

TM-0978

並列コンピュータの概念

平田 圭二

January, 1991

© 1991, ICOT

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03)3456-3191 ~ 5
Telex ICOT J32964

Institute for New Generation Computer Technology

並列コンピュータの概念

平田 圭二

1990 年 11 月

1 はじめに

最近、特に並列コンピュータへの期待が高まっているが、その利点を幾つか考えてみよう。(1) 汎用大型計算機の高速度化(今までは4年で約2倍)が、ハードウェア実装技術の面アーキテクチャの面からも難しくなってきたが、さらに高速大容量なコンピュータが必要とされている。(2) VLSI 技術が進み集積度が上がった反面、回路設計、レイアウトには莫大なコストがかかるようになった。従ってできるだけ同じハードウェアモジュールからコンピュータを構成しコストを下げる必要がでてきた。(3) オンラインでかつ高信頼性が要求されるような分野(金融、バンキング、航空、軍用等)で需要が高まり、並列コンピュータがそのようなシステムを構築するための有効な方法であることが認識されてきた。(4) スーパーコンピュータ等の利用経験の蓄積、並列処理・並列コンピュータ自体の研究が進んだことも大きく寄与している。

他にも、(5) 逐次コンピュータのアーキテクチャの改良(パイプライン実行、仮想メモリ、キャッシュ、RISC等)は、並列コンピュータの高速度化にも十分役立つ技術である。(6) 一部の並列コンピュータに実装されているプログラミング言語は、従来の言語(FORTRAN, COBOL等)に対して上位互換性を保っており、過去のソフトウェアのスムーズな継承が可能であった、などが考えられよう。

しかし、技術的にはまだ未解決の問題も多く、ここ数年でようやく商用機が登場してきたという状態である。

本稿の前半では、その脚光を浴びる並列コンピュータについての解説を行い、後半では、その並列コンピュータ研究開発の一例として、現在通産省の第五世代コンピュータプロジェクトで開発中の並列推論マシンを紹介する。

2 並列コンピュータと並列処理

並列コンピュータとは、要するに並列処理のできるコンピュータを指すのだが、並列処理について説明する前に、まず従来の逐次処理につ

いての説明から始めよう。

まず最初に、個々の命令から成るプログラム全体と、計算に必要なデータが1個のメモリに格納されている。1台の制御装置が個々の命令をメモリから1個ずつ取りだし順に実行していく。制御装置は、計算の途中結果をおぼえたりするための作業領域も必要とするので、それらのデータもメモリ中に蓄え、必要に応じて読み出したり書き込んだりする。またユーザや外部記憶との入出力をするための命令も実行し、周辺機器を駆動する。これら一連の処理を1台の制御装置が順に(逐次的に)処理していくことを逐次処理と呼ぶ。この逐次処理を素直に実現したのが逐次コンピュータである。逐次コンピュータを、その発明者の名前を取ってノイマン型コンピュータと呼ぶこともある。

これに対して並列処理とは、複数の制御装置が複数の計算¹を同時に実行する²ことを意味する。そして物理的に並列処理が可能なコンピュータを並列コンピュータと呼ぶ。一般に、並列コンピュータのハードウェア構成は、一定台数以上の等均質のCPUが何らかのネットワークによって結合され、CPU同士はそのネットワークを経由して通信を行うという形態をとる。

3 並列コンピュータの分類

本章では、並列コンピュータをハードウェアの観点に立って、SISD, SIMD, MISD, MIMD という4つのタイプに分類する方法を紹介する。どのタイプの名前も“*I*D”のような形をしており、*印の所にはSかMが入る。IはInstruction Stream(命令列)、DはData Stream(データ列)の略であり、SはSingle(単一)、MはMultiple(複数)の略である。これらは、命令列を複数にして並列処理するか否かと、データ列を並列処理するか否かという $2 \times 2 = 4$ 通りの組み合わせに対応している(表1)。例えばSISDは単一命令列単一データ列タイプのコンピュータのことであり、これはまさしく

