卒業研究報告書

題目

MPIを用いた並列計算

指 導 教 員 石水 隆 助教

報告者

05-1-037-0069

鍵谷幸洋

近畿大学理工学部情報学科

平成21年1月31日提出

概要

計算機の高速処理化といったことは、近年ますます発展している分野であり、大容量の記憶デバイスの登場や、ネットワークの高速化などにより、大量のデータを高速に処理することが求められている。現在様々な角度から計算機の向上を目指している。高速処理化を実現させる手段として、並列計算機(Parallel Computer)の実装が注目されている。だが、並列計算機というものは一般的には高価なものであり、個人で使用するにはあまりに困難である。だが、ネットワーク接続を介して1台の並列計算機と考える、仮想並列計算機(Parallel Virtual Computing)を用いればコストをかけずに並列計算をできる環境を整えることができる。

本研究では、仮想並列計算機を可能とさせるソフトウェアであるMPI(Message Passing Interface)を用いて評価する。MPIとは、分散メモリ型の並列計算機において、複数のプロセッサ間で、データのやりとりをするために用いる、メッセージ通信操作の仕様標準である。評価方法として、wav(RIFF waveform Audio Format)形式の音声データファイルをmp3(MPEG Audio Layer-3)形式の音声データファイルにエンコードした時の時間を測定してMPIを評価し、その結果により並列計算機の有用性を実験的に評価する。

目次

1 序論

1.1 本研究の背景

1.1.1 並列処理(Parallel Processing)

並列処理(Parallel Processing)とは、計算機において複数のプロセッサで1つのタスクを動作させることである。問題を解く過程はより小さなタスクに分割できることが多い、という事実を利用して処理時間の短縮になり、処理効率の向上につながるといった手法である。

1.1.2 並列計算機(Parallel Computer)

並列処理を行うために用いられるのが並列計算機(Parallel Computer)である。並列計算機は複数のプロセッサを持ち、各プロセッサが協調して動作することにより高い処理能力を得ることができる。

並列計算機による処理の方法は、共有メモリ(shared memory)型と分散メモリ(distributed memory)型の2つに大きく分類できる。共有メモリ型の並列計算機は、それぞれが同じメモリを通して計算するため同期やデータの送受信が対処しやすいという長所があるが、プロセッサの増加によりメモリにプロセッサを接続させることが困難になるといった短所がある。また、一方分散メモリ型による並列計算機はプロセッサが個々のメモリを持つといった特徴から、複数のプロセッサを接続させる並列計算機を構築しやすくなるといった長所をもっており、そのことから現在では分散メモリ型並列計算機が主流となっている理由となっている。短所としては、他のプロセッサの持つデータをすぐに参照できないといったことが挙げられる。

データの高速処理には、複数のプロセッサを持つ並列計算機は非常に有用であり、現存では並列計算機は共有メモリ型、分散メモリ型共に様々な機種が存在する。しかし一般に並列計算機は非常に高価であるため容易に利用できない。このため、近年、複数の計算機をネットワーク接続し、計算機群全体を1台の仮想並列計算機(Parallel Virtual Computer)として用いるクラスタ(Cluster)処理が注目されている。仮想並列計算機を構築するソフトウェアは様々なものが開発されており、無料で提供されているものもある。このため、仮想並列計算機を個人で使用することも容易になっており、今後並列計算機の重要性はより拡大していくと考えられる。

無料で提供されている仮想並列計算機を構築するソフトウェアとしては、PVM(Parallel Virtual Machine)[5]やMPI(Message Passing Interface)[1][2]といったものがある。以下に、これらについての説明を記述する。

1.1.3PVM(Parallel Virtual Machine)

PVM(Parallel Virtual Machine)[5]は、1991年に米国のオークリッジ国立研究所(Oak Ride National Laboratory)[5]が中心となって開発された、並列計算を行うためのソフトウェアである。

動作するマシンの種類が多いこと（Linux、BSD、WindowsのどのOSでも動作可能）や、入手方法が容易である為、研究機関などで広く利用されている。PVMソフトウェアシステムの構成は、大きく2つに分けられる。1つはデーモン、もう1つはルーチンライブラリである。

PVMはpvdmd3というデーモンを起動させて仮想並列計算機上のすべてにデーモンを常駐させる。PVMを利用したアプリケーションを実行する場合に、まず最初に仮想並列計算機を構成している計算機のうちのどれか1台のからpvmd3を起動する。pvmd3は独自のシェルスクリプトのようなものを持ち、ここから仮想並列計算機を構成するすべての計算機のpvmd3を起動する。全て起動した後、どれか一つの計算機に表示されたUNIXプロン

