリング
  - 実行時不確定性を含む MTG の実行状況の変化に適しており、条件分岐を含む MTG についてもスケジューリングを行なうことができる
  - スケジューリングを行なう際に生成される MT の大きさによってはオーバーヘッドが相対的に大きくなる可能性がある
- スタティックスケジューリング
  - 制御依存が確定している MT に対してコスト計算が正確な場合、良質のスケジューリングコードを得ることができる
  - データ転送や同期オーバーヘッドの最小化、ローカライゼーション、レジスタ利用の最適化が可能
  - 実行時不確定性が存在する場合、コンパイル時にそれを予測できず効果的なスケジューリングが期待できない場合がある

階層型マクロデータフローでは、階層毎に任意のスケジューリングを選択することができるが、スケジューリングオーバーヘッド等の問題を考慮すると、できるだけスタティックスケジューリングを行なう方が有効である。そこで、実行時不確定性に対応するためにスタティックスケジューリングを行なう際に、ダイナミックスケジューリングとの併用を考える。

3.1 SSMTG の分割生成手法の拡張

条件分岐を含まないマクロタスクグラフに対し適用されるスタティックスケジューリングでは、コンパイル時に MT を各 PC に割り当て、その順番通りに MT の実行コードを配置していく。PC 間においては MT 間に同期コードを挿入することにより各 PC 間で同期をとりながら実行する。本手法で用いる同期オーバーヘッドはダイナミックスケジューリングオー

---

*A Dynamic/Static Combined Scheduling Scheme for Hierarchical Macro-dataflow Computation

†Masaki KIRIHARA, Masami OKAMOTO, Hironori KASAHARA

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

‡Hideki AKASHIKA

‡NTT Corporation

バーヘッドに比べてかなり小さく抑えられる。しかし、スタティックスケジューリングはMT中に条件分岐のような実行時不確定性が存在する場合、制御依存を受けるMTはコンパイル時に効果的なスケジューリングができない。その場合、条件分岐を含まない部分でグループ化を行なう。以下このグループをSSMTG(Statically Scheduled Macro Task Group) [8]とする。

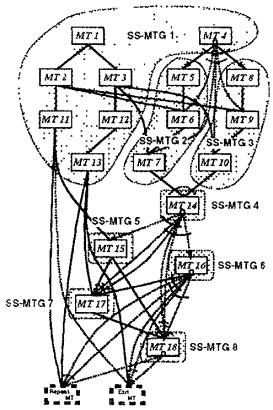

図 1: SSMTG

図1の網掛けで囲まれた部分がそれぞれ1つのSSMTGとして定義される。SSMTGを生成するためにマクロタスクグラフより得られるMT間のデータ依存や制御依存などの情報が用いられる。

SSMTGの分割生成は既にOSCARマルチグレインコンパイラ上に以下の手法で実装されている。

1. MTG中から制御依存が等しいMT集合を抽出する
2. 抽出されたMT集合中のMTの先行タスク、後続タスクに属さない集合がある場合にMT集合から分割して新しいMT集合を生成していく
3. 以上の分割により分かれたMT集合をSSMTGとする

しかし、グループ中に極端にコストの大きなMTが存在する場合には、SSMTG中でそのMTだけに負荷が集中してしまうのでSSMTGから外す必要がある。また、SSMTG内に並列性が全くないタスクが残る場合があり、そのような時はそこでSSMTGを分割してしまう方が良い。このようにこれまでのSSMTGの分割生成手法だけではSSMTGの分割が十分でない場合がある。

そこで、MT_iが自分の属するSSMTG以外に並列性がなくなるようにするため、筆者等はこれまでのSSMTGの分割生成手法を拡張した以下の手法を新たに提案する。

1. あるMT_iを集合S_gに加える
2. S_gの要素内にある各MTと並列実行が可能なMTを全て集合S_gに加える
3. 要素が追加できなくなるまで2.を繰り返す
4. 最終的に抽出された集合S_gを新たなSSMTGとして分割する
5. 以上を全てのMTがグループ化されるまで繰り返す

ただし、このSSMTGの分割生成手法はすでに制御依存によるグループ化が行なわれていることを前提とし、そのSSMTGに対して再分割を行なう形になる。また、SSMTG外へのデータ依存に関しては無視する。

