解　　説

PCクラスタの作り方

廣安　知之*・三木　光範*

1. はじめに

どの分野でもそうですが，PCでデータを整理したり文章やレポートを書いたりすることが当たり前になりました．オフィスにも研究室にもPCがあふれています．夜遅くまで仕事をしていて，ふと辺りを見渡すと，使用されていないPCが怪しくスクリーンセーバーを輝かせたりしています．それらのPCはもちろんネットワーク接続されています．さらに，CPUやハードディスクは日に日にその性能を劇的に向上させているのです．「これらのPCを統合して使用すれば高性能な計算機になるのでは？」とお考えになるのは自然なことだと言えるでしょう．

PCクラスタは，複数台のPCをネットワーク接続することにより構成された並列計算機です．PCクラスタの最大の特徴は市販の製品・部品でシステムを構築することにあります．これらの製品とLinux OSに代表されるようなフリーウエアを利用することで，PCクラスタの大きな魅力であるコストの削減が可能になるのです．

では，その性能はどうでしょうか．まずは，TOP500 Supercomputer SitesのTop 500 Sub List[1]をご覧ください．このページには，世界のスーパーコンピュータをLinpackとよばれるベンチマークソフトで計測した結果の上位500位がランキングされています．第35位にはthe University of HeidelbergのHelics Cluster[2]が，第47位には東工大・松岡研のPrest III PC　クラスタ[3]が，第50位にSandia National LabのC Plant Cluste[4]が位置しています．これらは実はPCクラスタなのです．個々の部品は市販製品ですが，CPU，ネットワークなどの性能の飛躍的な進歩に伴い，PCクラスタといえどもスーパーコンピュータと言える性能を有した並列計算機を構築することが可能なのです．

これらはHPC（High Performance Computing）と呼ばれる分野での例ですが，webサイトなどでの使用などのHA（High Availability）の分野ではさらに数多くのPCクラスタが稼動し，実績をあげています．みなさんがよくお使いになると思われる検索サイトGoogleでも多数のPCを活用しているようです[5]．Star Warsなどの映画の特殊効果で有名なIMLでは600台のPentium 4マシンを使用しているそうです[6]．もちろんこれらはネットワーク結合されて利用されているでしょうから，クラスタシステムと言えるでしょう．第1図は，同志社大学の私どもの研究室にある，128CPUのPCクラスタの写真ですが，多くのPCクラスタの外観は，このように市販のPCが並んでいるものになります．

このようにPCクラスタは近年，特に注目されており，ここ数年で広く使用されるようになりました．また，8台や16台からなるPCクラスタは価格的にも手ごろですし，使いやすいシステムです．さらに，Linuxマシンなどの管理をされた経験がある方であれば，誰でも基本的なPCクラスタシステムを構築することができるでしょう．

そこで，本稿では，小規模なPCクラスタシステムの構築についての概説を行います．PCクラスタシステムでは，いろいろな技術が組み合わさり利用されておりますので，すべてを詳細には説明できません．ここでは，PCにLinuxをインストールし利用した経験のある方を対象に，できるだけ必要な項目とその情報源を示すことを本稿の目的としたいと思います．よろしくお付き合い下さい．

第1図　Gregorクラスタ（同志社大学）

2. PCクラスタとは？

クラスタ(Cluster)とは英語で「ブドウなどの房，同種類のものの群れ」という意味です．

Pfisterによるとクラスタの定義は以下の様になります[7]．

クラスタ:

*同志社大学・工学部

Key Words: PC Cluster, Parallel Processing, Linux

クラスタは単一で稼働するコンピュータの集まりで，一つの計算機資源として使用可能な並列もしくは分散システムである．

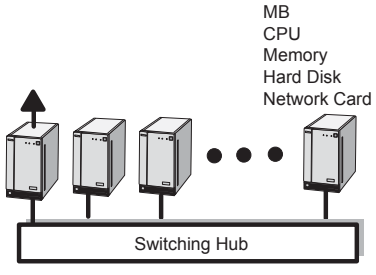
そのため PC クラスタは，パソコン（PC) を複数台集めてきてネットワーク結合して房を作ることになります．必要な機材 (Hardware) は次のようなものです．

