計算機アーキテクチャ 151－2
（２００３．１．３１）

マルチグレイン並列性向上のためのインライン展開手法

白子　準[†]　長澤耕平[†]　石坂一久[†,††]
小幡元樹[†,††]　笠原博徳[†,††]

マルチプロセッサシステムの実効性能の向上, 使い易さの向上のため, 基本ブロック, ループ、サブルーチン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・基本ブロック内ステートメント間の近細粒度並列処理を階層的に組合せ, 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重要となっている. マルチグレイン並列処理において階層的に並列性を抽出し, 効率よい並列実行を実現するためには, 各々の階層 (ネストレベル) の並列性に応じたプロセッサ数を割当てる必要がある. また, 階層の異なるサブルーチンをインライン展開により同一階層になるようリストラクチャリングすることで, マルチグレイン並列性を高めることが可能となる. 本稿ではプログラム中の各階層の並列度を定義し, これを用いマルチグレイン並列性を高めるために有効なインライン展開すべきサブルーチンを選択する手法を提案する. 本手法の有効性を SMP サーバ IBM RS6000 PowerPC 604e High Node 8 プロセッサシステム上にて, SPEC95FP ベンチマークの 103.su2cor, 107.mgrid を用いて評価を行った結果について述べる.

Inline Expansion for Improvement of Multi Grain Parallelism

Jun shirako,[†] Kouhei Nagasawa,[†] Kazuhisa Ishizaka,[†,††]
Motoki Obata[†,††] and Kasahara Hironori [†,††]

To improve performance and usablity of multiprocessor systems, a multi-grain parallel processing which exploits coarse grain parallelism among loops, subroutines and basic blocks, conventional medium grain parallelism among loop-iterations in a parallel loop and near fine grain parallelism among statements inside a basic block, is important. In order to extract the parallelism over different nested layer in multi-grain parallel processing, it is necessary to assign the appropriate number of processors, according to the parallelism of the each layer. Also, the inline expansion scheme that inlines subroutines in different layers having high parallelism is effective to get more parallelism. This paper defines a degree of parallelisms and proposes a scheme for selective inline expantion for improvement of multi grain parallelism. Effectiveness of the proposed scheme is evaluated on IBM RS6000 SMP server with 8 processors using 103.su2cor, 107.mgrid of SPEC95FP.

1. はじめに

マルチプロセッサシステムの性能向上のためには, これまで用いられてきたループ並列化技術に加え, サブルーチン・ループ・基本ブロックの間の粗粒度タスク並列処理, またチップマルチプロセッサ等で有効な基本ブロック内ステートメント間近細粒度並列処理の階層的な利用が重要となっており, 筆者らが開発中の OSCAR マルチグレイン並列化コンパイラ[1),2)] ではこれらの並列性を利用することが可能である. 本コンパイラはプログラムを粗粒度タスクに分割し, 最早実行可能条件解析[3)] を用いて各粗粒度タスク間の並列性を抽出してマクロタスクグラフ (MTG) を生成する. 生成された粗粒度タスクがサブルーチンブロック, ループブロックの場合は, 階層的にその内部を粗粒度タスクに分割し, 階層的な MTG を生成することによりプログラムの各ネストレベルに対して MTG を定義し, プログラム全域の並列性を抽出する.

マルチグレイン並列処理においては, MTG の並列性や形状に応じ, 割当てるプロセッサ数や並列処理方法を適切に決めなければならない. また, 実際のプログラムでは, 内部に高い並列性を含むサブルーチンが異なる階層に分散していることが多く, 階層間の総合的な並列性の利用が困難な場合もある. サブルーチン間の並列性の解析, 抽出については従来よりループ並列性を抽出するための, 様々なインタープロシジャ解析 (IPA) 手法が提案されている. スタンフォード大学の SUIF コンパイラでは, サブルーチンを選択的にクローニングする手法[4)] により, サブルーチンを全てインライン展開することなく, 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いでいる. また, OSCAR コンパイラでは粗粒度並列性解析のための解析時インライニング及びフレキシブ

† 早稲田大学理工学部電気電子情報工学科
Dept. of Electrical, Electronics and Computer Engineering, Waseda University

†† アドバンスト並列化コンパイラ研究体
Advanced Parallelizing Compiler Reserch Group
http://www.apc.waseda.ac.jp/

