# 並列処理に対する一つの試み

吉村 晋
（東京芝浦電気(株) 総合研究所）

## 1. はじめに

計算機が出現して以来，並列処理は比較的古くから研究されて来た。<1,4>
並列処理とは，基本的には複数個あるいは単一個の処理をいくつかの並列に処理可能な部分に分解して，これを複数台のプロセッサで処理することである。　このような方法は，計算機以外の分野においては，すでに広く用いられている方法である。
近年，超LSI技術の進歩，汎用マイクロプロセッサの実用化に伴い，これらの組合せからなる並列処理が着目されつつある。<2,3>
他方今までの特殊用途の並列処理システムが高価な割に応用範囲が限定されていることから，柔軟な並列処理システムへの要求が強い。
そこで，我々はコストパフォーマンスの良い並列処理の為に次の事柄を目標に研究開発を行っている。

(1) 並列処理に適した応用ソフトウェアのための基本アルゴリズムの研究開発。

(2) 多様な処理対象に合った効率の良い並列処理ハードウェアと基本ソフトウェアの研究開発。

これにより並列処理技術の適用範囲の拡大をめざしている。
本報告では，柔軟な並列処理システムに対する一つの考え方を示すと共に，この一つの具体例として高精度な固定小数点数の計算処理，正確な計算処理に対する並列処理システムの適用について述べる。

## 2. アーキテクチャ

我々のシステムは，並列演算マシンPARACOM（PARAllel COmputing Machine）と呼ばれ，2種類のアレイプロセッサとマルチマイクロプロセッサから構成される。
全体として柔軟な並列処理システムを実現している。
マルチマイクロプロセッサは，並列プログラム処理実験装置<8>と呼ばれ，16台のマイクロコンピュータTosbac－40Lのワンボードcpuを使用している。
このシステム構成を図1に示す。

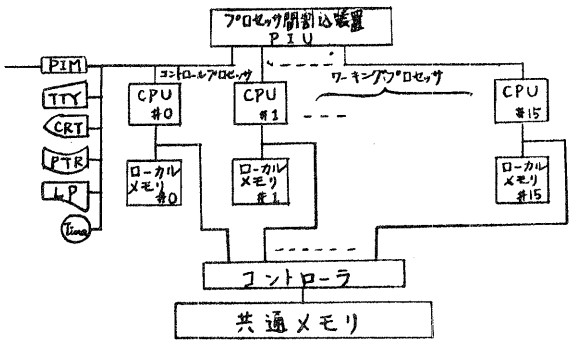


図1．並列プログラム処理実験装置の構成

cpuには，0～15の番号が付けられる。　0番のプロセッサをコントロールプロセッサと呼び，他の15台のcpuをワーキングプロセッサと呼ぶ。

16台のプロセッサは，それぞれ

32K Byteのローカルメモリが接続される。
各CPUのアドレス空間の最初の32K Byteがローカルメモリに割り当てられる。
共通メモリは，32K Byte記憶容量を持つ16ポートのメモリである。

プロセッサ間の通信のためにプロセッサ間の通信は，プロセッサ間割込み装置（PIU）及び共通メモリのMailboxで行われる。

ソフトウェア <7,8,9>

(1) プログラムの動作方式
この実験装置で実行される並列処理のためのプログラムは，一つのprogramと，いくつかのprocessより構成される。 programは主としてプログラム全体の制御を行う部分で，これは必ずコントロールプロセッサによって実行される。 processはprogramがシステムマクロactivateをコールした時に起動されるタスクで，ワーキングプロセッサで実行される。 通常は並列処理のために同じprocessが別々のローカルメモリにローディングされ，同時に実行されるが，processは一種類とは限らないので，異なるprocessが，異なるCPUで同時に実行されることもある。

