《講　座》

並列処理技術　—並列処理の概要—

笠　原　博　徳*

ABSTRACT　Parallel processing technology has attracted much attention in various research fields on simulation, such as electronic circuit simulation, nuclear reactor dynamics simulation and neural network simulation. In order to give the readers a concept of parallel processing, this paper which is the first one of four survey papers on parallel processing describes needs of parallel processing and a brief overview of parallel processing hardware so far proposed. More concretely speaking, first of all, it describes definition of parallel processing computers and purposes of parallel processing. Next, various parallel processing schemes which have been so far proposed are surveyed together with brief history of parallel processing technology. Finally taxonomy of parallel processing schemes relationship between needs of parallel processing technology and advancement of semiconductor technology are explained.

1. はじめに

本号より4回にわたり，並列処理技術に関する解説を行う．この並列処理技術[1)~15)]は従来より気象シミュレーション，流体シミュレーション，構造シミュレーション，電子回路シミュレーション，原子炉動特性シミュレーション，ニューラルネットシミュレーションを初めとした各種シミュレーション分野[6)~8),16)~18)]，あるいはシミュレーション結果を高速にグラフィック表示するための画像処理等多くのシミュレーション関連分野で注目を集めている．本講座はこの並列処理技術を，計算機アーキテクチャの専門家以外にも分かりやすく平易に解説を行うことを心がけているので，コンピュータの苦手な方も是非一度目を通して頂きたい．

本講座第一回目の今回は，並列処理技術の概要を理解して頂くために，並列処理のニーズ及び現在までに提案されている並列処理技術のハードウェア全般について簡単に解説する．

まず本論に入る前に，並列処理コンピュータとは何かということを定義しておかなければならないが，ここでは直感的に，同時に複数の演算装置を動作させながら処理を行うコンピュータと理解して頂きたい．

またこの並列処理の目的であるが，これには以下のようなものが挙げられる．

(1)　スーパーコンピュータのような超高速コンピュータの開発

現在手にはいる最高速の演算装置を結合し並列に動作させることにより，超高速な計算を可能とする．

(2)　価格性能比の優れたコンピュータシステムの開発

低価格の演算装置（例えばマイクロプロセッサ等）を複数結合・並列動作させることにより，同一の処理性能をもつコンピュータを低コストで開発する．

(3)　信頼性の高いコンピュータシステムの開発

冗長な演算装置を用意することにより単一あるいは複数の演算装置が故障したときでも，平常時と同様な処理あるいはコンピュータ全体がダウンすることなく性能を多少落とした状態で処理を続けること（グレースフル・デグラレーション）を可能とする．

このような目的のために並列処理技術は，従来より30年近くも研究利用されている．しかしこの並列処理技術がまだ完成されておらず現在も話題になっていることを考えると，いかに難しい技術であるかがお分かり頂けると思う．

以上のようなことを踏まえ，まず2章ではこの計算速度を重視したコンピュータを中心に並列処理技術の歴史を簡単に追いながら，現在までに開発されている各種並列処理方式について解説を行う．3章では並列処理方式の分類を行い，4章で半導体技術の進歩と現時点に置ける並列処理の必要性について説明する．

Parallel Processing Technology —Overview of Parallel Processing—. By *Hironori Kasahara* (Department of Electrical Engineering, Waseda University).

*早稲田大学理工学部電気工学科

2. 現在までに開発された並列処理技術

本章では現在までに開発された主要な並列処理コンピュータを紹介しながら現在までに研究開発されている並列処理技術について簡単に説明する．まず最初に並列処理を導入したコンピュータとしては，1960年代の初めにCPUの演算速度と入出力装置の速度のギャップを埋めるために入出力用のプロセッサを別にもたせ機能分散的な並列処理を行ったスペリー社のLARC（1960年）あるいはIBMのSTRETCH（1961年）が挙げられる[12]．これらのコンピュータは通常のコンピュータの世代的には第2世代に属するもので半導体トランジスタと磁気コアメモリを使用して製作されており，その処理速度は加算4μs，乗算8μs程度であった．

次に並列処理技術を用いたコンピュータは1960年中旬から後半にかけて開発されたCDC6600（1964年）[20]，CDC7600（1968年），IBM360/91（1967年）[21)~23)]のようなコンピュータであり，これらのシステムは単一のプロセッサ（CPU）の中に命令パイプライン（複数の命令をオーバラップして処理する方法）や同時動作可能な複数の演算ユニット（複数の浮動小数点加算器・乗算器等）を持っていた[1),2),10),12)]．