プトに対して特定のコマンドを入力することにより、PVMアプリケーションを実行する。複数のユーザは、互いに計算機をオーバーラップさせて仮想並列計算機を構成でき、また、各ユーザは一人で複数のPVMアプリケーションを同時に実行することも可能である。

また、libpvm3.aといったルーチンライブラリによってメッセージパッシング、プロセスの生成、タスクの協調などの必要な関数を提供する。PVMを利用したアプリケーションを実行する場合には、アプリケーションプログラムとこのライブラリをリンクする必要がある。

1.1.4MPI(Message Passing Interface)

MPI(Message Passing Interface)[1][2]とは並列計算機を実装するための標準化された規格であり、実装自体を指すこともある。

複数のCPUが情報をバイト列からなるメッセージとして送受信することで協調動作を行えるようにする。ライブラリレベルでの並列化であるため、言語を問わず利用でき、プログラマが細密なチューニングを行えるというメリットがある一方、利用にあたっては明示的に手続きを記述する必要があり、ロックの対処などもプログラマ側が大きな責任をもたなければならないといったことが挙げられる。業界団体や研究者らのメンバからなるMPI Forum によって規格が明確に定義されているため、ある環境で作成したプログラムが他の環境でも動作することが期待できる。

MPIは専用の並列計算機からワークステーション、パーソナルコンピュータに至るまで幅広くサポートしている、無料で提供されている主な実装はMPICHやLAMといったものがある。

1.1.5PVM と MPI の相違点

PVMとMPIの大きな違いとして動的なプロセス生成・停止やまた動的ノード数の増減がPVMでは可能であったがMPIでは不可能であったことが挙げられる。しかし、最近MPIの新規格であるMPI-2の仕様において動的なプロセス管理の機能が取り入れられたことから、PVMの優位性の1つが失われている。しかし、MPI-2は前の規格であるMPI-1の関数も使用できるが、新規格であるMPI-2の機能を完全にサポートしている実装が無いという問題が残っている。

PVMの利用時の大きな欠点は移植性に乏しいということがいえ、MPIはPVMと違い世界的な標準が目的とされ開発されたものであるため移植性が高く評価されている。だが、PVMは異機種間における処理を目的で作成されているので、異機種の計算機による並列処理に適しているといった長所をもっている。

一方、MPIの欠点は、MPIは高レベルのバッファ操作を可能とするために同一のワークステーション・クラスタを対象としていることである。MPIは仮想計算機を構成する全ての計算機が同じOSで無ければ動作がしないため、異機種間での並列処理ができないことである。MPIは同機種における並列処理が望ましいため近くにある同機種の計算機でのクラスタ処理が、PVMはデーモンを使用してのTCP/IP通信ができ、異機種の計算機を使用できるという特性を持っているため、様々な異機種の計算機がある場所や通信を利用し離れた計算機も使用できるという点からクラスタ処理とグリッド処理にそれぞれ利用できる。現在の両方の開発状況は、進んでおり、MPI Forumにおいて頻繁にマイナーチェンジが行われている。

1.2 本研究の目的

本研究では、仮想並列計算機の有用性を検証するためのMPIを用いて、複数台の計算機をネットワークにおいて構成し、仮想並列計算機の性能を実験的に評価する。評価方法としては、MPIを構築するソフトウェアの1つであるMPICH2を用いて音声ファイルであるwav(RIFF waveform Audio Format)[8]からmp3(MPEG Audio Layer-3)[3]にエンコードさせる並列エンコーダを作成しエンコードにかかる時間を計測することによって、検証を行っている。

2 準備

2.1 使用機器

本研究を始めるにあたり、まずMPIによる仮想並列計算環境を構築する必要がある。本研究では、その実装にはMPICH2[4]を用いた。MPICH2はMPIを実装した無料のソフトウェアであり、Windows OSに対応しているといった点を特徴としている。今回、Windows OSに導入可能という点を重要視したのは、企業や個人で使用するOSで幅広く使用されているといった点であり、またMPIは異なったOS間の実装はできないためOSの統一とした目的でも重要である。

本研究にて、使用した計算機のスペック(spec)を表1に記す。

表 1 本研究で使用する計算機

PC1	OS	Microsoft Windows XP
	CPU	Intel Pentium M processor 4 3.20GHZ
	Memory	2038MB
PC2	OS	Microsoft Windows XP
	CPU	Intel Pentium M processor 1.5GHZ
	Memory	502MB
PC3	OS	Microsoft Windows XP
	CPU	Intel Pentium M processor 1.5GHZ
	Memory	502MB
PC4	OS	Microsoft Windows XP
	CPU	Intel Pentium M processor 1.5GHZ
	Memory	502MB

