計算機アーキテクチャ 83－6
（1990．7．18）

TOP-1における流体問題解析

大澤 暁
日本アイ・ビー・エム株式会社 東京基礎研究所

Abstract

TOP-1はCPUを10台使用した共有メモリ型マルチプロセッサ・ワークステーションである。このようなシステムを用いて並列に実行できるアプリケーション・プログラムの開発・研究を行なうことは、今後の並列処理システムの開発にとって意義がある。本報告では、アプリケーションの一例として流体問題をとりあげ、TOP-1を用いて並列に実行した結果について述べる。TOP-1には簡単なパフォマンス測定ツールが用意されており、これらを用いて実際に評価測定した結果から、並列化に伴うさまざまな効果についても検討する。

Solution of Fluid Dynamics on TOP-1

Gyo OHSAWA
IBM Research, Tokyo Resrarch Laboratory, IBM Japan, Ltd.
5-19, Sambancho, Chiyoda-ku, Tokyo 102 JAPAN

Abstract

TOP-1(TOkyo research parallel Processor-1) is a high-performance multiprocessor workstation which has 10 CPUs connected to a shared memory. Developing of parallel applications with such a workstation is indispensable for mini-super computing environments in near future. This paper shows executing results of a real parallel application, Flow-solver on TOP-1 which analyzes two-dimensional flow problems. TOP-1 has some performance monitoring tools and we also discuss effects of parallel calculations with the execution results.

1 はじめに

近年の計算機に関するめざましい技術革新により、高性能なマイクロプロセッサを多数用いたマルチプロセッサ・システムが開発されるようになってきた[1]。これらのシステムには、小規模なものから大規模なものまでいろいろなものが、アプリケーション用途別にまたは用途を問わず発表されている。一方ではワークステーションにおける計算能力の進歩もめざましく、最近では高性能なRISCチップを用いたシステムが発表されている。さらには、複数の処理装置(CPU)を持つ並列処理システムの研究も盛んに行なわれている。従って近い将来、非常に計算量の多い科学技術計算もこのようなシステムを用いて手近に行なえるようになると考えられる。

そのような意味から、実際に並列処理システムを用いてアプリケーション・プログラムを実行し、並列処理にともなうさまざまな問題の研究を行なうことは、今後のシステム開発の上で大きな意義がある。本報告ではアプリケーションの一例として2次元の流体問題をとりあげ、これをTOP-1上で解いた結果について述べる。TOP-1は東京基礎研究所で開発された並列処理ワークステーションで、10台のIntel 80386を用いたシステムである。このシステムにはパフォーマンスを測定するツールがいくつか用意されており、実際にアプリケーション・プログラムを実行しながら、並列化効率に関するいくつかの項目について測定した。本報告ではこれらの結果をもとに、効率に関する問題について考察する。

以下2章において、TOP-1のシステム構成について簡単に述べる。3章では、並列プログラミング・並列実行環境について概説し、4章で実際に実行した流体アプリケーションの説明を行なう。5章において、評価測定を行なうための観点とその方法について概説する。そして6章で、実行結果およびその考察を行なう。

2 システム構成

マルチプロセッサ・ワークステーションTOP-1にはプロセッサカードが10枚あり、その各々にはIntel 80386の他に、浮動小数点演算プロセッサとしてIntel 80387およびWeitek 1167が用意されている[2]。

各プロセッサは、128kバイトのキャッシュメモリを有している。そのキャッシュのモードには、次の2通りがある。

- 更新型プロトコル
  共有データへの書き込みが生じた時、システム中の他のすべてのコピーを更新する。

- 無効化型プロトコル
  共有データへの書き込みが生じた時、システム中の他のすべてのコピーを無効化することによって、一致性を保つ。

この2つのプロトコルは同時にサポートされており、実動作時にソフトウェアによって切替えることができる。

共有メモリのサイズは32Mバイトで、187.5nsのサイクル・タイムで使用される。これらのプロセッサカードとメモリが、64ビット幅・2ウェイのバスで接続されたいわゆる共有メモリ型密結合マルチプロセッサ・システムとなっている。

次に、ソフトウェアの構成について述べる[3]。現在のTOP-1オペレーティング・システムは、UNIX[1]をベースにマルチプロセッサ用に拡張したもので、主に次の特徴がある。

- あるひとつの特定のプロセッサ(カーネル・プロセッサ)は、カーネル・コードのみを実行する。

- もうひとつ別の特定のプロセッサは、システムのモニタリングのために使用される。

- 残りの8個のプロセッサ(ユーザ・プロセッサ)は、ユーザ・プロセスを実行する。従って同時に並列実行できるプロセスの数は、最大8となる。

ユーザ・プロセッサで実行していたプロセスがシステムコールなどを発すると、カーネル・プロセッサに移動する。そしてカーネルでの処理が終了すると、ユーザ・プロセッサに戻るが、これらの制御はオペレーティング・システム内で自動的に行なわれる。また科学技術計算において、四則演算などの計算を行なっている限りは、プロセッサ移動は起きない。

