論　文

UDC 681.322.012-185.4

超高速科学技術計算向きデータフロープロセッサアレイ計算機のアーキテクチャ

正員　高橋　直久†　　正員　雨宮　真人†

A Data Flow Processor Array System for Large-Scale Scientific Calculation

Naohisa TAKAHASHI† *and* Makoto AMAMIYA†, *Regular Members*

あらまし　本論文では，超高速科学技術計算を指向したデータフロープロセッサアレイ計算機の構成と特性評価について述べている．この方式では，プロセッサ間転送制御とプロセッサ内の実行制御をデータ駆動概念で統一して高度の非同期並列処理を実現することをねらっている．方式上の特徴は，通信の局所性を保つように結合系を構成してデータ転送遅延を小さくするとともにハードウェアコストを低くしたこと，及び，実行時の適応的な資源割付と実行前の静的な資源割付とを組合わせた柔軟性が高くオーバヘッドの少ない資源割付機構としたことにある．偏微分方程式の解析プログラムの実行シミュレーション結果により，この方式がプログラムの持つ並列性を十分に引き出せることを示している．また，プロセッサ間の転送遅延が性能に与える影響が小さいことを明らかにしている．

1. まえがき

最近，大規模科学技術計算を行なうために，現在の高性能計算機のもつ性能をはるかに上回る数GFLOPS（10^9 Floating point operations per second）の超高速計算機の必要性が強く叫ばれている．このような要求に応えるためには，超高速素子の開発とともに並列計算機アーキテクチャの開発が必要である．

並列計算機としては，これまで，科学技術計算でのデータの従属関係が局所的かつ規則的である点に着目してSIMD（Single instruction stream multiple data stream）型計算機が開発されている[1]．SIMD型計算機では，非局所データの参照や規則性の乱れなどが演算器の稼動率に敏感に影響するので性能低下が深刻となる場合がある．また，同一操作を多数のデータに対して施すような構造のプログラムを記述しなければならないことがプログラム作成を難しくしている．

一方，システムの柔軟性を高めて規則性の乱れを吸収する方式としてマルチマイクロプロセッサ構成のMIMD（Multiple instruction stream multiple data stream）型計算機が提案されている[2],[3]．しかし，これらのシステムでは，要素プロセッサ（PE）間のデータ転送処理とPE内の演算処理との間に制御概念上の断層がある．即ち，PE間のデータ転送処理がデータの発生を契機としたデータ駆動型であるのに対して，PE内の実行制御はデータの発生とは独立に予めスケジュールされる逐次型である．このため，高性能を得るためには，PE間の同期・通信の制御が複雑となる．

他方，プログラムの持つ並列性を自然な形で引き出し得る方式としてデータフロープロセッサが注目され活発な研究が行なわれている[4]～[7]．しかし，高速処理の観点からは，データパケットの通信，動的な資源割付，及び，並列実行のために加わる制御命令により生じるオーバヘッドの問題などの解決がまだ十分ではない．

本論文では，科学技術計算の分野において十分に柔軟性を確保してプログラムの作成を容易にするとともに高速処理を可能にするためのアーキテクチャとしてデータフロープロセッサアレイ方式を提案する．この方式では，プロセッサ間転送制御とプロセッサ内の実行制御をデータ駆動概念で統一して高度の非同期並列処理を実現することをねらっている．方式上の特徴は，通信の局所性を保つように結合系を構成してデータ転送遅延を小さくするとともにハードウェアコストを低

† 電電公社武蔵野電気通信研究所，武蔵野市
Musashino Electrical Communication Laboratory, N.T.T., Musashino-shi, 180 Japan

くしたこと，及び，実行時の適応的な資源割付と実行前の静的な資源割付とを組合わせた柔軟性が高くオーバヘッドの少ない資源割付機構としたことにある．

本論文では，まず，データフロープロセッサアレイ計算機のアーキテクチャを示す．次に，応用プログラムの実行特性を示して方式の有効性を明らかにする．

2. データフロープロセッサアレイ計算機

データフロープロセッサアレイ計算機は，問題の持つ並列性を自然な形で引き出し得るデータ駆動方式の特徴と科学技術計算に共通するデータ従属関係に適合した結合形態のプロセッサアレイ方式の特徴を融合させた並列計算機である．ここでは，この計算機の設計方針と構成を示す．