ルクローニング手法[5] の研究も行われいる．ただし，この手法では実際にインライン展開を行うため，多くの計算機資源を必要とし，全てのサブルーチンを考慮することが難しい．本論文では，推定したプログラム中の全階層の並列度をもとに下位階層に存在する並列性の高いサブルーチンを上位階層にインライン展開することによって，マルチグレイン並列性を向上させる選択的インライン展開手法を提案し，共有メモリ型マルチプロセッサシステム上で評価した結果を述べる．

2. 粗粒度タスク並列処理

本章では，逐次プログラムを階層的に粗粒度タスク分割して階層的マクロタスクグラフを生成し，粗粒度タスク間並列性解析を行う手法について述べる．

粗粒度タスク並列処理とは生成された MT をプロセッサ (PE)，もしくはプロセッサグループ (PG) に割り当てて実行することによりマクロタスク間の並列性を利用する並列処理手法である．

2.1 粗粒度タスク生成

粗粒度タスク並列処理では，逐次プログラムは基本ブロックまたはその融合ブロック (BPA)，繰り返し (ループ) ブロック (RB)，サブルーチンブロック (SB) の 3 種類のマクロタスク (MT，粗粒度タスク) に分割される．RB や SB に対しては，そのボディ部を階層的に粗粒度タスク分割し，この処理をプログラム全域に適用する．また RB が並列処理可能な DO ループである場合，これをタスク分割し，RB のループ並列性を粗粒度並列性に変換して並列処理を行っている．

2.2 粗粒度並列性抽出

マクロタスク生成後，各階層において，MT 間のデータ依存とコントロールフローを解析し，階層的マクロフローグラフ[1] を生成する．次に，コントロールフローとデータ依存を考慮し MT 間の並列性を抽出するために，各 MT の最早実行可能条件を解析し，階層的マクロタスクグラフ (MTG)[1] を生成する．この最早実行可能条件は，コントロール依存とデータ依存を考慮したマクロタスク間の粗粒度並列性を表す．

2.3 プロセッサグループとプロセッサエレメント

階層的に定義された MTG を効率よく処理するためにはプロセッサも階層的な集合形態をとらねばならない．OSCAR コンパイラでは，複数のプロセッサエレメント (PE) をソフトウェア的にグループ化し，1 つのプロセッサグループ (PG) と定義し，この PG に MTG 上のマクロタスク (MT) を割り当てる．割り当てられた MT 内で更に MTG が定義されている場合は，内部 MTG の並列性に応じて PG 内の PE を階層的に PG 化する．これを階層的に行うことでプログラム全域にわたる粗粒度タスク並列性を利用することができる．

3. MTG の並列性による自動インライン展開

粗粒度タスク並列処理においては，MT の逐次処理コストが相対的に大きい，より上位の (ネストレベルの浅い) 階層の並列性を利用する方が，同期やスケジューリングオーバーヘッドを相対的に小さく抑えられる．このため，並列性に応じて各階層に割当てる PG 数，PE 数を決定する並列処理階層自動決定手法[6] では，上位階層の並列性を優先している．しかし，並列性の高い MTG が異なるネストレベルに分散している場合，単に上位階層の並列性を優先するだけでは並列性の高い下位階層を効果的に並列処理するために必要なプロセッサ数を確保できなくなる場合がある．よって本論文では，階層自動決定手法による各 MTG の PG 数・PE 数より，割当てられるプロセッサ数よりも高い並列性を持つ SB を見つけインライン展開を行う手法を提案する．

3.1 並列処理階層自動決定手法

並列処理階層自動決定手法では，各 MTG の逐次処理コストと CP 長 (クリティカルパス長:入り口ノードから出口ノードまでの最長のパス長) を用いて各階層の並列度を推定する．MTG_i の逐次処理コストを Seq_i，MTG_i の CP 長を CP_i として，粗粒度タスク並列度 $Para_i$ を

$$Para_i = Seq_i \ / \ CP_i$$

と定義する．ここで，Seq_i は，算出した各 MT のコストをコントロールフローに従って総和した値，CP_i はクリティカルパス上にある MT のコストの総和である．この $\lfloor Para_i \rfloor$ は MTG_i を CP_i で処理するために必要な PG 数の下限である．