3.2 SSMTGのスケジューリング手法

SSMTG内のMTに対してはスタティックスケジューリングを適用する。OSCARマルチグレインコンパイラではスタティックスケジューリングアルゴリズムとしてデータ転送を考慮したスタティックスケジューリング手法であるDT/CP/MISF法 [1]を用いる。

また、SSMTG自身は1つのMTとして扱う。これによってSSMTG間の実行を制御する際にSSMTGのためだけに別のダイナミックスケジューリングコードを生成するというような例外的な処理をする必要がなく、他のMTと同様にSSMTG間でマクロデータフローを適用することができる。

OSCARマルチグレインコンパイラではダイナミックスケジューリングアルゴリズムとしてスタティックスケジューリングアルゴリズムであるCP法をダイナミックスケジューリングのために拡張したDynamic-CP法 [6]を用いている。

スケジューリング併用時におけるSSMTGの実行の様子を図2に示す。SSMTGはスタティックスケジューリング適用部分で次のように実行される。

まず、SSMTGの分岐が決定した時点でPEが次に実行するSSMTGを同一階層内の全PEに通達する。各PCに割り当てられたSSMTG内の全MTの実行が終了したら同一階層の全PC間でバリア同期を行なう。そして、バリア同期が終了したら各PEは通達されたSSMTGの実行を行なっていく。

また、適用するスケジューリングが変わる時には同一階層の全PC間でバリア同期を行う。

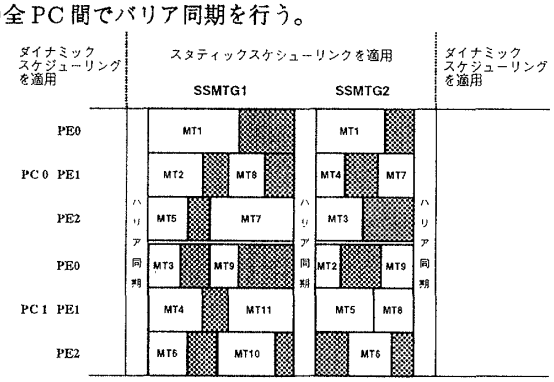

図 2: SSMTGの実行の様子

4 むすび

本稿ではOSCARマルチグレインコンパイラ上で、階層型マクロデータフロー処理においてダイナミックスケジューリングとスタティックスケジューリングを併用して適用する手法について述べた。提案手法によって階層型マクロデータフローの実行の際にスケジューリングオーバーヘッドを軽減することが可能である。

今後、各階層において最適なプロセッサのクラスタ構成を自動算出する手法を開発し、SSMTGにこの手法を適用する予定である。

本研究は一部、文部省科学研究費補助金（一般研究（C）No.07680372）の補助で行なわれた。

参考文献

[1] 笠原, 並列処理技術, コロナ社 (1991-06).
[2] H.Kasahara, H.Honda, S.Narita, "A Multi-Grain Compilation Scheme for OSCAR," Proc. 4th Workshop on Languages and Compilers for Parallel Computing (Aug. 1991).
[3] 本多, 岩田, 笠原, Fortranプログラム粗粒度タスク間の並列性検出手法, 信学論, J73-D-I(12) (1990-12).
[4] H.Kasahara, H.Honda, M.Iwata, M.Hirota, "A Compilation Scheme for Macro-dataflow Computation on Hierarchical Multiprocessor Systems," Inter. Conf. on Parallel Processing (Aug. 1990)
[5] 岡本, 合田, 宮沢, 本多, 笠原, OSCARマルチグレインコンパイラにおける階層型マクロデータフロー処理, 情処論, Vol.35(4) (1994-4).
[6] 本多, 合田, 岡本, 笠原, Fortranプログラム粗粒度タスクのOSCARにおける並列実行方式, 信学論, J75-D-I(8), (1992-8).
[7] 笠原, 本多, 橋本, OSCARのアーキテクチャ, 信学論D, Vol.38,No1, (1989-1).
[8] 赤鹿, 岡本, 宮沢, 安田, 笠原階層化マクロデータフロー処理のためのマクロタスクスケジューリング手法, , 情報処理学会第52回全国大会, 1L-1, (1996-3).