2.1 設計方針

データフロープロセッサアレイ計算機アーキテクチャの設計方針は次の通りである[8]～[10]．

(1) 制御概念の統一

プロセッサ間のデータ転送制御とプロセッサ内の演算実行制御とをデータ駆動制御の概念で統一的に実現してマルチプロセッサの制御概念を簡明にする．これにより，プロセッサ間のデータ転送や同期，及び，プログラムの実行スケジュールに関するプログラマの負担を軽減する．

(2) 専用化

偏微分方程式の解析問題を中心とした科学技術計算のプログラムの構造にアーキテクチャを合わせて，処理の高速化と制御機構の簡単化を図る．具体的には，配列処理や繰返し処理を高速化する．また，規則性が強く，データ従属関係が比較的簡単な処理に合わせた資源割付を実現してオーバヘッドを小さくする．

(3) 柔軟性の確保

科学技術計算の分野において十分に柔軟性を確保してプログラムの作成を容易にするために，多くの高級言語で許されている条件式，関数呼出し，繰返し文，或いは，マルチプロセスの制御などの高速実行を支援する．

2.2 基本構成

データフロープロセッサアレイ計算機は，図1に示すように，ホストプロセッサと2次元アレイ状に結合した多数の要素プロセッサ(PE)とから成る．ホストプロセッサは，ファイル管理，及び，全PEとのプログラムやデータの転送の機能を持つ．各PEは，データ駆動制御により自律的に動作し，それぞれデータフロープログラムとオペランドデータ用の専用メモリを持ち他PEと交信しながら処理を進める．

PEは，図2に示すように，命令メモリ(IM)，オペランドメモリ(OM)，演算部(OU)，通信制御部，リンクメモリ(LM)から成るサーキュラパイプライン型のデータフロープロセッサである．以下にPEの動作概要を示す[9],[10]．

データフローグラフの各ノードは，演算の種類を指定するオペレーションノードと演算結果の分配先を指定するリンクノードに分解され，それぞれIM，LMに格納される．ここで，データフローグラフは，本システムの機械語である．グラフの各ノードは，高々2個の入力オペランドを持ち，任意個のノードに演算結果を渡すことができる．

IMは，オペランドパケット(OP)が到着するとそ

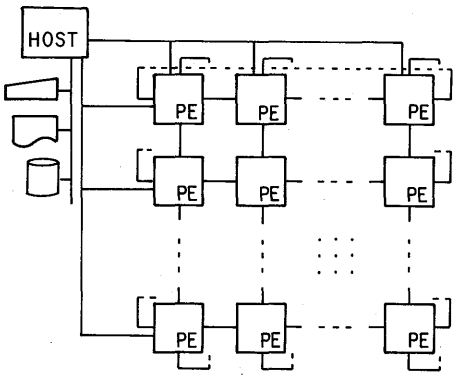

図1 データフロープロセッサアレイ計算機の構成
Fig. 1-Data flow processor array structure.

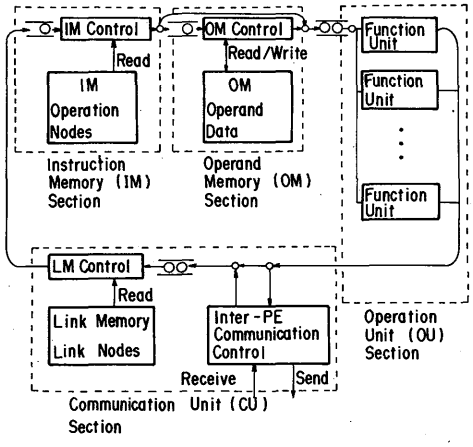

図2 要素プロセッサの構成
Fig. 2-Processing element structure.

のパケットで指定されたオペレーションノードをフェッチして到着パケットとともにOMに送る．OMでは，送られたパケットによりオペレーションノードの入力オペランドが揃うと，オペレーションノード，オペランドデータ，及び，OPから命令パケット(IP)を作成し演算部に送る．他方，入力オペランドが揃わない場合には，OPをOMに格納して待ち合わせる．

演算部は，複数の演算部から成り，実行可能なオペレーションノードを並列実行する．演算の結果，結果パケット(RP)が作られ，通信制御部に送られる．

通信制御部は，演算部或いは隣接PEからRPを受けとりLMに送る．但し，RPに他PEへの転送が指定されている場合には，そのRPを隣接PEへ送り出す．LMでは，リンクノードをフェッチして演算結果の分配先のオペレーションノード名を求め，オペレーションノード毎にオペランドパケット(OP)を作成し，IMに送る．

