4J-03

解析時インライニングを用いたマルチグレイン自動並列化手法 *

吉井 謙一郎 †, 松井 巌徹 ‡, 小幡 元樹 †, 熊澤 慎也 †, 笠原 博徳 †
† 早稲田大学理工学部電気電子情報工学科, ‡ 松下電器産業 (株)

1 はじめに

マルチプロセッサシステム上の Fortran 自動並列化コンパイラにおいては，従来よりループ並列化手法 [2, 3] が広く用いられている．しかし，ループ並列化手法では単一ループのイタレーション間の並列性しか利用できないという問題がある．この問題点を解決するために，基本ブロック，ループ，およびサブルーチンコール間の粗粒度並列処理，ループイタレーション間の中粒度並列処理，ステートメント間の近細粒度並列処理を階層的に組み合わせてプログラム全域の並列性を利用するマルチグレイン並列処理が提案されている [4]．この粗粒度並列処理 [5] においては，粗粒度並列性を抽出するためにインタープロシージャ解析 (IPA) が必須となる．IPA 手法には，配列添字の詳細とループ範囲をコールサイトへ伝搬するアトムイメージ法 [6] や，ループ境界とループストライドからサブルーチン内での参照配列部分を決定しコールサイトへ伝搬する *Bounded Regular Section* 解析法 [7]，そしてコールサイトコンテキストの相違による解析精度の低下を防ぐためにサブルーチンを選択的にクローニングする手法 [8] などが提案されている．しかし，アトムイメージ法では実引数及び仮引数整合配列の次元が異なる場合サブルーチン側での情報をコールサイト側での情報に変換できず解析できない．また，*Bounded Regular Section* 解析法ではコールサイトのコンテキストが大きく異なるものが存在する場合に解析精度が低下し，選択的クローニング手法ではコードサイズが増大するなどの問題がある．これらを解決するためにアトムイメージ法と配列の一次元化を組み合わせた手法 [9] も提案されているが，粗粒度並列性を考慮していない．そこで本稿では，既存の解析手法で粗粒度並列性の解析が行えないプロシージャコールに対して，コンパイラにより解析時にのみインライン展開し，並列性解析を行った後元のサブルーチンあるいは他の形のサブルーチンに戻し，解析精度を向上させつつコードサイズを小さく抑える手法を提案する．

2 マルチグレイン並列処理

OSCAR マルチグレイン自動並列化コンパイラ [4] では，プログラムを次の 3 種類のマクロタスク (MT) に分割する．

- BPA　基本ブロック及びこれを複数融合したブロック
- RB　最外側ナチュラルループ
- SB　サブルーチン

粗粒度並列処理とはこの MT 間の並列性を用いた並列処理手法である．また中粒度並列処理はループイタレーション間の並列性を用いた並列処理であり，近細粒度並列処理とは BPA 内の各ステートメントをタスクと定義してその間の並列性を用いた並列処理である．この 3 つの粒度の並列処理を組み合わせたものがマルチグレイン並列処理である．各階層の MT は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 (PE) により階層的に並列処理される．例えば MT が RB ならば，DOALL などの中粒度並列処理，ループボディの近細粒度並列処理，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粗粒度並列処理を適用する．本手法は，効果的な粗粒度並列処理を行うために，サブルーチン内外の並列性を抽出することを目的とした IPA 手法である．

3 解析時インライニングによる IPA

本稿では，従来の手法では解析できないサブルーチンコールに対し，インライン展開した中間コードを用いてサブルーチン内と外の MT に対する粗粒度並列性の解析を行い，プロシージャ内外にわたる並列性が存在すればそのままインライン展開したコードを生成し，部分的に存在すればその部分はインライン展開したままとし，存在しない部分を元のあるいは他の形のサブルーチンに変換する（フレキシブルクローニング）自動並列化手法を提案する．この解析時インライニング及びフレキシブルクローニングにより，生成コードの過度な増大を抑えつつ粗粒度及びプロシージャコールを含むループの並列性を最大限に引き出すことができる．以下では，解析時インライニングによる自動並列化手法について述べる．

3.1 適用する SB の選定とインライニング

解析時インライニングは，従来の代表的な IPA 手法であるアトムイメージ法で粗粒度並列性の解析ができなかった SB に対して行う．手法を適用する SB の選定は以下の順に行う．