次に，粗粒度並列性とループ並列性を考慮した総合的な並列性を表す $Para\_ALD : Para\ After\ Loop\ Division$ を定義する．OSCAR コンパイラにおけるマルチグレイン並列処理では，並列処理可能 RB(DOALL ループ) をイタレーション方向に分割して，複数の小並列ループを生成し，各分割ループを MT として定義することにより，ループ並列性を粗粒度タスク並列性に変換して処理を行う．ここでは，ループ並列処理可能 RB を並列処理効果が期待される最小コスト T_{min} となるようにタスク分割した際の粗粒度並列性を等価粗粒度並列性と定義する．MTG_i 中の並列処理可能な RB をコストが T_{min} となるよう分割した際の CP 長を CP_ALD_i とし，

$$Para\_ALD_i = Seq_i \ / \ CP\_ALD_i$$

と定義する．

$Para$，$Para\_ALD$ は 1 階層が持つ並列性の大きさを表す指標であるが，注目階層 MTG_i とその下位階層の全並列性を表すために，

$$Hierarchical\_Para\_max_i = Convented\_Para_i \times \lfloor Para_i \rfloor \times \max_j(Hierarchical\_Para\_max_{i\_j})$$

を導入する．ここで $Convented\_Para_i$ は，MTG_i を内包する MT_i が並列処理可能 RB の場合は，コストが T_{min} を上回るように MT_i を分割した場合の分割数を

表し, MT_i を並列処理する場合の等価粗粒度並列性である. MT_i が並列処理可能 RB 以外の場合は 1 となる. $\max_j(Hierarchical\_Para\_max_{i\_j})$ は MTG_i の MT のうち, 最大の $Hierarchical\_Para\_max$ である.

これらの $Para, Para\_ALD, Hierarchical\_Para\_max$ を用いて, 各階層に割当てる PG 数・PE 数 の組合わせを決定する. まず MTG_i の粗粒度タスク並列性を十分に生かすための PG 数 N_{PGi}, PE 数 N_{PEi} は $Para_i \leq N_{PGi} \leq Para\_ALD_i$ かつ $N_{PGi} \times N_{PEi} = N_{Avail\_PEi}$ を満たすものとする. ここで N_{Avail_PEi} は MTG_i で利用可能な総プロセッサ数である. 上位階層の並列性を優先し, 割当てられたプロセッサを最大限に利用するため, これらの式を満たす N_{PGi}, N_{PEi} のうち, N_{PGi} が最大となる組合わせを用いる. ここで MT_{i_j} が MTG_i の j 番目の内部マクロタスクを表すとすると, N_{PEi} が MT_{i_j} に割当てられる総プロセッサ数となる. $N_{PEi} > Hierarchical\_Para\_max$ である場合, 無駄な同期やスケジューリングのオーバーヘッドを増加させる可能性がある. よってこの場合, MT_i に割当てられるプロセッサ数 N_{PEi} が $Hierarchical\_Para\_max_i$ と等しくなるような組合わせの N_{PGi}, N_{PEi} を決定する. これを上位階層から順に繰り返していき, 全階層の PG 数, PE 数を決定する.

3.2 MT が内包する並列度

3.1 節で MT が内包する全並列性を十分に利用するプロセッサ数の上限値として $Hierarchical\_Para\_max$ を定めた. これに対して, MT が内包する全並列性を利用するために必要なプロセッサ数の下限値を表す $Hierarchical\_Para$ を以下のように定義する. MT_i 内の全下位階層に対して, 並列処理可能 RB を T_{min} 以上のコストとなるようタスク分割し, 各 MTG の CP 長を下位階層から順に総和したものを $Hierarchical\_CP_i$ とする. つまり, MTG_i の内部マクロタスク MT_{i_j} の $Hierarchical\_CP_{i\_j}$ を, MT_{i_j} を処理する際の最小コストと考えたとき, この最小コストの和が最長となるパスが $Hierarchical\_CP_i$ である. ここで MT_i 自体が並列処理不可能な RB の場合, その回転数を考慮し, MTG_i の CP 長に回転数を乗じたものが, MT_i の $Hierarchical\_CP_i$ となる. MT_i が並列処理可能 RB の場合, MTG_i の CP 長が T_{min} を上回るコストである場合は, その CP のコストを $Hierarchical\_CP_i$ とし, T_{min} を上回らない場合は, T_{min} が $Hierarchical\_CP_i$ となる. $Hierarchical\_CP_i$ は MT_i 内の全並列性を利用した際の, 1 プロセッサ当りの最大処理コストを意味し, これを用いて,

$$Hierarchical\_Para_i = Seq_{MT_i} \ / \ Hierarchical\_CP_i$$

と定義する. 式中の Seq_{MT_i} は MT_i 自体のコストであり, MT_i が RB の場合は MTG_i の逐次処理コストに RB の回転数を乗じたものが Seq_{MT_i} となる.

