技術解説

PCクラスタ　－手作り並列計算環境－

萩原兼一*

PC cluster －Homemade Parallel Computing Environment－

Key Words：PC cluster, Parallel Computer, Parallel Program, registration

1．はじめに

本解説では，並列計算機の概要，最近よく使われだした“手作りの並列計算機”PCクラスタ，そして並列プログラムを作成するときの留意点を３次元画像の位置合わせ問題を例にして説明する．“計算機を使ってこんなことできればよいな”というアイデアはあるが，それを計算機処理させると長い計算時間がかかるので棚上げになっているものはないだろうか．そのようなときに，今回説明する手作り並列計算機の適用を考えるとよいかもしれない．

2．高速計算のための並列処理

パーソナルコンピュータ(PC)の性能が向上し，PCを用いて実際的な計算時間で処理できる仕事が増えてきた．それでも，数時間～数日間の計算時間が必要な仕事もある．それが，例えば100分の１の計算時間に短縮できれば，何が起きるだろうか．手術の分野ではCT画像などを利用することが多い．その画像処理に要する時間が１台のPCを用いて10時間(＝600分)ならば，医者は手術中にその画像処理を行ってできる(動的な)対応はあきらめ，手術前に行う計画などの(静的な)応用に限るであろう．それが100分の１の時間，すなわち６分，で計算結果が得られるならば，手術中に利用する可能性が広がり，手術の質を高めることができる．

計算時間を100分の１に短縮するにはどうすればよいであろうか．PCの性能を100倍に上げれば，そのような短縮は可能であろう．PCの性能を年に２倍上げることは非現実なことではなく，その調子で性能向上すれば8年後には128倍の性能に達することになる．ただ，８年も待てない場合に，現在のところ可能な対策は，プロセッサを同時に100台程度使用して“人海戦術的に”その処理を行うことである．これを**並列処理**(Parallel Processing)あるいは**並列計算**(Parallel Computation)という．プロセッサとは，コンピュータでデータを処理する頭脳部分を表す用語である．

3．並列計算機

並列に処理を行える計算機を**並列計算機**(Parallel Computer)という[1]．１台の並列計算機の中に複数(数台から数千台)のプロセッサを装備している．並列計算機に対して，プロセッサを１台装備している計算機を**逐次計算機**という．プロセッサを２台あるいは４台装備しているPCもなくはないが，一般家庭で用いるPCはプロセッサを１台装備している逐次計算機である．一般家庭で“けいさんき(計算器)”というと電卓のことだが，ここでは“コンピュータ”の意味で使っている．

3.1．共有メモリ型vs分散メモリ型

並列計算機は，メモリの形態により**共有メモリ型**(図１)，**分散メモリ型**(図２)，およびそれらの混合型に大分類できる．最近は分散メモリ型のものが増えている．後述のPCクラスタは分散メモリ型である．

比喩で説明しよう．プロセッサを仕事を行う人で，メモリを仕事をするための机と考える．各人はそれぞれ個人用机を使って仕事をする．そして，並列計

*Kenichi HAGIHARA
1952年１月生
1979年大阪大学・大学院基礎工学研究科・博士後期課程修了(物理系専攻情報工学分野)
現在，大阪大学・大学院情報科学研究科・コンピュータサイエンス専攻，教授，工学博士，並列処理工学
TEL 06－6850－6595
FAX 06－6850－6599
E-Mail hagihara@ist.osaka-u.ac.jp