1) 従来手法で解析できない SB 中推定コストが最大のもの
2) 1) の SB とデータ依存のある SB

2) では従来手法で解析できた SB も解析時インライニングの対象として選ぶこともある．これは，1) に適合した SB から呼び出されるサブルーチン内部の MT と 2) に適合した SB から呼ばれるサブルーチン内部の MT との粗粒度並列性を抽出する目的のためである．この条件に適合し，解析時インライニングを適用することが決定した全 SB は，インライン展開ルーチンによって中間語レベルで展開される．

3.2 並列性解析

解析時インライニングが適用された SB は，中間コードを用いたグローバルデータ依存解析及び並列性解析の対象となる．

解析時インライニングにより，引数などの隠れた情報が明らかになることや，定数，シンボリック伝搬などにより，インライン展開を用いない従来の IPA に比べより詳細かつグローバルなデータ依存解析が可能となり，より粗粒度並列性の抽出が可能となる．また，粗粒度並列性の解析には OSCAR マルチグレイン自動並列化コンパイラにおいて従来より採用されている最早実行可能条件解析法 [1, 5] が使用される．

3.3 フレキシブルクローニング

並列性の解析が終了した段階で，このインライニングにより並列性が増加したかどうかを判断する．この検査においては，並列性解析時に行った最早実行可能条件解析の結果から作成されたマクロタスクグラフ (MTG) に対して，その MTG の推定並列度 Para [10] を計算し，展開前の Para と比較することにより並列性が増加したかどうかを判断することができる．次に，展開によりできた各 MT について並列性の向上に寄与したかどうかを検査する．これは，推定並列度 Para が MTG 全体における並列度を示すもので，Para が増加していても展開した SB の中にはそれに寄与していないものが存在

---

*A Multi-Grain Automatic Parallelizing Compilation Scheme with Analysis-Time Procedure Inlining

Ken'ichirou YOSHII †, Motoki OBATA †,
Shinya KUMAZAWA †, Hironori KASAHARA †
†Dept. of Electrical, Electronics and Computer Engineering, School of Science and Engineering, Waseda Univ.
Gantetsu MATSUI ‡
‡Matsushita Electric Industrial Co., Ltd.

する可能性があるからである．この検査は，作成されたMTG上の各エッジを出口ノードからたどりながらMTのグルーピングを行い，グループ内の推定シーケンシャルコストを入口MTのクリティカルパス長で割ることにより部分的な並列性を計算し，MTG全体のマクロな並列度と比較するものである．この検査によって並列性の増加に寄与していないと判断されたMT群はフレキシブルクローニングされる．このフレキシブルクローニングの際には，そのコールサイトの情報を伝搬することにより無用コードの除去などを施し最適化する．

例えば，図1(a)のコードにおいて，解析時インライニングの際にDOループをアンローリングして並列性の解析を行うと図2(b)のようになるが，元のDOループ内でYPENTA及びYPENT2の呼び出し部分以外ではループインデックスが使われていないためコードが全く同じであり，その上MT1からMT4，MT6からMT9，そしてMT11からMT14までは他のサブルーチン内のMT群と並列性が存在する．そこでフレキシブルクローニングを適用し同じサブルーチンを呼ぶように変換すると図2(c)の様なMTGとなり，グローバルな粗粒度並列性を維持したままコードサイズを抑えることができる．

4 解析の適用例

この解析の適用例をPerfect Benchmarksに含まれるベンチマークプログラムARC2Dを用いて説明する．実際の解析においては，本解析を適用する前に解析をする部分を選定する必要がある．これにはコンパイラによる推定タスクコストが用いられる．すなわち，効果的な並列処理を実現するために推定コストが最も大きい部分を選び，その部分に対して本解析を適用することになる．ARC2Dの場合，サブルーチンINTEGRが選ばれる．サブルーチンINTEGRは実際に実行時間の約95%を占めており，妥当な決定と見ることができる．

4.1 解析適用部分の決定

3.1節で示した基準によりサブルーチンINTEGR内でこの解析を適用するSBを選定する．すると，サブルーチンSTEPFYを呼ぶSBが3.1節の1)に適合する．これは以下による．

- STEPFYからYPENTAを呼ぶ部分が図1(a)の様になっており，STEPFY側実引数配列Sが3次元であるのに対し対応するYPENTA側仮引数整合配列Fが2次元であり，アトムイメージ法ではこの配列に関する情報をYPENTA側からSTEPFY側へ伝搬できず解析できない
- 推定コストがINTEGR全体の約23%を占め最も大きい

