1P－2

データフローグラフ変換による並列度抽出

吉瀬謙二*1, 中村友洋*1, 金指和幸*2, 田中英彦*1
*1東京大学工学部, *2富士通(株)

1　はじめに

プログラムには分岐やループ等の制御命令が存在するためにプログラム全体から並列度を抽出するには限界がある。しかしトレースデータにおいては、入力データとプログラム実行時の状態に特化しているが、動的な命令は静的な命令へと変化するため、実行された命令全体から並列度を抽出することが可能になる。

本研究では、トレースデータから得られるデータフローグラフに変換規則を用い、理想的な実行が行なわれた時の並列度を抽出する。

2　トレースデータとデータフロー解析

以下実際に例を挙げ、並列度を抽出する過程を説明する。

2.1　C言語によるソースファイル

x の値によって4つの値の和または差を求める関数が $x=1$ で実行された時の例を以降説明する。

```
1:  function( x , a , b , c , d)
2:    {
3:      if (x==1)
4:        return a + b + c + d;
5:      else
6:        return a - b - c - d;
7:    }
```

2.2　実行ファイルとトレースデータ

左側の命令列がコンパイラによって作られた(SPARC用)実行ファイルで、対応する右側の命令列がトレースデータの出力を簡略化したものである。トレースデータの右に現れる命令番号は命令の依存関係を示しこの依存関係がデータフロー解析を可能にしている。

実行ファイルに対応するフローグラフを図1に示す。この図で破線は制御依存を表している。

トレースデータに対応するデータフローグラフを図2に示す。$x=1$ に特化した時の実行であるため制御依存は存在しない。

```
1:    ST    x,   [44]          1:  ST
2:    ST    a,   [48]          2:  ST
3:    ST    b,   [4c]          3:  ST
4:    ST    c,   [50]          4:  ST
5:    ST    d,   [54]          5:  ST
6:    LD    [44],$0            6:  LD    1:
7:    CMP   $0,  0x1           7:  CMP   6:
8:    BNE   L20                8:  BNE   7:
9:    LD    [48],$0            9:  LD    2:
10:   LD    [4c],$1            10: LD    3:
11:   ADD   $0,  $1, $0        11: ADD   9: 10:
12:   LD    [50],$2            12: LD    4:
13:   ADD   $0,  $2, $0        13: ADD   11: 12:
14:   LD    [54],$3            14: LD    5:
15:   ADD   $0,  $3, $0        15: ADD   13: 14:
16:   B     L18                16: B
17: L20:
18:   LD    [48],$0
19:   LD    [4c],$1
20:   SUB   $0,  $1, $0
21:   LD    [50],$2
22:   SUB   $0,  $2, $0
23:   LD    [54],$3
24:   SUB   $0,  $3, $0
25:   B     L18
26: L18:
```

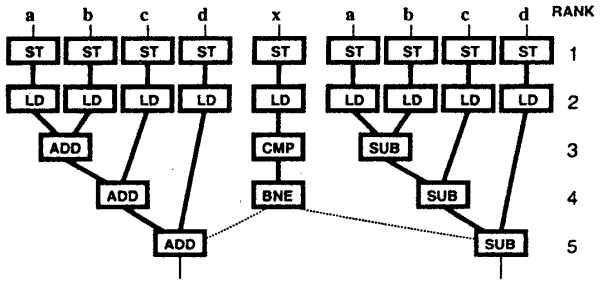

図1: 実行ファイルに対応するフローグラフ

2.3　並列度

データフローグラフにおいて、各命令は深さを表すRANKという値を持つ。またRANKの最大値をRANK値と定義する。この時RANK値は実行時間に、同じRANKを持つ命令の数は, その深さの並列度に対応する。また総命令数をRANK値で割ったものは平均並列度を表す。

2.4　変換規則の適用

図2に示すデータフローグラフに以下の変換規則を用いる。

Finding Parallelism using Dataflow Graph Transformations
Kenji KISE*1, Tomohiro NAKAMURA*1,
Kazuyuki KANAZASHI*2,Hidehiko TANAKA*1
*1Faculty of Engineering, University of Tokyo
*2Fujitsu Limited

