再帰的手続きの自動ベクトル化における幅優先法の拡張

6G－3

岡山 藤治　上原 哲太郎　津田 孝夫

京都大学工学部情報工学教室

1 はじめに

Fortran 90 の登場により、単純 DO ループ以外の様々な繰り返し構造に対する自動ベクトル化技法への要求が高まっている。本稿では、再帰的手続きの自動ベクトル化手法について述べる。我々は、当研究室で開発中の自動ベクトル化並列化コンパイラ V-Pascal[1] において、既に再帰的手続きの幅優先法によるベクトル化手法 [2] を提案しているが、本稿では、この手法を拡張し、様々な再帰的手続きに対する自動ベクトル化手法を提案する。この手法により、再帰呼出し間に部分的な依存が存在する手続きや、相互再帰を行う手続きのベクトル化が可能となる。

2 幅優先法

各手続きのために実行時に生成される局所変数の記憶域を環境と呼ぶことにする。手続きが再帰的手続きであった場合、各環境において実行される演算は同じものとなる。手続き中に再帰呼出し文が複数存在するとき、実行時の環境の構造は図 1 のような木構造をなす。

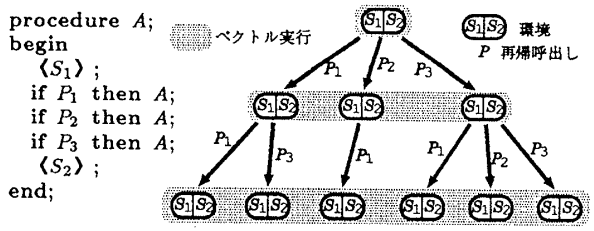

図 1: 再帰的手続きの環境木

呼出し側と呼出された側の環境間では、引数と帰り値の授受のため、実行順序の変更はできない。手続き中の全ての再帰呼出し文について相互に実行順序の交換が可能な場合、これらの呼出しは同時に行うことができ、これらの呼出しから生成される環境木の部分木上の演算を同時に実行することが可能である。再帰的手続きにおいて、このような手続き呼出し間の依存関係解析を行い、依存が存在しなければ、環境木の各深さ毎に、手続き中の演算をその深さにおける環境数をベクトル長としてベクトル実行するように手続きを変更する。このようにしても、元の手続きの意味を変えることはない。

変換後の手続きでは、各々の環境について、各呼出し実行の可否を決定する条件式 (呼出し条件) が全て偽となるまで、引数を計算する部分が実行される。その後、帰り値を計算する部分が実行される。また、再帰呼出し文は通常ある条件が成立したときのみ実行されるので、変換後の手続きでは、再帰の深さ毎に環境の収集/拡散を行い、選択実行を行う。

3 呼出し間の依存

手続き中の全ての呼出しの間に依存が存在した場合、環境木上の実行順序は全て確定しているので、本質的に並列実行は不可能である。しかし、手続き中の呼出しのうち、ある幾つかについて依存が存在しない場合、部分的に幅優先実行を行うことで、並列性を最大限に引き出すことが可能である。

手続き中の呼出しを、依存の無いものについてまとめ、幾つかの呼出し群に区分する。ある群中の各々の呼出しが生成する環境は並列実行が可能である。これらの環境を 1 つの縮退された環境として置き換えると、縮退環境中の演算はベクトル実行可能である。環境木にこのような変更を行ったものを、縮退化環境木として定義する。図 1の手続きにおいて、呼出し群 (P_2, P_3) は並列実行可能で、また (P_1) と (P_2, P_3) の間には依存が存在する場合の縮退化環境木を、図 2 に示す。

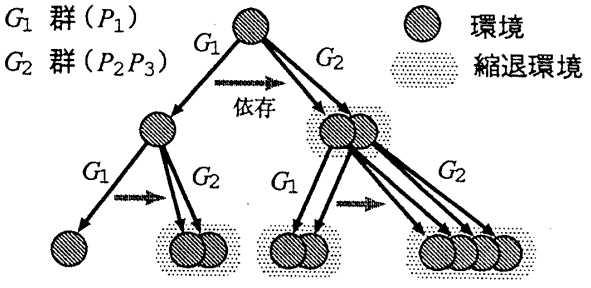

図 2: 縮退化環境木

縮退化環境木上では、もはや並列実行可能な呼出しは存在しないので、通常の再帰呼出しを行う。則ち、縮退化環境木上の深さ優先順で手続きを実行する。縮退環境上では、呼出し元の縮退度、則ちベクトル長に、呼出しが

Extentions of the Breadth-First Method for Vectorization of Recursive Procedures
Touji OKAYAMA, Tetsutaro UEHARA, Takao TSUDA
Kyoto University