本研究では表1のスペックをもつ計算機計4台での検証を行なう。また、仮想並列計算機を構築する際にLAN(Local Area Network)やルータ(Router)そしてハブ(Hub)を使用し計算機をネットワークでつないでいる。この構成図を図1に示す。

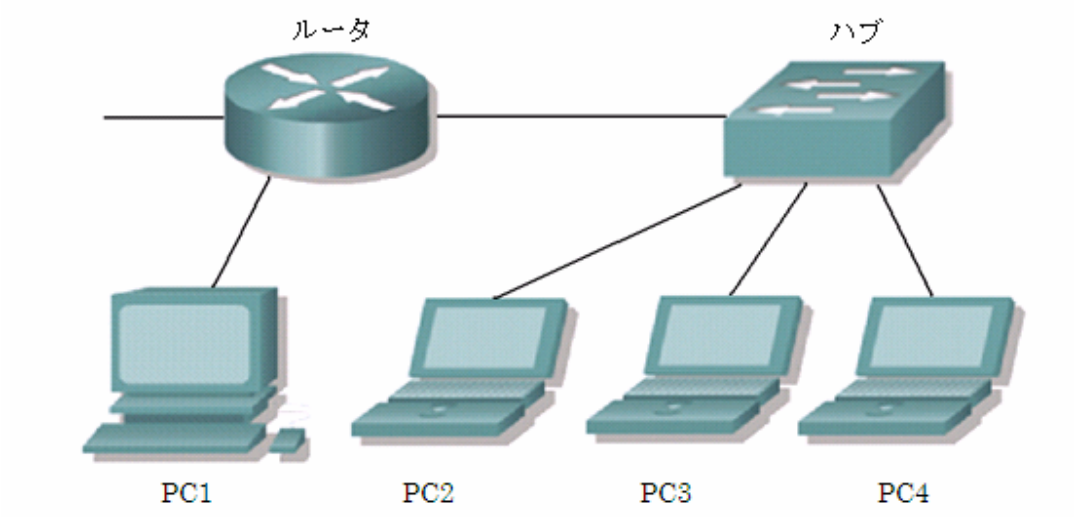

図1　仮想並列計算機の構成

2.2MPICH2 の導入

MPICH2を使用するためにまず、計算機にMPICH2をインストール(Install)する。MPICH2はインターネット上のMPICH2[4]のウェブページにおいて無料で配布されているので使用するOSに合うものをダウンロードし、インストールする。本研究でのWindows用の最新のソフトmpich2-1.0.6p1-win32-ia32.msiを各計算機にダウンロードし、C:¥Program Files¥MPICH2の下にインストールを行った。

Windowsで計算機を使用するにあたって、全ての使用する計算機に管理者権限を持つ同名のアカウントを予め設定しておき、各計算機で同一のパスワードを持つ必要がある。また、各計算機上に共有フォルダを作成しておくことで、実行ファイルの受け渡しを円滑にすすめることができる。

インストールが完了すると、各計算機のMPICH2のバイナリのあるフォルダに対して環境変数のPATHの指定をしておく必要がある。PATHが通っているかどうかを確かめるには新規でコマンドプロンプトを立ち上げた後に”mpiexec”と入力し、引数のUsageが表示されているかどうかで判断できる。

本研究では、Visual C++2005をコンパイラとして使用している。こちらは、Microsoftの公式ホームページ[7]からダウンロードできる。Visual C++2005をインストールした後、これを用いてMPICH2による並列プログラミングを行うには以下の設定をする必要がある。

まずVisual C++2005を起動し、起動画面の上部にあるツールバーからツール、オプションを選択し、オプションダイアログを開く。オプションダイアログで、ツリービューのプロジェクトおよびソリューションを開く。その後、ディレクトリを表示するプロジェクトからインクルードファイルにMPICH2のインクルードファイルを追加する。同様に、ライブラリファイルも追加する。初期設定だと、追加するフォルダはそれぞれ”C:Program Files¥MPICH2¥include”,” C:Program Files¥MPICH2¥lib”となっているので環境に合わせて適切なフォルダを選択する。また、追加する依存ファイルとして、デバッグの場合”mpi.lib cxxd.lib”と入力しリリースの場合”mpi.lib cxx.lib”と入力する。