- PC 数台（ネットワークカードを有しているもの）
- スイッチングハブ
- ネットワークケーブル

次に，小規模（8 台程度）の PC クラスタを構築する手順は以下の通りです．

(1) 上記の機材を用意する．
(2) 上記の PC をネットワーク接続する．
(3) 互いに通信可能で，プログラムをコンパイルおよび実行可能な環境を構築する．そのためには OS，いくつかのツール，ネットワーク関連ソフト，コンパイラ，および並列通信ライブラリなどをインストールする．
(4) PC の電源を入れる．
(5) プログラムをコンパイル，実行．

非常に簡単です．それでは，以下に必要なハードウエアとソフトウエアについて説明を行います．



第 2 図 PC クラスタの構成

3. ハードウエア

ハードウエアとしては，PC を構成しているマザーボード，CPU，メモリ，ネットワークカード，筐体およびスイッチングハブなどを選択します．それぞれのパーツを自分で選択し構築することもできますし，すでに組み立てられた市販の PC を利用することももちろん可能です．

選択の方法としては，まず全体の予算を決定し，そのうちいくらをハードウエアに充てられるのかを見積もります．ノード数が少ない場合は，一般的なネットワークカードの価格，稼動するアプリケーションが必要とするメモリ量の価格を考えながらノード数を決定すると良いでしょう．

3.1 CPU

年々，市販の CPU の選択肢が狭まっています．今では Intel の Pentuimu や AMD の Athron などの x86 系の CPU が無難な選択でしょう．旧コンパック（旧 DEC）のアルファチップは魅力的な CPU でしたが，DEC はコンパックに買収され，コンパックがアルファチップをインテルに売却し，コンパック自体も HP と合併してしまいました．現段階では Pentium 4 が非常に優秀な成績をおさめています．次回の Top500 でも Pentium4 系の Xeon を搭載したクラスタが多くランクインされるものと考えられます．AMD は次期 CPU の Hammer で挽回を狙っているものと思われます．年々処理速度は上がっていますが，それに応じて大きな消費電力が必要となり，発熱量も問題となっています．

今後は，処理速度だけではなく，後述する設置スペースなどとの兼ね合いで，モバイル系に搭載されるような消費電力の少ない CPU も注目されて行くでしょう．

3.2 メモリ

PC クラスタシステムにおいて，ネットワークが問題になる場合が多いのですが，実は，メモリもシステムの性能を大きく左右します．できるだけ速いメモリを大容量搭載することが重要となります．メモリは CPU と同様，現在，過渡期にあり，今後はどのようなメモリが主流になるのかは不透明です．

負荷の高い計算を連続して行うような場合には，良いメモリを使用されることをお勧めします．その場合には，サーバー系のマザーボードが必要となるかもしれません．ワープロや表計算を行うような，一般のユーザーが使用する場合には，安いメモリでも稼動しますが，負荷の大きな計算を長時間行う場合には，メモリが原因で稼動しないことがあります．筆者らの経験では，メモリのベースクロックが 100MHz と表示されていたのに実際には 133MHz のもので，問題となったことがありました．

3.3 ネットワーク

ネットワークハードウエアは PC クラスタシステムのトータル性能を左右する大きな要因の一つです．安価な PC クラスタを構築するためには FastEthernet を利用します．しかし，どのボードも同一性能ではなく，ドライバとの相性などによって帯域やレイテンシの性能が十分に得られない場合があります．ネットワークの性能を高めるためには，ネットワークカードを複数枚用意する方法と，性能の良いカードを利用する方法があります．2 枚のネットワークカードを利用して，1 枚では NFS や NIS などの管理用のデータを流し，2 枚目に MPI などのデータを流すようにしてやるとネットワークのトラフィックが低減します．

Myricom 社 [11] の Myrinet は HPC 分野での PC クラスタシステムの標準的なネットワークです．最近では SCI も注目されており，Dolphin 社 [12] などが発売しています．大規模なクラスタを構築する場合には，これらのネットワークが必要となります．これらのカードとスイッチは高価ですので，大規模なクラスタを構

築する際には，ネットワークのコストが総コストの中で占める割合が高くなり，場合によっては半分近くになることもあります．

また，最近では最大 1000Mbps のスループット実現可能なギガビットネットワークも価格がこなれてきており，使いやすくなっているようです．しかしながら性能をフルに発揮するには，ネットワークカードとスイッチの相性問題を解決する必要があるようです．