行われた群の要素数を乗じたものが新しい縮退度になる。

このような変換を行った場合、各呼出し群のうち、複数の呼出しを含む群が少なくとも１つ存在すれば、元の手続きよりも高速化される可能性がある。

4 手続きの正規化

幅優先法による実行は、再帰的手続き中の複数の手続き呼出しを並列に行うことと同義である。このような呼出しの並列実行を行うためには、対象となる呼出し間に依存が無く、且つこれらの呼出しが手続き中で連続していなければならない。また、実行効率を向上させるためには、手続き中の呼出し群の数が少なく、各々の呼出し群の要素数が多い方が良い。このため、手続き変換を行う前に、手続きの正規化を行う。

手続き間の大域的な依存が存在しないとき、呼出し間の依存の有無は、その呼出しの条件、引数、帰り値の３変数によって定まる。よって、正規化を行うためには、まず手続き全体に対して依存関係解析を行い、その結果に従って、呼出し文が可能な限り隣接するように文の移動を行う。その結果、隣接する互いに依存の無い呼出し文を、群とする。

また、群の数の最小値は、手続き中の呼出し間の依存関係を DAG として表したときの 最長パス長＋1 で表される。

5 相互再帰

幅優先法の若干の拡張によって、互いに呼び合うような複数手続き間の再帰に対してもベクトル化が可能である。拡張された手法では、環境木上で、異なる手続きの環境をその節に持つことを許し、各呼出し群は、それぞれある１つの手続きに対する呼出しだけを含むようにする。その後、環境木を縮退し、各々の呼出し群に応じた手続きの環境を生成しながら幅優先実行を行うような手続き変換を行う。相互再帰の場合の環境木を 図３ に示す。

```
procedure A;
begin
  ⟨S1⟩;
  if P1 then A;
  if P2 then B;
  if P3 then B;
  ⟨S2⟩;
end;
procedure B;
begin
  ⟨S3⟩;
  if P4 then B;
  if P5 then A;
  if P6 then A;
  ⟨S4⟩;
end;
```

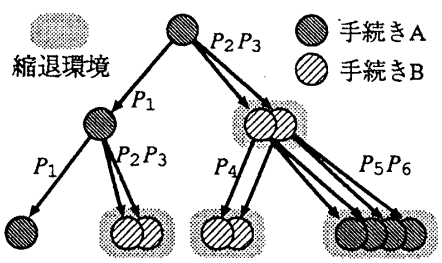

図 3: 相互再帰の環境木

正規化は、相互再帰を行う各々の手続きに対して行う。同一の手続きに対する呼出しを隣接させる作業が加わるため、正規化は若干複雑になる。正規化を行った全ての手続き中で、複数の呼出しを含む群が少なくとも１つ存在すれば、元の手続きよりも高速化される可能性がある。

相互再帰を行う複数の手続きをただ１つの手続きからなる再帰的手続きに変換することが可能である。この変換を行った手続きに対して幅優先ベクトル化を行うことも考えられるが、記憶領域と実行の効率の点で、上で述べた手法が優れている。

6 対象手続き

幅優先法によるベクトル化でのベクトル加速度は、環境木の形に大きく依存する。理想的な場合では、再帰が深くなるにつれてベクトル長が飛躍的に増大し、大きな加速が望める。しかし実際には、ベクトル長は呼出し条件の真偽によって大きく変化するので、各群の呼出し条件の値によっては、幅優先化のオーバーヘッドのため、変換前の手続きよりも実行速度が遅くなることが考えられる。呼出し条件を静的に解析することは非常に困難で、幅優先法によるベクトル化を適用するべきかどうかをコンパイル時に判定することは難しい。

幅優先法によるベクトル化は、呼出し条件の真率が高い場合に有効に働くため、配列などのデータ構造に対する全解探索のようなアプリケーションで、効果を発揮すると思われる。

7 おわりに

再帰的手続きのベクトル化によって、Algol 系言語等を用いてアルゴリズムに忠実に書かれたプログラムを、ベクトル計算機の能力によって高速に実行することが可能となる。この目的のもとに、幅優先法によるベクトル化を自動ベクトル化コンパイラ V-Pascal Ver.3 上に実装中である。

今後の課題としては、ループ中に再帰呼出しが含まれるような再帰的手続きの自動ベクトル化手法の開発が挙げられる。これについては幅優先ベクトル化法の改良によって行う手法を現在考察中である。

参考文献

[1] Tsuda, Takao and Kunieda, Yoshitoshi: V-Pascal: an Automatic Vectorizing Compiler for Pascal with No Language Extensions. *The Journal of SUPERCOMPUTING(Kluwer Academic Publisher)*, Vol.4, pp.251–275, 1990.

[2] 上原哲太郎, 津田孝夫：「幅優先法」による再帰的手続きの自動ベクトル化・並列化, 並列処理シンポジウム JSPP'93, pp.135–142, 1993.