PVMによる並列処理システム構成の試み

On a small PVM system connecting some PCs with linux

小池 慎一[†] 山住 富也[††]
Shin-ichi KOIKE Tomiya YAMAZUMI

Abstract

We constructed a small Parallel Virtual Machine(PVM) system with some old PCs. Then we tried a calculation of multipling matrixes of 500 by 500. Comparing two real processing times, the cases of one PC and 4 PCs, we took the efeciency 0.55. Where, we used 3 PCs with linux and a Macintosh with MkLinux and the first machine's MIPS is 300 and the slowest is 35.9.

1　はじめに

PCの普及に伴って，研究室にも何台ものPCが存在するようになった．商用のOSは相次ぐ機能強化とバージョンアップで数年前のPCでさえ実用にならない廃物にしてしまっている．粗大ゴミになりかねないPCを有効に使用するために，OSとしてフリーであるlinuxを導入している．計算処理のアプリケーション開発では十分に使用可能であり，活躍している．

linuxと同様にフリーなソフトに，PVM(Parallel Vertual Machine)がある[?]．PVMはアメリカのオークリッジ国立研究所(Oak Ridge National Laboratry)とエモリー大学(Emaory University)で開発が始められた．それは，複数のコンピュータをメッセージ通信を介してが並列処理するためのソフトウェアである．使用言語はFORTRANとC++である．

PVMは，PCとTCP/IPの環境さえ在れば，フリーのPC UNIX(当研究室ではlinuxとMkLinux)を用意するのみで，ハードウェアの投資はほとんどゼロでとりあえずテストができる．そこで，4台のPC(3台のDOS/V機と1台のMacintosh)を用いて導入してみた．

まだ60-70%の程度しかPVMの能力は引き出せてないようであるが，行列の積のような簡単なプログラムで，処理速度の向上をみた．

2　経過

2.1　使用した機材

はじめに使用するPCを用意した．詳細は付録に示すが，最新のDual PCのPentiumII 300MHz×2のものから，K5 166Mhz, 6x86 133MHzの旧型機，PowerPC 604のMacまで新旧入り交じっている．K5搭載のマシンは研究室の共用計算サーバとなっており，常時電源ONになっている．公称のCPUのMIPS値はPIIの300からK5の36までと幅が広い．当然，もっとも遅いK5のマシンに足を引っ張られる結果となることが予想される．

OSは，linux 2.0.21, linux 2.0.34, MkLinux2.0.28の3種類である．このテストに当たってDOS/V機のOSのバージョンを統一することはしなかった．なお，MkLinuxはMacintosh用のlinuxである．

2.2　PVMの概要

PVMでは，処理対象となるタスクを複数のタスクに分割し，そのタスクをPVMデーモンが制御する．いくつのタスクに分割するかは指定する必要がある．それらのタスクが同一のホスト(PC)の上にある場合には，TCPソケットを介してタスク間の通信が行われる．別々のホストの間にあるタスクは，UDPを介してタスク間通信がなされる．この様子を図1.に示す．

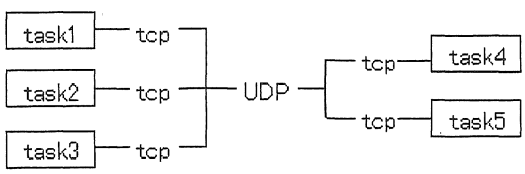

図１．PVMにおけるタスク間通信
host1に3個のタスク，host2に2個のタスクがある場合

プログラムは，通常マスターとスレーブの2個からなる．マスターは，主たる処理をするもので，スレーブが構成するタスクの初期設定しデータを配布したり，その結果を集めて全体を初期の要求にまとめる仕事などをする．それらは，PVMデーモンと呼ばれる制御プログラムのもとで行われる．PVMデーモンは利用するすべてのホストおよびタスクを管理下におき，それらを統合して，shellのように働く．

言語はlinux用のgnuのgccである．並列処理用の特別なものではない．と言うことは，目的とする処理をどのように並列化するかはユーザが自分の責任ですることであ

[†]愛知工業大学計算センター（豊田市）
[††]名古屋文理短期大学情報処理学科（稲沢市）

り，それにはスキルが要求される．

PVMそのものは，国内のサイトからダウンロードした[?]．また，使用方法などの情報もほとんどWEBから得た[?, ?, ?, ?]．PVMについて紹介程度以上に書かれている書籍は利用できなかった．使用したPVMのバージョンはPVM3.4 $\beta 7$ である．

2.3 テストした処理の内容

並列処理の内容を次の二つにわけて考える．

1. 別々に処理してあとからそれらを集めてまとめることが可能なもの

   例えば，数値積分は区間を分割して積分しその和を求めることにより最終結果を得る．また，ここで報告する行列の積も，被乗算される行列を行に分割して分割すれば，並列処理された個々の結果を集めるだけで最終的な結果を得る．ただし，メモリ仕様効率の点では問題がある．
2. 分割され並列動作しているタスク間で途中結果を受け渡ししながら処理を進めるもの

   このタイプの処理は，通信回数が増すのでその分オーバーヘッドが増すが，応用性が高い．例としてラプラスの偏微分方程式の解を求めようとしたが，トラブルが発生し，本報告には含められなかった．