各パケットの構成は，次の通りである．

RP=〈env, l-name, val (, PEaddr) (, sys)〉

OP=〈env, o-name, val (, sys)〉

IP=〈env, opc, l-name, val 1 (, val 2)〉

ここで，( )内の項は省略可能であることを示す．envは，実行環境名で，タグ付きトークンの概念[5]を実現するためのデータ識別名である[9]．また，sysは，プログラムのローディングやOMデータの消去などのシステム制御のために用いる．各項の記号の意味を表1に示す．

2.3 特　徴

データフロープロセッサアレイ計算機アーキテクチャの主な特徴は次の通りである．

(1) 高並列処理

自律的に実行を進める多数のPEの並列実行による高度なMIMDシステムである．PE内では複数の演算器による並列処理を行ない，システム全体では多数のPEが並列動作する．

表1　パケットの種類とその構成要素

記号	意　味	記号	意　味
RP	結果パケット	env	実行環境名
OP	オペランドパケット	val	演算結果の値
IP	命令パケット	val1	オペランド1の値
l-name	リンクノード名	val2	オペランド2の値
o-name	オペレーションノード名	PEaddr	PEアドレス
		sys	システム制御データ

(2) アレイ結合

通信の局所性を活かすように転送路を分散化したデータフローマルチプロセッサ構成である．流体シミュレーションや行列計算など多くの科学技術計算で有効なアレイ結合を実現して近傍PE間の転送を特に高速にするとともにハードウェアコストを低くする．各PEの通信制御部が中継することにより余分な命令なしで遠隔PE間の転送が可能である．

(3) 準動的な資源割付

実行時に適応的に行なう動的な資源割付と実行前の静的な資源割付とを組合わせることにより，柔軟性が高く，かつオーバヘッドの少ない資源割付方式を実現する．たとえば，配列処理のプログラムでは，図3のように規則的なマッピング法に従い実行前に配列要素をPEに割付け，実行時にはデータ駆動原理に従い個々の演算命令にPEの演算器群の中から一つの演算器を動的に選び割付ける．

(4) 非同期制御

関数呼出しや繰返し処理，及び，配列処理に対してタグ付きトークンの概念[5]を適用してプログラムの非同期並列実行を実現する[9],[10]．特に，配列処理では，各配列要素に対してユニークな名前を付与して，配列データの値の計算と参照を配列の各要素間で完全に非同期に行ない並列性を高める．また，配列データを分解して，配列要素の値を直接演算命令に送る方式をとり，配列処理におけるポインタ操作のオーバヘッドやメモリアクセスの競合をなくす．

(5) 状態の保存

通常のオペランドは，演算を行なうとその値が不要となるのでOMから消去される．しかし，ループの入口で値が定まりループ内では値が不変のデータ(ルー

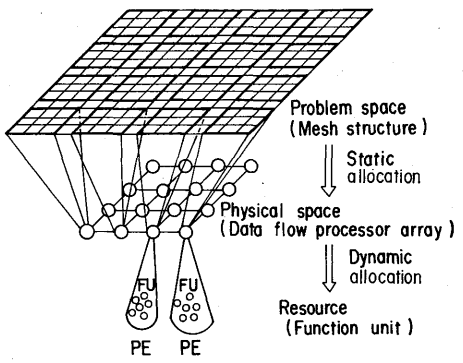

図3　資源割付方式
Fig.3-Resource allocation.

プ内定数）などは値を一定期間（たとえばループの実行中又は関数の実行中）OMに保持して状態を持つことができる．これによりオーバヘッドなく特定データを複数の実行環境下の演算で繰返し使用できる．

3. 評　　価

評価実験により，本方式の動作特性を解析してアーキテクチャの有効性を明らかにする．

3.1 実験システムEddy

提案方式の有効性と問題点を定量的に明らかにすること，システムの設計パラメータを得ること，及び，関数型言語Valid[12]の実験環境を与えてValid の有効性を検証することを目的として，実験システムEddy（Experimental system for data driven processor array）を開発した[9]．Eddyでは，PE 機能の実現容易性，及び，実験環境としての柔軟性を重視して，マイクロプロセッサを用いてPEを構成した．

(1) ハードウェア構成

Eddyのハードウェアは，図4に示すように8隣接PEと直接結合された4×4のPE，及び，2個のブロードキャスト制御装置（BCU）から成る．Eddyでは，前述のように，各PEをマイクロプロセッサで構成した．各PEにおいて，データ駆動実行制御用ユニット（NCU：Nucleus Control Unit）と転送制御用ユニット（TCU：Transmission Control Unit）としてそれぞれ専用のマイクロプロセッサを使うことによって処理速度の向上を図った．また，BCUは，データフロープログラムや初期データなどを各PE に同時にロードする機能を持つ．

