計測自動制御学会東北支部 第195回研究集会 (2001.6.22)
資料番号 195-6

PVM実装とPVM上での並列アルゴリズム実行

PVM's Mounting and Execution of Parallel Algorithm on the PVM(Parallel Virtual Machine)

○川崎 拓弥，奈良 久

○Takumi Kawasaki, Hisashi Nara

八戸工業大学

Hachinohe Institute of Technology

キーワード： PVM (Parallel Virtual Machine)，並列アルゴリズム (Parallel Algorithm)，
複区分予測子法，レイテンシー (Latency)

連絡先： 〒031-8501 八戸市妙字大開88-1 八戸工業大学 システム情報工学科 奈良研究室
川崎拓弥，E-mail: m00204@stud.hi-tech.ac.jp

1. はじめに

VLSI技術の目覚しい発展に伴い，並列処理計算機の研究が盛んに行われている．高速演算が必要である科学技術用数値計算の分野で大いに期待されているなか，近年，汎用の並列処理計算機システムが市販され普及し始めている．しかし，実際には非常に高価であり，日常レベルで使用できる環境には必ずしもない．そこで，既存のネットワーク上で仮想的に並列計算機を実現するソフトウェア「PVM(Parallel Virtual Machine)」[1-3]を利用して，前回提案した並列アルゴリズムを用いてプログラムを作成し，PVM上で実行したい．

本研究では，実際に使用するPVM環境をネットワーク上に実現したので，その概略を述べ，従来のアルゴリズム[4-7]と前回提案している常微分方程式数値解法並列アルゴリズム[8,9]の比較検討した結果について報告する．

2. PVM

PVMとはParallel Virtual Machineの略であり、ネットワークに接続された異機種UNIXコンピュータ群を，仮想的に単一の並列コンピュータとして利用する事を可能にするソフトウェアシステムである．このPVMはWeb上でフリーウェアとして公開されており，先端科学技術分野における大規模計算のためのソフトウェアとして世界中で利用されている．また，FreeUnix系のOS(Linux，FreeBSDなど)で動作させることができるので，低コストで仮想的な分散メモリ型並列計算機を構築することができる．

現在，研究室では9台の計算機にPVMを導入している．各計算機のCPUはPentium^{TM}166MHzが4台，PentiumII^{Xeon}450MHz が1台，PentiumIII800MHzが2台，PowerPC-G3-400MHzが1台，Alpha600MHzが1台である．それぞれ，計算機へのインストールと，各種サンプルプログラムによる動作確認は成

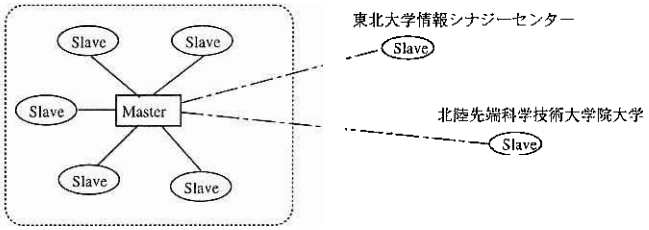
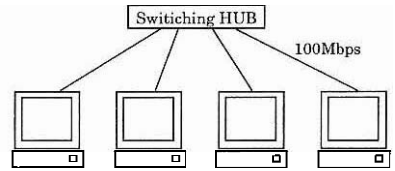


Fig. 1　PVM環境の概略図



CPU : Pentium 166MHz
Memory : 96MB
OS : Vine Linux2.1(kernel 2.2.17)

Fig. 2　実際に使用したPVM環境

功している．

また研究室内の計算機以外に，東北大学情報シナジーセンター情報教育研究部の計算機1台(CPU: Pentium233MHz，メモリ160MB，OS:VineLinux2.1 CR)がインターネット経由でPVMを利用できる．さらに，八戸工業大学は，ギガビットネットワークJGNに参加しており，北陸先端科学技術大学院大学の計算機とギガビットネットワークを用いたPVMの利用を検討しており，実際に実験する段階まで来ている．八戸工業大学－東北大学間，八戸工業大学－北陸先端科学技術大学院大学間の通信には，両大学間の地理的位置関係(数100kmから1000km程度)により，物理的にミリ秒単位の時間がかかる(latency，レイテンシー)．また，このレイテンシーのほかに通信線の容量，実際に選択された経路，その他の理由によるレイテンシーも加わる．このレイテンシーの実体の解明とその改善も重要な研究テーマと考えている．

3.　複区分予測子法

ここで前回提案した並列アルゴリズムについて，簡単に説明する．

初期値問題$y' = f(x,y)$，$y(x_0) = y_0$を満足する特殊解$y(x)$を求める場合，パラメータMを導入することにより，積分して形式的に，

$$y_{n+1} = y_{n-M} + \int_{x_{n-M}}^{x_{n+1}} f(x,y)dx \qquad (1)$$

と解くことが出来る．ここでMは正の整数または0である．(1)式の$f(x,y)$に対してNewtonの後退補間多項式で展開して整理すると，

$$y_{n+1} = y_{n-M} + \sum_{k=0}^{K} hP_k f_{n-k} \qquad (2)$$

$$\left(P_k = \frac{1}{k!} \int_{-M}^{1} u(u+1)\cdots(u+k-1)du \right)$$

となり，この式を複区分予測子法と呼んでいる．こ

Table 1　プログラム全体の並列プログラムの実行時間(単位は[μSec.])