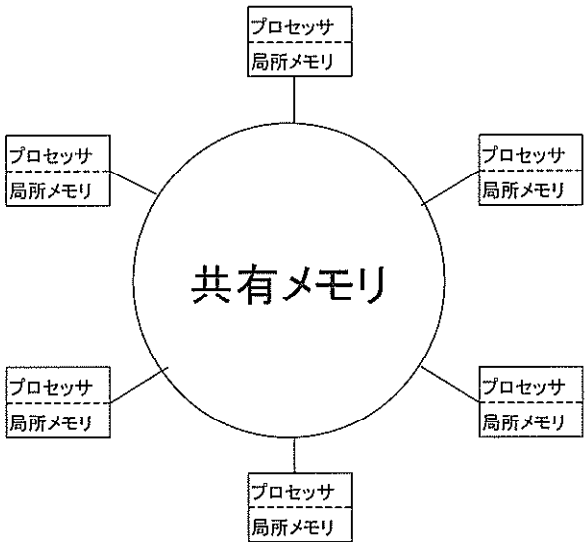


図1　共有メモリ型並列計算機

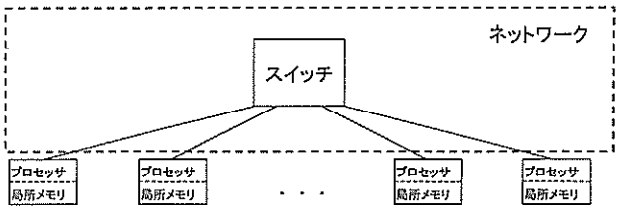


図2　分散メモリ型並列計算機

算は多くの人が協同作業している状況と考える．協同作業には，ある人A氏の仕事結果Dを他の人B氏が必要とする状況が一般に生じる．これを情報伝達と呼ぶことにする．共有メモリ型と分散メモリ型とでは，この情報伝達の方法が異なる．

共有メモリ型の並列計算機は，各人の個人用机とは別に，共有の机(共有メモリ)があり，その周りに複数の人が座って仕事をしている状況である．情報伝達の方法は，全員が机を共有しているのでA氏はDをその共有机の上に置きさえすればB氏に限らず誰でもDを利用できる．ただし，大きな共有机を作る技術は難しく，そのため共有机の大きさには限度がある．また，共有机の周りにそう多くの人が座ることはできない．現在のところ数十～百人が限度であり，数百人以上もの人が座ることはできない．

一方，分散メモリ型の並列計算機は，共有机はなく，情報伝達の方法は，A氏がDをメッセージと呼ぶ“手紙”にしてB氏に伝える(通信する)．メッセージには，伝えたい内容D以外に，送り先名，差出人名，メッセージの通し番号なども書く必要がある．メッセージをやりとりする通り道(ハードウェア)全体をネットワークと呼び，やりとりするためのソフトウェア(関数の一式)を通信ライブライと呼ぶ．メッセージを作成するには手間がかかり，したがって，結果のやりとりは，共有メモリ型に比べて不便である．また，やりとりの速度も速いとは言えない．多くのメッセージをやりとりする場合は，ネットワークの通信性能が悪いと，協同作業に支障がでる．全体での人数を制限する技術的制約はあまりなく，千人程度の協同作業は行える．そして，机の総面積(総メモリ量)は(個人机の面積×人数)であるので，それを大きくするには，個人机の面積を大きくする，あるいは人数を増やせばよいので，比較的容易である．

3.2．専用品vs汎用品

逐次計算機のアーキテクチャは“フォン・ノイマン型”と呼ばれ，各種の逐次計算機で同種のものである．したがって，逐次計算を議論するための理論モデルは同一のものを利用できる．逐次計算機用のプログラム(逐次プログラム)の性能は，そのモデルのもとだけで評価すればよく，その性能評価結果は任意の逐次計算機で有効である．直観的に言えば，ある逐次計算機で性能が2倍になるプログラムは，任意の逐次計算機でも2倍の性能向上が期待できるということである．

一方，並列計算機は，それぞれ独自のアーキテクチャをもつ．したがって，ハードウェアの開発期間が長くなり，ソフトウェアも個別に開発しなければならないことがある．そのため並列計算機は高価である．

計算機ハードウェアの歴史で，Lisp処理など特定用途専用の計算機が開発されることがある，そして，特定用途向けハードウェアと汎用ハードウェアとのどちらを利用するのが価格対性能がよいかが議論された．一般に，特定用途向けプロセッサは，開発期間が長いので性能向上は緩やかである．そして，利用者が少ない場合は高価なものとなる．一方，汎用プロセッサは，利用者が桁違いに多く，しかもメーカ間での競争が激しいので，その性能向上は速くしかも安価である．したがって，単に価格対性能だけを比較すると，汎用の普及品の性能向上を期待してシステム開発する方に軍配が上がることが少なくない．