3.4 ハードディスク

PC クラスタの構成方法として，マスターとなる PC1 台だけがハードディスクを持ち，他のノード（スレーブ）はディスクを持たず，マスターのハードディスクを共有して利用するディスクレスクラスタというものがあります．筆者らは，過去にはディスクレスクラスタを推奨していたこともありますが，最近では，各ノードはディスクを持つべきだと考えています．ハードディスクは PC クラスタを構成する部品の中でも壊れやすいものであるという印象がありますが，実はそうではありません．安定稼動後はそれほど壊れるものではありません．しかも，近年では IDE ハードディスクが非常に安価になったために，これをけちっても価格にあまり反映しません．それとは逆に各ノードにディスクがあるほうが，インストールは簡単ですし，アプリケーションを稼動させるときにも，ローカルな tmp ファイルに高速に結果を書き込むことができるので便利です．

3.5 筐体

ハードウエア面での最近の流行は，ラックマウント形の筐体です．PC クラスタは価格性能面で非常に優れているわけですが，ノード数を多くしていくと，電力を多く消費する，空調がよけいに必要な場合があるという問題に加えて，設置の場所が多く必要だという欠点があります．

これは HPC 向けの PC クラスタだけの問題だけではなく，web サーバー系でも大きな問題となっているため，各社ともにできるだけ少ないスペースに多くの CPU を詰め込むことを競いあっています．ラックマウント型システムはその答えの一つです．1 ラックあたりにできるだけ多くのシステムを搭載することが課題です．ただし，ラックマウント型のシステムでは，発熱の問題があり，搭載可能な CPU が限られています．たとえば，Pentium4 などのような熱を多く発生する CPU は搭載されていなかったり，Myrinet といったような特別なネットワークを敷設することができないなどといったこともあります．よって，これらの CPU やネットワークを選択する場合には注意が必要です．最近，Apple もこの分野に参入し，話題を集めています [14]．

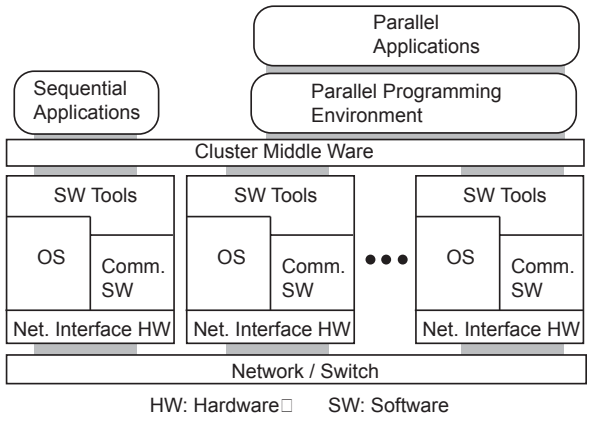
3.6 どのような頻度で故障するのか

一般に PC クラスタは壊れやすいイメージがありますが，いったん安定稼動し始めますと，故障はそれほど頻繁におこりません．初期不良は確かにありますので，最初のチェックは肝心です．筆者らの経験では，最初にできるだけ丹念に負荷のかかるテストを行い，初期不良を洗い出すことが肝心だと思います．簡単なプログラムでは動いたのに，負荷をかけたとたん，ブレーカが落ちたこともありました．負荷をかけることによって初めてメモリに問題があることがわかったこともあります．経験が浅かったために，初期にテストを行わず，購入してから 1 年以上も格闘しているクラスタもあります．

PC を構成する部品はその開発サイクルが非常に短いために，構築後しばらくしてから部品のどこかに障害が起こり交換が必要となった場合，同一の部品を購入することが不可能な場合がよくあります．たとえば，CPU のクロック数が上がってしまっているとか，ハードディスクの容量が大きなものしか購入できないなどです．実際の運用では，問題ないと思いますが，このような場合でも，どうしても同一のスペックで各ノードを揃えたい場合には，PC クラスタの導入時に，あらかじめいくつかの予備の部品を購入しておくことが必要となります．

4. ソフトウエア

PC クラスタを構成するソフトウエアは第 3 図 に示す通りです．



第 3 図　クラスタにおけるソフトウェア

これらのそれぞれについて管理者はインストールする必要があります．

4.1 OS

現在，PC クラスタの OS として Linux が広く利用されています．その理由は，フリーウエア，オープンソースであることが挙げられます．PC クラスタはまだまだ発展途上のシステムであるため，OS のソースが公開されていることが必要となるわけです．Windows で

