4P-3

“発注／納品モデル”に基づく並列処理記述言語 P³L

須崎有康、冨澤眞樹、五十嵐智、阿刀田央一、齋藤延男
(東京農工大学)

1. はじめに

筆者らは、並列計算機の制御機構に“発注／納品モデル”[1] を提案した。これは、並列実行するテンプレート毎にキューを持ち、制御フローの分岐はこのキューに発注を積むことで実現する。発注側では、発注に対し受領するためのアドレスを用意し、発注の完了を書かせる。発注されたテンプレートはインスタンスを生成し、各々独立の環境を割当て実行する。インスタンス内で並列手続きの結果待ちや共有資源の取合いが起こるとディスパッチを起こしてキューに積まれている発注に対し新たにインスタンスを生成するか、今まで結果待ちのまま止まっていたインスタンスを再実行させる。インスタンス終了後は完了を発注者に納品して、割当てられていた環境を解放し、キューに積まれていた発注を取り除く。

この発注／納品モデルに基づいて、並列計算機の動作を抽象化した手続型言語P³L(Parallel Process Programing Language)を作成した。この並列言語は既存のC言語を土台にし、関数を並列関数と逐次関数に分けた。並列関数の実行は並列関数の呼出し時に制御をフォークし、実行結果は必要になったときにまでに返されればよい。引数の渡し方は、並列関数の呼出し側が引数渡し用のフレームを作る。呼ばれた並列関数はこのフレームから引数を取出し、実行結果もこのフレームを介して渡される。この関数呼出しを“前倒し呼び(advance call)”と呼ぶ。

並列関数呼出しに前倒し呼びを用いることにより、プログラムが物理的な制約なしに書けるようになった。この抽象化されたプログラムを実行に移すには各々の並列関数を割当てるプロセッサを指定しなければならない。また、並列関数を実行時に複数用意して人海戦術的に処理を進める指定をすることで並列度の抽出を高め高速処理が期待できる。

2. 並列関数呼出し

言語P³Lでは、関数宣言・関数定義において、記憶クラス指定子“template”をつけることにより並列関数と逐次関数との識別を行う。並列関数のインスタンスの識別には、インスタンス名をつける。インスタンスの名前を受取る型として型指定子“tag”を用いる。並列関数のインスタンス名を付けられたタグは並列関数の実行結果の受取り時に引換え札として用いることで並列関数の実行結果を受取ることができる。実行結果を受取った時点でタグの名前付けが解放される。このtag型変数には並列関数の状態が入り、この状態にはインスタンスの名前付けがされていないempty状態と名前付けがされているfull状態を取る。full状態はさらに、並列関数の実行未終了であるexecute状態と実行終了であるcomplete状態をとる。状態を変える手続きとして、名前付けするときはtag変数に単項演算子“¥”を付けfull状態に変え、結果の受取りのときはtag変数に単項演算子“%”を付けempty状態に変える。execute状態からcomplete状態への変化は並列関数の未終了・終了によって変化する。この変化はシステムのディスパッチで使用される。以下に前倒し呼びによる並列関数の例を示す。

言語P³Lによる制御例

テンプレート1

```
main()
{
    template int func();
    tag a;
    int b;
    ...
    ¥a   func(); /* 名前付け */
    ...
    b   %a;      /* 結果の受取り */
    ...
}
```

テンプレート2

```
template int func()
{
    int a;
    ...
    return a;
}
```

Parallel Process Programing Language P³L based on “DEMAND/DELIVER Model”

Kuniyasu SUZAKI , Masaki TOMISAWA , Satoshi IGARASHI , Oichi ATODA , Nobuo SAITO
Tokyo University of Agriculture and Technology

3. 変数の渡し方

言語P³Lにおいて並列関数の引数渡しは、呼出し側に引数渡しの用のフレームをつくり、このアドレスを渡す。引数の内容は、C言語の値渡しとアドレス渡しのほかに結果渡しを用意した。呼出し側での書式は

func(a, &b, $c)

と書き、単項演算子なしは値渡し、単項演算子"&"はアドレス渡し、単項演算子"$"は結果渡しを意味する。並列関数においてアドレス渡しを行うとき、呼ばれた並列関数側で書換えを行うと、いつ書換えが起こるか呼出し側では判らないので、その値を参照するときに書換え前の値か書換え後の値か判らない。結果渡しでは、書換えする時を関数の結果を受取り時に特定する。このようにすることで、参照する値を書換え前の値に特定することができる。

4. 実行制御ファイル

実行にあたって並列計算機では、プロセッサの台数によって実行形態が異なる。言語P³Lでは実行する計算機の物理形態にこだわらずにプログラミングできるが、実行させるためには計算機別にコードを作らねばならない。計算機別にコードを作るために、計算機の物理形態情報が必要である。これにはプロセッサの数や実装アドレスや発注／納品モデルのキューの数などある。さらに、並列関数をどのプロセッサに配置するか、並列関数のクローンを作り人海戦術的に処理を行うかなどの実行方法を指定することにより、目的プログラムが効率よく実行することができる。これらの情報を記したものを実行制御ファイルと呼び、コンパイラはこのファイルをもとにコードを作成することができる。プロセッサ4台の例を以下に示す。

実行制御ファイル例

```
ﾌﾟﾛｾｯｻ.ｾｸｼｮﾝ          /* 並列計算機の物理情報…プロセッサの数、実装アドレス、キューの数 */
    ﾌﾟﾛｾｯｻNO      ﾒﾓﾘ        ｷｭｰ
       1          200000      10
       2          220000      15
       3          240000       5
       4          260000       5
ﾌｧﾝｸｼｮﾝ.ｾｸｼｮﾝ       /* 並列関数の配置プロセッサ情報 */
    関数名      配置ﾌﾟﾛｾｯｻ
    main        PE1             /* main関数をPE1に割当てる */
    funca       PE2             /* funca関数をPE2に割当てる */
    funcb       PE2, PE3, PE4   /* funcb関数をPE2, PE3, PE4に割当て人海戦術処理をする */
```

5. その他の機能

言語P³Lでは上記で述べた並列関数を支援する機能として、各々のインスタンスごとにスタティックなistatic記憶クラス指定子、テンプレートから生成されたインスタンス間でスタティックなtstatic記憶クラス指定子、共有資源の取合いを制御するlock文、複数のインスタンスから演算が終わったものの1つを選択するamong文などを付加した。これらの機能によって、並列関数で用いる変数管理や動的な実行制御が可能となった。

6. まとめ

言語P³Lを作成することにより、いままでのC言語の関数による発注／納品モデルの実現のような物理的制約を気にすることなしにプログラミングすることができるようになった。また、言語の記述レベルと実行レベルを分離することにより、並列計算機の物理形態によらず、同一プログラムの実行が可能となった。さらに、制御ファイルのみを変更することで最適実行な並列関数の配置、クローンの配分を求めることができる。

参考文献

(1) 富澤、五十嵐、阿刀田、齋藤：“手続きフロー型並列計算機における分散型組込み制御機構”，信学論(D)，J71-D，10，pp1921-1930(昭63-10)