命令パイプライン

ここで“命令パイプライン”とは，図1のようにCPUでの命令の実行過程を命令フェッチ（取り出し）ステージ，命令デコード（解読）ステージ，オペランド(被演算データ)フェッチステージ，実行ステージ，結果ストアステージのように各ステージの処理時間t1からt5がほぼ等しい時間を持つように複数のステージ（別々に動作する専用ハードウェア）に分割し，さらにt1からt5のうち最大のものを1単位時間［u. t.］とし，命令を1［u. t.］毎にI1, I2, …I7, …のように左端から連続してパイプに流し込み，同時に各ステージの処理が終わった命令を右側のステージに移すという方式である．この方式ではもし常にパイプ中を命令がスムーズに流れているならば，n個の命令を処理する時間は通常の逐次形処理では$5n$［u. t.］かかるのに対し$5+(n-1)$［u. t.］で終了することになる．すなわちnの数が十分大きければ処理時間が1/(パイプラインのステージ数）に近付くと言うわけである．この命令パイプラインはこれ以後スカラ計算を高速化するための基本的技術として汎用コンピュータ，スーパーコンピュータを初めとする多くのコンピュータで使用されている．しかしこの命令パイプライン方式の場合，実際の命令には条件分岐命令（条件分岐があるとパイプ中に入れられている分岐以降の命令が必ずしも実行されるとは限らず，実行されない場合にはそれらの命令を無効にしなければならない）があったり，浮動小数点演算のように1単位時間で終わらないような実行時間の異なる命令があったり，すぐ前の命令の結果を使用する命令があったりするため，理想通りスムーズには命令が流れず理想通りの性能を得ることが難しい．したがってこのような場合に対処するためにこれ以後のコンピュータでは多くの工夫が行われている．例えば条件分岐命令がある場合にはその命令解読時に分岐後の別の命令の流れも高速に命令を読み出すことができる命令バッファに先読みしておいたり[38)~41)]，それに加え過去の分岐履歴を保持し実行される確率が高い命令の流れをパイプラインにいれること[41)]により性能の劣化を避ける方法がIBM3090（1985年）[38)]，日電ACOS1500（1985年）[39)]，日立M680H（1985年）[40)]，富士通M780（1986年）[41)]のような最近の大型汎用コンピュータを初め多くのコンピュータでとられている．

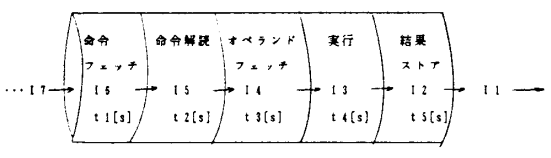


図1 命令パイプライン

複数演算ユニット方式

また命令による実行ステージの処理時間の変動に対しては，例えば前の命令が複数単位時間を要するのに次の命令は1単位時間で実行を終了するような場合には，浮動小数点演算ユニットや固定小数点演算ユニットを複数もたせ（前述の“複数演算ユニット方式”で例えばCDC6600は10個の演算ユニットを持つ）先行命令と後続命令が同時に実行できるようにすることにより性能の劣化を防ぐことができる．但しこの際命令の実行順序が変わってしまうことがあるためデータの代入参照関係等により計算結果が間違わないように制御する必要がある．この制御をCDC6600ではスコアボード，IBM360/91ではリザベーションステーション，ビジィビット，共通データバスというハードウェアを用いて実行時に制御している[1),2),20)~23)]．

次に出てきた並列処理コンピュータのうち代表的なものは1970年前半にIC, LSI等の集積回路を用いて開発されたイリノイ大学のILLIAC IV（1972年）[25),26)]，カーネギ・メロン大学のC.mmp（1973年）[27)]等であ

る．これらのコンピュータで用いられた並列処理方式は，それぞれプロセッサ・アレィ方式，マルチプロセッサ方式である．

プロセッサアレィ方式

まずILLIAC IVで採用された“プロセッサアレィ方式”とは，**図2**のように各種演算を行うことができるプロセシングエレメント（PE）をPE間インタコネクションネットワーク（相互結合網）で規則的に並べ（ILLIAC IVでは縦8台×横8台の計64台のPEを格子状に並べた），単一のコントロールユニットの制御下で全てのPEが同一の演算を同時に行う空間的な並列処理方式である．この方式は単純な配列演算には適しているが，それ以外の計算に効果的に使用するのが難しい，また単純な配列計算でもその配列の大きさによっては処理効率が落ちる，データのPEへの入出力に時間がかかる等の問題点がある．このような理由からILLIAC IV自身はあまり多く使用されることはなかったが，バイポーラを用い最高150 MFLOPS（Mega FLoating Operation Per Second）の処理能力を達成するために研究開発された素子，多層基盤，プロセッサ間結合，プログラミング等に関する技術はそれ以後の並列処理コンピュータに大きな影響を与えた．またこのプロセッサ・アレィ方式はこの後も大量の単調なデータ処理を高速に行うために利用されており，その例としては，Goodyear Aerospace社が人工衛星からの画像データを処理するために1983年に開発した128×128の計16384台のPE（1ビットプロセッサ）を格子状に並べ最高6000 MOPS（Mega Operation Per Second）を達成したMPP（Massively Parallel Processor）[35]や，Thinking Machineが1985年にAIを含めた種々の利用法を求めて65536台の1ビットプロセッサ2進12キューブネットワーク（詳細に関しては次回解説する予定）を用いて接続したConnection Machine[42]等が挙げられる．またMPPのようなプロセッサ・アレィコンピュータは，セルラアレィプロセッサ（CAP）とも呼ばれ，現在進行中の通産省のスーパーコンピュータの大型プロジェクトの中でも研究開発されている．

