4H−1　並列処理システム−晴−における条件分岐文の並列処理とその効果

山名早人，　安江俊明，　神舘　淳，　村岡洋一
（早稲田大学　理工学部）

１．はじめに

本報告では，プログラム内の条件分岐を並列処理することによるプログラム実行時間の短縮について述べ，我々の提案している並列処理システム−晴−上での条件分岐並列処理手法の性能予測を示す．

プログラム中の条件分岐を並列処理しようという試みは，VLIW型計算機を中心にこれまでに数多く行われている[1-3]．しかし，これらの方式は，大規模な並列処理計算機を対象とした方式ではないため，条件分岐の先行評価段数が小さく，得られる並列性も小さい．

これに対して，−晴−は1000台規模の要素プロセッサを持つため，先行評価段数を大きくし，十分な並列性をプログラムから抽出する．先行評価段数を大きくする手法として，我々はこれまでにフローグラフ展開を提案している．フローグラフ展開とは，条件分岐点における同期をとらず，条件の成立・不成立によって分かれる全ての制御フローについて演算を同時に実行し，後で制御に基づいて有効となったフローを選択する手法である．これまでの評価では，フローグラフ展開の対象となる部分について，1.5倍-5.2倍の処理速度の向上を確認している[4]．

本稿では，まず，(1)条件分岐の並列処理による処理速度向上をいくつかの科学技術計算プログラムのシミュレーション結果を用いて示し，次に(2)フローグラフ展開による処理速度の向上が，プログラム全体として考えた時に，どの程度期待できるかについて評価した結果を示す．

２．条件分岐の並列処理による効果

条件分岐を並列処理することによる効果を調べるため，以下にしめす８つの科学技術計算プログラムを用いてステートメントレベルでのシミュレーション評価を行った．なお，いづれも10×10のデータを利用した．

(1)HB（固有ベクトル逆変換）	(2)BL（平衡化）
(3)HE（ﾍｯｾﾝﾍﾞﾙｸﾞ行列変換）	(4)TS（マラー法）
(5)LM（一次関数極小化）	(6)AK（補間）
(7)MI（多変数関数極小化）	(8)NO（ﾏﾙﾀｰﾄ法）

シミュレーションにおける仮定を以下に示す．

(1)計算機資源は無限大．
(2)各演算実行時間を1単位時間に設定．
(3)データ通信時間は0．
(4)データ駆動計算機を仮定．
　・データが揃った時点で演算を実行．

シミュレーション結果を図１に示す．図１は上記８つのプログラムのシミュレーション結果の平均・最大・最小値を示す．図中の横軸は先行評価された条件分岐数（IF文の数）である．例えば，５であれば，常に５つ先の条件分岐まで先行評価することを示す．縦軸は，先行評価を行わない場合の実行時間を基準（＝１）とした時の処理速度向上比を示す．図１より，２０段以上の条件分岐を超えて並列処理を行うことにより，平均３倍以上の処理速度向上が得られることがわかる．また，先行評価の対象となる条件分岐数を無限大とした場合，平均５倍程度までの処理速度向上が得られることを確認した．

以上から，条件分岐を多段に渡って先行評価することにより，プログラム全体の実行時間を最大1/5に短縮することが可能であることがわかる．

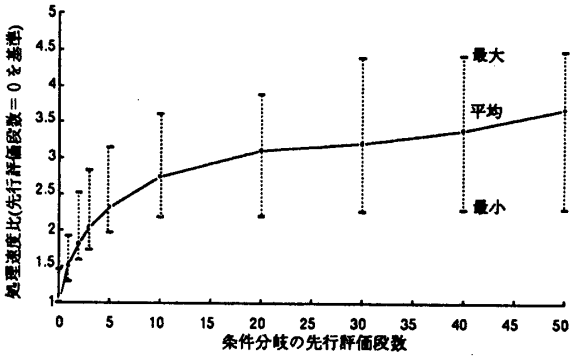

図1 条件分岐の先行評価による処理速度向上

３．フローグラフ展開[4,6]

