2Q-7

並列処理記述言語ＰＡＲＡＧＲＡＭにおける並列性検証

出本学　山本富士男　㈱日立製作所・中央研究所

１．はじめに

数値シミュレーション向き並列処理記述言語であるＰＡＲＡＧＲＡＭ[1]は手続き型の言語であるために、ユーザは誤って計算結果の再現性の無い現象であるハザードやデッドロックを引き起こすプログラムを記述する可能性がある。そのため、記述されたプログラムの並列性の正当性を検証する必要があり、それを行うために並列性解析を行う。並列性解析には、ソースプログラムに対する静的解析とプログラム実行時の情報を用いる動的解析があるが、本報告では静的解析のハザードチェック法について述べる。

２．ハザード(Hazard)

ハザードは並列実行関係にある実行文の間で同一変数の値を使用・定義、定義・定義する時に、使用と定義の実行順序が不確定になることで計算結果の再現性が無い場合を言う（図１）。ＰＡＲＡＧＲＡＭには、並列実行部を指定するための並列実行文としてpcalc for（同種手続きの並列実行）とpcalc case（異種手続きの並列実行）があり、その並列実行文の並列ブロック（並列実行の処理単位）内の文の間でハザードが発生する可能性がある。なお、スカラ変数および添字無しアレイ変数は図１のようにハザードとなる場合が明らかなので、以後は添字付きアレイ変数におけるハザードを対象とする。

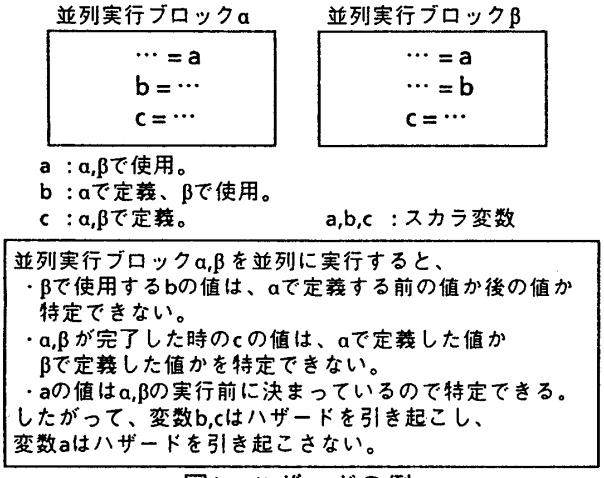

図1　ハザードの例

３．ハザードチェック法

並列実行文をループ文に置き換えるとハザードチェックは、依存関係グラフ[2]のフロー依存、逆依存、入力依存、出力依存のうち、ループブロックに渡る入力依存以外の依存関係の有無を求めることに相当する（依存関係が存在する場合がハザード）。そこで、互いに並列実行の関係となる並列ブロックの間で同一名称へのアレイ変数の参照関係が使用・定義、定義・定義である場合に、アレイ変数の添字が一致する場合の有無を求めることでハザードチェックを行う（存在する場合がハザード）。なお、逐次型言語の自動並列化では多重ループの場合は特定のループについてのみ並列化可能性を調べればよいのに対し、ハザードチェックでは多重並列の場合は全ての並列実行文について調べる必要がある。

現時点のハザードチェックの主な機能を以下に示す。

機能１：アレイ変数の添字式がインデックス変数の線形式で、インデックス変数の範囲が定数値の場合に添字比較を行う。

機能２：インデックス変数の範囲が定数値でない場合にアレイ変数の要素範囲を用いてインデックス変数の範囲を予測する。

機能３：send/receiveによる同期関係を含む場合のハザードチェックを行う。

機能４：ＰＡＲＡＧＲＡＭのサブアレイ変数の構造を考慮する。

これらのうち、ここでは基本的な機能である機能１の添字比較のアルゴリズムについて説明する。

４．添字比較アルゴリズム

添字比較アルゴリズムの概要とその適用例を以下に示す。なお、本アルゴリズムの適用には、アレイ変数の添字式はインデックス変数の線形式で、インデックス変数の範囲は定数値という制限を置く。（並列ブロックの制御変数を並列ブロック変数、並列ブロック変数とループ変数を合わせてインデックス変数と呼ぶ）。

Validity Checking Method for Palrallel Description Language PARAGRAM

Manabu DEMOTO , Fujio YAMAMOTO

Central Research Laboratory, HITACHI Ltd.