マルチプロセッサ方式

次にC. mmpで使用されている“マルチプロセッサ方式”[29)~31)]とは，実際にはC. mmp以前からバローズ社のD-825（1962）[24]でも使用されていた方式であり，**図3**に示すように独自に別々の演算を行うことができる複数のプロセッサをバス，クロスバースイッチ，多段スイッチングネットワーク等のインタコネクションネットワーク（前述のように詳細は次回解説する）を用い必要に応じ共有メモリなどを介して接続したものである．例えばC. mmpは8kバイトのローカルメモリを持つPDP11/40Eのインストラクションに多少手を加えたコンピュータモジュール16台を16個の共有メモリにクロスバースイッチ（スイッチの切り替えにより同時に全てコンピュータモジュールを全ての共有メモリに接続することができる接続方式）で接続した構成であった．この方式ではプロセッサアレィと異なり各プロセッサが異なる命令系列の処理を並列に行うことができるので多様な処理に柔軟に対応できるため，これ以後カーネギ・メロン大学の階層型マルチプロセッサシステムCM*(1980年)，命令パイプラインを発展させ複数の命令の流れ（異なるジョブ中の互いに独立な命令の流れ）を順番にパイプに入れ処理効率の向上を図るMIMDパイプライン方式を用いたプロセッサ16台をスイッチングネットワークで共有メモリに結合したデネルコア社のHEP（最高性能160 MFLOPS, 1981年）を初め，最近のスーパーコンピュ

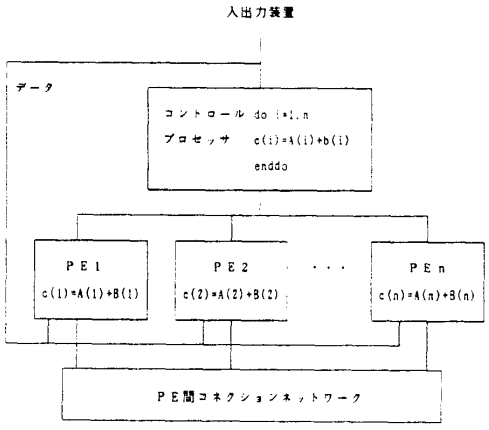


図2 プロセッサ・アレィ

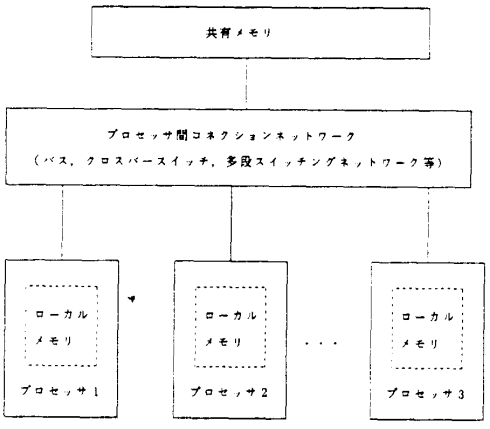


図3 マルチプロセッサ

ータ例えば4台のプロセッサ（各プロセッサは後述するベクトルプロセッサでCRAY1のような従来のスーパーコンピュータに相当する）を共有メモリで接続したクレイ社CRAY X-MP (840 MFLOPS, 1983年), CRAY 2 (2 GFLOPS, 1985年), 8台のプロセッサを接続したETA社のETA10 (10GFLOPS), あるいはElexsi6400 (12プロセッサ, 159 MFLOPS), Alliant FX80 (8プロセッサ, 188 MFLOPS), FPS-Tシリーズ（最大4096台のトランスピュータをハイパキューブ結合可能）等の多くのミニスーパーコンピュータで使用されている．しかし，このマルチプロセッサ方式はハードウェア的にはかなり進んで来たがまだソフトウェアが追いついておらず，従来より多くのソフトウェアの蓄積を持つFortran等の逐次形言語で書かれたプログラムからどのように並列性を引き出すか，また並列処理可能な部分をどのようにプロセッサに割当てどのような順序で処理すれば最小の処理時間が得られるか（マルチプロセッサスケジューリング問題），どのように同期をとるか等ソフトウェアに関連した問題が多く残されている[31]．このようなソフトウェアに関しては現在IBMのRP3プロジェクト[44),45)]，イリノイ大のCEDARプロジェクト[46)]，早稲田大のOSCARプロジェクト[47)]等で研究中である．なおこのマルチプロセッサシステムのハードウェア，ソフトウェアに関しては今後の並列処理技術の展開の中でも特に重要であるので第2回目以降でも詳しく説明していく予定である．

次に重要な並列処理コンピュータは演算パイプライン(ベクトルパイプラインまたはベクトルプロセッサ)を用いて1976年に開発されたスーパーコンピュータCRAY-1（160 MFLOPS，クロックサイクル26.5 ns，素子ECL）である．この演算パイプラインを用いたコンピュータはCRAY-1以前にもTI社のASC (30 MFLOPS, 1972年) やCDC社のSTAR-100 (50 MFLOPS, 1973年) 等があったが，真にベクトルプロセッサを実用的なものにし今日のようなベクトルプロセッサ方式スーパーコンピュータの全盛期をもたらす基礎を作ったのは，CRAY-1である．

演算（ベクトル）パイプライン

ここで“演算（ベクトル）パイプライン”とは，基本的には前述の命令パイプラインと同一の概念であるが，配列演算のように異なったデータに対して繰り返し同一の演算を行う際に，その演算ユニットをパイプライン化し，処理を高速化しようとするものである．たとえばそれぞれn個の要素からなる1次元実数配列

