5D−5

並列化支援システム「Parassist」の試作
- プログラム解析方法 -

飯塚孝好*，梅原清美*，木村和幸**，黒澤隆**
*（株）日立製作所　中央研究所，**（株）日立東北ソフトウェア

1．　はじめに

既存のプログラムの並列化支援のためのシステムParassist (Parallelization Assist System) [1] の設計に当たって，典型的なFortran並列化ベンチマークに対して並列性の調査をしたところ，手続き内の解析に加えて，手続き間にまたがった配列データフロー解析，特に，手続きで作業的に使われる配列の検出が重要であることが分かった。従来の手続き間配列データフロー解析[2]では，手続き内での配列の定義／使用の可能性を検出するフロー非依存な解析が主体で，定義と使用の順序関係や定義の必然性を考慮したフロー依存な解析は行なわれておらず，作業配列の検出には不十分であった。

そこで，Parassistでは，フロー依存な手続き間配列データフロー解析を行なうこととし，試作したので報告する。

2．　解析情報

作業配列とは，ループの各回内での計算のために他の回とは独立に定義され，使用される配列である（図1a）。既存の逐次プログラムでは，ループの各回で同一の作業配列を用いることが多い。そのため，そのままではループを並列に実行することができない。このようなループを並列化するには，配列の次元を1次元増やして，ループの各回毎に別々の部分配列を参照するようにすればよい（図1b）。

```
   PARAMETER (N=100)
   REAL A(N,N),W(N)
   ...
並列化不能 (Wは、作業配列)
   DO 10 I=1,N
   CALL REV(A(1,I),W,N)
10 CONTINUE
   ...
   END
   ...
   SUBROUTINE REV(V,W,N)
   REAL V(N),W(N)
   DO 10 I=1,N
   W(I)=V(N-I+1)
10 CONTINUE
   DO 20 I=1,N
   V(I)=W(I)
20 CONTINUE
   END
```

図1a

```
   PARAMETER (N=100)
   REAL A(N,N),W(N,N)
   ...
並列化可能
   DO 10 I=1,N
   CALL REV(A(1,I),W(1),N)
10 CONTINUE
   ...
   END
   ...
   SUBROUTINE REV(V,W,N)
   REAL V(N),W(N)
   DO 10 I=1,N
   W(I)=V(N-I+1)
10 CONTINUE
   DO 20 I=1,N
   V(I)=W(I)
20 CONTINUE
   END
```

図1b

手続き間配列データフロー解析の解析結果は各手続き毎に，参照される部分配列（リージョン[2]）の和として表すこととし，リージョンを用いて表現する情報の種類としては，作業配列の解析および手続き内で定義される配列の手続き出口でのLive性の解析も行なえるよう，表1の5種類とした。

表1　解析情報

RMOD	手続きの実行により、値が定義される可能性のあるリージョン
RUSE	手続きの実行により、値が使用される可能性のあるリージョン
RKILL	手続きの実行により、値が必ず定義されるリージョン（いずれの制御フローでも定義される）
REUSE	手続きの実行により、値が露出使用される可能性あるリージョン（定義より前に使用される）
RLIVE	手続き呼び出し元で、値が使用される可能性のあるリージョン

手続き呼び出しを含むループに対して，呼び出し先手続きに対する上述のリージョン情報を用いて，並列化判定を行なう方法については[4]を参照のこと。

3．　解析方法

3．1　解析部の構成

手続きを含むファイルが書き換えられた場合，再解析する対象を最小限に抑えて，インクリメンタルな解析が可能なように，解析部は手続き内の解析を行なうモジュールアナライザと手続き間にまたがった解析を行なうプログラムアナライザに分離した。

また，解析自体を変数解析と，配列解析に分離した。この理由は，変数解析で求めた手続き間定数伝播情報を配列解析で利用することにより，配列解析の精度を向上させるためである。

以上により，解析部は，図2のように，4つの部分から構成される。本稿では，このうち，配列解析の部分のみについて報告する。（手続き間変数解析については，[3]で発表済みである。）

Prototyping of Parallelization Assist System "Parassist", - Program Analysis Method -
Takayoshi Iitsuka*, Kiyomi Umehara*, Kazuyuki Kimura**, Takashi Kurosawa**
*Central Research Lab., Hitachi Ltd., **Hitachi Touhoku Software Ltd.

