PIMOSの評価

2P-7

佐藤裕幸[1], 星田昌紀[2], 近山隆[2], 藤瀬哲朗[3]
1: (財) 新世代コンピュータ技術開発機構 (現在は三菱電機 (株) 情報電子研究所)
2: (財) 新世代コンピュータ技術開発機構 3: (株) 三菱総合研究所

1 はじめに

第5世代コンピュータ・プロジェクトでは、知識情報処理を行うに当たって必要とされる高度な計算能力を提供するために、並列推論マシン PIM [1] の研究が進められている。この並列推論マシンの能力を最大限に引き出すためには、高並列に動作するプログラムを効率的に制御するオペレーティング・システム (OS) が必要不可欠である。

PIMOS(Parallel Inference Machine Operating System) [2, 3] は、この目的のために設計された並列推論マシン用の OS であり、現在、その最初の版がマルチ PSI [4] 上で動作している。PIMOS では、ユーザが消費するさまざまな資源をこれらの並列マシン上で効率良く管理するために、従来の OS のように集中的に管理するのではなく、それぞれの資源を分散して並列に管理している。

本論文では、この PIMOS の資源管理方式の評価について報告する。

2 PIMOS の資源管理方式

PIMOS では、「タスク」と呼ばれる資源管理単位の階層に沿って、ストリームでお互いに結合されたプロセス [5] の木構造 (資源木) によって、全ての資源を管理している。

タスクの階層構造及び資源木の構造の例を図 1に示す。この図の右側の部分が資源木であり、資源木の各ノードは、タスクの階層にそって木構造になっており、各タスク内の資源は、それぞれが輪構造で結ばれている。以下に、資源木を構成する各プロセスについて説明する。

タスク・ハンドラ: タスクの実行及びタスク内で使用されている資源を管理しているプロセスである。タスクの実行は、図でタスクの上につながるストリームにより制御している。また、このタスク内で使用されている入出力装置や子タスクなどの資源は、それぞれ対応するモニタ (後述) を環状に繋げることで管理されている。

入出力ハンドラ: 入出力装置を資源としてユーザに見せるためのプロセスであり、一般の入出力要求に応じる機能と、この資源に対するメタな問い合わせ (例えば、どのような種類の資源なのかなど) に答える機能を持っている。前者のようなメッセージはユーザからフィルタ (後述) を通して送られ、更にデバイスに近いプロセスに転送される。また、後者のようなメッセージはモニタを通して送られ、このプロセスで処理される。

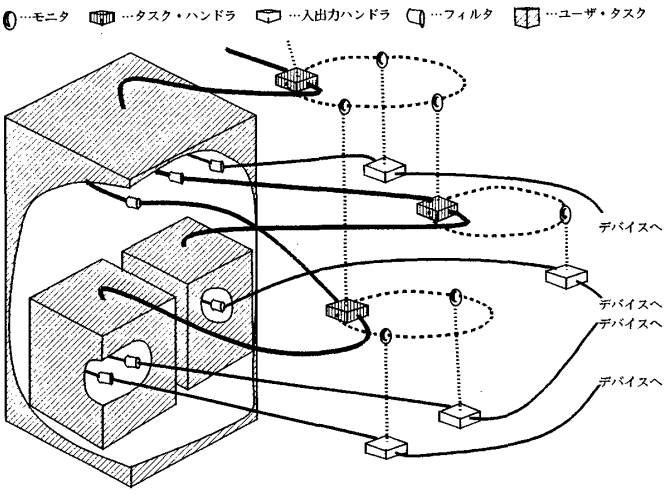

図 1: 資源木とユーザ・タスクの構造

モニタ: タスク・ハンドラからのメッセージを自分が保持しているハンドラに送るかどうかを決めるプロセスである。

フィルタ: ユーザの誤りからシステム全体を保護するためのプロセスである。ユーザと PIMOS が通信する場合には、必ずこの種の保護フィルタが付けられる。この保護フィルタはユーザ・タスク上で実行されるが、プログラム・コードとしては PIMOS が提供するものである。

このように PIMOS では、機能の異なるいくつかのプロセスをストリームで結合して、資源を管理している。そのため、例えばユーザの入出力要求は、保護フィルタ, 入出力ハンドラなどのプロセスを介してデバイスに送られる。

3 評価

この PIMOS で採用している資源管理方式で、保護フィルタや資源ハンドラなどの各プロセスがどの程度のオーバヘッドになっているかを評価する。

Evaluation of the PIMOS

Hiroyuki SATO[1], Masaki HOSHIDA[2], Takashi CHIKAYAMA[2], Tetsuro FUJISE[3]
1: Institute for New Generation Computer Technology (Present affiliation: Mitsubishi Electric Corporation).
2: Institute for New Generation Computer Technology. 3: Mitsubishi Research Institute.