今回の流体問題アプリケーションは上記のシステムを用いて行ない、実行時のプロセス数を1ないし8に変えることにより処理時間を測定した。

[1] UNIXはAT&T社が開発し、ライセンスしています。

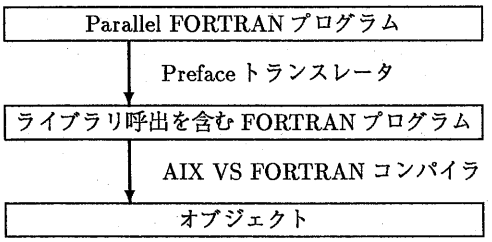

図 1: FORTRAN プログラム実行環境

3 並列実行環境

3.1 並列記述言語

IBM[2]では、Parallel FORTRAN という言語が IBM 3090 上に用意されている。VS FORTRAN の拡張として、タスクの生成・消滅・スケジュールや、ループの並列実行、ブロック単位の並列実行をプログラマが陽に記述することができるようになっている。TOP-1 ではそのサブセットとして、次のシンタックスが使用できる。

- PARALLEL LOOP
  通常の DO 文の代わりに PARALLEL LOOP 文を用いることによって、ループ内が iteration 単位で同時並列実行される。

- PARALLEL CASES
  いくつかの実行文のかたまりを単位として、同時並列実行する時に使用する。

この他、並列実行部分における排他的処理の指定などの必要最小限の機能が用意されている。従って、比較的並列化し易いアルゴリズムを用いている、流体問題などの科学技術計算では、Parallel FORTRAN を用いて簡単に並列プログラムを記述することができる。

3.2 並列実行環境

Parallel FORTRAN で記述されたプログラムを TOP-1 上で実行できるように、このプログラムを VS FORTRAN に変換するトランスレータが用意されている。このトランスレータは **Preface** と呼ばれる。変換されたプログラムには、並列実行のための各種ライブラリ・ルーチンを呼出すような記述が埋め込まれている。TOP-1 の上では AIX[3]VS FORTRAN コンパイラが使用できるので、後は通常と同じ操作でコンパイルすることによって、実行モジュールを作成することができる。これをブロック図で示すと、図 1のようになる。

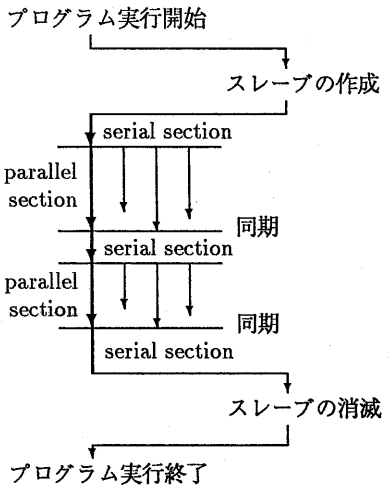

図 2: Preface 実行モデル

3.3 並列実行モデル

通常、プログラムには並列に実行できない部分 (serial section) と並列に実行できる部分 (parallel section) が存在する。複数個のプロセスでこれを扱う方法として、主に次の 2 通りの方法がある。

1. serial section はすべてのプロセスが同時に同じコードを実行し、parallel section は、それぞれ個別に対応するコードを同時並列的に実行する。

2. serial section はある特定のプロセスのみが実行し、残りのプロセスは何もしない。parallel section は、すべてのプロセスによりそれぞれのコードを同時並列的に実行する。

TOP-1 は汎用目的のワークステーションであるから、あるアプリケーション・プログラムを実行している間に不必要なコードを実行することは効率の低下を招き好ましくない。そこで TOP-1 では、2. の

[2]IBM は米国 IBM 社の登録商標です。

[3]AIX は米国 IBM 社の商標です。

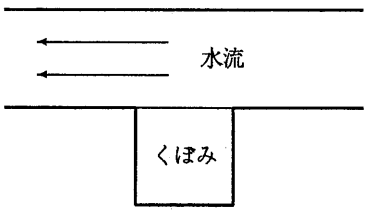

図 3: Cavity Flow

方法をとっている。この実行モデルは Preface トランスレータおよび Preface ライブラリにより実現されているので『Preface 実行モデル』と呼ばれ、図2はこれを図にしたものである。