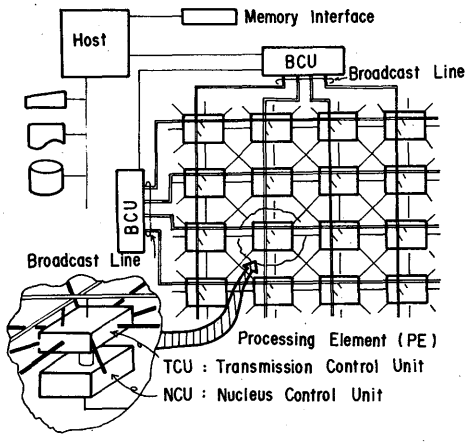

図4　実験システムEddyの構成
Fig.4-Experimental system Eddy organization.

(2) ソフトウェア機能

Eddyでは，各PEはマイクロプロセッサを用いて構成されており，各PE の機能は制御プログラム（PEシミュレータ）により実現される．各PE を専用ハードウェアで構成した場合の特性データを得るために，図2に示したPE の各セクションの機能を分解して14段（隣接PE間転送時は18段）の機能モジュールで構成しPE のパイプライン処理を詳細にシミュレートしている．

図5に示すように，PE シミュレータは，起動制御部（AC），特性収集部（OCC），及び，各セクション対応の機能モジュール群から成る．PEを専用のハードウェアで構成した場合の実行特性を実験結果から容易に予測できるように，AC は論理的なシミュレーションクロック（以下では単にクロックと呼ぶ）に従って他PE と同期をとりながら各機能モジュールの起動制御を行なうことができる．OCCは，各機能モジュールの稼動時間（実際にパケットを処理した期間のクロック数）やキューの長さなどの特性量を収集しモニタに送る．

3.2 評価項目

応用プログラムの実行特性により，次のような項目を明らかにする必要がある．

(1) システムの並列度

データの従属関係や扱う配列の大きさなどの特性に応じて各種応用プログラムでどの程度の個数の演算器が並列実行するか．ここでは，システムに潜在する並列処理能力のうち実際にどの程度の並列性が引き出されたかを示す指標として稼動率ρを次式で定義する．

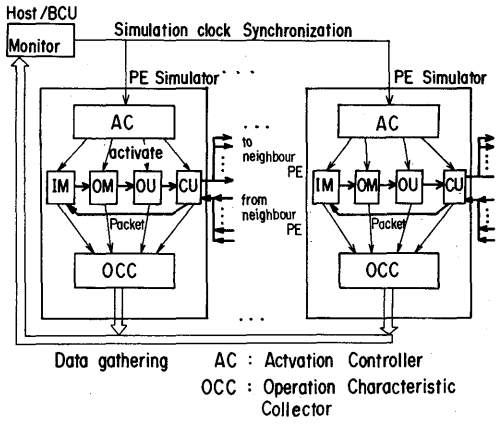

図5　ソフトウェアシミュレータの構成
Fig.5-Software simulator structure.

$$\rho=\frac{\sum_{i=1}^{N_f}T_i}{T_e\cdot N_f} \tag{1}$$

ここで，T_e はプログラムの実行時間，N_f はシステム内の演算器総数，T_i は演算器 i の稼動時間（$i=1, \cdots, N_f$）である．

(2) 通信路の構成と通信遅延

通信の局所性を活かしたり，パイプライン効果を引き出すことにより通信遅延による性能の劣化を実際の応用でどの程度に抑えることができるか．また，結果パケットを分割転送した場合に生じる転送路の実質的な速度低下に対しても高速性を維持できるか．

(3) 実行時の負荷の偏り

前述のようにPEの割付けを静的に行なう場合にマッピング法の選択により負荷を時間，空間的に分散させることができるか．

3.3 例題プログラム

楕円型偏微分方程式を差分法により反復計算する問題を例題とした．二次元問題では，空間座標を x, y 方向の格子に分割し各格子点の解を $X_{i,j}$ とすると，次のような差分方程式の組を解く問題に帰着される．

$$a_{i,j}X_{i,j-1}+b_{i,j}X_{i-1,j}+c_{i,j}X_{i,j}+d_{i,j}X_{i+1,j}+e_{i,j}X_{i,j+1}=q_{i,j} \tag{2}$$