フローグラフ展開とは，プログラム中の低並列度部分で，idle状態のPEを有効に利用し，全体の実行時間を短縮する手法である．具体的には，プログラム中に存在するデータ依存（オペランドが有効になるまで命令の実行ができないという依存）と制御依存（条件分岐で条件が決定するまで分岐先の命令が実行できないという依存）の内，制御依存を排除してプログラムを実行する方式である．例えば，５つのマクロブロック[4]間（−晴−におけるタスクの単位）のコントロールフローが図２(a)で示される時，同図(b)のように制御依存を排除する．この時，変換して出来た各スレッドを同時に実行させる．これによって，idle状態のPEを使用できると共に，制御依存がなくなった分，全体の実行時間を短縮することができる．

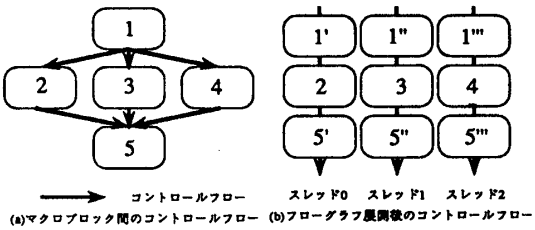

図2 フローグラフ展開例

ここで，マクロブロックの定義を以下に示す．マクロブロックとは，−晴−におけるタスクの単位であり，(1)他のマクロブロックからの制御の入口を１つだけ持つブロックであり，(2)DOループについては，DOSERIAL型のループをIF GOTO文による分岐と考えDOSERIAL型のループを消去した時，残った最

A Parallel Execution Scheme of Conditional Branches and its Evaluation for the Parallel Processing System -Harray-
Hayato YAMANA, Toshiaki YASUE, Jun Kohdate, Yoichi MURAOKA
Waseda Univ.

外側のDOALL/DOACROSS型のループを１つのブロックとしたものである．

フローグラフ展開を多段の条件分岐に渡って行うと，スレッド数が指数倍で増加する．このため，−晴−では実行確率の高いパスをスレッドとして用意し，他は，下方展開[4]と呼ぶ変換をした後，通常のマクロブロックとして扱う．また，フローグラフ展開はDOALLループを含まない低並列度部分に対して適用し，高並列度部分を含む部分に対しては適用しない．これは，高並列度部分を含むコードに対してフローグラフ展開を行った場合，各スレッドに与えられた少数のPE上で高並列度部分を実行することになり，逆に実行時間が増加してしまうからである．

従って，フローグラフ展開による処理速度向上を求めるためには，DOALL等の高並列度部分にまたがった条件分岐の先行評価を制限しなければならない．図３にDOALLを超える先行評価を制限した場合の処理速度向上率（先行評価をしない場合を基準）の平均を示す．また，マクロブロック単位で先行評価を行った場合の結果も合わせて示す．ただし，マクロブロック内は，ゲート後置手法[5]により先行評価が既に行われているものと仮定している．

図３よりわかるように，DOALLにまたがった先行評価を行わない場合，処理速度向上比が約２．５倍（先行評価を行わない場合の約２／５の実行時間）で飽和している．これは，DOALLループが障害となり，多段に渡る条件分岐の先行評価が制限されているためである．また，マクロブロック単位での先行評価の方が処理速度の向上の立ち上がりがよいのは，１つのマクロブロック内に複数のIF文が存在し，ゲート後置手法によってこのIF文以降の演算が先行評価されているためである．