も PC クラスタの構築は可能ですが [13]，情報が少ないため，問題に直面したときに解決に時間がかかること，各ノードに OS が必要なため，ノード数が増加すると OS のコストが馬鹿にならないことなどから，最初に PC クラスタを構築される場合にはお勧めしません．フリーウエア/オープンソースの選択として Free BSD[16] や NetBSD[17] なども選択肢として挙げられますが，ユーザー数が多いのは Linux です．

Linux kernel およびいくつかのアプリケーション，インストーラなどはまとめてディストリビューションと呼ばれています．最近ではすでにコンパイルしてあるバイナリをパッケージとしてインストールすることが一般的です．Red Hat[18] や Debian GNU/Linux[19] がその代表です．ここではそれらのインストールについてはスキップしますが，いくつもの参考書が存在します [20–23]．最近のディストリビューションには，後述する MPICH などといった並列計算に必要なパッケージが付属しています．

4.2 開発環境

プログラムをコンパイルするために，コンパイラが必要となります．最初の段階としては GNU コンパイラ [24] で十分でしょう．

しかしながら CPU の性能を十分に引き出すためには，それに対応したコンパイラが必要となります．Pentium 4 に搭載されている SSE2[25] や Athlon の 3DNow[26] などの命令の活用が重要です．

PGI 社製 [27] や NAG 社製 [28] および Absoft 社製 [29] などのコンパイラは定評のあるところです．富士通製 [30] も OpenMP に対応し，HPC での技術を Linux 上で稼動する形でクラスタにポーティングしつつあります．最近では Intel もコンパイラを提供しています [31]．

並列アプリケーションの作成における問題点の一つが，デバッグが簡単に行えないという点です．各ノードで間違い無く動くアプリケーションでも通信が発生すると正常に動作しない場合があるからです．しかも複数ノードを効率よくデバックを行うためにデバッガーは GUI 表示できることが必須です．GNU デバッガ [32] や strace[33] などのツールが多少の手助けになるかもしれません．TotalView[34] は広く使用されている GUI ベースのデバッガーです．これは，一般的に使用されている言語やメッセージパッシングライブラリに対応しています．一方，どの程度の通信遅延がどこで生じているか，あるいはどの関数でどれだけの時間がかかっているのかを知ることはプログラムを高速化するために必要なこととなります．それらを教えてくれるのがプロファイラーと呼ばれるツールです．フリーウエアとしては後述する MPICH に含まれている MPE や Paradyn[35] が比較的有名です．市販のものとしては Vampier[36] が広く使用されています．

4.3 ネットワークソフトウエア

ネットワークは PC クラスタシステムのボトルネックとなるためネットワーク関連のソフトウエアに注意を払わなければなりません．特にネットワークドライバの性能によって，通信性能が大きく異なることを知っておく必要があります．すなわち，ドライバのバージョンが上がったにもかかわらず特定のネットワークカードとの相性が悪いために通信性能が下がってしまうこともあり得るのです．さらにネットワーク性能の向上を図りたい場合や，Myrinet を使用している場合には，通信プロトコルを TCP/IP から他のものに変更することを考慮すべきです．代替案としては，Myrinet+GM もしくは Myrinet, Ethernet+SCore です．SCore[37,38] は RWC で開発された Cluster でのネットワークシステムソフトウエアで非常に性能の高いソフトウエアです．現在では Score 型 PC クラスタともいうべきクラスタシステムとなっており，全世界のクラスタで利用されています．開発は PC クラスタコンソーシアム [39] に引き継がれています．

4.4 管理ソフト

PC クラスタを構成する各マシンにはそれぞれ，個人の情報やディレクトリが存在するため，ノード数が増大するとその管理が大変煩雑になります．そのためにそれらを一元管理する仕組みが必要となります．

Network File System (NFS) はファイルを一元管理します．これを利用することで，ユーザーはどのマシンにログインしても，同一のディレクトリを参照することができます．しかしながら，Linux の NFS はまだまだ性能が低いと言われていますし，ネットワークを介してサーバーディスクに書き込み/読み込みを行うわけですから，ディスクアクセスを頻繁に必要とするようなアプリケーションの場合は，ローカルディスクにまず書き込みを行い，後でデータを整理するような方法を選択した方が無難でしょう．