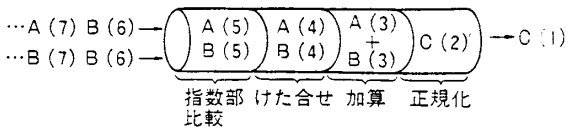


図4 演算（ベクトル）パイプライン

$A(i)$と$B(i)$を加算しその結果を$C(i)$とする計算では，これを表現するDOループを$C(1:n)=A(1:n)+B(1:n)$という1つのベクトル命令に置き換え（ベクトル化），この命令により$A(i)$と$B(i)$の要素を，**図4**のように指数部比較，桁合わせ，加算，正規化等の複数ステージに分割された浮動小数点加算パイプにクロックサイクル（ほぼ等処理時間に分割された各ステージの処理時間のうち最大の処理時間）毎に流し込むことにより，nの長さ（ベクトル長）が十分長ければ逐次形処理の1/(パイプラインのステージ数)の時間で処理が行える．しかしこの方式ではベクトル長が短いとアドレスの設定時間を含め最初のデータがパイプを通過するまでの時間（初期パイプ遅延）が大きなオーバヘッドになり実行効率が悪くなるという問題点がある．これはパイプの段数を増やせればピーク（最高）性能は上げることはできるが，平均的な性能（実効性能）とのギャップが大きくなる可能性があることを意味している．従ってこのパイプラインの性能を評価する際単にピーク性能のみを議論するのではなく，実効性能も議論しなければならない．このようなことを含めて演算パイプラインの性能を評価する指標としてピーク性能の半分の性能を達成するのに要するベクトル長（半性能長）[10),11)]を用いることも提案されている．

CRAY-1以後，CDC社Cyber205 (400 MFLOPS, 11.0 ns, ECL, 1981年)，日立S810 (857 MFLOPS, 8.9 ns, CMOS, 1983年)，CRAY X-MP (800 MFLOPS, 14.6 ns, ECL, 4プロセッサ, 1983年)，富士通VP200/400 (1.1 GFLOPS, 7 ns, CMOS, 1984年)，日電SX1/2 (1.3 GFLOPS, 6 ns, CMOS, 1985年)，CRAY-2 (2 GFLOPS, 4.1 ns, ECL, 4プロセッサ, 1985年)，ETA10 (10 GFLOPS, 5 ns, CMOS, 8プロセッサ, 1987年) 等多くのスーパーコンピュータがこの演算パイプライン方式を使用している．1988年現在単一ベクトルプロセッサの最高性能は富士通VP2000の4 GFLOPS (ECL, 1988年) である．ただしこれらのスーパーコンピュータは単なる演算パイプライン方式だけではなくパイプのチェイニング（リンケージ），マルチパイプライン等という方法も導入している．まず

チェイニングはCRAY-1が初めて導入した技術であり，同一の配列に対して順序関係のある複数の演算を行いたい場合に，最初の演算の結果をメモリにストアしてから新たにロードして次の演算を行うのでは，前の演算がすべて終了するのを待つ時間とメモリアクセス時間がオーバヘッドとなってしまうので，前の演算パイプの出口を次の演算パイプの入口に接続して，配列の最初の要素（データ）に対する最初の演算が終わったら，すぐデータに対してその次の演算ができるようにしようとするものである．またマルチパイプラインとは，単一のプロセッサ中に並列動作可能な複数の演算パイプを用意し，プログラム中に同時に処理可能なベクトル命令があればそれらを同時に別々のパイプで処理しようとするものである．

この演算パイプライン方式は，ハードウェア的にはかなり完成されてきており，日本製のスーパーコンピュータが出てきた辺りからソフトウェアの優劣がシステムの性能を左右するようになってきている．具体的には，コンパイラが，Fortranプログラム中のDOループのうちベクトル化可能な部分をどれだけ見つけることができるか，データの依存関係が複雑なループをどれだけ自動変換してプログラムをベクトル化できるか，あるいはユーザがプログラムをベクトル化しやすくするためにどのようなチューニング情報をユーザに与えられるかというところが勝負となっている．このようなベクトル化のためのソフトウェアに関しても紙面が許せば第2回以降で触れる予定である．

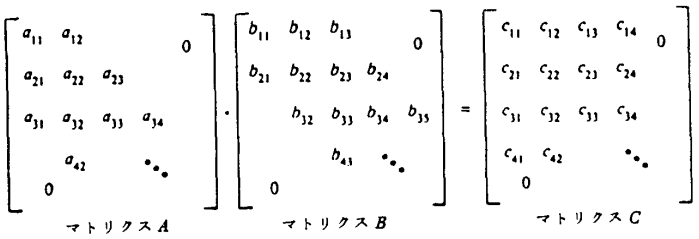
VLIW方式

また以上で紹介した方式とは異なる並列処理方式でMultiflow社のTraceというミニスーパーコンピュータで利用されている“VLIW（Very Long Instruction Word）”と呼ばれる方式がある．このVLIW方式は従来よりエール大のELI-512や京大のQA-2で研究されていた水平型マイクロプログラミングを用いてプロセッサ内に用意した複数の演算器を並列動作させようとする方式である[2]．すなわちコンパイラによりプログラム中より並列に計算可能な演算部分を見つけだし，その並列計算可能な複数の演算命令を256ビットあるいは512ビットという1つの非常に長い命令の中に埋め込み，その長い命令を直接複数の演算器を用いて処理しようという方式である．この方式ではいかに並列計算可能な命令をプログラム中より見つけ，さらに条件分岐等も考慮して1命令中に埋め込むかがキーポイントとなり，このための方法としてMultiflowではトレーススケジューリング[48]という方法が用いられており，今後の性能評価が注目される．