次に, MTG_i に存在する全 SB をインライン展開した際の並列性 $Para\_inl_i$, $Para\_ALD\_inl_i$ を定義する. ここで, MTG_i 内の全サブルーチンをインライン展開した MTG_i を想定し, 並列処理可能 RB をタスク分割しない MTG_i の CP 長を CP_inl_i, RB の等価粗粒度並列性を考慮した MTG_i の CP 長を $CP\_ALD\_inl_i$ とする. これらを用いて,

$$Para\_inl_i = Seq_i \ / \ CP\_inl_i$$

と定義する. MT_i が SB であるとき, $Para\_inl_i$ は MT_i をインライン展開した際に上位階層に持ち越される粗粒度並列性を表す. 同様に,

$$Para\_ALD\_inl_i = Seq_i \ / \ CP\_ALD\_inl_i$$

と定義する. $Para\_ALD\_inl_i$ は MT_i をインライン展開することで上位階層に持ち越せる粗粒度並列性とループ並列性の総合値である.

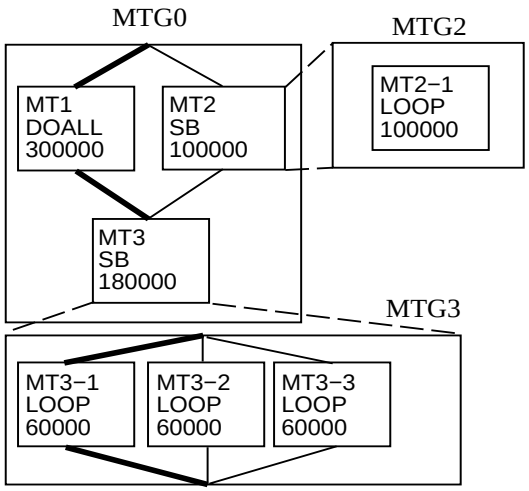

図 1 $Hierarchical\_Para, Para\_inl, Para\_ALD\_inl$ の例

例として図 1 に示す MTG を用いて, 並列度を計算する. 図 1 中, ノードは MT を表し, ノード内の数値はその MT の逐次処理コスト, DOALL は並列処理可能 RB, LOOP は逐次処理ループを表す. また, エッジはデータ依存を表し, 太線はクリティカルパスを表す. ここでは, ループの 1 イタレーション当りの逐次処理コストを 1000 とする. また, 並列処理の効果が見込める最低コスト T_{min} を 10000 とする. まず MTG_3 について考える. $Seq_3 = 180000$, $CP_3 = 60000$ であり, $Para_3 = 180000/60000 = 3$ となる.MTG_3 には, 並列処理可能 RB が無いため, $CP\_ALD_3 = 60000$ となる. よって $Para\_ALD_3 = 180000/60000 = 3$ である. また $Hierarchical\_CP_3 = 60000$, $Hierarchical\_Para_3 = 180000/60000 = 3$ である. 次にインライン展開時の並列度を考えると, MTG_3 内には SB が存在しないので $CP\_inl_3 = CP_3 = 60000$, $CP\_ALD\_inl_3 = CP\_ALD_3 = 60000$ より $Para\_inl_3 = 180000/60000 = 3$, $Para\_ALD\_inl_3 = 180000/60000 = 3$ となる. 次に MTG_2 については, 図 1 より逐次処理ループ 1 つ

表 1　1 の各並列度の値

MTG	MTG0	MTG2	MTG3
$Seq.cost$	580000	100000	180000
CP	480000	100000	60000
CP_ALD	280000	100000	60000
$Hierarchical\_CP$	160000	100000	60000
CP_inl	360000	100000	60000
CP_ALD_inl	160000	100000	60000
$Para$	1.21	1	3
$Para\_ALD$	2.07	1	3
$Hierarchical\_Para$	3.63	1	3
$Para\_inl$	1.61	1	3
$Para\_ALD\_inl$	3.63	1	3