¹計算の種類が同じでも異なっても構わない。

²始まる時間や終わりの時間が揃っていても揃ってなくても構わない

表 1: 並列コンピュータの分類

命令列 \ データ列	単 一	複 数
単 一	S I S D	S I M D
複 数	M I S D	M I M D

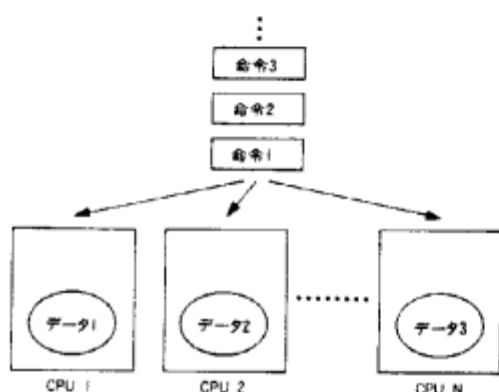


図 1: SIMD 型コンピュータ

逐次コンピュータを差している。MISD は複数命令列単一データ列タイプのコンピュータのことであり、ベルトコンベア上の流れ作業的な処理を思い浮かべて頂きたい。並列コンピュータでは特に SIMD と MIMD が重要である。SIMD とは単一命令列複数データ列タイプのコンピュータのことであり（図 1）、各 CPU にはあらかじめデータがうまく分割して置いてあり、全 CPU で同時に同じ命令を次々と実行することにより、CPU 台数分の並列度を得るハードウェア方式である。一人のかけ声によって皆が同じ動作をするラジオ体操のようなものを連想すれば良いだろう。MIMD とは複数命令列複数データ列タイプのコンピュータのことであり、各 CPU はマイペースで勝手に仕事を実行する方式である（図 2）。人工知能アプリケーションに代表される記号処理の分野では、その処理が非定型でありかつ実行時のプログラムの挙動予測も大変難しいため、効率的な実行のためには、MIMD 型コンピュータで実行時に適切な制御をその場で

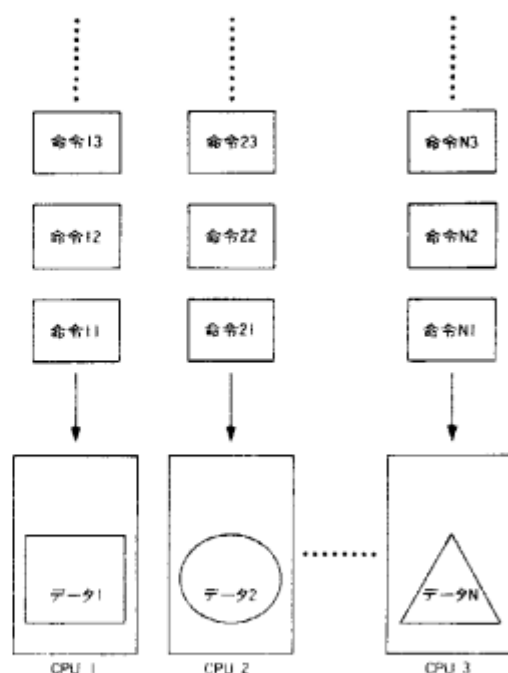


図 2: MIMD 型コンピュータ

決定するしかない。従って後で述べるように、MIMD 型の並列コンピュータでは特にソフトウェアが大きな比重を占めるのである。

4 並列コンピュータ実現上の問題点

1 台の並列コンピュータが N 個の CPU から成っていたとすると、その並列コンピュータ全体では CPU 1 台の速度の N 倍が期待されるが、通常実際の速度比は N より小さい値にとどまる。この実際の速度比を「台数効果」と呼んでいる。複数の計算を並列に実行すると言っても、元々は大きな 1 つの仕事を適当な大きさに分割して処理するのであるから、台数効果ができるだけ N に近づくように仕事をうまく分割して個々の CPU に割り振らなければならない。次のような本質的問題点が浮かび上がってくる。

- (1) 複数のプロセッサが複数の仕事を同時に実行するので、メモリに対するアクセスの競合を起こす可能性がある。
- (2) プロセッサ同士を接続しているネットワーク上の通信が混雑し、性能が低下する可能性がある。
- (3) 逐次コンピュータならば同一メモリ中に存在するデータでも、並列コンピュータでは