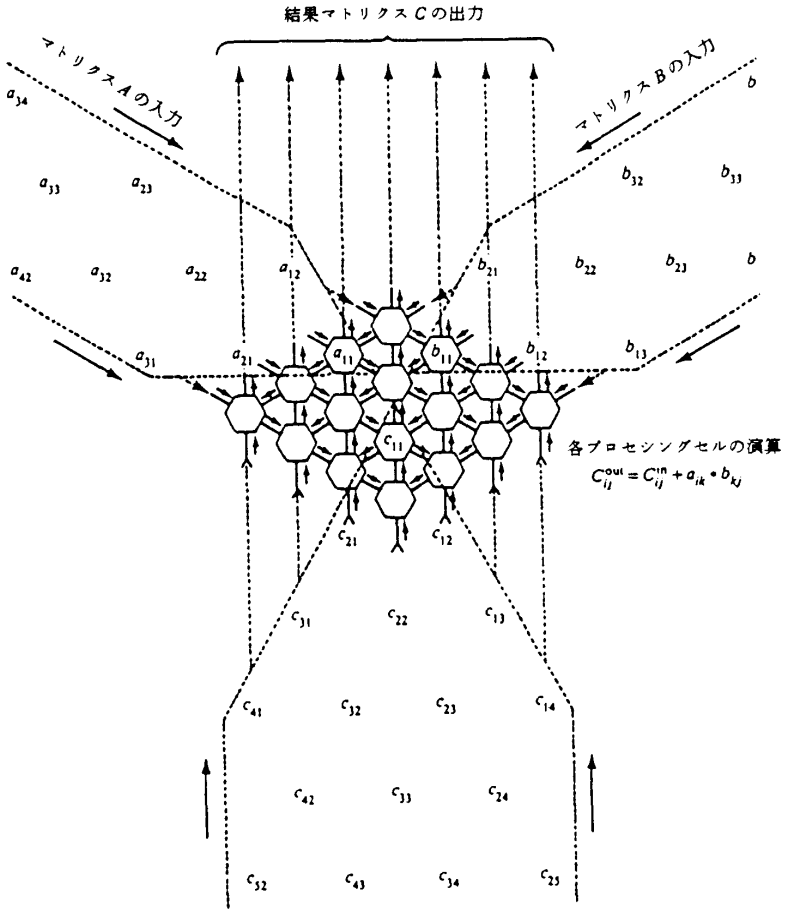
またこのほかにまだ完全な実用化には至っていないが，従来より提案されている重要な並列処理方式としてカーネギメロン大の*Kung*により提案されたシストリックアレィ[37]とMITの*Dennis*[32]により提案されたデータフローマシンがある．以下ではこれらについて簡単に説明する．

シストリックアレィ

“シストリックアレィ”（シストリックと言う語は本来循環器系の脈動を意味する）は大量のデータに対して同一な演算を行う専用目的並列演算器を作製するのに適した方式であり，VLSI（Very Large Scale Integration）でのインプリメントを前提としている．すなわちVLSIでインプリメントし易いように，比較的単純な演算を行うプロセシングセル（演算器）を1次元あるいは2次元アレィ状に規則的に配列し，そのセル間の結合も隣接するもの同士を接続し，全てのセルは同期をとりながら同一の計算を行い結果を隣接するセルに送るという操作を繰り返し演算を進める．例えば図5(a)に示すようなバンドマトリクスの乗算を行うシストリックアレィ[12]は(b)のようにプロセシングセルを2次元状に並べた構成となる．入力データとなるマトリクスAとBの各要素は図のように並べられそれぞれ左右上部からクロック毎に同期してアレィに注ぎ込まれる．各セルは，左右上部からきたAとBの要素を掛け合わせさらにその結果を下部から来る演算結果マトリクスCの部分演算結果に加えその演算結果は上部に接続されたセルに送り，Aの要素はそのまま右下のセルにまたBの要素は左下のセルに送るという動作を，クロック毎に繰り返す．シストリックアレィはこのように一種のパイプライン演算を行う並列処理方式である．またどの様な機能を持つセルをどのように配列・接続し，データをどのように入力すれば所望の演算を行うことができるかというアルゴリズムをシストリックアルゴリズムと呼び現在までに各種マトリクス演算のほか信号処理用フィルタ用のものなど多く提案されている．*Kung*等は，このシストリックアレィの実用性を検証するために10個のセル（Weitekの浮動小数点演算チップ使用）を1次元状に配列したWARPを製作している[49]．このシストリックアレィは，対象とする計算の種類あるいは規模（例えば前述のバンドマトリクスの演算ではバンド幅）が異なる毎に異なるVLSIを作製しなければならないというところが問題点であり，今後実用化されていくかどうかは同一のVLSIを多量に使用するアプリケーションがど