のみであるため, 各並列度は全て 1 となる. 最後に, MTG_0 の各並列度を算出する. まず MTG_0 の逐次処理コストは $Seq_0 = 580000$ であり, CP は MT_1, MT_3 を通るパスで, $CP_0 = 480000$ となる. これより $Para_0 = 580000/480000 = 1.21$ である. また MTG_0 内では, 並列処理可能 RB である MT_1 の等価粗粒度並列性を考慮した場合の CP は MT_2, MT_3 を通るパスとなる. よって $CP\_ALD_i = 100000 + 180000 = 280000$, $Para\_ALD_0 = 580000/280000 = 2.07$ となる. 次に $Hierarchical\_Para$, $Para\_inl$, $Para\_ALD\_inl$ を算出する. まず $Hierarchical\_CP$ を考える. MT_1 に関しては等価粗粒度並列性を考慮し $Hierarchical\_CP_1 = 10000$, MT_2 については $Hierarchical\_CP_2 = 100000$, MT_3 については $Hierarchical\_CP_3 = 60000$ となる. よって, コストが最小となるパスは MT_2, MT_3 を通るパスであり, $Hierarchical\_CP_0 = 100000 + 60000 = 160000$ となるため, $Hierarchical\_Para_0 = 580000/160000 = 3.63$ となる. また CP_inl_0 は MT_1, MT_3 を通るパス長であり, MT_1 はループであるのでコストは $Seq_1 = 300000$, MT_3 は SB であるので, これをインライン展開したと想定した場合の, MT_3 に対応するコストは $CP\_inl_3 = 60000$. よって $CP\_inl_0 = 300000 + 60000 = 360000$ となる. これより $Para\_inl_0 = 580000/360000 = 1.61$ である. 同様に $CP\_ALD\_inl_0$ は MTG_0 内 SB をインライン展開し, 並列処理可能 RB の等価粗粒度並列性を考慮したときの最長パスであるので, $CP\_ALD\_inl_0 = CP\_ALD\_inl_2 + CP\_ALD\_inl_3 = 100000 + 60000 = 160000$ である. これより, $Para\_ALD\_inl_0 = 580000/160000 = 3.63$ となる. 以上のように算出した各 MTG の並列度を表 1 に示す.

3.3　PG, PE 構成に基づくインライン展開階層の決定

3.1 節では各階層の PG 数, PE 数の決定手法について述べ, 3.2 節で各 MT が内包する並列性, インライン展開で上位階層に持ち越せる並列性を定義した. 本節では, PG 数, PE 数が決定された MTG_i 内の各サブルーチンブロック MT_{i_j} に対して, MT_{i_j} を MTG_i にインライン展開する場合に利用可能な並列性とインライン展開しない場合に利用可能な並列性を比較することによって, 実際にインライン展開すべきかを判断する手法について述べる. 本手法では, インライン展開する SB を選択する段階では実際の展開は行わず, プログラム全体に対してインライン展開すべき SB を選択した後に, 実際の展開を行う. 以下に本手法のアルゴリズムを示し, 図 1 の MTG を用いて MTG_0 を 4 プロセッサで処理する際に, MT_2, MT_3 をインライン展開すべきかの判定を行う.

MTG_i の PG 数が 2 以上のとき, MTG_i のマクロタスク MT_{i_j} に割当てられるプロセッサ数は, MTG_i に割当てられるプロセッサ数より少ない. このようなケースにおいて, MT_{i_j} のインライン展開によって実際に利用することができる並列性が向上するかどうかによって, インライン展開すべきかを判断する. 図 1 の MTG_0 の PG 数・PE 数を定めるため, $Para_0$, $Para\_ALD_0$ を四捨五入した $1 \leq N_{PG0} \leq 2$ を満たす最大の PG 数となる組み合わせを考える. $N_{Avail\_PEi} = 4$ より $N_{PG0} = 2$, $N_{PE0} = 2$ という構成が選択される. しかし仮に全ての MT_{i_j}(SB) が, 割当てられるプロセッサ数よりも高い並列性を持たない, もしくはインライン展開により上位階層に並列性を持ち越せない場合には, をインライン展開を利用しても効果はない. よって全てのサブルーチンブロック MT_{i_j} が $Hierarchical\_Para_{i\_j} \leq N_{PEi}$ もしくは $Para\_ALD\_inl_{i\_j} = 1$ を満たすとき, MTG_i 内の SB のインライン展開は行わない. 逆にこの式を満たさない SB が存在する場合はインライン展開により MTG_i の並列性を高めることができる. 図 1 の MT_2 は, $Para\_ALD\_inl_2 = 1$ よりインライン展開の効果はない. また, MT_3 は, $Hierarchical\_Para_3 = 3$ よりインライン展開しない場合に割当てられるプロセッサ数 $N_{PG0} = 2$ より大きく, さらに $Para\_ALD\_inl_3 = 3$ であるのでインライン展開により MTG_0 に並列性を持ち越すことが可能である. よって MTG_0 をインライン展開元の階層とし, 実際にインライン展開するかどうかを以下の手順で判定する. まず, インライン展開で MTG_i の並列性が変わるため階層決定手法で定められた PG, PE 構成は適切でない場合があるので, インライン展開後の PG 数と PE 数を, $Para\_inl_i \leq N_{PG_i}' \leq Para\_ALD\_inl_i$ かつ $N_{PG_i}' \times N_{PE_i}' = N_{Avail\_PE_i}'$ を満たす組み合わせのうち, PG 数が最大となる N_{PG_i}', N_{PE_i}' と仮定する. 図 1 中の MTG_0 について, インライン展開後の PG 数, PE 数の組合わせは $2 \leq N_{PG_0}' \leq 4$ より $N_{PG_0}' = 4$, $N_{PE_0}' = 1$ となる.