従来のアルゴリズム	並列アルゴリズム	
(Runge-Kutta法)	nproc=2	nproc=4
357	468180	559026
356	470493	568405
359	469220	581554
357	476828	567998
357	468768	588025

こで2次まで取る(M=3) とすると，

$$y_{n+1} = y_{n-3} + \frac{4h}{3}(2f_n - f_{n-1} + 2f_{n-2}) \quad (3)$$

が得られる．ただし，ここで初期近似解$\{y_n^{(0)}\}$は刻み幅をnh (hは通常の刻み幅)とする．この初期近似解$\{y_n^{(0)}\}$がメモリに記憶されているとすれば，割り当てたプロセッサの数だけ並列に計算できる．ここで，第1近似解以降，第3番目までの近似値$\{y_1^{(i)}, y_2^{(i)}, y_3^{(i)}\}$は未知の値を含んでいるため，代表的な陰的解法のアダムス・モルトン法

$$y_{n+1} = y_n + \frac{h}{12}(5f_n + 8f_{n+1} - f_{n+2}) \quad (4)$$

を採用した．

4. 並列プログラムの作成

PVM上でのプログラム作成は，PVMライブラリをリンクすることによりFortran，C言語ともに可能であるが，今回はC言語を用いてプログラムを作成した．また、今回作成したプログラムはMaster用のプログラムとSlave用のプログラムの2つをそれぞれ別々に作成する，いわゆるマスタースレーブプログラムである．これにより，マスターの計算機自身もスレーブの計算機と同様に動作させることが出来る．以下に今回作成したプログラムの大方の流れを示す．

1) master(Host側)で，初期近似解$y_n^{(0)}$を求める．

2) masterで求めた初期近似解$y_n^{(0)}$をslave(Client側)に送信する．

3) slaveは受け取った初期近似解のデータを複区分予測子法(3)式に適用して，新しい近似解を求める．

4) slaveで求めた近似解のデータを，masterに送信する．

5) masterはslaveからデータを受け取り，誤差判定を行う．

6) 条件を満足したらループを抜け，プログラムを終了する．

7) 条件を満足できなかったら，求めた近似解(今の場合第1近似解)をスレーブに送信し，以下，3)〜6)までの処理を繰り返す．

今回はデータの通信量や経路については一切考慮せず，Slave側で計算されたデータをとりあえずMaster側の計算機で受信し，収束条件を満たすまで誤差判定を行った．なお収束条件に関しては，相対誤差を

$$|y_n - y^{(i)} \quad (5)$$

と定義し，最後の分点での誤差が10^{-6}以下になるまで処理を繰り返した．

5. 実験結果

本実験では，我々が提案した並列アルゴリズムの有用性を調べるため，CPUが§Pentium^{TM}166MHz，メモリが96MB，OSはVineLinux2.1 (kernel2.2.17)

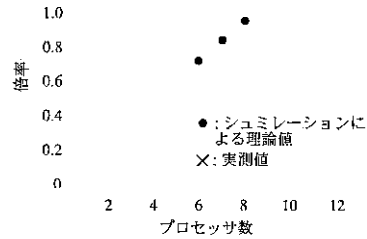


Fig. 3　シュミレーションおよび実測による並列計算の計算速度向上比

の計算機を4台(すべてGateway2000)用意し(Fig.2参照)，PVM上でプログラムを実行した．

今回取り上げた初期値問題は$y' = -xy, y(0) = 1, y = \exp(\frac{-x^2}{2})$である．前回，プロセッサ数が100個だと仮定したシミュレーション実験を行った結果が，Fig.3である．横軸がプロセッサ数，縦軸が何倍早くなったかを表す倍率である．このグラフのように，理論値はプロセッサ数に比例して増大していくものと予想される．また，Fig.3からも分かる通り，最低でも9個のプロセッサを使用しなければ，従来のアルゴリズムに比べて遅くなるということだが，今回はプロセッサ数が2つの場合と4つの場合しか測定できないため，最低でもプロセッサ数が10個以上の場合のプログラム実行を今後行っていきたい．

