9F−1　並行代入機能を持つ並列プログラミング言語の設計と実現

前田雅之　飯塚　肇
成蹊大学

1．はじめに

近年各種の並列計算機が実用化されているが、そのソフトウエア環境はまだ不十分である。本稿では、「代入文が複数同時に発生する」という概念で記述するモデルに基づく並列プログラミング言語の実現について述べる。

これまでの並列計算機用のプログラミング言語は、多くの場合並列動作するプロセスを導入する方法が取られている。この場合プロセス同士での情報交換はメッセージ通信か共有変数への代入で行われる。どちらの場合もハードウエアアーキテクチャをベースとしたモデルである。そのためハードウエアに依存した概念が言語上へ現れる。またプロセスの生成、消滅、同期といったアルゴリズムとは直接関係のない問題をプログラマ自身で解決する必要があった。これを解決するにはアーキテクチャと独立し、かつ各種アーキテクチャでの実行可能なモデルに基づいた言語が求められる。

2．　並列性

2．1　同時代入

プログラムはメモリ上のあるアドレスへ格納されているデータに定められた操作を行い、定められたアドレスへ格納することを繰り返すことで実行される。これらはプログラミング言語では代入という形で表現される。この操作が同時に起こる事が並列と考えられる。よってプログラミング言語では代入が同時に起こるという事を指示する事によって並列性が記述出来る。

2．2　UNITYモデル

上記の考え方に基づくモデルとしてはChandyとMisraのUNITYモデル[1]がある。これは並列アルゴリズム記述のために提案された並行代入を基本としたモデルで、以下の二つの要素にその特徴がある。

1）並行代入

UNITYではプログラミング言語における並列性の記述を、並行代入（concurrent　assignment）によって行う。x,y := y,x;のような記述で行い、この式が実行されると変数x，yの内容が交換される。

また、略記法として以下の様な並行for文がある。

```
<|| 1 <= i <= 10 ::
      a[i] := i;
>
```

この例ではa[i]に対する十の代入文が並行して行われる。

2）不動点計算

UNITYモデルでは、変数の値が変化しなくなるまで計算を繰り返すことが仮定され、そのような状態に達したとき不動点に達したという。これは通常言われている不動点計算とは異なる。例えば以下の様な記述で行う。

```
<| 1 <= i <= 9 ::
   if a[i] < a[i+1] then
     a[i],a[i+1] := a[i+1],a[i];
>
```

この例では、a[i]の内容に変化が起こらなくなったら計算が終了する。この場合aは降順にソートされる。

3．　言語設計

従来の並列プログラミング言語では、並列性を記述する際に、以下の様な問題をプログラマが自分で解決しなければならない。

・プロセスの管理
・変数へのアクセスに際する問題
・プロセス間での同期

UNITYモデルににはこれらの概念はないので、このUNITYモデルを言語へ導入する事ことでこの問題を解決する事が考えられる。本稿ではPASCALへUNITYの考え方を導入する事を試みた。

3．1　言語仕様

UNITYモデルをPASCALへ導入する際に仕様を以下のように決定した。

・UNITYモデルのうち取り扱うのは並行代入、不動点計算のみとする。
・変数宣言はPASCALと同様とする。今回は基本型として整数型、実数型、複合型として配列を利用できる事とする。また並行for文、不動点計算の為にindex変数を用意する。これは非負の整数である。
・関数は値呼出しのみとする。関数内には逐次手続きしか書けない。
・標準入力としてread、readln、標準出力としてwrite、writelnを用意する。
・二次記憶利用としてパシステントデータを導入する。パシステントデータの言語への導入は、文献[4]で述べた。

A design and implementation of parallel programing language with concurrent　assignment

Masayuki MAEDA and Hajime IIZUKA
Department of Imformation Sciences, Seikei University