次に, MTG_i 内の SB である MT_{i_j} をインライン展開しない場合に, MT_{i_j} 中で実際に利用可能な並列性 $Para\_E\_no\_inl_{i\_j} : Para\_Effective\_no - inl$ を定義する. MT_{i_j} をインライン展開しない場合, MT_{i_j} には N_{PE_i}' のプロセッサが割当てられ, プロセッサ数以上の並列性を利用することはできない. 一方, MT_{i_j} に内包される全並列性は $Hierarchical\_Para_{i\_j}$ であるため, MT_{i_j} に内包される並列性のうち, 実際に利用される並列

性は割当てられたプロセッサ数と $Hierarchical\_Para_{i\_j}$ の最小値であり,

$$Para\_E\_no\_inl_{i\_j} = min(N_{PE_i}', Hierarchical\_Para_{i\_j})$$

と定めることができる. 図 1 の MT_3 は, インライン展開されない場合に割当てられるプロセッサ数は $N_{PE_0}' = 1$, 内包される並列性が $Hierarchical\_Para_3 = 3$ となるため, 利用可能な並列性は $Para\_E\_no\_inl_3 = min(3,1) = 1$ となる. 次に, インライン展開を行った場合に利用可能な並列性について考える. インライン展開により MT_{i_j} の内部階層 MTG_{i_j} は上位階層 MTG_i に持ち越され, 最大で N_{PG_i}' の PG(プロセッサグループ) によって並列処理される. また, インライン展開後の MTG_{i_j} 内のマクロタスクには N_{PE_i}' のプロセッサが割当てられる. ここで, 展開後の MTG_{i_j} を N_{PG_i}' で並列処理する際に, MTG_{i_j} で実際に利用可能な並列性を $Para\_E\_PG_{i\_j}$, 展開後の MTG_{i_j} 内の MT 群を N_{PE_i}' のプロセッサで処理する際の利用可能な並列性を $Para\_E\_PE_{i\_j}$ とする.

$Para\_E\_PG_{i\_j}$ は MTG_{i_j} 内の全 SB をインライン展開したと想定した際の並列性 $Para\_ALD\_inl_{i\_j}$ が上限であるが, N_{PG_i}' 以上の並列性は利用できない. よってこれらの最小値をとり,

$$Para\_E\_PG_{i\_j} = min(Para\_ALD\_inl_{i\_j},\ N_{PG_i}')$$

となる. 図 1 の MT_3 について考えると $Para\_ALD\_inl_3 = 3$ の並列性を $N_{PG_0}' = 4$ のプロセッサで処理するので, $Para\_E\_PG_3 = min(3,4) = 3$ である.

次に $Para\_E\_PG_{i\_j}$ を求めるため, 下位階層全ての SB をインライン展開したと想定した際の MTG_{i_j} 内のマクロタスク群に内在する並列性を $Para\_Include_{i\_j}$ とし, 以下のように求める. MTG_{i_j} の全並列性は MT_{i_j} が SB であるため $Hierarchical\_Para_{i\_j}$ である. このうち MT_{i_j} とその内部の全 SB をインライン展開した際に MTG_i に持ち越される粗粒度並列性 $Para\_inl_{i\_j}$ は MT 間の並列性であり, MT 内に内包される並列性ではない. よって

$$Para\_Include_{i\_j} = Hierarchical\_Para_{i\_j} \ / \ Para\_inl_{i\_j}$$

となる. しかし, インライン展開を考慮した MTG_{i_j} 内に並列処理可能 RB がある場合, この RB の等価粗粒度タスク並列性も MTG_i に割当てられた PG によって利用される. よって, PG によって利用することのできる並列性が $Para\_E\_PG_{i\_j} > Para\_inl_{i\_j}$ である場合,

$$Para\_Include_{i\_j} = Hierarchical\_Para_{i\_j} \ / \ Para\_E\_PG_{i\_j}$$

であり, これらをまとめて

$$Para\_Include_{i\_j} = Hierarchical\_Para_{i\_j} \ / \ max(Para\_E\_PG_{i\_j}, Para\_inl_{i\_j})$$

