

分割・再構成によるループの並列化変換について

2G-6

十山 圭介

大西 裕二

神野 俊昭

(株)日立製作所 システム開発研究所

1. はじめに

マルチプロセッサシステムにおいて並列処理を行う場合に、最も実行速度の向上が期待できるものはループを並列化して実行するものである。ループの並列化手法では、これまで do all, do across, cycle shrinking 等の手法が提案されている。[1][2]

ループを並列に実行可能な要素に分割する場合、各要素が割当てられるべきプロセッサ間のデータ転送をできるだけ少なくすることが望ましいが、ソースプログラムの静的な構造だけから変換を行う手法では不要なデータ転送や待機が入り込むことがある。

ここでは、ソースプログラムの形式から直接に導かれる変換で、1つのループを多数の並列小ループとして分割し、データ転送や待機が少なくなるように再構成する手法について提案する。

2. 方式の概要

本方式においては、1重の DOループを対象とする。また、割当てべき(仮想)プロセッサは同一の性能のものが無限個あるものとする。ループ本体の異なる繰返しの中から実行文を選び、基本フロー単位と名付ける文の集合を構成する。この集合を、新たな小ループの本体(1回の繰返し)とし、この基本フロー単位間の依存関係によって複数個の並列ループを構成する。方式は以下のとおりである：

- 1) 基本フロー単位を決定する。
- 2) 基本フロー単位間の依存距離によって(仮想)プロセッサへの割り付けを行う。
- 3) 各々のプロセッサ上でループを構成する。
- 4) 各々のループにおいて初期処理と終了処理を加える。

2.1 基本フロー単位

基本フロー単位は元のループ本体の文から構成され、ループを展開した空間内の文の集合で、繰返し回数の違いを除いて本体の全ての実行文を含む単位である。

また、本体の文 S_i と文 S_j との依存距離とは、文 S_i

の m 回目の実行が文 S_j の n 回目の実行結果に影響を与える場合の繰返し回数の差 $(n-m)$ である。次のループ

```
DO i=1,N
... S1
... S2
END
```

において、文 S_i の j 回目の実行を $S_i[j]$ と表すとき、 $S_1[j]$,

$S_2[j+\alpha]$ (α は S_1 と S_2 との依存距離)

がこのループの基本フロー単位である。簡単のため、ここではループの制御変数を含まない文は考えないものとする。

2.2 基本フロー単位の割り付け

依存グラフとは、ループ本体の各文をノードとし、それらの依存関係をエッジとする有向グラフである。エッジには前述の依存距離が付加されている。基本フロー単位の候補は一般に複数個存在するが、ループの分割においてはその1つを選択する。

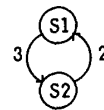
基本フロー単位を新たなループの本体とすると、文間の依存距離を基本フロー単位での依存関係に変換することができる。これを基本フロー単位間の依存距離と呼ぶことにし、以下のようにして求める：

2.1で述べた基本フロー単位を選択した場合に、依存グラフ上で選択されなかったパスが残る。このパスの依存距離とこのパスに含まれる基本フロー単位内の文間の依存距離の和が基本フロー単位間の依存距離であり、基本フロー単位間の依存距離の最小値を D とすると、 D はループを分割したときに、並列に稼働するプロセッサの台数となる。

3. 例

例として以下の DO ループを考える。(例1)

```
DO i=3,N
A(i)=B(i-2) ... S1
B(i)=A(i-3) ... S2
END
```



この場合、基本フロー単位として $S1[j]$, $S2[j+3]$ と $S2[j]$, $S1[j+2]$ とが考えられる。ここでは、 $S2[j]$, $S1[j+2]$ を採用する。この例では無関係であるが、プロセッサ間のデータ転送による待機を少なくするため、一般に依存距離の和の小さいものを選択する。

基本フロー単位間の距離は5であるので、プロセッサ5台でこれらの基本フロー単位のループが並列実行できることになる。

プロセッサ 1	プロセッサ 2	...	プロセッサ 5
DO i=3,N,5	DO i=4,N,5		DO i=7,N,5
S2[i]	S2[i]	...	S2[i]
S1[i+2]	S1[i+2]		S1[i+2]
END	END		END

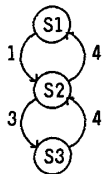
この例ではデータ転送や待機が生じないので、並列度が5とループのもつ並列性が十分に生かせる。

プロセッサ間でデータ転送を生じる例では次のものがある。(例2)

```

DO i=1,N
  A(i)=B(i-4)      ... S1
  B(i)=A(i-1)+C(i-4) ... S2
  C(i)=B(i-3)      ... S3
END

```



この例では $S1[j]$, $S2[j+1]$, $S3[j+3]$ が基本フロー単位となる。基本フロー単位間の依存距離は $1 + 4 = 5$ であり、前例と同様プロセッサ5台で並列実行することが可能である。この場合、基本フロー単位間でのデータ転送が発生し、他のプロセッサからの出力データを待機し、他のプロセッサへデータを出力する。

プロセッサ 1	プロセッサ 2	...	プロセッサ 5
DO i=3,N,5	DO i=4,N,5		DO i=7,N,5
S1[i]	S1[i+1]		S1[i+4]
S2[i+1]	S2[i+2]	...	S2[i+5]
S3[i+4]	S3[i+5]		S3[i+8]
Dtr P3 S2[i+8]	Dtr P4 S2[i+9]		Dtr P2 S2[i+12]
END	END		END

Dtr Px Sx[j] はプロセッサxの Sx[j] 文へのデータ転送を表す。

4. 考察

プロセッサ間のデータ依存を伴わない場合、例1に見られるように、高い並列性を得ることができる。cycle shrinking [2] による変換では以下のようになり、並列度が4で、かつ外側ループの繰返しごとに同期が必要となる。

```

DO j=3,N,2
  DOALL i=j,j+1
    A(i)=B(i-2)
    B(i)=A(i-3)
  END
END

```

一般に、ループを基本フロー単位に従った部分ループに分割することにより、各々のループのストライドが大きくなり、並列度が向上する。プロセッサ間でのデータ転送を伴う例2の場合でも、基本フロー単位の決定時にデータ依存関係を反映しているので、転送の影響を小さくできる。

5. おわりに

ループの形式に着目し、データの依存関係を反映した基本フロー単位を設定してループを小ループに分割、再構成することによって転送を少なくし、並列度を大きくするループ変換法について提案した。最後に、本方式について御討論を頂いている立教大学島内剛一教授に感謝いたします。

参考文献

- [1] Padua, D. and Wolfe, M. 'Advanced Compiler Optimizations for Supercomputers' CACM 29-12
- [2] Polychronopoulos, C. 'Automatic Restructuring of Fortran Programs for Parallel Execution' in LNCS 295