このプログラムの実行を制御する為の並列処理オペレーティングシステムは，シンプルであるだけでなく並列処理プログラムの形式とうまく合致している。

(2) 並列処理言語（PPL）
並列処理のための言語は，並列処理システムのアーキテクチャとの適合性に重点を置いて考案されたものと，汎用性に重点を置いた高級言語の二種類からなる。 最近は高級言語的なアプローチが多い。 しかし，基本は並列処理に適したアルゴリズムの実行にあり，システムアーキテクチャとの適合性が重要である。 PPLは，システムアーキテクチャとの適合性に重点を置いているが，構造プログラミングに適し，レジスタ管理等をユーザに開放した形式となっている。
PPLでは，並列処理プログラムはデータ宣言部，process部およびprogram部よりなる。 データ宣言部はprogram部やprocess部の中で使用されるデータのうち，共通メモリにおくものを定義する部分である。
process部では，ワーキングプロセッサで実行されるプログラムの手続きと，ローカルメモリに置かれるデータの記述を行う。 program部では，コントロールプロセッサで実行されるプログラムの手続きと，そのローカルメモリに置かれるデータの記述を行う。

## 3. 柔軟な並列処理への考え方

従来，並列処理は，特殊の超高速処理システムの代名詞のように言われる事が多い。 しかし並列処理は，あくまで直列処理に対する相対的な概念である。 従って一般的な処理業務から工業プロセスのあらゆる分野において潜在的な要求を持たれている。しかも一般業務において効果が生じるものの大部分は，単純な形の処理形式で良い。

### 一時的な負荷の吸収

事務管理，販売管理においては，月末，年度末等に特に処理業務が増大す

る。　これが汎用機の本来の計算業務を圧迫する場合が多い。

これらの計算業務の多くは，並列処理可能なものが多い。<4>　このように一時的な負荷を吸収する為の効果的な手段の一つが，並列処理である。

## コストパフォーマンスの考え方

従来は，CPU，Memoryが周辺装置に対して高価であった。　現在では，これらは単なる素材部分となっている。　従ってCPU　メモリの冗長性は，コストアップ要因とならない時代が来ていると言って良いであろう。　この場合，高価な大型機の場合を除き，性能の考え方を考えることが重要となる。　逆に信頼性の向上や，部分的な用途に対する能力の可変性が可能となる点から，付加価値が増す。

## 並列化の方法　（直接法と間接法）

並列処理に対して，ハードウェア，基本ソフトウェアが優れているだけでは効果的な能力を得ることが困難な場合が多い。　その為には，問題，処理アルゴリズムの並列化の方式が重要になる。

通常行われている並列化の方式では，並列向きのアルゴリズムを選び出し，それを直接に並列に実行可能な部分に分解して実行させる方式が最も多い。　このような直接法では，同期及び並列に実行できない部分のオーバヘッドが無視できない場合が多い。

他方，統計計算における最尤度法分類のように，同一の処理をn回繰り返す場合は，問題自体に並列性を有している為に好ましい。

この考え方を一歩進めて，問題対象のベース自体を並列性を有するベースへ変換することにより，結果として並列処理を実現する方法が考えられる。

例えば，基本的算法自体に内包する並列性を利用して並列処理を実現する方法である。　この算法の適用できる範囲であれば，直接の問題，アルゴリズムに依存せずに並列化可能である。　我々は，これを間接法と呼んでいる。

このような方法は，今までほとんど考慮されてはいなかったが，今後その重要性が大になるものと思われる。

最近データフローマシン<14>への試みが多いが，問題自体の並列性がネックとなる。　従ってこの考え方は，データフローマシンにおいても有効となろう。

## 4. Residue Arithemetic <11,12,13,15>

計算機の計算の精度，桁数は，その計算機により定まっている。

しかし，多重精度計算の場合は，桁数の多い数を分割し，多重精度のサブルーチンを使って計算されるが，これは大量の計算量が要求される。

多重精度の計算に対して，簡単化する方法がある。　これはResidue Arithmetic（またはModular Arithmetic）と呼ばれる。

この計算法に関しては，高橋<11>，N.S. Szabo <12>等により，1960年代に研究され，近年においても各種の試みがなされている。　特にこの計算法と並列処理との関連については，D.E. Knuth <13> に述べられている。