3.3. ＰＣクラスタ

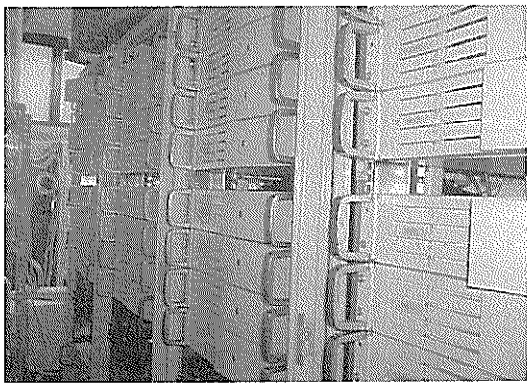
類似の構図が並列計算の分野でも起きている．並列計算機が特殊用途向けハードウェアとすれば，汎用ハードウェアに相当するものをPCクラスタと言ってよい．

PCクラスタは，ハードウェアとソフトウェアともに普及品を組み合わせて構成する．ハードウェアは複数(数十～数百)のPCをネットワークで相互接続したものである．PCは，特別な計算機ではなく，普及品であり，安価である．ネットワークは，低性能でよいならばファースト・イーサーネットFEを用い，高性能が必要ならばギガビット・イーサーネットGbEやミリネットMyrinetを用いることが多い．FEは普及品なので安価であるが，GbEとMyrinetは現時点では高価である．ネットワークを構成するには，スイッチと各PCごとPCとスイッチを結合するためのネットワーク・インタフェースボードおよびケーブルが必要である．現在のところ，高性能のPCクラスタを構成するには高性能のネットワークが必要で，総額の半分程度の予算をネットワークに当てることになる．

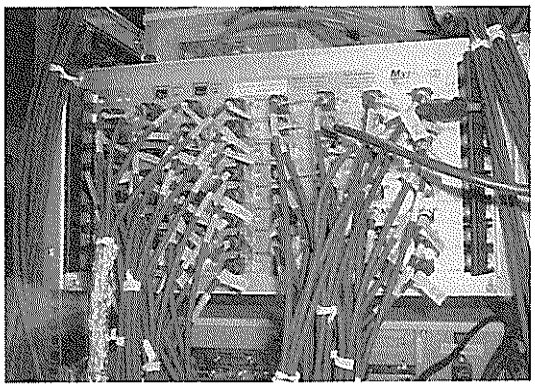
PCクラスタのソフトウェアも普及品を用いる．オペレーティングシステムはLinuxを，通信ライブラリはMPI(Message Passing Interface Standard)を用いることが多い．MPIには，プログラム言語FORTRAN用，C用，C++用のものがあり，１対１通信，１対多通信，多対多通信のための約120個のメッセージ通信用ライブラリ(関数)が用意されている[2]．実質は10個ほどの関数で，実用的な並列プログラムを書くことができる．これらのソフトウェアは無料である．

このように，ハードウェアとソフトウェアを安価あるいは無料の汎用品で構成しているという意味で，PCクラスタは「手作りの並列計算環境」と言える．さらに，予算に応じてPC台数を追加できる．PCクラスタは分散メモリ型である．

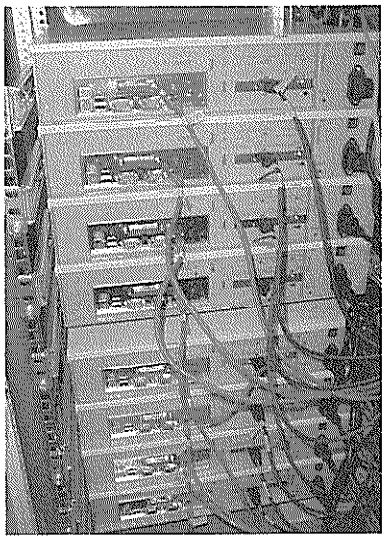
PCクラスタの一例として，筆者の研究室にあるPCクラスタを写真１に示す．30台ほどのPCが写っているが，実際には64台の構成である．写真２はMyrinet2000という高速ネットワークのスイッチであり，64台のPCを接続できる．写真３はPCのインタフェースボード(各PCの右側のケーブル部分)である．



写 真 1



写 真 2



写 真 3

4．大量メモリを使う計算のための並列処理

並列計算機を使うもう一つの目的は，逐次計算機の主記憶では格納しきれない規模のデータを扱うことである．使用可能な主記憶領域は，現在広く普及しているIntel 4などの32ビットプロセッサであれ

