プレディケート付き依存グラフを用いたバージョニング手法

4U－7

桑原 睦子† 古関 聰‡ 小松 秀昭‡ 深澤 良彰†
†早稲田大学理工学部 ‡日本 IBM(株) 東京基礎研究所

1 はじめに

ループに対して複数のバージョンのコードを生成して、実行時に最良のバージョンを選択するコードを付加する手法（バージョニング手法）は以前から提案されている。バージョニング手法では、多数のバージョンから少数の有望なバージョンだけを選択することが重要である。従来のバージョニング手法では、バージョニングの後に続く並列化に向いたコードを得るための詳細な依存情報がないために、効率的なコード生成を行なうことができなかった。本稿では、詳細な依存情報を、依存の種類や方向がある条件によって変わる依存があるとき、それを決定するときの条件であると仮定している。また、この条件とそれによって決定される依存の対をプレディケートとよぶ。

本研究では、プログラム依存グラフ (PDG) を利用し、そのエッジに対してプレディケートを付加したグラフをプレディケート付き依存グラフとよぶ。このプレディケート付き依存グラフを使うことにより、依存の種類や方向が限定されるので、バージョニングの後に行なわれる並列化の並列性を引き出すのに有効なエッジを容易に選択できるようになる。

本発表では、ループ中の配列間の依存について述べる。特にパイプライン並列性を引き出すためのバージョニング手法のアルゴリズムを提案する。

2 プレディケート付き依存グラフ

プレディケート付き依存グラフにおけるプレディケートを求めるために、まず依存解析を行なう。

依存解析のために、ループ中の配列について依存の抽出を行なう。依存が存在する二つの配列の添字式は、同じ次元では、一つの繰返し変数しか含まない (これを SIV : single index variable とよぶ) ことが多く、また、ある次元における添字式中に現れた繰返し変数は他の次元の添字式中で使われない (この性質を分離可能であるとよぶ) ことが多い [1]。

また、配列の添字式中で配列参照がなされる場合がある。バンドマトリクスやスカイラインマトリクスを使った計算はその代表的な例である。これらは、ある次元における添字式中で参照される配列の特性を利用することによって、依存を取り除くことができる場合がある。

そこで、本研究では、依存解析対象の配列が SIV かつ分離可能であるときおよび配列の添字式中で配列参照がなされるときに、プレディケート付き依存グラフを作成する。このとき依存情報は依存探索ツリーを用いることによって得ることができる。依存探索ツリーは、場合わけの条件を節にもち、依存の種類と依存距離（あるいは依存方向）を葉にもつ。依存が存在する二つの配列の添字式と繰返し変数を与えると、依存探索ツリーが作られる。依存探索ツリーが作られるときの場合わけの全ての条件と依存の対が、その依存エッジに対するプレディケートとなる。

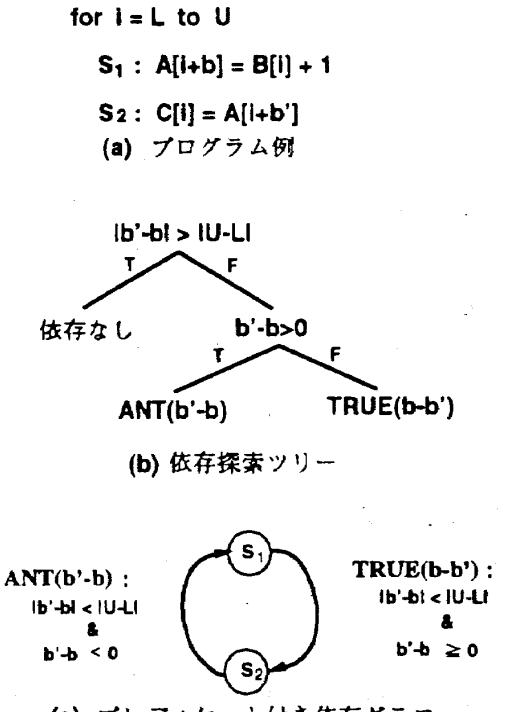

図 1: 依存探索ツリーとプレディケートつき依存グラフ

図 1 (a) のプログラムは SIV で分離可能であり、S_1 の配列 A[i+b] と S_2 の配列 A[i+b'] の間に依存が存在す

A Loop Versioning Technique using Predicated Dependence Graph
Mutsuko Kuwahara†, Akira Koseki‡, Hideaki Komatsu‡, and Yoshiaki Fukazawa†
†School of Science & Engineering, Waseda University
‡IBM Japan.Ltd. Tokyo Research Laboratory