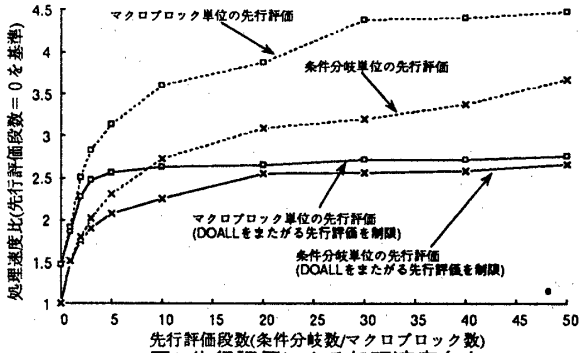

図3 先行評価による処理速度向上

次にデータ規模（扱うデータの大きさ）をパラメータとした処理速度比の推移を，３つのプログラムを例にとり図４に示す．横軸にデータの規模を，縦軸に先行評価をしない場合を基準とした時の処理速度向上比（先行評価をしない場合を１とし，先行評価段数＝無限大時の処理速度向上比）を示す．図４より，データの規模が大きくなるにつれ，(1)フローグラフ展開の効果が顕著に表れてくるプログラムと，逆に(2)フローグラフ展開の効果が薄れてくるプログラムが存在することがわかる．これは，データ量が増大すると，プログラム全体に占める逐次ループの実行時間の割合が増加し，(1)逐次ループ内に条件分岐が存在すればフローグラフ展開の効果が表れるが，(2)条件分岐が存在しない場合には，フローグラフ展開の対象とならず，逐次ループの実行時間が短縮されない．従って，プログラム全体としてみた時の処理速度向上率が下がることに起因している．すなわち，フローグラフ展開は，逐次ループが内部に条件分岐を持っており，さらにその逐次ループの実行時間が全体の実行時間（計算機資源を無限大とした時の実行時間）に占める割合が大きいほど効果が大きいことがわかる．

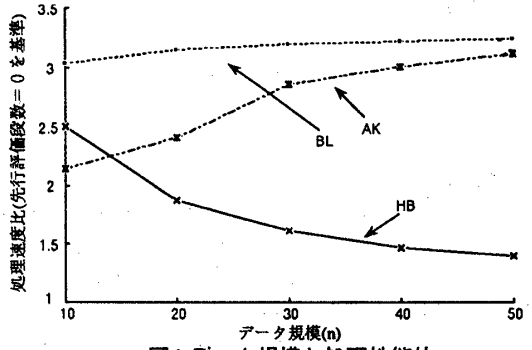

図4 データ規模と処理性能比

４．おわりに

本報告では，シミュレーション評価によりフローグラフ展開の性能予測を行った．その結果，フローグラフ展開を行うことにより，プログラム全体の実行時間が平均２／５に短縮されることがわかった．また，フローグラフ展開は，内部に条件分岐を持つ逐次型ループが多いほど効果が上がることを確認した．なお，本シミュレーションでは，各種オーバヘッドを考慮していない．これは，フローグラフ展開の前後でこれらのオーバヘッドを等しいと考えることができるからである[6]．

現在のフローグラフ展開は，DOALLにまたがって適用することができない．これは，DOALLにまたがって適用した場合，DOALL部分の計算を十分なPEを用いることができなくなり，逆に遅くなってしまうからである．しかし，図３の結果よりDOALLをまたがって適用できれば，さらに２倍以上の高速化が望めることがわかる．ここで，(1)PEが十分な数あり，扱うデータ量が十分に大きい場合，プログラム全体の実行時間は，SERIAL型のDOループの実行時間で決定される．このため，(2)DOALLを含んだスレッド数を（実行確率により）10程度に抑えることができたと仮定すると，DOALL部分の実行時間の増大自身は10倍であるが，プログラム全体に対する実行時間の増大は小さくなる．従って，(3)この割合が展開部分を増やしたことに伴う効果より小さくなれば，DOALLをまたがる展開による効果が表れる．今後は，以上の点を考慮して，DOALLをまたがったフローグラフ展開手法について検討を進めていく．

参考文献

[1]A.K.Uht:"Requirements for Optimal Execution of Loops with Tests," Proc. of ICS,pp.230-237 (1988)

[2]B.R.Rau et al.:"The Cydra5 Departmental Supercomputer," Computer, Vol.22,No.1,pp.12-35 (1989)

[3]J.A.Fisher:"Very Long Instruction Word Architectures and the ELI-512," Proc. of 10th Ann. Int'l Symp. on Computer Arch.,pp.140-150 (1983)

[4]H.Yamana et al.:"A Preceding Activation Scheme with Graph Unfolding Scheme for the Parallel Processing System -Harray-," Proc. of Supercomputing'89, pp.675-684 (1989)

[5]萩本他:"並列処理システム−晴−における実行時エラーの処理",第39回情処全大,6W-5,pp.1800-1801 (1988)

[6]山名他:"並列処理システム−晴−におけるフローグラフ展開を用いた条件分岐の並列実行",早稲田大学情報科学研究教育センター紀要,Vol.12 （掲載予定）