ば1プロセス当たり最大4GBである．

3次元画像では，1次元当たりの解像度が2倍の高解像度データを扱おうとすると，8(=2×2×2)倍のデータ量を扱わなければならないことになる．したがって，よりきれいな画像を目指して画像データの解像度を上げ出すと，1台のPCで扱うには自ずと限界がある．その解決策の一つが複数のプロセッサで画像データを分割して保持して処理することであり，並列計算機の出番となる．1台のPCでは計算できない(計算時間∞)処理が計算できる(有限の計算時間)という意味で，これも高速計算の一種と考えられないこともない．

5．並列プログラムの開発例

上手に並列計算するためには，プログラムおよびハードウェアを含むシステムのバランスが必要である．そのためには，まずプログラムのどの部分にどの程度の計算時間がかかっているかを把握することから始まる．3次元画像の位置合わせ逐次プログラムを並列プログラムにする例を用いて説明しよう[3]．

5.1．3次元画像位置合わせ問題

身体の同一部分を，異なる時刻に撮影した2つの3次元画像RとFを考える．一般に，RとFは微妙に位置がずれていて，その位置ずれが解消されるように，Fを変形する(RとFとの対応関係を求める)操作を位置合わせ(registration)と言う．この位置合わせは，患者の姿勢や呼吸などの影響により生じる画像間の位置ずれを解消し，医療における診断および手術で重要な計算機処理である．

5.2．位置合わせアルゴリズムのアイデアの概略

［制御点］

3次元の対象はボクセル(voxel)と呼ぶ小さな立方体で表現する．3次元画像が高解像度であれば，位置合わせは数時間におよぶ計算となる．ここでは，RおよびFのボクセル数として512×512×159の場合を例とする．これは各画像約4168万という大量のボクセル数となり，すべてのボクセルを位置合わせの直接の対象にするには多すぎる．それで，3次元空間における数個おきの代表的な位置を基準にして位置合わせを行う．この基準位置を制御点と呼ぶ(図3)．

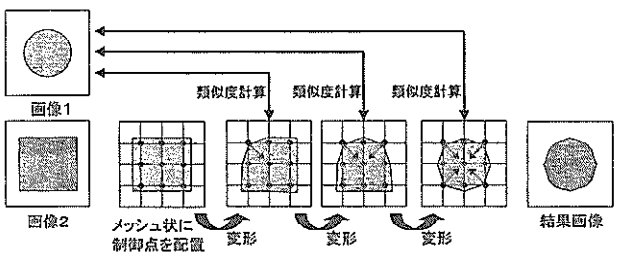


図3 位置合わせアルゴリズム

［繰り返し計算］

FをRに近づくように変形する．変形後の画像をF'と表し，F'が許容される程度にRに類似するまで，この操作を繰り返す．

［変形計算］RおよびFの制御点に着目して，Fの各制御点の周辺部分を3次元方向にどの程度移動させるべきかのベクトル値を計算すること．ベクトル値は一般に制御点ごと異なる．

［類似度計算］F'とRとがどの程度類似しているかを計算すること．

［階層的処理］

制御点の個数を，少なくとると計算時間は短いが精度の悪い位置合わせになり，多くとると計算時間は長くかかるが精度はよくなる．そこで，少ない制御点で大まかに位置合わせし，徐々に制御点の数を多くして細かく位置合わせする方法は，最初から多い制御点で細かく位置合わせするより，収束するまでの時間が短くてすむ．ここでは，制御点数は3段階に分けて行う．

5.3．逐次計算のを観測

変形計算と類似度計算の計算法はいろいろある．ここでは，変形計算でB-FFD(B-Spline Free-Form Deformation)を，類似度計算で正規化相互情報量を用いた方法を観測した．全体の計算時間約10時間40分のうち，変形計算に93.0%，類似度計算に6.9%の時間を占めている．残り(0.1%)はデータの入出力関係の時間である(表1)．

表1 逐次計算と並列計算の各フェーズの実行時間

フェーズ	逐次計算(秒)	比率	並列計算(秒)	比率	スピードアップ率
画像の初期化	67	0.1%	59.7	8.6%	1.12
変形計算	59458	93.0%	578.2	83.6%	102.83
類似度計算	4409	6.9%	53.91	7.8%	81.78
合計	63934		691.81		92.42