これらの設定を行なうことでMPICH2による並列プログラミングが可能となる。

2.3 音声ファイル

2.3.1 wav

wav(RIFF waveform Audio Format)[8]は、MicrosoftとIBMによって開発された音声ファイルの形式である。WAVE形式などとも呼ばれる。音声信号をデジタルデータに変換したものを記録するための保存形式などを規定している。圧縮(符号化)方式については規定しておらず、任意のものを利用することができる。デフォルトではPCM(無圧縮)方式やADPCM方式などの圧縮方式に対応している。

2.3.2 mp3

mp3(MPEG Audio Layer-3)[3]は音声圧縮方式のひとつであり、音質をほとんどそのままにして、データのサイズを約十分の一程度まで小さくできるといった大きな長所がある。このため、インターネットを介して音楽を配信する場合においても、時間をかけずに相手に送信することを可能としている。また、記録媒体により多くの音声データをいれることができるといったことがメリットとなる。だが、短所としてmp3は不可逆圧縮であり、圧縮したファイルは展開して元に戻しても全く同じにはならない、といったことが挙げられる。これは人間の耳に聞こえにくい部分を削ってデータのサイズを小さくしているために起こる。

mp3は、大きく分けて5つの行程を経て圧縮される。まず、最初にサブバンド分解と呼ばれる、元の音楽データを32分の1のサンプリング周波数でサンプリングし直すことから始めている。次に心理聴覚評価といったものを行っている。これは、人間の耳が大きな音を聞いたあとに後続する小さな音が聞こえない等の特性を利用したものであり、どの周波数域に何ビット割り当てるかを決めている。次に、MDCTを使ってサブバンド分解されたデータをさらに細かく周波数単位のデータに変換する。MDCTとは変形離散コサイン変換の略である。このデータを量子化という過程を経て周波数領域ごとに、適当に除算処理が施される。いくつの値で除算を行うかは、心理聴覚評価の結果を基に決定する。最後に、可逆圧縮であるハフマン符号化を行い、圧縮される。

2.3.3 本研究で使用する音声ファイル

本研究ではMPICH2を用いて、wav形式[8]の音声ファイルをmp3[3]形式のファイルに音声変換を行っている。4台の計算機にあらかじめ元のwavファイルを均等のサイズに分割したwavファイルを置き変換処理させている。処理時間決定の際には計測誤差を避けるために数回同一の処理を行いその時間の平均値を採用する。

本研究では、サイズの異なる4つの音声ファイルを準備した。表2に本研究で使用した音声変換前のwavファイルの詳細を示す。

表 2 本研究で使用した wav ファイル

サイズ	155MB	258MB	421MB	531MB
ビットレート	1411kbps	1411kbps	1411kbps	1411kbps
オーディオサンプルレート	44KHz	44KHz	44KHz	44KHz
オーディオサンプルサイズ	16 ビット	16 ビット	16 ビット	16 ビット
時間	15:24	25:35	41:46	52:39

3 結果

今回の検証では、4つの音声ファイルを使用してwavファイルをmp3にエンコードする。1台での処理時間および、2台、4台と計算機を増やしていったときの処理時間を計測し、並列処理によりどれだけ処理時間が短縮できるかを検証する。

3.1 検証方法

まず、入力となるwavファイルおよびそれを2分割、4分割したwavファイルを各計算機のメモリに置く。次に、1台の計算機を用いてwavファイルをmp3に逐次エンコードし、変換時間を測定する。また、2台、4台の計算機を用いて、それぞれ2分割、4分割されたwavファイルをmp3に並列エンコードし変換時間を測定する。

本研究の検証では、サイズの大小にかかわらず並列計算が有効であるかどうかということを評価するために変換前のwav形式の音声ファイルのサイズに幅をもたせた。また、計算結果に誤差が出ないように、検証を行うにあたって複数回同様の処理を行い、その平均値をもって計測結果としている。

3.2 計測結果

本検証で得た音声変換の実行時間および変換後のファイルのサイズを表3および図3に示す。

表 3 MPI による音声変換の実行時間(秒)

	変換前のサイズ	155MB	258MB	421MB	531MB
	変換後のサイズ	14.1MB	23.4MB	38.2MB	48.1MB
計算機数	1	77.13	119.31	186.08	231.10
	2	39.22	61.66	97.17	121.13
	4	20.02	31.01	51.24	64.15

