3L－3　並列処理計算機Datarolマシン用プログラム抽出アルゴリズムの計算量

日下部 茂　立花 徹　谷口 倫一郎　雨宮 真人
(九州大学総合理工学研究科)

1．はじめに

我々は、データフロー・アーキテクチャの問題点を解決するアーキテクチャとして、Datarolマシン・アーキテクチャを提案している[1][2][3]。Datarolプログラムは、データフロー・プログラムからペア・オペランドの存在チェックやゲート演算などの余分なデータフロー制御を取り除いて最適化したマルチスレッド・コントロールフロー・プログラムである[1]。関数型言語Validのプログラムを Datarolプログラムにコンパイルするアルゴリズムについては既に述べた[3]。しかし、コンパイラを実現するに当たっては、そのアルゴリズムの計算量が問題となる。そこで本稿はDatarolプログラムを抽出するアルゴリズムについて述べその計算量について考察する。

2．Datarolプログラムの抽出

以下のようなステップを経て、データフロー用の関数型言語Validのプログラムを Datarolプログラムに変換する。

① 関数型プログラムを中間言語表現に変換する。
② ステップ①で得られた中間言語表現からマルチスレッド・コントロールフローを抽出する。
③ プログラム中の変数の生存区間を解析する。
④ プログラム中の変数にレジスタを割り付ける。

このうち、ステップ①に関しては既に存在する字句解析、構文解析等の手法で可能であるのでここでは述べない。ここではステップ②,③,④のアルゴリズムとその計算量について述べる。

3．コントロールフローの抽出

3.1 抽出アルゴリズム

まずコントロールフローを抽出するアルゴリズムについて述べる。Datarolプログラムでは、親子関係がある命令間でも子が親からの継続点となるとは限らない[3]。その例として、(w op u v)という命令について考えると、この命令が必ずしも u を生成する命令、 v を生成する命令の両方からの継続点になるとは限らない。例えば、u が v の先祖となっていれば、命令(w op u v)は u を生成する命令からの継続点とはならない(図1)。

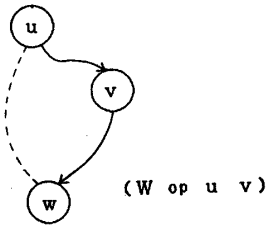

図1　継続点

このような継続点を決定するアルゴリズムの概略を以下に示す(図2)。

```
1. begin
2.   L←(x1,x2,・・・,xn) : 継続点を決定したい命令のリスト
3.   各命令に Cフラグ,Pフラグ
4.   while L が空でない do
5.     begin
6.       Lから1つの要素xi を取り出す
7.       xi の子q1,q2,・・・,qkのCフラグを立てる
8.       Q←(q1,q2,・・・,qk)
9.       R←(q1,q2,・・・,qk)
10.      while Qが空でない do
11.        begin
12.          Qの1つの要素qj の子をたどる
13.          if Cフラグ then そこはxi の継続点ではない
14.          if not(Pフラグ) then qj の子をQに加える
15.            else qj のPフラグを立てRにqj の子を加える
16.        end
17.      Rに含まれる命令のCフラグ,Pフラグを戻しておく
18.    end
19. end
```

図2　継続点決定のアルゴリズム

3.2 抽出アルゴリズムの計算量

全命令数N個のプログラム中、継続点を解析したい命令がn個だとする。上述のアルゴリズムではある命令 x_i についての解析時、11.～16.の部分を最悪約N回繰り返さなければならない。この時11.～16.の計算量が一定としてO(N)の計算量となる。これをn個の命令について行えば、n＜Nなのでその計算量は$O(N^2)$となる。

4．生存区間の決定

4.1 生存区間決定のアルゴリズム

Datarolアーキテクチャでは、変数の生存区間が終了した時点で、その変数に割り付けられているレジスタは解放することができ、他の変数に割り付けることが可能である。レジスタ割り付けの際にはこの特性を利用しプロセッサ内のレジスタファイルを最大限に利用するプログラムを抽出するようにしなければならない。

ある変数の生存区間の終点を求めることは、その変数の値を生成する命令の全ての子の共通の子孫に含まれる命令で、その親が共通の子孫に含まれない命令を見つけることである(図3)。そのような生存区間の終点を求めるためその変数の値

Complexity of Compilation Algorithm for parallel computer "Datarol machine"

Shigeru KUSAKABE, Tohru TACHIBANA, Rin-ichiro TANIGUCHI, and Makoto AMAMIYA
Graduate School of Engineering Sciences, Kyushu University

を生成する命令の子に番号を付け、その番号を渡しながら次々に継続点をたどる。マッチングが必要な継続点では受け取った番号データをマージする。マージの結果、全ての子の番号が揃っていればそこは生存区間の終わりであることが分かる。その生存区間決定のアルゴリズムは以下のとおりである(図4)。