Preface 実行モデルでは、ユーザが使用するプロセスの数は、プログラムの実行時に環境変数を用いて指定できるようになっている。これにより、プロセス数を変えてもコンパイルし直す必要がないという利点がある。

アプリケーション・プログラムの実行が開始されると、まず Preface のライブラリは次の操作を行なう。

1. 必要な共有変数領域を確保し、データの初期化を行なう。
2. 環境変数で指定された数だけ、プロセスを作成する。この時、親プロセスがマスタ、fork された子プロセスがスレーブとなる。
3. マスタ・プロセスはその他の初期化を行ない、プログラム本体の実行に移る。
4. スレーブ・プロセスは、最初の serial section の間はアイドル状態となる。

図2からもわかるように、serial section においてはスレーブ・プロセスはアイドル状態であり、parallel section に入ると実行状態になる。parallel section の終わりで同期がとられ、parallel section 内のすべてのコードが終了するとマスタ・プロセスは次の serial section の実行に移る。このようなプロセスの管理は、すべてマスタ・プロセスが Preface のライブラリを用いて行なっている。

4 流体問題プログラム

今回 TOP-1 上で実行した流体問題プログラムは2次元非圧縮性粘性流体問題で、次に示す方法で離散化・計算を行なっている。

1. 対象とする物理領域をスタッガード・メッシュにより分割する。
2. 運動方程式を差分近似により陽的に解く。
3. 反復計算により、連続の式を満たすように圧力と速度を修正する。

対象とした問題は、いわゆる Cavity Flow である。これは図3のように、流れの底にくぼみがあった時に上方を流れる水流によって、くぼみの中の水流がどのように誘発されるかを計算によりシミュレートするものである。計算条件や境界条件などは、次に示すものを用いた。

- 計算対象領域は、くぼみの部分の正方領域とする。
- メッシュ・サイズは 48×48 とする。
- シミュレーションの計算時間ステップ幅は 0.005 秒、計算終了時刻は 10 秒とする。
- レイノルズ数は 400 とする。
- くぼみの部分の境界条件は「スリップせず」とする。

5 性能評価測定法

2章でも述べたように、現在 TOP-1 のキャッシュ・モードには更新型と無効化型の2つのプロトコルが存在する。そこでこれらのプロトコルによる差を調べるために、同じプログラムを両方のキャッシュ・モードで実行して計算所要時間を測定した。

また今回のアプリケーション・プログラムはコード上 parallel section が5ヶ所存在し、それらが反復計算により何回も繰り返し実行される。従ってその前後で Preface ライブラリ・ルーチンがあると、ある程度のオーバヘッドが予想される。そこで各 section 単位で実行時間を計測し、serial section・parallel section・オーバヘッドなどの割合を調べた。実行時間の計測は、各計測点での utime および stime を調べることにより行なった。

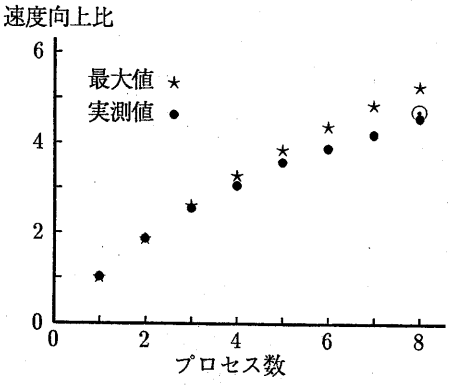

図 4: 処理速度向上比

プロセス数	1	8
無効化型	3461.0 (37.1)	743.9 (1.4)
更新型	3475.7 (15.8)	767.5 (9.0)

表 1: 平均処理時間 (秒) (カッコ内は標準偏差)

さらにはTOP-1 上に用意されているキャッシュの統計用ルーチンを同時に動かし、いろいろなキャッシュ情報を収集した。以上の実行ではすべて、浮動小数点演算プロセッサとして Intel 80387 を用いた。次章から、これらの結果について示す。

6 実行結果

図 4は、同じプログラムをプロセス数を変えて実行し、処理速度向上比を調べた結果である。この図では、プロセス数 1 の時の処理時間を 1 とした。また、キャッシュは更新型プロトコルを用いた。プロセス数が増えるに従い性能の劣化がみられるが、主に次の原因によるものである。

- serial section の存在
- 並列実行にともなうオーバヘッド

表 1は、キャッシュ・モードによる処理時間の違いを示している。この表から更新型プロトコルを用いると、無効化型より約 3%時間のかかることがわかる。並列計算を行なうと処理時間には多少のバラツキが存在するが、今回は同じプログラムを 5 回実行して、その平均をとった。