(a) バンドマトリクス A, B の乗算



(b) (a)の乗算用シストリックアレイ

図5 シストリックアレイ

れだけあるかに依存する．

データフローマシン

“データフローマシン”の根本的な概念は，従来のノイマン形コンピュータのようにプログラムで上から下に書かれた順すなわちプログラムポインタが指す順に命令を逐次処理するのではなく，書かれた順序とは無関係にその演算で必要なデータが揃ったらすぐに実行を開始し，その時同時に複数の命令が実行可能になったら複数のプロセッサで並列に処理しようというものである．データフローマシンのプログラムは本質的に命令間のデータの依存関係（データの定義と使用の関係）を表すデータフローグラフ（ノードが命令を表し，ノード間のエッジはデータの流れすなわち依存関係を表す）と呼ぶ有向グラフで表現される．すなわちデータフローマシンの実行は，データフローグラフ上のノードの全ての入力エッジにその命令が必要とする全てのデータ（データトークン）が揃ったら，その命令を実行し，その実行結果をデータトークンとして出力エッジを通し次の命令に送るという動作の繰り返しであると解釈できる．従ってデータフローマシンの研

究では，このデータフローグラフを並列に効率よく処理するためのアーキテクチャが研究されており，従来提案されている方式は *Dennis* によって提案されているスタティックデータフローマシンと MIT の *Arvind*，マンチェスタ大学，電総研等で開発されているダイナミックデータフローマシンの2つに大きく分かれる．スタティックとダイナミックの違いは，データフローグラフの各エッジにデータトークンが2つ以上存在してもいいかどうかの違いであり，スタティックの場合には1つのみの存在しか許されず，ダイナミックの場合には2つ以上の存在が許される．この違いはループとか再帰の処理の際に顕著に現れ，ダイナミックデータフローマシンの方がプログラム中の並列性を最大に引き出すことができるが，各データトークンにカラーあるいはタグと呼ばれる識別子をつけて正しいデータを用いて演算を行うための制御をしなければならず，オーバヘッドが大きくなるという問題点がある．データフローマシンは，実行形態，アーキテクチャ及びデータフロー言語（関数型言語，単一代入規則等が大きな特徴）によるプログラミングを含めて従来より多くの長所短所が指摘されているが[12),34)]，方式的な最も大きな問題点は並列処理の単位が命令レベル（ファイングラニュラリティ）で小さすぎるため実行に伴うオーバヘッドが相対的に大きくなってしまうことである．従来データフローマシンは次世代の並列処理方式として最も有望であると思われていたが，最近のマルチプロセッサシステムの進歩により両者の差が無くなっていく傾向がみられ[31)]，逆に従来のプログラムとの親和性，処理オーバヘッドを考慮するとマルチプロセッサシステムの方が有望とも考えられる．

3. 並列処理システムの分類

2章で述べたような種々な並列処理方式を分類する方法として，従来より多くの分類法が提案されているが，分類法を提案してもすぐそれに適合しない並列処理方式が提案されるなどのため，現在の所全ての処理方式を完全に分類することができる方法は知られていない．しかし一般的には命令の流れとデータの流れに注目した *Flynn* の分類[36)]が使用されるのでこれを紹介する．*Flynn* の分類では並列処理コンピュータは SISD (Single Instruction Stream Single Data Stream), SIMD (Single Instruction Stream Multiple Data Stream), MISD (Multiple Instruction Stream Single Data Stream), MIMD (Multiple Instruction Stream Multiple Data Stream) の4つに大別される．ここで SISD とは1つの命令系列により1つのデータの流れを処理する逐次形の処理方式を表しており，2章で述べた命令パイプライン*はこの分類にいれることができる．また SIMD は単一の命令により複数のデータが処理される処理方式であり，プロセッサアレィ，演算パイプライン（ベクトルプロセッサ)*，シストリックアレィ*などがこれに分類される．また MISD は複数の命令系列が1つのデータの流れを処理するもので一般的にはこれに属するものは無いとされている．MIMD は複数の命令系列がそれぞれ別のデータの処理を行う柔軟性の高い方式であり，マルチプロセッサ，データフローマシン，VLIW*をこれに分類することができる．ただし上述のようにこの *Flynn* の分類も完全なものではなく，上で分類した各方式(特に星印をつけたもの)も見方によっては他の分類に属すると考えることができる．

4. 半導体技術と並列処理の必要性

2章で述べた並列処理技術の歴史を見ると分かるように並列処理の進歩は半導体技術の進歩に密接に関連している．例えば現在のスーパーコンピュータの主流であるパイプラインコンピュータを例に考えてみると，従来のパイプラインスーパーコンピュータのピーク速度向上の多くは半導体技術のお陰である．根本的に単一のパイプラインプロセッサを用いてピーク速度の向上を図るためには，パイプのステージ数を多くするか，各ステージの処理速度を上げるすなわち素子速度を上げる，あるいは並列に動作可能な演算器（パイプ）の数を増やすかのいずれかである．しかしパイプの段数は通常8段ぐらいが物理的に限度と言われており，もし仮に増やすことによって表面上ピーク性能が向上したとしても，実際に使用する時の平均的な性能（実効性能）は上がらず，場合によっては逆に低下してしまうことも有り得る．またパイプの本数を増やすことに関しても最近のスーパーコンピュータは既に多くのパイプを保持しておりこれ以上単に増やしても，それらのパイプ全部を有効に使用することは難しいという問題点がある．従ってパイプライン方式のみを用いて実効性能も上げることができる真の速度向上の方法は，素子速度を上げることと言うことになる．