遠くの別のメモリ中に存在する可能性があり、データアクセスに遅延が生じる。

- (4) プロセッサ間で忙しさが不均一になった時、仕事の再配置をすること自体大きなコストがかかる。

これらを解決するには、遠距離の通信が少なくなるよう適切に仕事を分散し、各プロセッサが協調して仕事を進めるように実行を制御しなければならない。換言すれば「できるだけ処理の局所性を保って仕事を分散させる」ということである。これは一見矛盾するようにも聞こえる。普通、一つの問題をより多くの CPU 間にバラまいて並列分散処理すると、並列にしたための無駄がより多く生じるものであるが、一方ではより多くの計算パワーを手に入れることができる。上の文は、処理を分割する時、場所的にも時間的にもできるだけ局所的に納まるように分割しておけば、CPU 台数 N を大きくして行った時でも、並列にしたための無駄をそれほど増やさずに十分な台数効果が得られるだろうということを示唆している。もちろん、プログラムに内在している並列性が元々 CPU 台数 N を下回っている時には台数効果を N にすることは不可能である。そのような場合は、問題の定式化、アルゴリズムの見直しまで立ち戻って考え直す必要があるだろう。

ある特定の狭い応用分野、例えば画像処理、科学技術計算、シミュレーション等、に限れば専用的高速並列コンピュータはすでに実用化の段階にある。さらにチェス専用的高速並列コンピュータというのも開発されている。これらの応用プログラムでは、実行前にプログラムの挙動をかなり予測することができ、分割された個々の処理が定型的で、プログラムの並列性が CPU 台数 N より十分大きく、アルゴリズムの構造が並列ハードウェアにマッピングし易かったために、効率的な仕事の分散や並列実行制御の計画を立てることができたのである。

しかし、汎用の並列コンピュータという視点に立つと、どんな問題に対しても有効であるような一般的な仕事の分散や並列実行制御の方式というものが必要となってくる。しかし、ある種の応用には大変有効な処理方式でも、別のタイプの応用には全く歯が立たないというのが通例である。また事前に十分な計画を立てられないような問題に対しては、実行中の動的な制御しかないが、そのための一般的な方式も見出されていない。ハードウェア構成的には汎用であっても、ソフトウェアも含めたシステム全体として十分に性能が発揮されて汎用と言えるような並

列コンピュータは残念ながらまだ見当たらないのである。これは、まさにシステム全体の問題であり、逆にここが並列コンピュータシステム設計の醍醐味であるとも言えよう。

5 第五世代コンピュータと並列プログラミング

通産省の第五世代コンピュータプロジェクト(1982年から1992年までの10年計画)の推進母体である(財)新世代コンピュータ技術開発機構(ICOT)では、並列推論マシン(Parallel Inference Machine, PIM)の開発を進めている。PIMは、やはりICOTが開発したKL1というプログラミング言語専用のMIMD型コンピュータである。このKL1という言語は、大規模な非定型問題を記述でき、並列コンピュータ上での効率的な実行を可能とするように設計されている。またKL1は、Prologと同じく論理に基礎をおく論理型プログラミング言語の一つでもある。並列推論マシンの「推論」という言葉は、KL1が論理的推論をその基本演算に持ち、PIMがそれを機械語レベルで高速に実行できる専用マシンであるところに由来する。

KL1で書かれた個々のプロセスは基本的に並列に実行され、明示的に指定した時だけ逐次処理が行われる。また、プログラマは、メモリ管理、並列に実行されるプロセス間のスケジューリング、プロセス間のデータ受け渡しのための同期操作をわざわざ記述する必要が無い。このためプログラマは、処理の本質的な部分だけを簡潔に記述することができる。この処理の本質的な部分は、問題を解く並列アルゴリズムの部分と、前章で述べたような動的な実行制御指定の部分から成る。これらがともにKL1プログラムとして陽に書けることは、並列アルゴリズム及び動的な実行制御の研究が大変容易に行えることを意味する。実際、PIMのオペレーティングシステムや多くの応用プログラムがKL1で記述され、これらの研究に大きく貢献している。並列応用プログラムの典型的な挙動もある程度解明されつつある。

6 ICOTにおける並列推論マシンの研究開発

汎用(記号処理用)のMIMD型並列コンピュータを構築するには、どういったアーキテクチャが適しているかを明らかにし、MIMD型並列コンピュータの制御方式を確立する必要がある。しかし、現時点ではこれらに対する最適解は見つかっていない。ICOTではアーキテクチャ