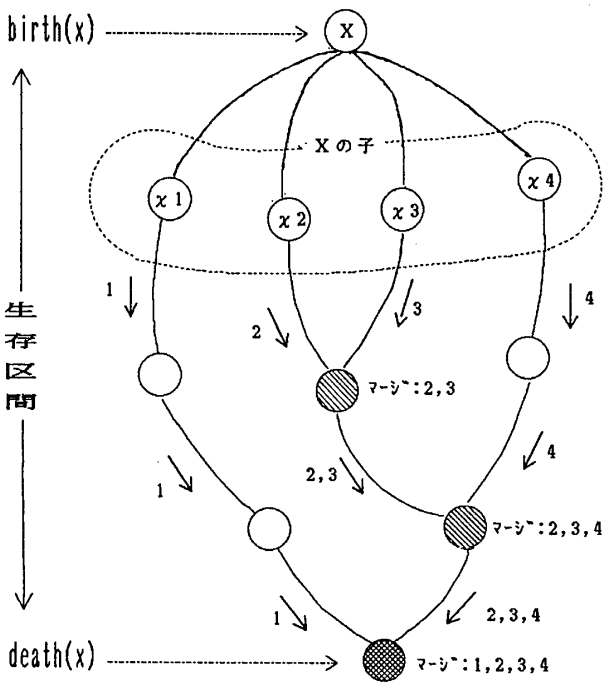

図3　生存区間

```
1. begin
2.   L←(x1,x2,・・・,xn) : 生存区間を求めたい変数のリスト
3.   Lの要素のうちマッチングが必要なものにフラグを立てる
4.   while Lが空でない do
5.     begin
6.       Lから1つの要素xiを取り出す
7.       xiを生成する命令の子p1,p2,・・・,pkに番号を与える
8.       Q←(p1,p2,・・・,pk)の継続点の集合
9.       R←(nil)
10.      while Qが空でない do
11.        begin
12.          Qの1つの要素qjを取り出す
13.          if qjの継続点はマッチング不要 then
14.            qjに親と同じ番号を与えqjの継続点をQに加える
15.          elseif フラグが立っている then
16.            フラクを戻してqjをRに加える
17.          else
18.            begin
19.              2つの親の番号のマージを行う
20.              if マージ結果の長さがk then
21.                Q←(nil) : そこがxiの生存区間の終わり
22.              else マージ結果をqjに与え継続点をQに加える
23.            end
24.        end
25.      Rに含まれる命令のフラグを立てる
26.    end
27. end
```

図4　生存区間決定のアルゴリズム

4.2　生存区間決定のアルゴリズムの計算量

生存区間を求めたい変数の数がn個である場合に、このアルゴリズムでの生存区間の解析に必要な手間について以下に考察する。プログラム中のN個の命令によって生成される全ての変数に対して生存区間を求める必要はなく、N＞nとなる[4]。変数 $x_1,x_2,\cdots,x_n$ の生存区間を求めるために、$m_1,m_2,\cdots,m_n$ 個の命令をそれぞれ調べる必要があるとする。また、変数 $x_1,x_2,\cdots,x_n$ を生成する命令の子の数をそれぞれ $d_1,d_2,\cdots,d_n$ 個とする。このとき、ある変数 $x_i(1\leqq i\leqq n)$ の生存区間解析の11.～24.の部分を m_i 回繰り返さなくてはならない。その部分で最も多くの計算量を必要とする19.の解析での最大の計算量は d_i に比例する。以上の解析に必要な計算量は$O(d_i*m_i)$以下となる。Datarolプログラムではある命令の親の数は最大2個なので、プログラム全体の合計では2N個より少なくなる。よって子の数についても$\sum d_i<2N$が成り立つ。このときm_iがNに比例するとして全解析に必要な手間は$O(\sum d_i*m_i)\leqq O(N^2)$となる。

$$\because\ \sum d_i*m_i = d1*m1+d2*m2+\cdots+dn*mn$$
$$= c1N*d1+c2N*d2+\cdots+cnN*dn \quad \{0<ci<1\}$$
$$< CN(d1+d2+\cdots+dn) \quad \{C=\max(ci)\}$$
$$= CN(\sum d_i) < CN*(2N)$$

5．レジスタ割り付け

ここではレジスタの割り付けについては紙面の都合上簡単な説明にとどめる。既に生存区間が決定していればレジスタの割り付けは次のように行うことが出来る。

- プログラムをたどりながら必要な変数にはレジスタを割り付けていく。
- その際、たどり先の命令で生存区間が終わっている変数があればその変数にに割り付けられているレジスタを解放し再利用可能にする。

詳しい説明は省略するが、Datarolアーキテクチャではプロセッサ内のレジスタ数は一定であり[1]、上述の割り付けは命令数に比例した計算量で行うことが出来る。

6．まとめ

本稿ではDatarolプログラム抽出過程のうちの、コントロールフローの抽出,生存区間の決定,レジスタ割り付けのアルゴリズムとその計算量について述べた。結論としてそれらの解析に必要な計算量は$O(N^2)$だということが分かった。しかし、一般にプログラム中の変数には参照の局所性があると言われている。$O(N^2)$は最大の計算量であり、一般のプログラムで参照性の局所があるとすれば、ほとんどのプログラムではもっと少ない計算量で解析できる可能性がある。今後は多くのプログラムをコンパイルし、必要とする平均計算量について考察を行う。

参考文献

[1]雨宮:超多重並行処理のためのプロセッサ・アーキテクチャ,情報処理学会「コンピューター・アーキテクチャ」シンポジウム,1988,pp99-108
[2]上田,谷口,雨宮:Datarolプロセッサのアーキテクチャについて,情報処理学会第38回全国大会講演論文集,1989
[3]立花,谷口,雨宮:データフロー解析による関数型プログラムのコンパイル法,情報処理学会第38回全国大会講演論文集,1989
[4]日下部,立花,谷口,雨宮:Datarolプログラムの変数の生存区間の解析について,第42回電気関係学会九州支部連合大会講演論文集,1989