（$i=1, 2, \cdots, N_x$，$j=1, 2, \cdots, N_y$，$a_{i,j}\sim e_{i,j}$，$q_{i,j}$ は係数）

ここでは，以下に示す基本的な解法についてデータフロープログラムを作成し，一定回数だけ反復させて特性データを得た．

(1) 陽解法（点ヤコビ法，red-black SOR法）

陽解法では，第（$k+1$）近似 $X_{i,j}^{(k+1)}$ を与える式において右辺のすべての値が $X_{i,j}^{(k+1)}$ を決める際に既知となっている[13]．このためデータ従属関係が簡明であり並列度も高い．たとえば，データ従属関係の最も簡単な点ヤコビ法では，$X_{i,j}^{(k+1)}$ の値が周囲の格子点の値 $X_{i+1,j}^{(k)}$，$X_{i-1,j}^{(k)}$，$X_{i,j+1}^{(k)}$，$X_{i,j-1}^{(k)}$ を使って求められるので図6(a)のように全格子点の値を並列に求められる．

また，点ヤコビ法よりも収束が速く，かつ並列計算機に適した解法の red-black SOR では，図6(b)のように，$i+j$ ＝偶数，$i+j$ ＝奇数の格子点が交互に周囲の格子点の値を使って求められる[14]．このように2回の反復で全格子点が計算され，全格子点の1／2が並列に処理される．従って，red-black SOR 法で格子点数を点ヤコビ法の場合の2倍程度にすれば，演算器の稼動率や負荷の分散の度合などの実行特性は両解法のプログラムでほぼ等しいと予想される．

(2) 陰解法（ADI法）

陰解法では，一般的には $X_{i,j}^{(k+1)}$ の値を決めるために逆行列方程式を解く必要がある[13]．このためデータ従属関係が複雑となり，並列度も小さい．実際に広く用いられているADI法では，単純な三重対角行列の逆行列を求める問題に帰着され，計算は図6(c)のように進められる[14]．即ち，まず図6(c)①のように水平方向に往復するように各行並列に進められる．次に，図6(c)②のように垂直方向を往復するように各列の計算が並列に進められる．

3.4 評価条件

次のように条件を設定して評価実験を行なった．

(1) マシン条件

- PE台数… $n\times n$（$n=1\sim4$）
- PE当りの演算器個数… f（$f=1\sim4$）
- 各機能モジュールでの1パケット当りの処理時間（クロック数）
  - 演算器… T_f（$T_f=5\sim80$）
  - LM制御部… l（l は分配数であり，1クロック毎に1パケット分配する）
  - 転送制御部… T_t（$T_t=1\sim50$）
  - その他の機能モジュール… 1
- キューの最大制限長
  - LM用キュー… Q_{LM}（$Q_{LM}=16\sim\infty$）
  - OU用キュー… Q_{FU}（$Q_{FU}=16\sim\infty$）
  - その他のキュー…無限長

(2) プログラム条件

- 格子の大きさ… $M\times M$（$M=4\sim32$）
- 反復回数

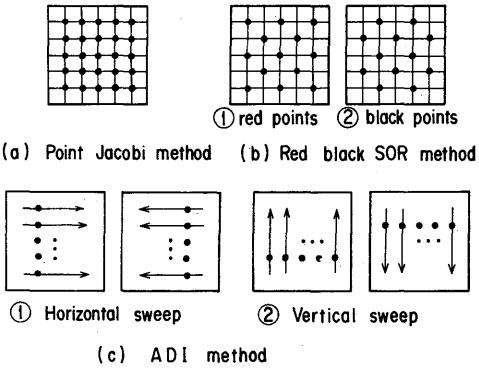

図6 例題プログラムでの並列実行
Fig.6-Parallel execution of sample programs.

・点ヤコビ法… 32回
・ red-black SOR法… 16回
・ ADI法… 16回

(3) マッピング法

次の2つの規則的なマッピング法に従い格子点をPEに割付ける.

(a) 隣接マッピング (Adjacent mapping)

図3に示したように互いに隣接する$\lceil M/n \rceil \times \lceil M/n \rceil$の格子点を1つのPEに割付ける.

(b) モジュロマッピング (Modulo mapping)

$k = i \bmod n$, $l = j \bmod n$を満す格子点(i, j)を$PE_{k,l}$に割付ける.

3.5 実験結果と考察

Eddyでの実験結果を示し, 方式の有効性に関して考察を加える.

