並列ゼミ'95: H. Morse, Practical Parallel Computing,Chap. 1

報告: 久野

1995.9.5

前書き

「これくらいの計算量をこなさなければならない。並列計算機を使うべきかどうか? 使うとしたら、どの機種?」

- どんな種類のハードがあるの?
- 計算内容や性能目標にどれくらい適合する?
- 解きたい問題は並列処理に向く/向かない?
- 既存のシステムとうまく組合せられる?
- アルゴリズム、OS、開発ツール、環境、テスト、ベンチマーク?
- リスク?
- I/O?

著者は実際にそういう立場に立たされてきたので、「こういう本があったらなあ」と思って書いたのがこの本。

- Part 1: 基本概念
- Part 2: ソフトウェア
- Part 3: マネージメント

本書はアジではない。本当に並列システムを採用すべきかどうかについて、バランスのとれた視点を提供する。

1 MPPの現状と将来

並列処理は…「苦難に満ちている」が「避けられない」。現状は伝統的なアーキテクチャのメーカと並列マシンのメーカの綱引きだが、伝統派のメーカも並列の研究はしている。なぜ「避けられない」か?

1.1 技術的なトレンド→並列指向

「並列」とは? 異なる部品どうしの並行動作? それだったら10万円のPCでもやっている。

- 計算機の能力 = 1CPUの能力× CPU台数。
- MPPとは、比較的遅いCPUを多数使って高性能を達成。
- 伝統的なアプローチでは、比較的少数の1CPUを高速化して済ませる。

どちらも利点と欠点がある。

1.1.1 クロックの限界

- クロック …6MHz(1985) → 50MHz(1993) → 1GHz(2001?)
- 同一構造のCPUでは、性能はクロック数の比率以上に向上することはない。
- メモリ帯域やバス容量が制約しないなら、クロック数に比例して性能が向上。
- クロック数向上の流れに乗っていれば、必要な性能は待っていれば得られ、従ってMPPは不必要???
- 逆に、クロック数向上には物理的上限があるから、ゆえに1CPU性能の向上は限界があり、ゆえにMPPは不可欠。
- 簡単にいえば、パッケージングと冷却の問題。クレイが偉大なのは、速く動くことではなく「融けずに動く」こと。

スーパーコンピュータの世界では、クロックの壁は明らか。

- Cray Y-MP: 167MHz(1986) — Cary C-90: 250MHz(1991)。たった 50%しか向上していない。(CPU を 8→32 と増加させて性能をかせいでいる。)
- Cray-3 はガリウム砒素半導体で 1nsec(1GHz) クロックと言われていたが、2nsec にまで後退して未だに完成していない。

おそらく、1GHz あたりにクロックの壁。結局、TFLOPS マシンには CPU 数の増加しかない!

1.1.2 信頼性

MPP は信頼性の点でも有利。メインフレームやスーパーコンでは CPU は多数のチップを組み合わせて構成。1CPU の性能向上のため様々な代償。

- 部品数の増加
- 実装密度の低下
- 実装空間の増大
- 電力消費の増大
- 配線数の増加

MPP ではできあいの CMOS CPU と DRAM が使われる。

- 高密度、低消費電力、廉価、安定、量産。
- 結局、1CPU は遅くてもその分以上に多数詰め込める。

しかも、CMOS CPU/DRAM は市場の要求によってすごい勢いで進化する。現在は 0.7 μ 線幅で 2,000,000〜3,000,000 トランジスタ/チップ。2000 年には 0.3 μ 線幅でその分機能やオンチップメモリが増大し、クロックも向上する。DRAM もまもなく 256MB があたりまえに。これらはすべて MPP にそのまま利用できる。

> 例: nCube 2→1 ノードが 11 チップ (CMOS CPU + 10 DRAM) → 1.25”× 3.25” ボード。これを 64 個載せたマザーボード全体で、200MFLOPS/400MIPS、1GB メモリ。消費電力は 200W で、空気冷却。

そして、できあいの量産部品で点数も少ない→信頼性が高い。当然。

1.1.3 メモリ帯域

T_R: メモリのアクセス時間。この逆数→メモリ帯域。1 チップあたり DRAM だと 80nsec → 12.5MW/sec、SRAM だと 10nsec → 100MW/sec くらい。これにメモリ幅を掛ける。

> 例: 486 PC で 32bit バス (1 アクセス 4 バイト) とすると、$12.5MW/sec \times 4B/W = 50MB/sec$。これはクロック数や CPU 速度とは無関係。

メモリ帯域を増やすには、ワード幅を増やす。クレイは Y-MP から C-90 で 8 バイト→16 バイトにした。または、T_R を減らす。Y-MP で 15nsec、C-90 で 6nsec。しかし、DRAM では容量は順調に増えているが T_R は 80nsec とほぼ一定のまま。そのため、CPU 速度とメモリアクセス時間のギャップが増大。もう 1 つは、メモリバンクを増やすこと。

一般に、1FLOP あたり 2 ワードのメモリ帯域が必要といわれる。

> 例: 倍精度 (8 バイト/語) の 20GFLOPS マシンでは、$40GW/sec = 320GB/sec$ の帯域が必要。10nsec の SRAM を使うとして、$40GW/sec \div 100MW/sec = 400$ つまり 400 バンク必要。80nsec DRAM だと 3200 バンクになる。

しかし、どうやって多数のバンクを少数の CPU につなぐのか? 旧来の方法では、SRAM で必要なバンク数を減らした上で、10〜20 個の CPU と各バンクの間を高速のクロスバースイッチで接続する。(典型的な共有メモリマシン。)