ユーザーのログインネームやパスワード，グループなどの一元管理を行う一つの方法が，Network Information System (NIS) です．しかしながら，NIS を使用する場合もネットワーク通信の問題や，信頼性の問題が発生します．PC クラスタ内ではそれほど頻繁にパスワード変更することは考えられませんから，ユーザーが増えた時やパスワード変更が行われた際に，パスワードファイルをすべてのノードにコピーすることで事は足ります．ユーザー数が増加した際には，LDAP[40] を利用することも一つの方法です．

ローカルディスクが大きくなった場合にはジャーナリングファイルの導入の検討が必要でしょう．RedHat がサポートしているような ext3 などは非常に使い勝手のよいジャーナリングファイルシステムです．ジャー

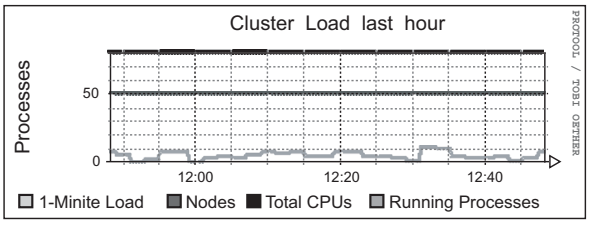
ナリングファイルを利用することで，突然の停止の際に再起動しなくてはいけない場合にもいらいらすることがなくなります．

Makefile などを使用する場合には，各ノードで時間がずれている場合には問題となりますので，統一するのが便利です．それを行うのが Network Time Protocol (NTP) です．

構築したクラスタを有効に利用するためには Resource Management and Scheduling (RMS) の導入が必要となる場合があります．RMS はプロセスの配分，ロードバランスの調整，複数タスクのジョブスケジューリングを行うため，ユーザーの助けを必要とせず，各ノードが自動的に効率良く使用されることになります．CONDOR[41][35] や PBS[42] はフリーウエアとして代表的なものです．LSF[43] は高価ですが，非常に高機能な市販のシステムです．最近では，GRID にも対応しているようです．

その他，クラスタシステム全体を一括して制御してくれるスクリプトやコマンドがあると便利です．簡単に自作も可能だと思いますが，Cluster Command and Control[44] などがあります．私たちのグループにも dsh というクラスタ用のスクリプトを開発している学生がいます [45]．

クラスタの稼動状態がわかるようなモニタがあると，どのノードが落ちているのか一目でわかるので便利であり，また，なにかと楽しいものです．Ganglia Cluster Toolkit[46] はそのようなモニタの一つです．



第 4 図　Ganglia のスナップショット

4.5　PC クラスタ構築ソフト

ここまでは，PC1 台ごとに Linux OS，開発環境，後述するメッセージパッシングライブラリ，管理ソフトなどをインストールして PC クラスタを構築する例でしたが，もちろん，PC クラスタ全体をセットアップするシステムもいくつか存在します．

Oscar[47]，Rocks[48]，Scyld[49]，および Lucie[50] といったシステムはその代表例です．

Lucie は Debian GNU/Linux をベースとした日本発のシステムです．アップデートするなら，最初からインストールという新しい発想を基にしたシステムです．先に紹介した Score[37] も日本発の優れた PC クラスタシステムです．

我々のグループも Lucie を見習ったセットアップシステムを現在，構築中です [51]．

5.　プログラミング

PC クラスタは PC が複数台ネットワーク結合されているわけですが，各 PC はノードと呼ばれます．各ノードは CPU およびメモリを有しています．各ノードの CPU はノード内のメモリのデータにはアクセスできますが，他ノードのメモリのデータにアクセスすることができません．これは分散メモリ型並列計算機と呼ばれています．そのため，あるノードの CPU が他ノードのデータにアクセスするためには，他ノードの CPU にデータを送るよう依頼し，データを受信する必要があります．これがメッセージパッシングです．PC クラスタにおいて効率的なプログラミングを行うためには，このメッセージパッシングを実装する必要があります．1 対多通信を行う時や頻繁に使用する命令を一々記述するのは非効率ですから，これらの通信はライブラリとして用意されています．その一つが MPI[52] です．

MPI(Message Passing Interface) はメッセージ通信のプログラムを記述するために広く使われる「標準」を目指して作られたメッセージ通信の API 仕様です．MPI の標準化への取り組みは Supercomputing ’92 会議において，後に MPI フォーラムとして知られることになる委員会が結成され，メッセージ・パッシングの標準を作り始めたことで具体化しました．これには主としてアメカ，ヨーロッパの 40 の組織から 60 人の人間が関わっており，産官学の研究者，主要な並列計算機ベンダのほとんどが参加しました．そして Supercomputing ’93 会議において MPI 標準の草案が示され，1994 年に初めてリリースされました．