しかし素子技術の進歩も従来よりの目覚しい進歩により，1988年現在最速の Si（シリコン）バイポーラ論理 IC の1ゲート当りの遅延速度が80 ps（集積度15000ゲート程度）と言うように高速になると，今後そう簡単にはスイッチング速度を上げることはでき

ず，できても後10倍程度であろうと言われている．またスーパーコンピュータ用高速素子としてはスイッチング速度だけでなく集積度と消費電力（冷却の問題）にも注目しなければならない．すなわち素子の高速化により，1 mm 当り 5 ps という素子と外部回路との接続配線による遅延が無視できなくなり，1 チップ当りの集積度を上げなければならないが，高速なバイポーラ素子（ECL 等）の集積度を上げると発熱が大きくなり，現在の冷却技術（1 チップ当り数ワットから数十ワットが限度）では冷却できないというわけである．これらを考え合わせると今後 Si 技術だけに頼りスーパーコンピュータの高速化を考えることは難しいことが分かる．

そこで GaAs（ガリウムヒ素），HEMT（High Electron Mobility Transistor），JJ（ジョセフソン結合素子）などの新素子の開発が重要となるわけである．GaAs はシリコンに比べ数倍の電子移動度を持つため高いスイッチング速度が得られることが期待されているが，MOS 形の製造が難しい（SiO_2 のような加工時の保護や電気絶縁につかえる良質の絶縁膜が無いため）ため，MESFET（Metal-Semiconductor Field Effect Transistor）を中心に研究が進められてきたが結晶成長技術や製造プロセスの技術が未熟のため，現在のところ集積度を上げるのが難しい．またそれらの技術が進歩しても，Si の 2〜3 倍の性能しか得られない可能性があるといわれている．また現在富士通等で研究されており実用化に最も近いと言われている HEMT は，基本的には GaAaMESFET と同様な性質を持ち，最近では常温でも高性能が得られることが確かめられたが，小規模集積回路では Si より優れた性能を示しているものの，大規模集積回路では大幅な速度向上が期待できない状況である．ジョセフソン素子は，液体ヘリウム温度の極低温での超伝導現象を利用しており高速（スイッチング速度 1 ps）かつ低消費電力（0.1 μw）のため高密度実装可能であるが，ヘリウム温度に冷却するためのエネルギーの消費に打ち勝つだけのメリットを引き出す超高密度実装は，現状では素子特性がばらつき製造歩留り悪いため難しい状態である[7]．

そこで素子技術だけに頼らずより速いスーパーコンピュータを開発するためには，現在最高速の素子を用いたプロセッサ（現在のスーパーコンピュータのようなパイプラインプロセッサ）を数十，数百，数千と接続して数桁の速度向上が得られる可能性のあるマルチプロセッサのような並列処理技術の導入が重要となるわけである．次回以降はこのマルチプロセッサシステムを中心にハードウェア，ソフトウェア，シミュレーション分野への応用について解説する．

5. むすび

本稿では講座第一回目として並列処理のニーズ，分類，各種の並列処理方式について簡単に解説した．第二回目では今後スーパーコンピュータ等でさらにその導入が活発になると予想されるマルチプロセッサシステムを中心に，プロセッサの結合方式（スイッチングネットワーク等）を初めとしたハードウェア技術をより詳細に述べ，三回目ではプログラム中の並列性や並列動作可能なプログラム間での同期を表現するために開発されている並列化言語及び逐次形言語で記述されたプログラムからの並列性抽出等の並列処理におけるソフトウェアについて解説を行い，最終回にあたる第4回では連続システムシミュレーション，ロボットシミュレーションの並列処理の具体例について説明する予定である．

参考文献

1) 村岡洋一：並列処理，昭晃堂（1986）
2) 富田眞治：並列計算機構成論，昭晃堂（1986）
3) 高橋義造，成田誠之助編：コンピュートロール，特集/パラレルプロセッシング，コロナ社（1987）
4) 唐木幸比古編：コンピュートロール，特集/スーパーコンピュータの現在，コロナ社（1987）
5) 情報処理，並列処理技術特集，27-9（1986）
6) 日本物理学会編：スーパーコンピュータ，培風館（1985）
7) 村田，小国，三好，小柳編：工学における数値シミュレーション：スーパーコンピュータの応用，丸善（1988）
8) 村田，小国，唐木編：スーパーコンピュータ：科学技術計算への適用，丸善（1985）
9) 村岡：並列処理，情報処理，20-4, 284/294（1979）
10) R. W. Hockney and C. R. Josshope: Parallel Computers: Architecture, Programming and Algorithms, Adam Hilger (1981)
11) R. W. Hockney and C. R. Josshope: "Parallel Computers 2: Architecture, Programming and Algorithms, Adam Hilger (1988)
12) K. Hwang and F. A. Briggs: Computer Architecture and Parallel Processing, McGraw-Hill (1984)
13) G. J. Lipovski and M. Malak: Parallel Processing, John Wiley and Sons, Inc. (1987)
14) R. G. Babb II, Ed.: Programming Parallel Processing, Addison-Wesley (1988)
15) K. Hwang, Ed.: Tutorial: Supercomputers: Design and Applications, IEEE Computer Society Press (1984)
16) 片山：気象分野への応用，情報処理，22-12, 1128/1138（1981）
17) 甘利俊一：バイオコンピュータ，岩波書店（1986）
18) 麻生英樹：ニューラルネットワーク情報処理，産業図