Table1に並列プログラムの実行結果を示す．同じ並列処理プログラムを5回実行し，それぞれの実行時間を測定した．なお，時間の測定にはgettimeofday関数を用いた．この関数は1970年1月1日を起点として，マイクロ秒での時間を返すので，プログラム中で調べたい部分の前後の時間を調べ，その差をとることによって，処理時間を測定できる．従来のRunge-Kutta法と単純に比較すると，並列プログラムの実行時間の方がはるかに大きい．この原因として，初期値問題$y' = -xy, y(0) = 1, y = \exp(\frac{-x^2}{2})$を解く場合の計算量が小さすぎるために，計算時間よりもむしろPVM関数によるデータのパック・アンパック・送信による処理時間の方が大きくなったためと考えられる．プロセッサの数nprocを2個から4個に増やしたときに処理時間が増えている(Table1)のはこのためであろう．要するに，ここで取り上げた初期値問題の例は単純すぎて，PVM独自の処理時間の中に埋没してしまったと考えられる．改善策としては，第1に初期値問題を最低でも1秒以上かかるような問題を取り上げること，第2にスレーブの数を増加することなどであろう．

Table2およびTable3は実際に並列数値計算にだけに費される時間のみを測定した結果を示したものである．プロセッサ数2個の場合の，倍率の理論値は0.244であり，実測値から求めた倍率とほぼ一致し，プロセッサ数4個の場合の倍率は理論値は0.488であるが，実測値は約0.37で，約0.1ほど低い．この理由としていくつかの可能性が考えられるが，プロセッサ数4個までの実測値から断定的な結論を引き出すのには無理があろう．

6.　まとめ

今回，我々が提案した並列アルゴリズムを用いてプログラムを作成し，実際にPVM上で数値実験を行った．その結果，プロセッサ4個までを用いた全体の並列処理速度としては従来の逐次アルゴリズムと比較して処理時間の改善は見られなかった．

Table 2　数値計算部分における並列プログラムの実行時間(単位は[μSec.])

従来のアルゴリズム	並列アルゴリズム(nproc=2)			
(Runge-Kutta法)	master	slave	total	倍率
357	298	1196	1494	0.239
358	297	1170	1467	0.244
355	297	1183	1480	0.240
355	298	1157	1455	0.244
357	297	1196	1493	0.239

Table 3　数値計算部分における並列プログラムの実行時間(単位は[μSec.])

従来のアルゴリズム	並列アルゴリズム(nproc=4)			
(Runge-Kutta法)	master	slave	total	倍率
359	298	663	961	0.374
358	297	637	934	0.383
355	297	650	947	0.375
359	298	650	948	0.379
357	297	663	960	0.373

しかし，PVMによるオーバーヘッド(データのパック・アンパック，あるいはデータの送信)を無視した純粋な数値計算における処理速度を調べてみたところ，前回のシミュレーションで求めた理論値に近いことが確認できた．全体としてのプログラムがシリアルな計算に比べて非常に遅くなった原因として，取り上げた初期値問題の計算量が極端に小さいために，PVM独自に関係する時間が実行時間のほとんどを占めたため，と考えられる．今後，今回取り上げた初期値問題より複雑な初期値問題を取り上げたい．

また，今回実験で用いたPVM環境は4台全て同じ処理能力を持った計算機同士であったが，OSや処理能力が異なる計算機どうしでも利用できるというPVMの長所を生かし，現在研究室で利用可能な7台の計算機を総動員し，さらに他研究室の協力も得てスレーブ計算機の台数を増やしていきたい．さらにそのPVM環境にあったプログラムの最適化(例えば，処理性能が高い計算機にはなるべく多くの計算を割り当てる，など)を行い，同様に処理時間を測定し，比較検討していきたい．

参考文献

1) Al Geist: PVM3 User's guide and reference manual, http://www.netlib.org/pvm3/book/pvm-book.html(1987)

2) 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，http://phase.etl.go.jp(1987)

3) 湯淺太一，安村通晃，中田登志之: 初めての並列プログラミング，共立出版(1994)

4) 奈良久，早川美徳，阿部亨: 数値計算法，朝倉書店(1991)

5) 玄光男: 数値計算とデータ構造，共立出版(1994)

6) 玄光男，井田憲一: 数値計算とデータ構造演習，共立出版(1992)

7) 小沢一文: 数値計算法，共立出版(1987)

8) 川崎拓弥，奈良久: 常微分方程式数値解法の並列アルゴリズム，計測自動制御学会東北支部論文，188 No.5，(2000)

9) 川崎拓弥，奈良久: 常微分方程式数値解法の並列アルゴリズムとパブリックドメインソフトウェアパッケージPVM，八戸工業大学システム情報工学研究所紀要，13，33−39，(2001)