とする. ここで, $Para\_E\_PE_{i\_j}$ は $Para\_E\_no\_inl_{i\_j}$ と同様に求まり,

$$Para\_E\_PE_{i\_j} = min(Para\_Include_{i\_j}, N_{PE_i}')$$

となる. 図 1 中 MT_3 が内包する並列性は, 内部階層 MTG_3 の粗粒度タスク並列性のみであるので, $Hierarchical\_Para_{i\_j} = Para\_inl_{i\_j} = 3$ である. MT_3 をインライン展開後はこの並列性が $N_{PG_0}' = 4$ の PG によって処理されるため $Para\_E\_PG_3 = 3$ となる. よって $Para\_Include_3 = 3/max(3,3) = 1$ であり, MT_3 をインライン展開した後に, MTG_3 内の MT_{3_1}, MT_{3_2}, MT_{3_3} に内包される並列性は 1 となる. これより, インライン展開後に $Para\_Include_3 = 1$ の並列性を $N_{PE_0}' = 1$ で処理する場合に利用可能な並列性は $Para\_E\_PE_3 = min(1,1) = 1$ である.

以上のように定義した,

展開前の並列性:$Para\_E\_no\_inl_{i\_j}$

展開後の並列性:$Para\_E\_PG_{i\_j} \times Para\_E\_PE_{i\_j}$

を比較し, [展開前の並列性] $\leq$ [展開後の並列性] を満たす MT_{i_j} をインライン展開を行う SB と判定する. 図 1 の MT_3 では $Para\_E\_PG_3 \times Para\_E\_PE_3 = 3 > Para\_E\_no\_inl_3 = 1$ となり, 展開後の方が, より効果的に並列性を利用可能であることが分かる. よって MT_3 はインライン展開すべきと判断される.

これを上位階層より順に全ての階層に適用した後, インライン展開すると判定された SB に対して実際にインライン展開を行う. インライン展開の後, 再び並列処理階層自動決定手法を用いて, 各 MTG の最終的な PG 数, PE 数を決定する. 図 1 に対しては MT_3 がインライン展開され, インライン展開後のプログラムに対して並列処理階層自動決定手法を用いると, MTG_0 は PG 数 4, PE 数 1 となる.

4. 性能評価

本章では, 提案するインライン展開手法を用いた並列処理階層自動決定手法を OSCAR マルチグレイン並列化コンパイラ上に実装し, その性能を 8 プロセッサ サーバ IBM RS6000 PowerPC 604eHigh Node 上で評価する. このマシンは 200MHz の Power PC 604e を 8 プロセッサ 搭載した SMP サーバである.

本評価では, 提案手法を組み込んだ OSCAR コンパイラ[1),2)] を並列化プリプロセッサとして用い, OpenMP API を用いた粗粒度並列化プログラムを出力した. 出力されたプログラムを IBM RS6000 上の IBM XL Fortran Version 7 によってコンパイル後, 実行する. 本評価においては, SPEC95FP 中の, su2cor, mgrid を用いた. 評価方法は本手法によるインライン展開と階層自動決定手法を用いた場合と, 階層自動決定手法のみによる場合の比較である. 参考として, XL Fortran の自動ループ並列化性能も示す. OSCAR コンパイラの出力を, XL Fortran でコンパイルする際のオプションは, 提案するインライン展開手法を

表 2 SPEC95FP 各ベンチマークの実行時間 [秒]

benchmark	su2cor	mgrid
逐次処理	504.49	596.44
XLF 最高性能	202.64(4)	193.84(4)
OFC 8PE	110.96	151.17
OFC(提案手法, 8PE)	98.78	84.09

用いた場合と用いない場合ともに “-O3 -qsmp=noauto -qarch=ppc” とし, XL Fortran のループ自動並列化の際のオプションは”-O5 -qsmp=auto -qarch=ppc” とした. また逐次性能評価のオプションは”-O5 -qarch=ppc” とした. OSCAR コンパイラ, XL Fortran コンパイラ共に, su2cor では配列リネーミングを行ったソースを入力とした.