(1)比較する添字式の値が一致するための条件式を立てる。ただし、同一インデックス変数が両方の添字式に存在しても別の変数として扱う。
(2)全ての並列ブロック変数について(3),(4)の処理を行う。
(3)着目した並列ブロック変数の値が異なり、並列ブロックより外側のインデックス変数の値が一致する条件を(1)の式に付与する。
(4)(3)に対し添字値が一致する場合が存在するかを求め、１つでも存在するならハザードと判定する。

このアルゴリズムを下記のプログラム例に適用する。

```
iter for I in [0:4];
 pcalc for J in [0:5];
  pcalc for K in [0:6];
   A(5*I+4*J+3*K+7)=A(4*I+3*J+2*K-6);
  end pcalc;
 end pcalc;
end iter;
```

(1)より代入文の左辺と右辺の添字式をα，βとする。

$\alpha=5*I+4*J+3*K+7$

$\beta=4*I'+3*J'+2*K'-6$

添字式の値が一致する条件式

$\alpha-\beta=5*I-4*I'+4*J-3*J'+3*K-2*K'+13=0$

($0\leqq I\leqq 4, 0\leqq J\leqq 5, 0\leqq K\leqq 6, 0\leqq I'\leqq 4, 0\leqq J'\leqq 5, 0\leqq K'\leqq 6$ I,J,K,I',J',K'は整数)を立てる。

並列ブロック変数Ｊで(3)より I=I',J≠J' とおくと

$\alpha-\beta=I+4*J-3*J'+3*K-2*K'+13=0$ (a)

となり、これを満たす J,J' の値が存在するかを調べる。まず、J,J' に注目した式を立てる。

$4*J-3*J'=-I-3*K+2*K'-13$ (b)

(b)に注目し、I,K,K'の変域より右辺の値域を求める。

$-35\leqq -I-3*K+2*K'-13\leqq -1$ (c)

(b)で J≠J' より J>J' 時の左辺の値域を求める。

J=J'+C ($1\leqq J\leqq 5, 0\leqq J'\leqq 4, 1\leqq C\leqq 5-J'$)を用いると、

$4\leqq 4*J-3*J'=J'+4*C\leqq 20$ (d)

同じく J≠J' より J<J' 時の左辺の値域を求める。

J'=J+C ($1\leqq J'\leqq 5, 0\leqq J\leqq 5, 1\leqq C\leqq 5-J$)を用いると、

$-15\leqq 4*J-3*J'=J-3*C\leqq 1$ (e)

(c),(d),(e)より、(b)の両辺の値が一致する場合があり、(a)を満たす J,J' の値があり得るので、ハザードが起きる可能性がある。

次に並列ブロック変数Ｋにて(3)より I=I',J=J',K≠K' とおくと

$\alpha-\beta=I+J+3*K-2*K'+13=0$ (f)

となり、K,K' に注目した式を立て、以下同様にすると(f)式を満たす K,K' の値が存在しないのでハザードは起きないことがわかる。

よって、プログラム例の代入文は並列ブロック変数Ｊの並列実行文でハザードが起きる可能性があることが分かる。

５．実応用プログラムへの適用結果

静的の並列性解析をＰＡＲＡＧＲＡＭで記述した実応用プログラムのデバッグ時に適用した結果を表１に示す。表１は、２つの実応用プログラムに対し、行数と並列実行部の数、解析できた並列実行部の数、解析に使用した機能を示している。適用結果より、合計１０３個の並列実行部のうち９４個の並列性の妥当性を検証できた。そのうち１１個をハザードと判定してデバックに寄与できた。

６．おわりに

ＰＡＲＡＧＲＡＭの並列性を検証する並列性解析のハザードチェック機能の中核である添字比較アルゴリズムを示し、その適用結果を示した。今後は、ハザードチェックの適用範囲を広げるために、インデックス変数の範囲を上位インデックス変数の線形式にするなどの機能拡張を行う。

なお本研究は、通商産業省工業技術院大型プロジェクト「科学技術用高速計算システムの研究開発」の一環として実施されたものである。

[参考文献]

[1]山本,梅谷,出本:"並列処理記述言語ＰＡＲＡＧＲＡＭ",信学論(D),Vol.J71-D,No.8,(1988),pp.1407-1414
[2]小原和博:"階層構造のＭＩＭＤ型スーパコンピュータ",情処Vol.25,No.5,(1984),pp.480-490

表1　実応用プログラム解析結果

		プログラムA	プログラムB
プログラム行数		1667	4133
並列実行部数*1		33	70
解析可能並列実行部数*2		33 (並33,ハ0)	61 (並50,ハ11)
使用機能	機能1	33	55
	機能2	0	31
	機能3	0	0
	機能4	0	0

*1: 多重並列の場合は、並列実行部は1つと数える。
*2: 並は正常な並列性、ハはハザードと判定されたことを示す。