この計算時間の比率を観測するときに，入力データ画像は実際に使用する大きさのものを用いること

が重要である．小さな画像を用いると一般に異なった比率になり，その比率が大きな画像でも同じであると想定すると，どの部分の並列化に力をいれるべきかの判断を誤ることになる．

5.4. 並列計算可能な部分

画像データを一つのファイルサーバFSから入力するならば，FSのハードディスクからネットワークを介してファイルが移動する．FSは1台であり，FS自体は並列に動作しないので，データ入力処理は本質的に逐次的なものであるので，サーバを多重化しない限り並列化は容易ではない．

一方，変形計算は制御点ごと，また類似度計算はボクセルごと，ほぼ独立に計算できるので，並列化の効果が期待できる．並列処理の基本的なアイデアは，画像の3次元空間領域を，プロセッサの個数に分割して，それぞれの領域を独立に(並列に)計算することである．

5.5. 並列計算の観測

計算時間の大部分を占める部分(3次元画像位置合わせ問題では，変形計算の部分)を並列化することが重要である．変形計算部を観測していると，ループが進行するにつれ，変形計算を行うべき制御点が徐々に減少していること，そして，その減少が領域的に均等には生じていないことがわかる．すなわち，各プロセッサが担当する仕事量が，最初は均等に分けたとしても，処理が進むと均等でなくなることになる．負荷が均等でないと，同時に処理したい仕事を，はやく終了するプロセッサと，なかなか終了しないプロセッサが生じる．この場合，はやく終了したプロセッサは，遅いプロセッサが終了するまで，行うべき仕事がなく遊んでしまうことになり，全体としての効率が悪くなる．したがって，変形計算に関しては，3次元領域の担当部分を，処理が進行するにつれ適宜変更し，プロセッサの負荷を均等にする工夫が必要である．このような工夫を動的負荷分散といい，性能のよい並列プログラムにするには重要な要素である．

6．並列処理のオーバーヘッド

逐次プログラムSAにおけるある計算部分qを，並列プログラムPAで並列処理する場合に，一般に並列処理するための前処理FPと後処理BPが必要となる．場合によっては，並列処理の途中での中間処理MPも必要となる．FPとしては，qを並列に処理するいくつかの部分に分けること，どのプロセッサがどの部分を担当するかを決めること(タスク割当)，そして各部分を計算するために必要なデータをその仕事を担当するプロセッサに配布すること，が代表的なものである．BPとしては，並列に求めた各部分の解を，全体の解にまとめる仕事が代表的であり，そのためにも一般には通信が必要である．MPとしては，同時進行中のプロセッサの状況や中間解の情報伝達が代表的なものである．

FP, BP, MPは本来SAには存在せず，並列処理を行うがゆえに生じるものである．その意味でこれらは“並列処理のためのオーバーヘッド”と呼ばれる．並列処理そのものでqに相当する部分自体の処理時間は短くなるが，それに付随するFP, BP, MPのための時間は増える．これらをすべて含めた計算時間が，並列処理におけるqの計算時間Tとなる．そして，SAでのqの計算時間よりTの方が大きくなってしまえば，その並列処理は，効果がないことなる．オーバーヘッドを少なくするためには，メッセージを少なくすることが重要である．そのためには，FP, BP, MPにおいて，なるべく情報伝達の必要がないようにデータの担当を決めること，そして同一宛先へのいくつかのメッセージを一つのメッセージにまとめること(メッセージの一括化)が効果的である．

7．スピードアップ率

並列プログラムが逐次プログラムよりどの程度速くなっているか示す指標としてスピードアップ率S(N)がある．

ある問題Qを解く逐次プログラムSAの計算時間をtとする．SAを並列化した並列プログラムをPAとし，PAをN台のプロセッサでQを並列計算する時間をT(N)とする．このときS(N)を次の式で定義する．

$$S(N)=t/T(N)$$

PAが高速であればT(N)の値が小さく(短時間に)なるのでS(N)の値は大きいほど並列プログラムの性能がよいことを表している．

PAを1台のプロセッサで動かすこともできるが(その計算時間はT(1)となる)，それはSAより効率

はよくない．すなわち，t≦T(1)であり，t/T(N)≦T(1)/T(N)となるが，S(N)はT(1)/T(N)で定義するものでないことに注意してほしい．

8．スピードアップ率の上限

SAを並列化した並列プログラムPAをN台のプロセッサで並列処理すれば，理論的にはどの程度速くなりえるかを考えてみよう．すなわち，S(N)の値はどの程度大きくなるか，その限界を考えてみる．