表3のそれぞれの実行時間の数値より、1台での逐次処理に比べて、計算機数を増加させるとそれに伴い音声変換にかかる時間も短縮されていることが示される。

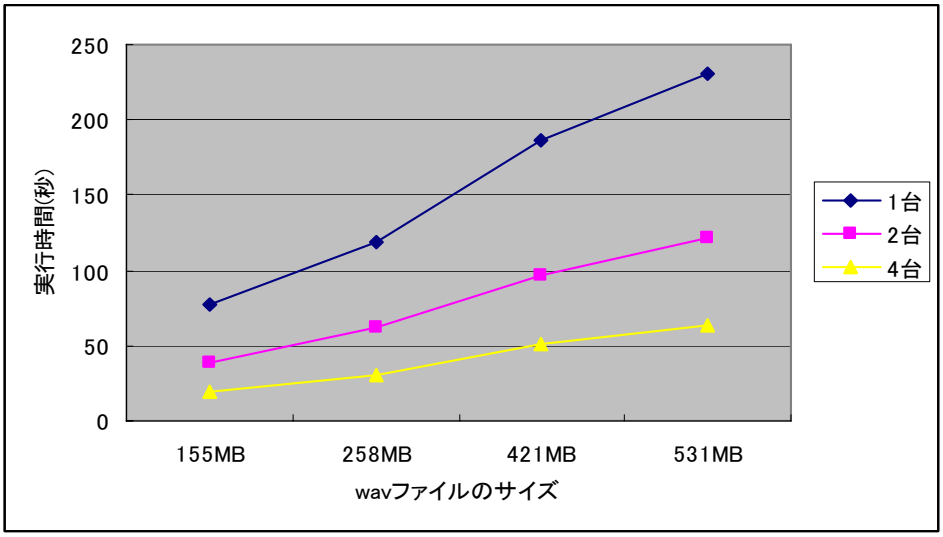

図3 各計算機台数の計測結果

4 考察

本研究の計測結果では、wavファイルからmp3のエンコードにおいて、並列計算機による高速化を実現している。この検証により、音声変換における並列計算機の有用性を証明することができた。また、サイズの大小に関わらず効率的に処理できていることからどのようなサイズであっても並列計算機は有効であることを示すことができた。

また、図3より計算機が2台、4台と増加するに伴い音声変換にかかる時間もおよそ2分の1、4分の1と処理にかかる実行時間が台数効果分短縮されていることが明らかである。このことより、計算機の数が多いほど高速化を実現しているといったことが示されている。

だが、本研究で使用したwavファイルの分割においては予め用意したものを使用している。そのため、エンコードの処理時間のみを計測結果として出力いるので分割にかかる時間を考慮していない。

5 結論・今後の課題

本研究の検証でサイズの大小に関わらず、音声変換において並列処理が有効であることが示せたが、仮想並列計算機の構成の中でスペックが低い計算機がある場合の並列計算を考慮していない。そのため、計算機のスペックに大きな違いがある時の有効的な処理方法についてはまだまだ研究の余地を残している。このことは、ファイル分割のサイズを決定する際に最も重要なことであるといえる。今回、音声ファイルの分割は各計算機に均等にサイズを割り当てたが、スペックの高い計算機に処理を多くさせることで、並列計算の高速処理化といったことになるのではないかと考えられる。

また、今回分割作業はMPIを用いていなかったが、この作業を並列計算に組み込むことで作業の効率化になり、より信頼できるデータを取り出すことができるのではないかと考えている。

これらの点より、計算機のスペックや計算機数に応じて、それに適したデータを割り当てることとファイルの分割作業を含めた音声変換処理をMPIで構築することが今後の課題である。

6 謝辞

本研究を行うにあたり、並列処理の基礎から自分達の研究内容まで時間を惜しまず、夜の遅くまでご指導をいただき、感謝の気持ちで一杯です。約 1 年半、ご迷惑をたくさんおかけしましたが先生のおかげで研究をすすめることができたと感じております。

また、同研究室の方々も私の様々な質問や相談にも快く答えてくださり、深く感謝しております。

参考文献

[1] P パチェコ 著,秋葉博 訳：MPI 並列プログラミング, 培風館（2001）
[2] 牛島省 著：OpenMP による並列プログラミングと数値計算(2006)
[3] 大澤文孝 著：たちまちわかる MP3(1999)
[4] MPICH2, http://www.mcs.anl.gov/research/projects/mpich2/
[5] PVM とは,http://m.weblio.jp/c/PVM
[6] 午後のこ～だ,http://www.marinecat.net/free/windows/mct_free.htm
[7] Visual Studio ホームページ, http://www.microsoft.com/japan/msdn/vstudio/
[8] WAV とは, http://e-words.jp/w/WAV.html
[9] MPICH2 on Windows local, http://ums.futene.net/wiki/Paralell/4D5049434832206F6E2057696E646F7773204C6F63616C.html