(1) 並列処理の効果

各解法のプログラムについて格子の大きさを変化させた時の稼動率ρの変化を調べると図7～9が得られた. ここで, $n^2 = 16$, $f = 4$であり, システムの演算器総数は, 64である.

図7, 8より, 点ヤコビ法とred-black SOR法では格子点が256程度以上になると稼動率が高率に達することがわかる. 特に, $T_f = 10$の時には稼動率がほぼ100%になり並列度が最大値64にまで達する. 従って, 1PE当り16格子点程度以上割付けると, 単一の演算器の場合に比べて実行時間が1／{演算器総数}になり, システムの持つ並列処理能力を最大限発揮しているといえる. なお, 上記両解法では, マッピング法による並列度の差異は見られなかった.

ADI法では, 図9のように, ここで測定した範囲($M \times M \leqq 24 \times 24$)では稼動率の飽和は見られない. これは, 格子の大きさに比べて演算器数が多いのでシステムに潜在する並列処理能力がプログラムの持つ並列度よりも大きくなり, 資源にまだ十分に余裕があることを示している. 見方を変えて, プログラムの持つ並列度をどの程度引き出しているかを調べるために1行(及び1列)当りの格子点数Mと実行時間T_eとの関係を表わしたものが図10である. 図より, モジュロマッピングの時にはT_eがMにほぼ比例する程度で増大することがわかる. 即ち, ADI法では処理量(格子

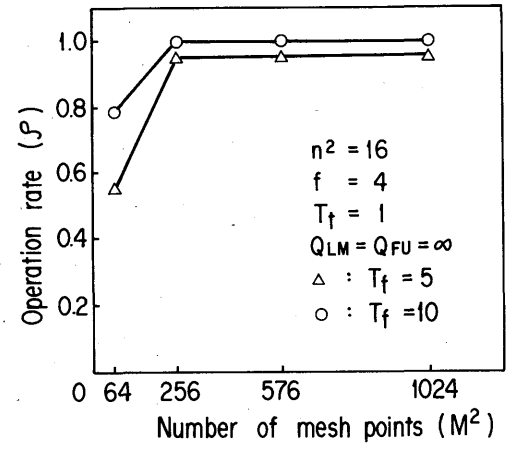

図7 点ヤコビプログラムの場合の稼動率の変化
Fig.7-Operation rate curves for Poin-Jacobi program.

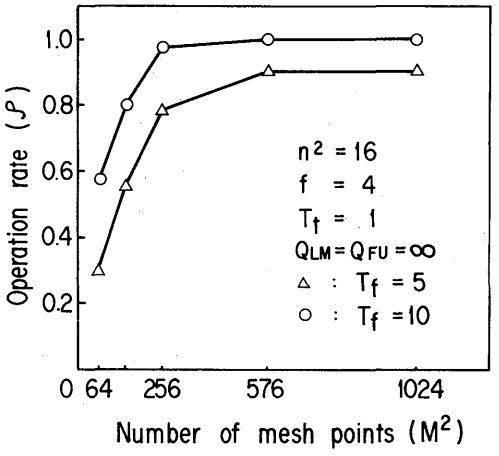

図8 red-black SOR プログラムの場合の稼動率の変化
Fig.8-Operation rate curves for red-black SOR program.

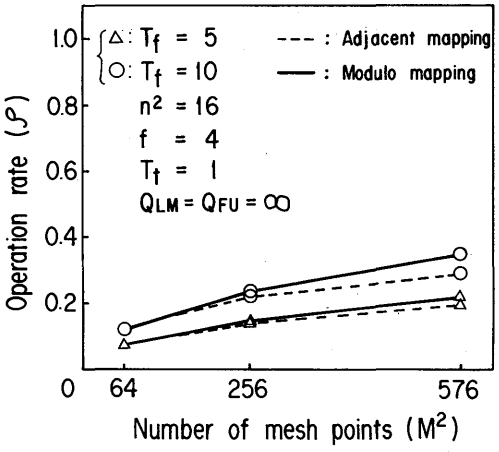

図9 ADI プログラムの場合の稼動率の変化
Fig.9-Operation rate curves for ADI program.