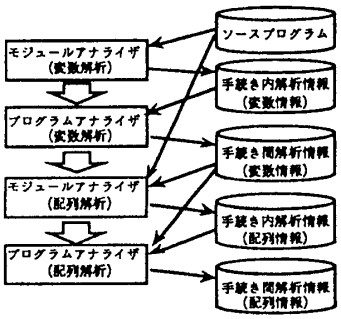

図2　解析部の構成

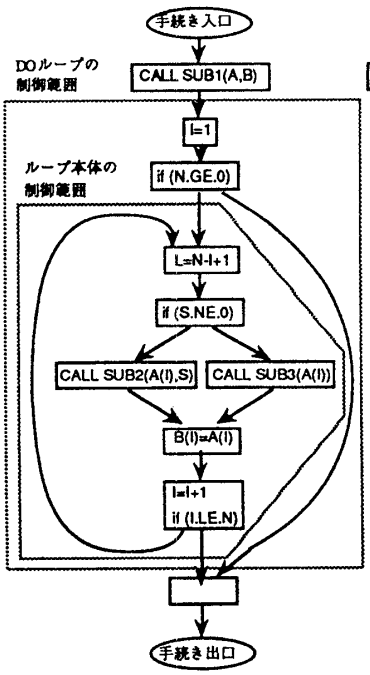

図3 a　フローグラフ

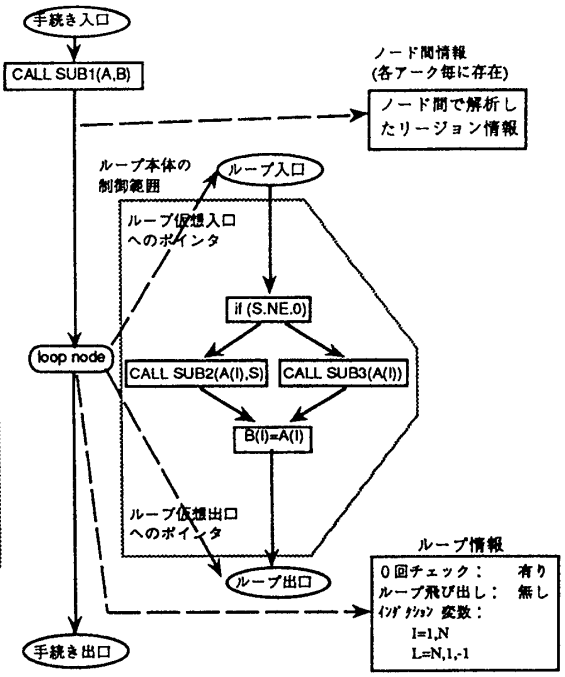

図3 b　リージョンプログラムサマリグラフ

3．2　リージョンプログラムサマリグラフ

リージョンについてフロー依存な手続き間解析を行なうために，変数についてのプログラムサマリグラフ（ＰＳＧ）[5]を拡張したリージョンプログラムサマリグラフ（ＲＰＳＧ）を考案し，これを用いて解析することにした。ＲＰＳＧは，ＰＳＧに，手続き呼び出し点（ＣＳ）を含むループの情報を追加し，各ＣＳ間のアークの情報として，アークに対応するフローグラフ上のパス上でのリージョン情報を付加したものである。解析の容易さを考慮して，ループは１つのＲＰＳＧノードにまとめ，ループ内のＲＰＳＧはループＲＰＳＧノードの付加情報として表現する階層的な構造を持つ（図３ａ，図３ｂ参照）

モジュールアナライザでは，各手続き内のＣＳ間の制御フローおよびＣＳ間のリージョン情報をＲＰＳＧに要約し，手続き内解析情報として出力する。プログラムアナライザでは，全手続きのＲＰＳＧ情報を元に，プログラム全体にわたるリージョン情報を計算する。

3．3　データフロー方程式

RMOD, RUSE, RKILL, REUSE, RLIVEのそれぞれのリージョン情報に対するデータフロー方程式は変数についての手続き間MOD, USE, KILL, EUSE, LIVEのものと[3]とほぼ同様である。

ただし，ＣＳがループ内にある場合は，ＣＳでのリージョンがループの実行に応じて変化するので，変数の場合と異った処理が必要である。

RMOD, RUSE, RKILLについては，ループ１回当たりのリージョン情報中のループ制御変数を，ループの制御変数の範囲で置き換えることにより，ループの実行に応じてリージョンを拡張する。

REUSEについては，ループ１回目とループ２回目の露出使用リージョンを求め，これらの和にループ３回目以降の露出リージョンを加える。ループ３回目以降の露出リージョンは，ループ３回露出しているリージョンをループ３回目以降のループの実行に応じて単に拡張することにより近似している。これは，リージョンの拡大近似であり，REUSE情報の安全サイドでの近似である。

4．　おわりに

フロー依存な手続き間配列データフロー解析について報告した。今後，これを用いた並列化変換の評価を行ない，ﾌｨｰﾄﾞﾊﾞｯｸしていきたい。

[参考文献]

[1]菊池ほか"並列化支援システム[Parassist]の試作　- 機能と構想 -"情報処理学会第44回全国大会(1991)

[2]R., Triolet, et.al., "Direct Prallelization of Call Statements", Proceedings of the ACM SIGPLAN '86 Symposium on Compiler Construction, pp.162-175

[3]木村ほか"手続き間解析機能の検討"情報処理学会第41回全国大会(1990)

[4]黒澤ほか"並列化支援システム[Parassist]の試作　- 手続き間並列性解析方法 -"情報処理学会第44回全国大会(1991)

[5]D., Callahan, "The Program Summary Graph and Flow-sensitive Interprocedural Data Flow Analysis", Proceedings of the ACM SIGPLAN '88 Conference on Programming Language Design and Implementation, pp.47-56