る可能性がある。このプログラム中の配列Aの添字式（i+bとi+b'）と繰返し変数iのループ範囲（LとU）に基づいて作られた依存探索ツリーを図1 (b)に示す。bやb'、UやLについては値がわからないが、図1 (b)の依存探索ツリーから

|b-b'| ≦ |U-L| & b'-b ≦ 0のとき　真依存
|b-b'| ≦ |U-L| & b'-b > 0のとき　逆依存
|b-b'| < |U-L| のとき　依存なし

という3つの場合に分けられる。これらが依存エッジのプレディケートであり、図1 (c)はこのようにして得られたプレディケート付き依存グラフの例である。

SIVかつ分離可能および配列の添字式中で配列参照をしているという関係をもたない配列の添字式の場合には、プレディケートは付加せず、以前のバージョニング手法で行なわれた依存テストだけを行なうものとする。配列が多次元配列であるときは、次元ごとに依存テストを行なえばよい。

3 パイプライン並列性の利用

パイプライン並列を利用したループ並列化にはいろいろな手法があるが、その中にステージング[2]と呼ばれる手法がある。

この手法では、ループ本体をステージと呼ばれるいくつかの実行段階に分割し、ステージを本体としたループを作る。それらを各プロセッサに割り当て、実行をオーバラップさせ、パイプライン並列を利用して実行する。ステージ間に依存関係が存在する場合、それを保証するためにプロセッサ間でデータ通信が行なわれる。DO-ACROSS型ループの中には、プロセッサ間でサイクルを形成するような依存が発生してしまうケースが存在する。このサイクル依存は、プロセッサ間の実行のオーバーラップを困難にし、並列実行を大きく阻害する。サイクルを形成する逆依存や出力依存はリネーミングなどの手法を使って回避できるので本質的な問題にはならない。したがって、プロセッサ間でサイクルを形成する真依存をバージョニングできればよい。

4 アルゴリズム

パイプライン並列性を引き出すためのバージョニングアルゴリズムについて述べる。

1. どのエッジからバージョニングを行なうべきかエッジ全体に順番づけを行なう。このアルゴリズムをエッジオーダリングアルゴリズムとよぶ。
2. 最も優先度の高いエッジをバージョニングする。
3. パイプライン並列のための最適化を行なう。
4. それぞれのバージョンがプロセッサ数のリソースを使いきるまで2、3を繰り返す。

＜エッジオーダリングアルゴリズム＞

プレディケート付きの依存エッジ（Predicated Dependence Edge）があり、そのエッジによってサイクルが形成されている場合、そのエッジをバージョニングする。つまり、PDEがあるノード間にサイクルを形成するような依存が存在してはならない。

1. それぞれのPDEに対して、このエッジを含むサイクルを数えあげる。このサイクル数をC_eとする。
2. PDEが存在する入力ノードと出力ノードの間にある全てのエッジを数えあげる。このエッジ数をE_eとする。
3. それぞれのPDEに対して$C_e - E_e$の値を求める。この値をPDEの重みといい、この値の大きい順に順番づけを行なう。
4. 重みが同じ場合、次の優先順位で順番づけをおこなう。
   (a) SIVかつ分離可能な二つの配列の添字式は繰返し変数をIとしたときに(a*I+b , a'*I+b)と表すことができる（a,b,a',b'は任意の整数）。このとき、プレディケートに次の条件を持つエッジ。a=0またはa'=0、およびaとa'の符号が異なる。
   (b) サイクルを作るノード数が最も大きいエッジ。

   (a)、(b)とも複数ある場合は、任意の順番でよい。

5 おわりに

今回はパイプライン並列についてのアルゴリズムだけを紹介したが、プレディケート付き依存グラフを用いることによって、あらゆる並列化に対して最適なバージョニングを行なうことができる。それぞれの並列化によって、どのエッジを選択してバージョニングすればよいかは異なっており、難しい問題である。バージョニングの後に行なう並列化ごとの最適なエッジオーダリングアルゴリズムを求めることは非常に有効である。

参考文献

[1] G.Gina, K.Kennedy, and C.Tseng: "Practical Dependence Testing", PLDI,Vol.26,No.6,1991,pp.15-29

[2] 金丸, 古関, 小松, 深澤, "共有メモリ型並列計算機における多重ループステージングによるパイプライン実行", 情報処理学会研究報告,96-ARC-117,1996,pp.43-48