3次元画像の位置合わせ処理を，画像の初期化，変形計算，類似度計算の3つに分類し，それぞれの部分の逐次計算および128プロセッサ(64台のdual PC，主記憶2GB/PC，Myrinte2000(2.0Gbps))による並列計算での処理時間，そして部分ごとのスピードアップ率を表1に示す．変形計算および類似度計算は，それぞれ約103倍および約82倍になっていてそれなりの並列化の効果がみられるが，画像の初期化は1.12倍と非常に低い．この理由は，1台のFSからデータを取り出しているので本質的に逐次的処理であり，並列化の恩恵をほとんど受けることができないからである．

このようにSAにおいて，並列化が比較的難しい部分SPと比較的容易な部分PPとが存在する．

［アムダールの法則］

SAの実行時間におけるSPとPPの実行時間をそれぞれsとpとする．PPをN台のプロセッサで“理想的”に並列処理できるならば，逐次実行時間がN等分されてp/Nとなる．ここで，理想的とは，並列処理のオーバーヘッドがなく．プロセッサ間の負荷がまったく等しい状況を意味する．実際にはほとんどありえないが，S(N)の理論的な上限(これ以上は速くできないという限度)を考えるうえでは，大きめの値が出ても許されるという立場である．SPの部分は並列化でも速くできないので，PAの計算時間は

$$T(N)=s+p/N$$

となる．SAの計算時間は

$$t=s+p$$

であるので，スピードアップ率S(N)の上限は次のようになる．

$$S(N)\leqq(s+p)/(s+p/N)$$

これを“アムダールの法則”という．sとpは実際にはSAの計算時間における比率でよく，したがって，

$$s+p=1$$

と仮定しても一般性を失わない．この法則によれば，比率sが小さくなるほどS(N)の上限は大きくなる．すなわち並列化に向くと期待できる．データ量が大きくなればなるほど比率sは小さくなる可能性が高いので，大規模計算ほど並列化の効果を期待できる．

(例)128台のプロセッサ(N=128)で，本質的に逐次な場所の計算時間の比率が0.1%(s=0.001，p=0.999)の場合は，S(128)=113.58となる．すなわち，128台のプロセッサを用いても，約114倍が限度であることを示している．ただし，114倍で必ずできることを意味しているわけではない．

9．並列処理の第3の効用

並列処理による大量の計算ができるようになると，物を考えるときの発想自体が変わる．これが並列処理の第3の効用である．物理現象を扱う場合を例にして説明する．大量計算が得られない場合は，“xxxの法則”などによる対象全体をまとめて見る統計的な処理を行わざるをえない．しかし，大量計算が使えるという想定では，対象をより細かい構成要素(極端な場合は，原子などの小さいもの)からなるシステムと見て，それらの構成要素を計算の対象とすることができる．すなわち，対象をあるがままに扱うことが可能となる．このような扱いにより，処理自体はより自然となり，予想外の発見を期待できる．

10．むすび

PCクラスタは価格に比べて高い性能が得られるので今後ますます普及するであろう．しかし，PCクラスタの管理と，PCクラスタで効率よく動く並列プログラムを開発することは容易ではない．

並列計算を必要とする学問分野をかかえる(大学の)学科では，“(逐次)プログラミング”の教育だけでなく，“並列プログラミング”の教育も重要となる．

参考文献

[1] D.E.Culler, J.P.Singh, and A.Gupta：“Par-

allel Computer Architecture", Morgan Kaufmann Publishers, Inc. San Francisco, 1998.
[2] W.Gropp, E.Lusk, A.Skjellum : "Using Mpi: Portable Parallel Programming With the Message-Passing Interface (Scientific and Engineering Computation Series) 2nd", MIT Press,1999.
[3] 大山寛郎, 竹内彰, 伊野文彦, 萩原兼一 : "分散メモリ型並列計算機を用いた非剛体レジストレーションの並列化", 電子情報通信学会技術研究報告, vol.102, no.577, pp.5-10, 2003.