プログラムを書くにあたって並列プログラムの実行の様子について概念的に理解しておく必要があります．手順を述べると，まずユーザはクラスタの１つのマシン，あるいは専用の並列計算機のホストとなっているマシンにログインを行います．そこでユーザはクラスタ，あるいは専用の並列計算機に対してジョブの投入を行います．ジョブの投入のために MPI ではスクリプト，専用の並列計算機では専用のコマンドが用意されています．プログラムが無事終了すると，ファイル，あるいはユーザのモニタに結果が出力されます．このように実際に並列に走る個々のプロセスの起動は，実行環境に任されています．プログラマはプログラムの中で，最初に通信ライブラリを利用するための簡単な手続きを記述し，その後で実際の並列処理の部分を書いていけばよいことになります．

MPI は API 仕様でしたので，実際に使用する際にはその実装を必要とします．MPICH[53] は，アメリカのアルゴーン国立研究所 (Argonne National Laboratory) が MPI（Message Passing Interface) の模範実装とし

て開発し，無償でソースコードを配布したライブラリです．移植しやすさを重視した作りになっているため盛んに移植が行われ，世界中のほとんどのベンダの並列マシン上で利用することが可能です．さらに，SMP, Myrinet などのハード面にも対応している上，DQS, Globus といった様々なツールを使用できることも大きな特徴の一つです．実装方法の中身は, 送信を行うためのドライバを取り替えることで対応するようになっています．たとえば, ワークステーションクラスタならネットワークを使った通信ドライバを, 共有メモリならメモリバスを使った通信ドライバをリンクするといった具合です．また，2002 年 9 月現在における最新バージョン MPICH 1.2.4 では，MPI-1.2 の全ての機能をカバーしており，MPI-2 に関しても幾つかの機能についてはサポートしています．

MPI によるプログラミングの方法についてはここでは述べません．MPICH をインストールしますと例題も付属してきます．代表的なテキスト [54,55] を参照されると良いでしょう．MPI では多くの API が用意されていますが，実際に使用するのはそのうちのいくつかです．

MPI ライブラリには，他にも多くの実装があります．Lam[56] は University of Notre Dame で開発された MPI のフリーなライブラリです．WMPI[57] は Universidade de Coimbra で開発された Windows 上で使用可能な MPI ライブラリです．一方，高い性能が必要な場合には市販の MPI ライブラリの選択もあります．

メッセージパッシングライブラリは MPI だけではありません．PVM は MPI と並ぶ代表的なメッセージパッシングライブラリです [58]．MPI が普及するまでは広く使用されていました．現在でも，その多機能性や柔軟性により人気があります．

その他，メッセージパッシングのみだけでなく，各ノードが SMP である場合に対応した技術もあります．MPI/PRO[59] は MPI Software Technology 製のライブラリです．ノード内が SMP の場合には，thread という技術を用いてプログラミングすることとなります．Open MP[60] は逐次型のプログラムの中に指示文を挿入することで並列化を行うことができます．

6. その他の項目

6.1 PC Cluster は本当に安価か？

PC クラスタの特徴の一つとして，そのシステムの構築の際にコストを抑えることが可能であることを述べました．しかし，価格を抑えるためにはそれなりに努力が必要です．

まず，ハードウエアの部分でコストを削減した場合，先に述べたようにシステムが不安定となり，負荷が高いアプリケーションを連続して運用できない場合があります．また，ノード数が多くなった場合には，スイッチングハブにコストをかけることが重要となります．一般に使用されているスイッチングハブは，オフィスで使用されるような場合を想定して設計されています．すべてのノードが同時にフルバンドでアクセスするようなことを想定していないために，ここの部分がネックとなり性能がでない場合があります．また，安いハードウエアを使用したために故障が多く発生し，逆にコストがかかる場合もあります．

ソフトウエアはフリーウエアのものを中心に使用することでコストを削減できます．一方で，これらのソフトウエアでの結果に満足できない場合には，良い結果が期待できる商用ソフトウエアを使用することが必要となります．たとえば，Pentium 4 の SSE2 の機能を十分に活用するためには，それをサポートするコンパイラを使用する必要があります．商用ソフトウエアは，場合によっては，ノード数分のライセンスが必要な場合もありますので，そのような場合にはソフトウエアのコストが馬鹿になりません．