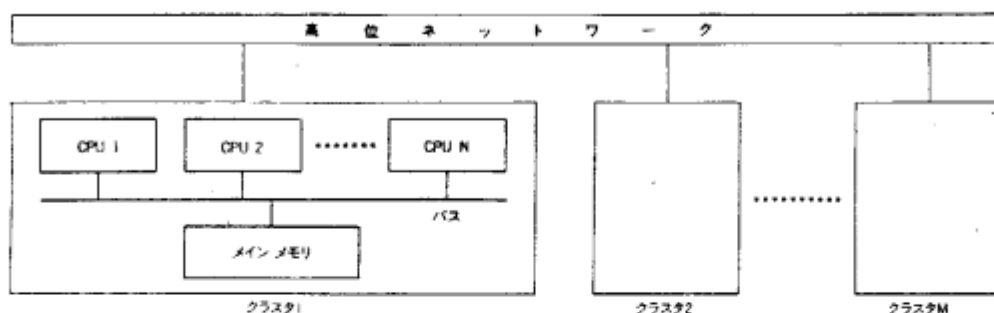


図3: PIMの基本アーキテクチャ

の異なるマシンを複数台開発し、その技術的選択肢を比較・評価することを試みている。複数台の PIM のプロトタイプは、ある基本的アーキテクチャに従うバリエーションとして位置付けられる。これら PIM に共通するアーキテクチャの特徴は階層性であり（図 ref:fig:pim）、これは「処理の局所性」をハードウェアで支援している。まず、1 台から 10 台程度の CPU が 1 本のバスで結合されメインメモリを共有し、クラスタと呼ばれる構造を作る（バスは下位のネットワークと見なせる）。さらに数個から数 10 個のクラスタが何らかの（高位の）ネットワークで結合される。クラスタとクラスタを結合する様々な高位ネットワークのパターン（トポロジ）を考えることができる。クラスタ内での CPU 間の通信は、バス・メインメモリ経由で行われるので、遅延を少なく転送量を大きくできる。一方、クラスタ間通信については、中継地点があったり距離的に離れているために、遅延が大きく転送量もあまり大きくできない。

このような階層的アーキテクチャを採用した背景には、数 100 台以上の CPU を一様に低遅延・高転送量で結合できるようなネットワークは物理的に実現不可能、という事実が存在するからである。従って、時間的・空間的に近くに存在するデータへのアクセスは、転送量がボトルネックにならない範囲で、できるだけクラスタ内で閉じるように実行を制御する。そして、転送量がボトルネックになって来た場合は、通信があまり生じないと予想される仕事から他のクラスタへ分散して行く。つまり、マシンパワーが許容する範囲で、処理を局所的に封じ込めるハードウェア構成法と言える。

我々が PIM を設計する際、技術的選択肢として考えたのは、主に、機械語命令セット、高位ネットワークのトポロジ、1 クラスタ内 CPU 数、一貫性キャッシュの方式、といった項目であった。各 PIM では、これらの選択肢について異なる

方式が採用されている。ここで一貫性キャッシュについて簡単にふれておこう。クラスタ内の複数の CPU がメインメモリ中の同一データにアクセスすると、バス上でアクセスの衝突が起きる。そこでアクセスの衝突を減らすために、CPU に（逐次コンピュータと同様の）キャッシュを導入するのだが、複数の CPU で同じデータにアクセスすることを考えると、各キャッシュ中の同一データの値が矛盾しないような工夫が必要になる。そのために一貫性キャッシュでは、バス上のアクセスを常に監視し、ハードウェア的にキャッシュ中のデータの一貫性を保つようにしているのである。また PIM では、KL1 を高速に実行するために、KL1 特有の演算操作や頻繁に実行されるシーケンスをハードウェア化している。

7 並列コンピュータの将来

実用的な並列コンピュータを目指すには、単にハードウェアだけでなくソフトウェアの研究開発も重要であることがご理解頂けたと思う。現在では、超並列コンピュータという 10 万台規模のものまで研究対象になってきており、ハードウェア、アーキテクチャはもちろん、特に並列ソフトウェア技術に対する期待や要求はますます高まるばかりである。

現在のソフトウェア教育によって、プログラマは自然と逐次型の思考に馴らされている。そのような教育を受けたプログラマが、後で並列型の思考を体得するのは非常に困難であろう。超並列システムは、従来とは全く異なる新しい並列ソフトウェア技術を必要としており、ICOT で研究開発しているような並列を基本とするソフトウェア技術がその候補の一つとなろう。将来、逐次型コンピュータを知らない並列ソフトウェア教育のみを身に付けたプログラマが活躍し出す頃、ようやく並列コンピュータ文化が開くのかも知れない。