変換規則 1 データ依存に関係無い, ロード, ストア, 分岐命令等の除去

変換規則 2 加法, 減法の結合則を用い RANK 値 を低くする

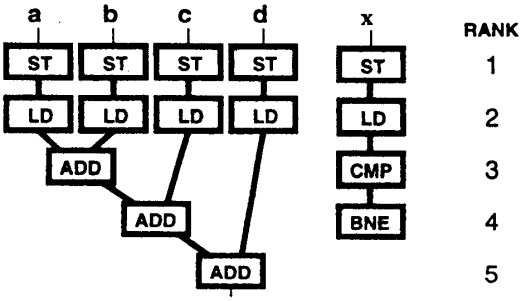

図 2: トレースデータに対応するデータフローグラフ

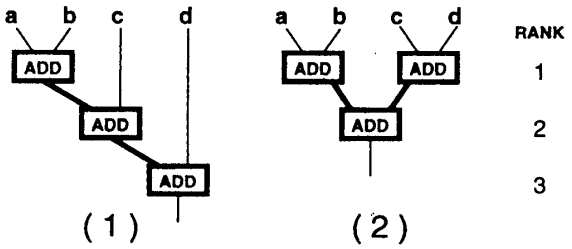

図 3: 対応する変換規則による変換結果

変換規則 1 によりプログラムの本質的流れの抽出が、変換規則 2 によりデータフローグラフの並列化が行なわれる。

2.5 平均並列度の意味

2 つの変換規則を用いて得られた平均並列度は、以下の条件を満たした理想計算機において取り出し得る最大の平均並列度とみなすことができる。

1. 演算器数 及びその組合せに制限がない
2. レジスタ数に制限がなく競合も存在しない
3. 分岐の投機実行が理想的なタイミングで行なわれかつ分岐予想が 100 % 成功する

また理想計算機における平均並列度を α , プロセッサ p 上での平均並列度を β , とした時 β/α はプロセッサ p がどれだけ理想的な実行を行なったかの尺度となる。

3 実験結果

```
1: MatrixProduct( a[n][n], b[n][n], c[n][n])
2:   {
3:     for (x=0; x < n; x++)
4:       for (y=0; y < n; y++)
5:         for (i=0; i < n; i++)
6:           c[x][y] = c[x][y] + a[i][x] * b[y][i];
7:   }
```

このプログラムで, 8 × 8 行列の積を求めた時の実験結果を示す。変換前, 変換規則 1 適用後, 変換規則 2 適用後のそれぞれの場合について、図 4 に各 RANK と並列度の関係を、表 1 に平均並列度の変化を示す。

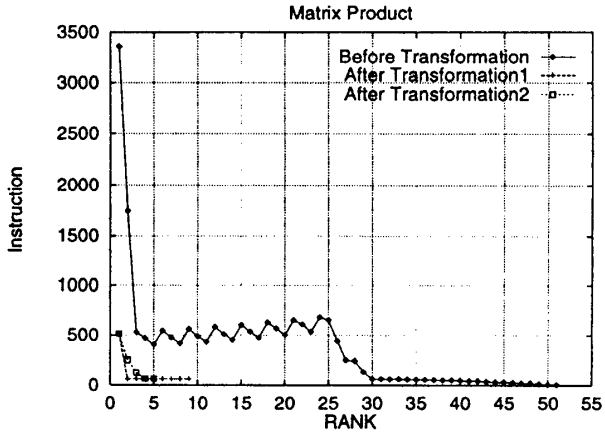

図 4: 行列積における各 RANK の命令数

表 1: 変換規則による変化

	総命令数	RANK 値	総命令数/RANK 値
変換前	19444	51	381.3
変換 1 後	1024	9	113.8
変換 2 後	1024	5	204.8

プログラム中の 3 重ループと 2 次元配列により、変換前のフローグラフは複雑なものになるが、変換規則 1 を適用したデータフローグラフではプログラムの本質的なデータ依存のみが抽出され, さらに変換規則 2 を用いることでデータフローグラフの並列化がおこなわれている。

4 まとめ

トレースデータに変換規則 1 を用いデータ依存関係のみを表すフローグラフを作成した。さらに変換規則 2 を適用して得られる並列度はプログラムから取り出し得る最大の並列度 (理想的なプロセッサを考えた時の並列度) であり、計算機の並列度をこの値に近づけることが目標となる。しかし理想計算機は極端に現実の計算機と異なるため、解析結果を実際のプログラムにフィードバックさせるためには、今回用いた変換規則をもとに現実の計算機の制約を反映させた変換規則の体系化をおこない、その変換規則による出力の解析を行なっていく必要がある。

謝辞

本研究の一部は文部省科学研究費 (一般研究 (B) 課題番号 07458052「大規模データパスプロセッサの研究」) による。

参考文献

[1] 許昭倫, 岡崎信一郎, 溝口正典. データフロー計算機向けフローグラフの最適化 並列処理シンポジウム JSPP '93 論文集, pp. 339-346, May 1993.