3．2　プログラム例

プログラム例を示す。

```
program test
declare
   i : index;
   a : array[1..10] of integer;
assign
init
 <|| i <= i <= 10 ::
    a[i] := i;
 >
end
 <| 1 <= i <= 9 ::
    if a[i] < a[i+1] then
       a[i],a[i+1] := a[i+1],a[i];
 >
 <|| 1 <= i <= 10 ::
    writeln('a[',i,'] is ',a[i]);
 >
end.
```

図1　プログラム例

このプログラムではaの値が降順にソートされる。

4．　実現

分散記憶型計算機nCUBE2上へこの言語の実現を試みた。実現上大きな問題となるのは次の3点である。

・計算のプロセッサへの割当
・変数の値の参照
・同期

これに対して以下のような方法によって解決をはかった。

4．1　代入文の割当

各プロセッサへ大域変数を分割して割り当てる。ここでは連続する大域変数領域をプロセッサが同じ量分担するように分割を行う。自分が割り当てられた変数に対する代入操作のみ実行する。index型を持つ変数は各プロセッサで個々に持つ。前述の例1を二台のプロセッサへ割り当てた例を示す。

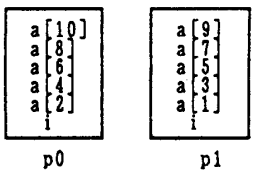

図2　変数の割当

4．2　変数の参照

上記の方式によると、他のプロセッサへ割り当てられた変数を参照する必要が生ずる。そのため、各プロセッサは1つの文の計算が終了する毎に自分の担当している変数の値を他のプロセッサへ送る。計算には1つ前の文が終了した際の値が参照されるので、それを識別する必要がある。よってプログラムの各文には先頭から絶対的な番号がつけられ、これを文番号と呼ぶ。値は文番号と一緒に他プロセッサへ送られる。他のプロセッサが担当する変数の参照は、通信されてきた値で行う。（現在計算している文番号－1）の値を参照する。前述図1のプログラムでは▒の範囲に0、▒の範囲に1、▒の範囲に2が文番号として与えられる。

4．3　同期

参照したい変数の値がまだ計算されていなければ、その値は送られていないのでそこでブロックする。よって同期は自動的に取られる。

4．4　並行代入

前述の方法による並行代入の実現を示す。

1）代入文の右辺値が自プロセッサ担当の場合。変数はそれぞれ一つ前の値を履歴として持つ。右辺の参照している変数のアドレスが自分の担当の場合はその履歴を参照する。

2）代入文の右辺値が他プロセッサ担当の場合。値は他プロセッサから送られてくるものを用いる。

4．5　不動点計算

大域変数の値が変化しなくなったかどうかをチェックする必要があるが、それは次のように行っている。

1）不動点計算の文に入る前に大域変数の値を退避する。
2）通常のfor文と同様に計算を行う。
3）for文の終了時に退避していた変数の値との変化をチェックする。
4）変化が有れば、また値を退避し、2）へ戻る。変化がなければ、終了。

5．　結び

並列計算機用言語として、UNITYモデルをPASCALへ導入したものを考え、その実現方式を示した。現在nCUBE2上で実現中である。今後はプロセッサ間の通信を減らすような変数域の割当方式、関数サーバ機能の実現なども考えていきたい。

参考文献

[1]K.Mani Chandy, Jayadev Misra:
"Parallel Program Design A Foundation",
Addison Wesley, 1988.

[2]Nicholas Carriero, David Gelernter:
"How to Write Parallel Programs:A guide to Perplexed",
ACM Computing Surveys, Vol.21, No.3, pp.323-357, 1989.

[3]Henri E.Bal, Jennifer G.Steiner,
Andrew S.Tanenbaum:
"Programming Languages for Distributed Computing System",
ACM Computing Surveys, Vol.21, No.3, pp.261-322, 1989.

[4]前田:"AnsiBASIC-永続性の導入",
データベースワークショップ論文集, pp.85-87, 1991.