上の1のタイプに属する行列の積を並列処理した．行列の積 $C = AB$ において，A を行に分割する．すなわち，この行列のサイズが $n \times m$ であったならば，分割数を N として，$n/N \times m$ の N 個の行列にする．それらを $A_i (i = 1, \ldots, N)$ とすると，$A_i \times m$ の積を求めるタスクを N 個生成し，並列処理させ，その結果をまとめて順序よくまとめれば結果 C を得る．言うまでもなく，n/N が整数値をとらない場合には，適当に丸めて整数の大きさにしておく．

例として，A が次の $10 \times m$ の行列で，4分割する場合，

$$A = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1m} \\ a_{21} & a_{22} & \ldots & a_{2m} \\ & \vdots & & \\ a_{101} & a_{102} & \ldots & a_{10m} \end{bmatrix},$$

この行列を，以下の $A_i (i = 1, \ldots, 4)$ に分割し，子タスクにそれぞれ送信する．また，行列 B はそのまま送信する．

$$A_1 = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1m} \\ a_{21} & a_{22} & \ldots & a_{2m} \\ a_{31} & a_{32} & \ldots & a_{3m} \end{bmatrix},$$

$$A_2 = \begin{bmatrix} a_{41} & a_{42} & \ldots & a_{4m} \\ a_{51} & a_{52} & \ldots & a_{5m} \\ a_{61} & a_{62} & \ldots & a_{6m} \end{bmatrix},$$

$$A_4 = \begin{bmatrix} a_{91} & a_{92} & \ldots & a_{9m} \\ a_{101} & a_{102} & \ldots & a_{10m} \end{bmatrix}$$

プログラムの概要を示す．

- マスタープログラム
  - step1. 分割数と行列のサイズの入力
  - step2. テスト行列の生成
  - step3. PVMデーモンに登録，子タスクの起動
  - step4. 分割サイズの算出，配列データの分配
  - step5. 結果の収集，まとめ
  - step6. 出力
- スレーブプログラム
  - step1. PVMデーモンに登録
  - step2. データ受信
  - step3. 計算処理
  - step4. 結果の送信

3 結果

500×500 の大きさの配列の積を求めた．6回繰り返した結果を表1に示す．

表1: 配列 500×500 同士の掛け算を行ったときの時間

PC台数	1台	2台	3台	4台
実時間 (s)	124.99	97.59	74.72	68.87
CPU時間 (s)	0.71	0.91	1.17	1.36
SYSTEM時間 (s)	0.39	0.45	0.63	0.73

実時間 (ターンアラウンド時間) は，1台の場合に比べて4台では0.55となって向上している．この値が1/4にならないのは，使用している機材の性能にばらつきがあり，もっとも遅いPCに足を引っ張られてしまうことが大きいと解釈する．どの機材にタスクが割り当てられるかはPVMデーモンに依存するので，今回は，これ以上の実時間の解析は出来なかった．

また，CPU時間とSYSTEM時間が台数の増加とともに増加するのは，台数が増えると本来の処理以外の処理が増えるためと推測される．

4 考察

処理する配列が小さければ，並列処理すればかえって処理時間は増加する．その境の台数は，上に述べた環境のもとでは，185×185 の配列の処理の場合に1台と4台の実

タスク間のデータ通信を伴うラプラス型の偏微分方程式のプログラムでもテストしたが，最終的にMkLinuxが存在するとPVMデーモンがタスクを異常終了させてしまうことが判明し，かつ，演算結果も誤っていたので，今後の課題としたい．

安価でかつ粗大ゴミとなるPCでもまとめれば最新式のPC以上の性能がでることが実証された．今後，環境を整え，研究室の計算サーバとして使用できるような構成でテストしてみたい．

謝辞

本システムのWEBからの情報の収集や実際のシステム構成においては，卒研生の福谷友宏君の尽力があったことを記して，謝意を表する．

参考文献

[1] "PVM Home Page" http://www.epm.ornl.gov/pvm/pvm_home.html (1998)

[2] まこ佐々木："PVMによる並列プログラミング" http://gaia.tokai.jaeri.go.jp/activity/mvp/mvp/parallel/PVM/index.html#what (1996)

[3] "Index for PVM3 Library" ftp://netlib.org/pvm3/index.html(1993)

[4] Richard Sevenich："The PVM Website Course" http://knuth.sirti.org/cscdx/pvm/index.html (1997)

[5] Al Geist,Jack Dongarra,Weicheng Jiang他 (訳　村田英明)： "PVM3　ユーザーズガイド&リファレンスマニュアル日本語版"(1995)　("PVMの使い方"http://www.cs.takushoku-u.ac.jp/dcl/PVMより入手)

[6] 富士通株式会社："UXP/V PVM使用手引書 V10" http://www.cc.kyushu-u.ac.jp/UXP_MANUAL/japan/pvm/index.htm (1995)

A　使用したPCの仕様

表 2: 使用したPCのスペック

ホスト名	Cyrix
CPU	6x86 L-P166 133MHz
MIPS値	132.71
メモリ	64M
ハードディスク容量	約 1 GB
OS	Linux 2.0.34
ホスト名	p2dual
CPU	Pentium II 300MHz × 2
MIPS値	299.83
メモリ	64M
ハードディスク容量	約 2 GB
OS	Linux 2.0.34
ホスト名	fmv590
CPU	K5PR 166MHz
MIPS値	35.94
メモリ	64M
ハードディスク容量	内蔵 約 500 MB, 外付け 約 2 GB
OS	Linux 2.0.21
ホスト名	MkMac
CPU	PowerPC 604
MIPS値	238.39
メモリ	16M
ハードディスク容量	約 600 MB
OS	MkLinux2.1(カーネル Ver.2.0.28)

（受理　平成11年3月20日）