## 原理

Residue Arithmeticは，大きな整数（固定小数点数）の演算を行う場合

に利点を持つ。これはChinese Remainder Theoremと呼ばれる整数論上の定理に基礎を置いた方法である。

互いに素な正整数 $P_1, P_2, \cdots, P_r$ があらかじめ与えられる。
大きな整数 $u$, $v$ に対して,

$$u = (u \bmod P_1, u \bmod P_2, \cdots, u \bmod P_r) = (u_1, u_2, \cdots, u_r)$$

$$v = (v \bmod P_1, v \bmod P_2, \cdots, v \bmod P_r) = (v_1, v_2, \cdots, v_r)$$

となるrタップルが対応する。これを Residue 表現という。この時, $u$ と $v$ の演算を直接行うのでなく, rタップルの各要素単位で演算を行えばよい。

$$(u_1, \cdots, u_r) \circledcirc (v_1, \cdots, v_r) = ((u_1 \circledcirc v_1) \bmod P_1, \cdots, (u_r \circledcirc v_r) \bmod P_r)$$

ここで, ◎: $+, -, *$ 。
この計算で扱いうるのは, 加, 減, 乗の3種類の演算である。しかし除算が含まれた計算であっても, 結果が最終的には割り切れて, 整数の答が出ることがわかっている場合に限り可能である。
除算が正統化される為には, 除算の一意性が保証される (Eulerの定理, Fermatの定理) 為に $P_i$ が素数であることが要求される。

基本的処理 (高橋, 石橋 <11,>)

(1) modulo $P_i$ での簡約
各演算は, mod $P_i$ の下で行われる。
この簡約処理は, 除算命令によってなされる。

(2) 逆数計算
residue arithmetic では, 除算 (制限がある) は, 逆数を掛けることによって行われる。
各 $P_i$ において, $v_i$ に対し

$$u_i y_i \equiv 1 \pmod{P_i}$$

なる $y_i$ を $u_i$ の逆数と言う。素数 $P_i$ に対しては, Fermat の定理により,

$$y_i \equiv u_i^{P_i - 2} \pmod{P_i}$$

によって求める。<11>

(3) Residue表現と通常の多重精度表現
多重精度 (高精度) の固定小数点数の計算機内部での表現には, 色々の形態があり, どちらがすぐれているとも言えない。しかし, 我々が日常使用している表現の方が有利であると考える場合が多い。
この場合, Residue表現と通常の多重精度表現との変換が必要となる。
この変換は, 以下のようになされる

<多重精度表現> ⇒ <Residue表現>
$u$ に対し, 各 $P_i$ で mod $P_i$ を求めることによりなされる。

<Residue表現> ⇒ <多重精度表現> <11>
$(u_1, u_2, \cdots, u_r) \rightarrow u$

$$0 \leq g_i < P_{i-1}$$

$$g_i \equiv (y_i - Y_{i-1})(P_1 P_2 \cdots P_{r-1})^{-1} \pmod{P_i} \quad \cdots (1)$$

（ここで，$(\quad)^{-1}$ は mod pi に関する逆数。）

これにより，uは次のようになる。

$$u=(((\cdots(g_r P_{r-1}+g_{r-1})P_{r-2}+g_{r-2})P_{r-2}+\cdots)P_1+g_1 \qquad \cdots(2)$$

ここで $g_1(=u_1)$，$g_2$，$g_3$，$\cdots g_n$ は(1)式により求められる。

この為には，多重精度の数と単一精度の数の乗算，および多重精度の数のmodulo Pによる簡約が必要となる。

特長

Residue Arithmetic の特長は，多重精度の計算を，単精度の計算処理に置き換えることができる点にある。

- 多重精度の乗算に要する計算量が減る。
- データのロード，ストアに要する不必要な時間が減少する。
- 各要素の計算は，互いに独立に実行可能。

注意を要する点は次のとうり

- overflowの判定が困難。
- 割算に制限がある。
- residue表現と通常の多重精度表現との変換が必要。

Residue表現で表しうる数の範囲は次のとうりである。

$$a \leqq u < a+P$$

ここで，$P=P_1P_2\cdots\cdots P_r$，　aは任意の数である。

aの定め方で色々な範囲を設定できるが，通常は，$a=0$（非負の最小剰余系），又は $a=-\frac{(P-1)}{2}$（絶対最小剰余系）とすることが多い。　これらは互いに同等であり，ハードウェア，処理系の都合の良い方を選択すれば良い。