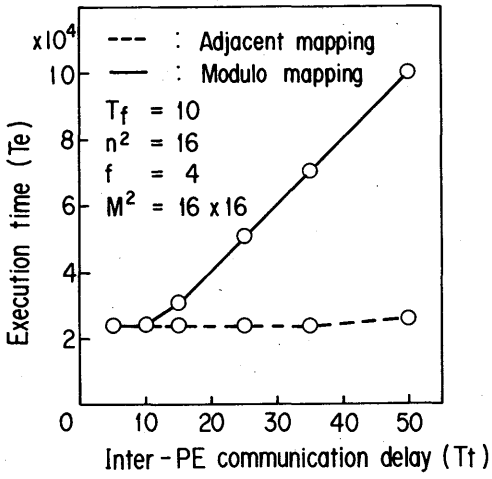

図10 ADI プログラムの場合の並列処理の効果
Fig.10-Parallel processing effects in ADI program execution.

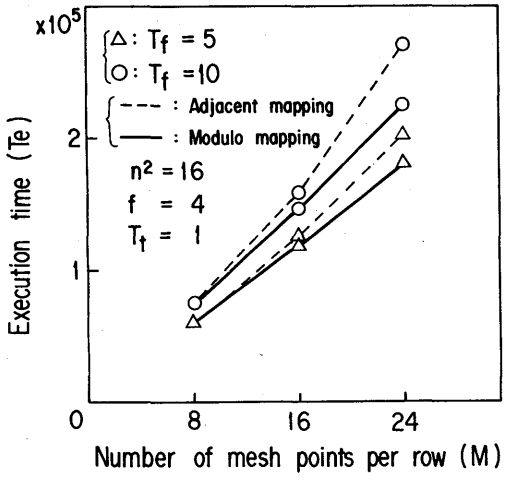

図11 点ヤコビプログラムの場合のPE間転送遅延の影響
Fig.11-Influence of inter-PE communcation delay in Point-Jacobi program execution.

点数）がM^2のオーダで増大するのに対して実行時間をMのオーダの増加程度に抑えられる．**3.3**で述べたように，ADI法は論理的には行又は列毎に並列実行の可能性を持つということを考えると，プログラムに内在する並列性を十分に引き出しているといえる．

図10のようにADI法ではマッピング法により実行時間の差が現われる．これは，図6(c)に示したように実行可能な格子点の位置が格子面を上下左右に1格子

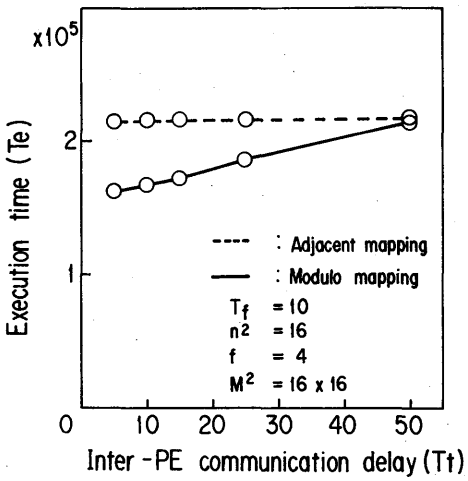

図12 ADI プログラムの場合のPE間転送遅延の影響
Fig.12-Influence of inter-PE communcation delay in ADI program execution.

点ずつ伝搬しながら繰返し動き回る場合に，隣接マッピングよりもモジュロマッピングの方が負荷が時間，空間的に全PEに分散されるためであると考えられる．

(2) 分散化の効果

通信の遅延が性能に与える影響を調べるためにPE間データ転送遅延の大きさT_tを変えた時の実行時間T_eの変化を各マッピング法について求めた．その結果を図11，12に示す．ここで，$M^2=16\times16$であり，1PE当り16格子点を割付けている．

図11より，点ヤコビ法ではモジュロマッピングの時にはT_tが10程度以上になるとT_eは急激に増大するが，隣接マッピングの時にはT_eはほとんど変化しないことがわかる．また，ADI法では，図12のように，いずれのマッピング法でもT_eの急激な変化は見られない．

図11，12より，PE間転送遅延が10クロック程度まではマッピング法によらず実行時間はほとんど変化しないことがわかる．いいかえると，PE間の物理的な転送路の速度をPE内のパイプライン1段の速度に等しくすればPE間転送用パケットを1/10程度に分割転送しても性能が劣化しない．このようにPE間の転送路幅を狭くできることは，VLSI化して小型化したPEを多数結合したシステムを構成する上で大きな利点となる．