書 (1988)
19) 広瀬：航空宇宙技術への応用，情報処理，22-12，1119/1127 (1981)
20) J. E. Thornton: Design of a Computer, the Control Data 6600, Scott, Foresman and Company (1970)
21) D. W. Anderson, F. A. Sparacio and R. M. Tomaslo: The IBM System/360 Model 91: Machine Philosophy and Instruction Handling, IBM Journal of Res. and Dev., 11-1, 8/24 (1967)
22) R. W. Tomsulo: An Efficient Algorithm for Exploiting Multiple Arithmetic Units, IBM Journal of Res. and Dev., 11-1, 25/33 (1967)
23) D. W. Anderson, J. G. Earle, R. E. Goldschmidt and D. M. Powers: The IBM System/360 Model 91: Floating-Point Execution Unit, IBM Journal of Res. and Dev., 11-1, 34/53 (1967)
24) J. P. Anderson, S. A. Hoffman, J. Shifman and R. J. Williams: D825-A Multiple Computer System for Command and Control, Proc. AFIPS Fall Joint Computer Conference, 22, 86/96 (1962)
25) G. H. Barnes, R. M. Brown, M. Kato, D. J. Kuch, D. L. Slotnick and R. A. Stokes: The ILLIAC IV Computer, IEEE Trans. on Comput., Vol. C-17, No. 8, 746-757 (1968)
26) W. J. Bouknight et al.: The ILLIAC IV System, Proc. of IEEE, 60-4, 369/388 (1972)
27) S. H. Fuller, R. Swan and W. A. Wulf: The Instrumentation of C. mmp: A Multi-miniprocessor, Proc. of IEEE COMPCON (1973)
28) A. K. Jones and E. F. Gehringer (Editor): CM* Multiprocessor Project: A Research Review, Tech. Rept. CMU-CS-80-131, Carnegie-Mellon Univ., (1980)
29) D. D. Gajski and Peir: Essential Issues in Multiprocessor Systems, IEEE Computers, Vol. C-18, No. 6, 9/27 (1985)
30) 星野力：PAX コンピュータ，オーム社 (1985)
31) 笠原：マルチプロセッサの研究動向，電気学会論文誌 C, Vol. 108-C., No. 2., 96/103 (1988)
32) J. B. Dennis: Dataflow Supercomputer, IEEE Computer, 13-11, 48/56
33) Arvind and K. P. Gostelow: The U Interpreter, IEEE Computers, 15-2, 42/50 (1982)
34) D. D. Gajski, D. J. Kuck, D. A. Padua and R. H. Kuhn: A Second Opinion on Dataflow machines, IEEE Computers, 15-2, 58/70 (1982)
35) K. E. Batcher: Design of a Massively Parallel Processor, IEEE Trans. on Comput. Vol. C-29, No. 9, 836/840 (1980)
36) M. J. Flynn: Some Computer Organization and Their Effectiveness, IEEE Trans. on Comput. C-21. No. 9, 948/960 (1972)
37) H. T. Kung: Why Systolic Architectures, IEEE Computers, Vol. C-15, No. 1, 37/46 (1982)
38) S. G. Tucker: The IBM 3090 System: An Overview, IBM Syst. Journal, 25-1, 4/19 (1986)
39) 馬場等：2 レベルのキャッシュパイプライン処理の工夫で速度を上げた大型コンピュータ ACOS1500，日経エレクトロニクス，233/279 (1985)
40) 小高等：演算パイプラインや 3 階層記憶により高速化を図った M-680/682H の処理方式，日経エレクトロニクス，228/267 (1985)
41) 槌本等：CPU を 1 ボードに実装した大型コンピュータ M-780，日経エレクトロニクス，179/209 (1986)
42) W. D. Hillis: The Connection Machine, MIT Press (1985)
43) K. Hwang: Tutorial Note on Parallel/Vector Processing on Supercomputers, 2nd Int. Conf. on Supercomputing (1987)
44) A. Gottlieb et al.: The NYU Ultracomputer-Designing an MIMD Shared Memory Parallel Computer, IEEE Trans. Comput. Vol. C-32, No. 2, 175/189 (1982)
45) G. F. Pfister et al.: The IBM Research Parallel Processor Prototype (RP3); Introduction and Architecture, Proc. 1985 Int. Conf. on Parallel Processing, (1985)
46) D. J. Kuck et al.: Parallell Supercomputing Today and the Cedar Approach, Science, 231, 967/974 (1986)
47) 笠原，成田，橋本：OSCAR (Optimally Scheduled Advanced multiprocessor) のアーキテクチャ，電子情報通信学会論文誌 D, Vol. J71-D, No. 8, 1440/1445 (1988)
48) J. A. Fisher: Trace Scheduling: A Technique for Global Microcode Compaction, IEEE Trans. C-30, No. 7, 478/490 (1981)
49) H. T. Kung et al.: Warp Architecture and Implementation, Proc. oh 13th Int. Symp. on Computer Architecture, 346/357 (1986)