MPP ではずっとエレガント。つまり、**各バンクに CPU をつけてしまう!**。つまり、多数の CPU がそれぞれローカルにメモリを持つ (分散メモリ)。

もちろん、弱点はローカルメモリ以外の場所をアクセスするのが大変なこと。これをどうするかはハードだけでなくソフトやアプリケーションの問題でもある→後で詳しく出て来る。

こういう考え方は RAID にちょっと似ている。

1.2 商業的な成功の阻害要因

MPP は現にまだ普及していない…スーパーコン市場が$1,500,000,000 として、その 20%程度が MPP。「もうかる」とは言えない。なお、CPU 数が 4〜10 程度の共有メモリマシン (SMP) を入れればずっと多いが、SMP は MPP とはだいぶ違う→後で詳しく出て来る。

しかし、分子モデリングとか OLTP など MPP が受け入れられている用途も多い。しかし「特殊用途マシン」と呼ばれてしまう。だがそういうことは決してない (伝統的マシンのメーカが MPP をけなすのに使うことば)。

MPP が効率良く使えるかどうかは、**問題の性質**よりは**問題のサイズ**による面が大きい (詳しく出て来る)。そして、どのような分野であれ、問題が十分大きなものであれば MPP で扱うのに適さない分野というのはほとんどない。例えば:

- AI — ニューロ、CBR、プロダクションシステム、ロボティクス、パターン認識
- SV
- 有限要素法 — 常/偏微分方程式、電磁方程式、流体なども
- ふつうの計算機アルゴリズム — ソート、サーチ、ツリー、グラフ等
- 最適化 — LP、DP、IP
- DB — SQL、OLTP
- シミュレーション — イベントドリブン、モンテカルロ法
- 信号処理 — FFT、イメージ処理、CG、文字認識、通信

結局、MPP に対する「ためらい」は「できない」からではなく、もっと実際的な点から来ている。具体的には:

- アプリケーションソフトがない
- 標準化されてない
- ベンダーの不安定さ

1.2.1 アプリケーションソフトの入手可能性

高性能 WS が売れるようになったのは、十分な機能/品質のサードパーティソフトが流通するようになったから。MPP が売れないのはそういうのがまだないから。で、結局作るしかない。なぜないか?

1. 台数が少ない→ソフトハウスは作っても売れない。
2. 移植が大変→旧来のソフトを MPP でまともに動くように移植するのは簡単でない。
3. 標準がない→ UNIX とネットワーク機能については OK だが、言語とアーキテクチャモデルについてはひどく不統一。

   > 例: 同じソートでも 2D メッシュのマシンではメッシュソート、ハイパーキューブマシンでは 2 分木ソートが向いている。並列マシンだからどれも同じ、とはいかない。

4. 検証→ MPP と旧来のコードで値が違ったらどうする? アルゴリズムが変わって計算順序が変わると (変えないと意味がない)、結果も変わってしまうことが多い。
5. ベンダーの不安定→せっかくソフトを作ってもベンダーが潰れたり、新機種のマシンが旧機種と全然違ってしまうと…しかしベンダーは「そのつどゼロから出発して速いマシンを」というのが多い。でも最近は互換性の考慮も行われる。

あと、ビジネスではなく「好きだから」MPP をやってる、という人がいるのも阻害要因ではある。

1.2.2 標準化

OS とネットワークについては、UNIX と NFS、EtherNet、TCP/IP、HIPPI、FDDI、ATM/SONET など標準が確立している。だからサーバマシンとして MPP を導入するのは容易。

言語については、ベンダーがでんでばらばらに C や Fortran の拡張版を提供しているので標準がないに等し

い。いくらかは共通言語のようなものもある。Express、Linda、STRAND、PVM など。これらはライブラリパッケージで、これらに基づいて書かれたプログラムは多くの MPP プラットフォームでそのまま動かせる。

もちろん、ハードウェアがひどく多様だから、というのが根本原因。アルゴリズムから変えなければならないことが多い→本書のテーマ。

1.2.3 ベンダーの不安定

つぶれた会社はいっぱい。AMetek、BiiN、Multiflow、WabeTracer、Alliant、BBN、FPS、Myrias。そもそも独立して採算が取れる状態でないので、スポンサーの意向を伺わなければならない。スポンサーの CPU を採用したり、スポンサーが提唱しているソフトを組み込んだりする。これからの課題。

1.3 傾向と課題

今後どうなるか?

1.3.1 ハイエンドシステム

スピード競争の時代。「世界最速のスーパーコン競争」は日本も結構がんばったが。現在は「世界初の TFLOPS マシン競争」。このマシンは MPP になることは間違いない。図 1-3 を見よ。

1.3.2 ローエンドシステム

ローエンドの世界は経済性。$100/MFLOP くらい (1993)。ということは、$100,000 で 2.5GFLOPS — Cray Y-MP の性能 — が買える。それをどう使うかはまだ未知だが、社会的インパクトはこちらが大。

1.3.3 政府と政治

1991、USA → HPCCI (High Performance Computing and Communications Initiative)。ゴア副大統領。高性能システムの開発にお金を出す。

賛成意見: (1) 資金を提供しなければ MPP 産業はつぶれてしまう。(2) 安保のため、高性能計算機技術を保持することは米国にとって重要。

反対意見: 政府資金をつぎこまないと成り立たない産業は不健全。

本当のところは…? 次第に、廉価なシステムを多数売って自立できる方向に?