評価を行ったプログラムの実行時間を表 2 に示す. 表 2 中, OFC は OSCAR FORTRAN COMPILER を表し, 8 プロセッサを用いた場合の処理時間である. XLF は XL Fortran コンパイラを表し,( ) 内は XLF が最高性能を示した際のプロセッサ数を表している. また, 表中の処理時間の単位は秒である.

su2cor では, 提案手法を用いない場合は処理時間が 110.96 秒であったが, 用いた場合 98.78 秒となり, 12 %の速度向上が得られた. また, XL Fortran コンパイラの 8 プロセッサまでの最小処理時間 202.64 秒と比べ, 2.05 倍の速度向上が得られた. 本手法によって, サブルーチン LOOPS の 3 重ループ DO 400 の最内側階層, 及びサブルーチン INT2V, INT4V の DO 100 の内部階層が本手法によりインライン展開元の階層として指定された. LOOPS-DO 400 は全実行時間中で大きな割合を占め, この最内側階層は粗粒度並列性 $Para = 2$ であり, この階層に 8 プロセッサ割当てられたため (2PG, 4PE) の粗粒度並列処理となる. この階層内部には, 内包する並列性 $Hierarchical\_Para$ が高い SB である PERM, MTAMAT, MATADJ, ADJMAT が存在し, これらが LOOPS DO 400 の最内側階層にインライン展開された. また, 4 プロセッサを割当てられたサブルーチン INT2V, INT4V の DO 100 内部もインライン展開前は (2PG, 2PE) という構成であったが, ループ内から呼ばれているサブルーチン BESPOL は高い並列性を内包し, これもインライン展開された. インライン展開により, これらのサブルーチン内の MT を処理するプロセッサ数が増えたため, 12 %の速度向上が得られたと考えられる.

mgrid では, 提案手法を用いることで 151.17 秒から 84.09 秒へと処理時間が短縮され, 79 %の速度向上を得た. これは XL Fortran コンパイラのみを用いた場合の, 8 プロセッサまでの最小処理時間 193.84 秒と比べ, 2.30 倍のスピードアップとなっている. この mgrid ではサブルーチン RESID, RPRJ3, PSINV, INTERP から呼ばれるサブルーチン COMM3 がインライン展開された. COMM3 を含むこれらの階層は, コストが大きな並列処理可能 RB を含むため 8 プロセッサが割当てられ, (8PG, 1PE) で処理される. このため階層自動決定手法のみの場合, COMM3 には 1 プロセッサしか割当てられず, COMM3 の並列性を十分に利用することができない. これに対し, 提案手法を用いると COMM3 がインライン展開され, 十分なプロセッサ数で並列処理されたため, 約 1.8 倍の速度向上が得られたと考えられる.

5. ま と め

本論文では, 階層的粗粒度タスク並列処理において, 並列性を高めるためのインライン展開手法を提案した. SPEC95FP ベンチマーク中の 2 本を用いて SMP サーバ IBM RS6000 SP 604e High Node 上で提案手法を評価し, 並列処理階層自動決定手法のみを用いた場合より su2cor で 1.12 倍, mgrid で 1.79 倍の速度向上となり, インライン展開すべきサブルーチンを適切に選択することができた. 今後の課題としては, プログラム中のデータの局所性を有効利用するローカライゼーション技術を考慮したインライン展開階層の自動選択が挙げられる. なお本研究の一部は, 経済産業省/NEDO ミレニアムプロジェクト IT21, および 早稲田大学理工総研プロジェクト研究「アドバンスト並列化コンパイラ」により行われた.

参 考 文 献

1) 岡本雅巳, 合田憲人, 宮沢稔, 本多弘樹, 笠原博徳: 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手法, 情報処理学会誌, Vol.35, No. 4,pp. 513-521 (1994).
2) H.Kasahara and et al: A Multi-grain Parallizing Compilation Scheme on OSCAR, *Proc. 4th Workshop on Language and Compilers for Parallel Computing* (1991).
3) 本多弘樹, 岩田雅彦, 笠原博徳: Fortran プログラム粗粒度タスク間の並列性検出手法, 電子情報通信学会論文誌, Vol. J73-D-1, No. 12, pp.951-960 (1990).
4) M.W.HALL and et al: Detecting Coarse-Grain Parallelism Using an Interprocedural Parallelizing Compiler, *Proc. of Supercomputing '95* (1995).
5) 熊澤慎也, 石坂一久, 小幡元樹, 笠原博徳: 粗粒度並列性抽出のための解析時インライニングとフレキシブルクローニング, 情報処理学会学会研究報告 ARC (2002).
6) 白子準, 神長浩気, 近藤巧章, 石坂一久, 小幡元樹, 笠原博徳: 並列処理階層自動決定手法を用いた粗粒度タスク並列処理, 情報処理学会研究報告 ARC2002-148-4 (2002).