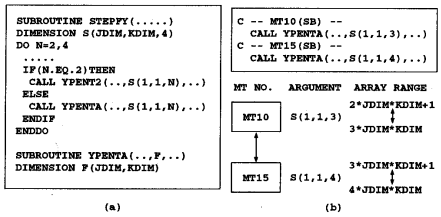

図1: STEPFY, YPENTAのコード片と依存解析結果

選定を進めると，INTEGR内の残りのSBは全て3.1節の2)に適合し解析時インライニングの対象となる．また，1回目の展開後もYPENTAを呼ぶSBが残る．これらも従来手法では解析できないため再度解析時インライン展開を行う．

4.2 並列性解析とフレキシブルクローニング

インライン展開後，シンボリック値（定数を含む）の伝搬やループアンローリングなど並列性増加のための各種最適化を行った後データ依存解析及び並列性の解析に移る．

INTEGRの場合，図1(a)に示した従来のIPA手法では解析できないSTEPFYからYPENTAの呼び出しが重要であるが，階層的に解析時インライニングを適用することにより図1(b)の様に各SB間に依存がないことが解析できる．この解析結果により，従来手法では図2(a)となっていたMTGが図2(b)の様により大きい並列性を持ったMTGに変換できる．さらに，3.3節で述べたフレキシブルクローニングを適用することにより，グローバルな並列性が存在するMT群はクローニングされ図2(c)の様になる．

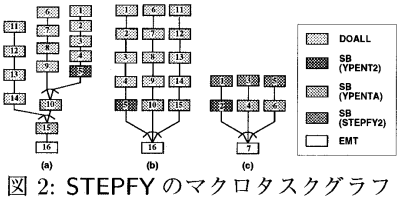

図2: STEPFYのマクロタスクグラフ

5 性能評価

プログラムARC2Dを用い，本手法とアトムイメージ法によるIPAのみの従来手法とをOSCARシミュレータ上で性能比較したものを表1に示す．なお，ARC2Dのメインループの回転数はINTEGRのMTG形状に無関係なため1回転とした．

表1: ARC2Dによる本手法の評価結果

PC数–PE数		1–1	3–1	3–2
実行時間（秒）	従来手法	179.6	160.7	109.5
	本手法	171.1	62.4	35.7
従来手法に対する速度向上（倍）		1.05	2.58	3.07

表1より，アトムイメージ法のみによる従来手法に比べ1PEにおいても最適化による速度向上が見られた．さらに3PC2PEにおいて3.07倍の速度向上が得られており，本手法によってさらに並列性が抽出できたことがわかる．

6 まとめ

本稿では，マルチグレイン自動並列化コンパイラにおける粗粒度タスク間の並列性抽出のためのインタープロシージャ解析手法として，従来解析できなかったサブルーチンに対して中間コードレベルでインライン展開した上で並列性の解析を行い，プロシージャ内外にわたる並列性が存在すれば展開したコードをそのまま残し，グローバルな並列性が存在しない場合は並列性の存在しない部分をクローニングする，解析時インライニング及びフレキシブルクローニング手法について述べ，ARC2Dにより本手法の有効性を確認した．

本研究の一部は，通産省次世代情報処理基盤技術開発事業並列分散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なわれた．

参考文献

[1] 笠原：“並列処理技術”，コロナ社 (June 1991).
[2] Banerjee, U. : “Loop Transformations for Restructuring Compliers”, *Kluwer Academic Pub.*(1993).
[3] Wolfe, M : “High Performance Compilers for Parallel Computing”, *Addison-Wesley Publishing Company* (1996)
[4] H. Kasahara, el al. : “A Multi-Grain Parallelizing Compilation Scheme for OSCAR”, *Proc. of the 4th Workshop on LCPC* (Aug. 1991).
[5] 本多，他：“Fortranプログラム粗粒度タスク間の並列性検出手法”，信学論，Vol.J73-D-I, No.12, pp.951–960 (Dec. 1991).
[6] Zyiyuan Li, et al. : “Interprocedural Analysis for Parallel Computing”, *CSRD Technical Report* (1988).
[7] Paul, H. et al. : “An implementation of Interprocedural Bounded Regular Section Analysis”, *IEEE Trans. on Parallel and Distributed Systems*, 2(3):350–360 (July 1991).
[8] M.W. Hall, el al. : “Detecting Coarse-Grain Parallelism Using an Interprocedural Parallelizing Compiler”, *Proc. of Supercomputing '95* (Dec. 1995).
[9] M. Hind, et al. : “An Empirical Study of Precise Interprocedural Array Analysis”, *Scientific Programming* 3(3):255–271 (1994).
[10] 山本，他：“OSCARマルチグレイン並列化コンパイラにおける階層的並列処理手法”，情報処理学会第58回全国大会，2D-04 (Mar. 1999).