3.1 測定方法

一般に入出力は、1文字ずつ行うのではなく、何文字かをバッファリングして行う方が、スループットが良くなる。PIMOS が扱うような並列計算機の場合も逐次計算機と同様にこのバッファリングの効果があり、PIMOS では、このバッファリングを行うユーティリティが用意されている。そこで今回は、ある固定のサイズの文字列をバッファのサイズを変えて入出力した時の実行時間の変化を

(1) オリジナルの PIMOS (フィルタ, ハンドラ付き)

(2) 保護フィルタを取った場合

(3) 保護フィルタ及び入出力ハンドラを取った場合

の 3 種類について測定した。

これらの測定を行う場合、実際の入出力装置 (例えばファイル) を使うと、それ自体の遅さが目立ってしまうので、今回はダミーの入出力装置を KL1 で作成した。PIMOS での入出力装置は、プロセスとして見えるので、KL1 でそれを摸擬するものを作成するのは容易である。このダミー装置は、あるサイズの文字列を出力したり、指定したサイズの文字列を入力する機能を持っている。

このダミー装置は測定プログラムとは別プロセサで実行させるのが実際の入出力装置に近い動きになる。しかし、これではプロセサ間の通信オーバヘッドのために、PIMOS プロセスのオーバヘッドが隠れてしまう可能性がある。そこで、入出力装置プロセスが測定プログラムと同一プロセサに存在する場合についても測定した。

3.2 測定結果 / 考察

1メガ文字 (1文字は2バイト) の文字列を読み込むのに、バッファ・サイズを変化させた時の実行時間の推移を図 2 に示す。このグラフから以下のようなことが言える。

- ダミー装置が同一プロセサにある場合には、バッファ・サイズ1キロ文字くらいまで保護フィルタや入出力ハンドラのオーバヘッドが確認できるのに対し、別プロセサの場合は無視できる程度である。これは、バッファ・サイズが1キロ文字くらいまではプロセサ間の通信オーバヘッドが PIMOS プロセスのオーバヘッドを打ち消してしまうほど大きいからである。

- バッファ・サイズが1キロ文字より大きい場合は、ダミー装置プロセスが別プロセサにあろうと同一プロセサにあろうと、PIMOS プロセスのオーバヘッドは無視できる。これは、バッファ・サイズが1キロ文字より大きい場合には、ダミー装置の仕事量が PIMOS プロセスのオーバヘッドを打ち消してしまうほど大きいためである。

- ダミー装置プロセスが別プロセサにある場合、バッファ・サイズが 500 文字近辺の処理時間が最小になっており、それを越えると急激に処理時間が上がっている。今回の測定で使用した KL1 処理系では、プロセサ間通信をパケットにより行っており、1度にプロセス間で通信できるデータのサイズが、500 文字辺りになっている。それを越えると1つのバッファでも複数のパケット (マルチ・パケット) に分割されるため、ここで実行時間が不連続になっている。

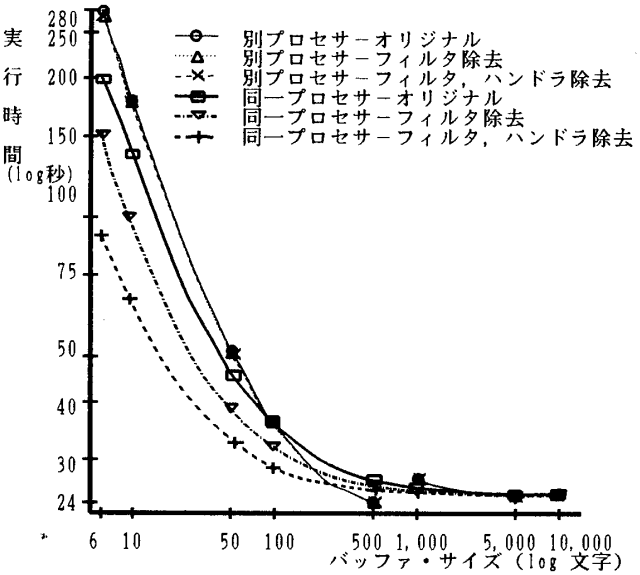

図 2: 各バッファ・サイズの実行時間と PIMOS プロセスのオーバヘッド

以上の測定の結果、PIMOS プロセスのオーバヘッドはあるが (同一プロセサによる測定より)、実際の入出力においてはどのようなバッファ・サイズをとっても、プロセサ間の通信オーバヘッドに比べて PIMOS プロセスのオーバヘッドは無視できる程度であると言える。また、バッファリングのサイズは、マルチ・パケットになる寸前である 500 文字近辺が最適であることが分った。このことから、現在バッファリング・ユーティリティでは、バッファ・サイズの既定値をこの辺りに設定している。

4 おわりに

測定の結果、一般の入出力においては、PIMOS を使用することによるオーバヘッドが、ほとんど無視できる程度しかないことを確認した。今後、他の項目 (例えば、タスクの状態問い合わせなど) についても測定 / 評価を行い、今後の拡張 / 改良に反映させていく予定である。

参考文献

[1] 後藤他: 並列推論マシン PIM– 中期構想 –, 第 33 回情処全国大会, 3B-5, 1986-10.

[2] T. Chikayama et al. : *Overview of the Parallel Inference Machine Operating System (PIMOS)*, In Proc. of FGCS'88, Vol. 1, pp. 230-251, 1988.

[3] 佐藤他: PIMOS の資源管理方式, 並列処理シンポジウム JSPP'89, S4-1, 1989-2.

[4] 瀧他: Multi-PSI システムの概要, 第 32 回情処全国大会, 3Q-8, 1986-3.

[5] E. Shapiro and A. Takeuchi, *Object Oriented Programming in Concurrent Prolog*, New Generation Computing, Springer Verlag Vol. 1, No. 1, pp. 25-48, 1983.