表 2は、section 単位で実行時間を計測した結果である。これによると、Preface によるオーバヘッドはそれほど大きくはないことがわかる。これはアプリケーション・プログラムおよび計算条件などにより異なるが、今回の場合はプロセス数 8 の時で、全体の 2.8%であった。表 2によるとプロセス数 1 の時、parallel section でない部分が全体の約 7.5%ある。この部分は並列に実行できないわけであるから、これを考慮して次式により処理速度向上比を算出すると、図 4の「最大値」のカーブが得られる。

$$\frac{1}{0.075 + \frac{1-0.075}{\text{プロセス数}}}$$

しかしプロセス数が 8 になると、これが 10.1%に増加する。この 10.1%という値で計算すると、最大値は⊙まで低下する。

次に TOP-1 上に用意されているキャッシュの統計用ルーチンで、キャッシュヒット率などを測定すると表 3となる。この表から次のことがわかる。

- データの「読み込み」・「書き込み」共に、ほとんど 100%ヒットしている。
- shared write 率は「 write ヒット時における shared write の割合」であるが、プロセス数 1 の時はこれが 0 となる。
- プロセス数 8 の場合、更新型の shared write 率の方が無効化型よりかなり大きくなる。

shared write 率が大きくなるのは、主に次の原因によるものである。

- 毎回反復終了後、serial section において全てのデータを更新している。

プロセス数	1	8
serial section	225.4 (6.5)	239.5 (6.5)
parallel section	3189.0 (92.5)	3308.8 (89.9)
Preface オーバヘッド	31.4 (0.9)	102.3 (2.8)
ロック待ち	0.9 (0.0)	30.8 (0.8)

表 2: 各 section における平均実行時間 (秒) (カッコ内は%)

プロセス数	1	8							
プロセッサ番号	1	1	2	3	4	5	6	7	8
read ヒット率	99.1	96.3	96.0	96.0	96.1	96.1	96.2	95.1	93.8
write ヒット率	99.3	98.7	99.7	99.8	99.6	99.8	99.8	99.4	98.2
shared write 率	0.0	43.2	54.4	48.8	54.3	54.5	54.3	45.6	39.1

(a) 無効化型プロトコル

プロセス数	1	8							
プロセッサ番号	1	1	2	3	4	5	6	7	8
read ヒット率	98.9	99.4	99.9	99.2	99.3	99.5	99.6	99.0	98.6
write ヒット率	98.8	99.1	99.5	99.9	99.9	99.9	99.9	99.4	97.1
shared write 率	0.0	90.6	95.1	98.4	97.9	98.4	98.4	94.1	91.3

(b) 更新型プロトコル

表 3: キャッシュヒット率 (%)

- parallel section ではいわゆる CHUNK オプションを用いて各プロセスがそれぞれの iteration 単位を実行するが、どの単位を実行するかは非決定的である。

このために、更新型プロトコルを用いると他のプロセッサ上にあるキャッシュのコピーをいつも更新することになり、結果として shared write 率が大きくなるわけである。一方キャッシュ・ラインが 8 バイトなので、無効化型では 4 バイト・データの書き込みが起こると、連続する次のデータは shared ではなくなる。無効化型で shared write 率が 50%前後なのはこのためである。表 1 において更新型の処理時間が長くなったのは、このようなキャッシュの特性によるものも大きい。

7 おわりに

本報告では並列処理ワークステーション TOP-1 を用いて科学技術計算を行なう一例として、2 次元の流体問題をとりあげた。そして TOP-1 上での実行環境を説明し、実行結果を述べた。TOP-1 上にはいくつかの評価測定用ツールが用意されており、それらを用いて簡単な評価を行なった。現在 TOP-1 では、浮動小数点演算プロセッサとして Weitek 1167 をサポートするのは C のみである。今回のアプリケーション・プログラムをいくつかの計算機・コンパイラを用いて実験した結果は [4] に報告されている。

また現在では、今回報告したものより細かな section 単位でキャッシュの統計情報を収集し、評価を行なっている最中である。これら使用経験をふまえて問題点を整理し、次のバージョンでオペレーティング・システムを含めた種々の改良を加え、実行環境の改善を図る予定である。

参考文献

[1] R. W. Hockney and C. R. Jesshohe, "Parallel Computers," Adam Hilger, Ltd., Bristol (1981)

[2] 清水, 他, "高性能マルチプロセッサ・ワークステーション TOP-1," 並列処理シンポジウム JSPP'89, pp.151~162 (1989)

[3] 穂積, 他, "TOP-1 オペレーティング・システム-基本方針," 第 39 回情処全大論文集, pp.1199~1200 (1989)

[4] 大澤, "TOP-1 における流体問題解析," 第 41 回情処全大論文集 (1990)