4. むすび

本論文では，超高速科学技術計算向き並列計算機の

アーキテクチャとしてデータフロープロセッサアレイ方式を提案し，その構成と動作特性を示した．この方式は，要素プロセッサ間のデータ転送制御と要素プロセッサ内の実行制御をデータ駆動概念で統一して高度の非同期並列処理を実現することをねらっている．方式上の主な特徴は，通信の局所性を保つように結合系を構成してデータ転送遅延を小さくしたこと，及び，実行時の適応的な資源割付と実行前の静的な資源割付とを組合わせて柔軟性が高くオーバヘッドの少ない資源割付機構としたことである．

本論文では，また，偏微分方程式の反復解法のデータフロープログラムを実験システム上で実行させ，提案方式の動作特性を解析して有効性を評価した．その結果，点ヤコビ法や red－black SOR 法などの陽解法のようにデータ従属性が弱い場合にはシステムに潜在する並列処理能力が最大限働くことを確認した．また，陰解法の ADI 法プログラムのようにデータ従属性が強い場合でもプログラムの持つ並列性を十分引き出し得ることを示した．更に，プロセッサ間のデータ転送遅延がある程度大きくなってもプログラムの実行時間が増大しないことを示して，プロセッサ間でパケットの分割転送が可能であることを明らかにした．

実験システム Eddy では，1クロック進めるのに1～10 msec 要する．これは，PE をマイクロプロセッサで構成し，更に，詳細な統計情報を収集しているためである．市販の TTL ICチップを用いて PEを専用ハードウェアで構成した場合には 200～500 nsecで1クロックの処理を行なえるという見通しを得て，現在 PE のハードウェアの設計・製作を進めている．今後，PE の速度評価，及び，有限要素法や LU 分解など広範囲の応用プログラムに対するシステム動作特性の解析を行なう予定である．

謝辞 本研究を進めるにあたり御指導頂いた畔柳功芳基礎研究部長，並びに，山下紘一第一研究室長に感謝します．また，実験システム Eddy のハードウェアを開発された吉田雅治主任，評価実験に御協力頂いた電気通信大学大淵竜太郎氏（現日本 IBM），並びに，御討論頂いた第一・第八研究室の皆様に深く感謝します．

文　献

(1) 加藤，苗村：“並列処理計算機”，オーム社（昭51）．
(2) Kober, R.：“Multiprocessor System SMS 201 -Combining 128 Microprocessors to A Powerful Computer”, COMPCON 77 Fall, pp.225-230 (1977).
(3) 山岡，伊藤，星野，佐藤：“科学技術専用並列計算機 PACS の開発(I)～(III)”，昭55信学総全大，1400, 1401, 1402.
(4) Dennis, J. B.：“Data Flow Supercomputers”, IEEE Computer, 13, 11, pp.48-56 (1980).
(5) Arvind and Kathail, V.：“A Multiple Processor Dataflow Machine that Supports Generalized Procedures”, Proc. of the 8th Annual Symposium on Computer Architecture, pp.291-302 (1981).
(6) Gurd, J. and Watson, I.：“Data Driven System for High Speed Computing”, Computer Design (July 1980).
(7) Amamiya, M., Hasegawa, R., Nakamura, O. and Mikami, H.：“A List-processing-oriented Data Flow Machine Architecture”, Proc. of the 1982 National Computer Conference, AFIP, pp.143-151 (1982).
(8) 高橋，雨宮：“超高速科学技術計算を指向したデータフロープロセッサアレイ計算機の提案”，信学技報，EC80-24（1980）．
(9) 高橋，吉田，雨宮：“データフロープロセッサアレイ計算機の構成と実験システム”，信学技報，EC 81-73（1981）．
(10) Amamiya, M., Takahashi, N., Naruse, T. and Yoshida, M.：“A Data Flow Processor Array for Solving Partial Differential Equations”, Proc. of International Symposium on Applied Mathematics and Information Science, Kyoto University (1982).
(11) 高橋，雨宮，大渕：“シミュレーションによるデータフロープロセッサアレイ計算機の評価”，情処学会計算機アーキテクチャ研究会，48-4（1983）．
(12) 雨宮，尾内：“データフローマシン用高級言語 Valid について”，信学技報，EC82-9（1982）．
(13) P. J. ローチェ（高橋他訳）：“コンピュータによる流体力学上，下”，構造計画研究所（昭53）．
(14) Ortega, J. and Voigt, R.：“Solution of Partial Difference Equations on Vector Computers”, Proc. of the 1977 Army Numerical Analysis and Computer Conference (1977).
(15) 山内，他編：“電子計算機のための数値計算法 I，II”，培風館（1965）．

（昭和58年4月30日受付，7月19日再受付）