## 5. Residue Arithmetic の並列処理

Residue Arithmetic の並列処理については，これ以上述べる必要はないであろう。

また原則的には，この算法が適用できる範囲であれば，問題やアルゴリズムに依存せずに効率的な並列処理が可能になる。

通常の場合，各々の要素の計算を各ワーキングプロセッサが担当するが，2つ以上の要素をワーキングプロセッサに担当させることも可能である。

我々は，16 bit プロセッサ（Tosbac 40L）を使用しているので，modulo $P_1\cdots\cdots P_r$ は $2^{15}$ 以下の素数を最大のものから順次選んでいる。

従って，

$P_1 \sim P_5$ で約 $\pm 1.8\times 10^{22}$
$P_1 \sim P_{10}$ で約 $\pm 6\times 10^{44}$
$P_1 \sim P_{15}$ で約 $\pm 2\times 10^{67}$

程度の範囲の計算をカバーする。

我々は，マルチマイクロプロセッサ上で，これらのための基本および応用プログラムを用意している。

もちろん，この算法と問題自体の並列性との融合および選択も可能であり，全体として柔軟な並列処理が実現され得る。

## 6. おわりに

本研究は，通産省の大型プロジェクト（パターン情報処理システム）の一環として行われたものである。

最後に，本研究に対してご協力いただいた日科技研 王 義孝氏，芝浦システム 岡林典明氏，ご指導，ご助言いただいた東芝総研 南川忠利氏，徳島大学 高橋義造教授に感謝する。

## < References >

(1) P.H.Enslow, Multiprocessor Organization — A Survay, Computing Surveys, Vol.9, No.1 (March 1977)

(2) W.A.Wulf, C.G.Bell, C.mmp-A Multi-Mini-Processor, AFIPS Conf. Proc.FJCC, Vol.41 (1972)

(3) S.H.Fuller, etal, Multi-Microprocessors: An Overriew and Working Exanple, Proc.IEEE, Vol.66, (1978)

(4) 村岡洋一訳（P.H. Enslow, JR），マルチプロセッサと並列処理，近代科学社，(1976)

(5) 金田悠紀夫，並列システムによる楕円形偏微分方程式の数値計算法，情報処理，Vol.16，No2 (1975)

(6) 金田悠紀夫，並列処理システムによる線型計画計算と数種行列の三重対角化計算，情報処理，Vol.19，No.1 (1977)

(7) 高橋義造，吉村 晋，並列プログラム処理実験装置とそのOS，情報処理学会計算機アーキテクチャ研資，31-1 (1978. 6)

(8) 高橋義造，吉村 晋，並列プログラム処理実験装置，情報処理，Vol.20，No.4 (1979)

(9) 吉村 晋，高橋義造，並列処理言語とそのクロスコンパイラ，情報処理学会 第20回 全国大会 (1979)

(10) 吉村 晋，高橋義造，マルチミニプロセッサによる並列処理，昭和54年度電子通信学会総合全国大会 (1979)

(11) 高橋秀俊，石橋善弘，電子計算機による exact な計算の新方法，情報処理，Vol 1，No.2 (1960)

(12) N.S.Szabo, R.I.Tanaka, Residue Arithemetic and its Applications to Computer Technology, Mc Grow-Hill, Inc. (1967)

(13) D.E.knuth, Seminumerical Algorithms, Addison Wesley Pub. (1969)

(14) J.B.Dennis and D.P.Misnas, A Preliminary Architectuer for a Basic Data-flow Processor, Proc. of the Second Annual Symp. on Computer Archtecture, IEEE, New York, (Jan. 1975)

(15) H.Takahashi and Y.Ishibashi, A New Method for "Exact Caluculation" by a Digital Computer (An Application of Modulo P Arithmetics), J.Information Processing Soc. Japan Vol.1 (1961)