これまでのスーパーコンピュータで最もコストのかかる部分は実は人件費と保守料金だったのではないでしょうか．大学などで PC クラスタを構築する際には，学生や若手の教員で構築することにより，人件費を削減することが可能となります．保守料金も，壊れることが覚悟の上であれば，削減することが可能です．それに対して，これらの人件費や保守料金を企業に依頼すると，PC クラスタはもはや低コストな並列機とは呼べなくなってしまうかもしれません．

しかしながら，我々が購入できる乗用車をいくらチューンアップしても，F1 カーにはなりませんが，我々が利用している PC の延長上には，スーパーコンピュータがあるのというのは大きな魅力です．

6.2 どこから情報を入手するか？

クラスタに関連する図書はいくつかあります [7–10,37]. しかしながら，ハードウエア/ソフトウエアともに日々進化しているために，やはり情報はネット上の方が新しい場合が多いと考えられます．

Debian GNU/Linux では Beowulf プロジェクトがあり，PC クラスタ関連ソフトの移植を行っています [15]. このプロジェクトのメーリングリストも情報源となるでしょう．

また，IEEE には Cluster Computing の Task Force グループ (TFCC)[61] があり，このサイトではニュースレターやホワイトペーパを公開しています．

日本でも，超並列計算研究会 [62] や SofTek の主催するメイリングリスト [63] などがあり，クラスタに関連する情報を得ることができます．

クラスタに関連する国際会議としては，Super Computing[64]，Cluster[65], CCGrid[66]，その他並列処理

に関連する国際会議が挙げられます．

国内でも，SACSIS（旧 JSPP)[67]，超並列計算研究会 [62]，NEC・HPC 研究会，IBM HPC フォーラム，Compaq HPTC ソリューション Forum などが開催され，クラスタに関する話題も提供されています．

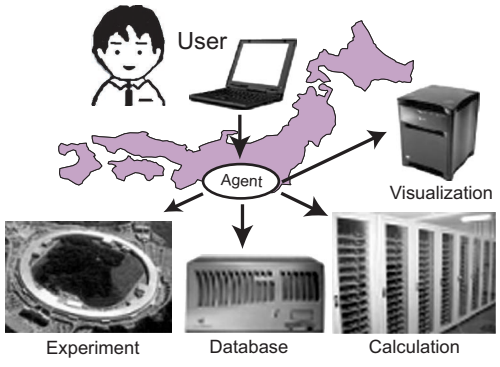
6.3 グリッドコンピューティング

PC クラスタは複数の PC をネットワーク接続して使用する分散システムでした．現在，あらゆるもののネットワーク化が進んでいます．計算機だけでなく，家電や乗用車，航空機，各種の実験装置までインターネットに接続されようとしています．

このような背景のもと，PC クラスタ同士をネットワーク接続することでさらに巨大なシステムを構築することができるようになるかもしれません．計算サーバーだけでなく，データサーバや可視化処理サーバーとも接続することで，多様なサービスを構築することができる可能性もあります．しかも，ユーザーの使用する計算機はそれほど大きくなくても良いのです．これがグリッドコンピューティングです．

グリッドとは電力網 (Power Grid) にその語源があります．電気は電気製品のケーブルをコンセントにさすだけで良質な電源が供給されます．しかも，その電源がどこで製造されたかはユーザーにはわからないのです．そのような知識がなくとも，ただ，ユーザーはサービスが享受できるのです．

グリッドコンピューティングもそのような状態が理想です．第 5 図の概念図に示すように，日本中（世界中）に存在するネットワーク接続されたあらゆる実験装置，データベース，計算サーバ，可視化装置などがユーザーは意識することなく利用可能となるわけです．



第 5 図　Grid コンピューティングの概念図

HPC の分野での利用を考えたとき，もっとも便利なツールが RPC によるシステムでしょう．代表的なミドルウエアが NetSolve[68] と Ninf[69] です．これらのミドルウエアはほぼ，同等のソフトウエアで，最近ではこれらのインターフェースを統一し Grid RPC としようという動きがあります．

グリッド環境下でのシステムを構築しようとする際に，現在もっとも広く使用されているのが Globus Tool Kit(GTK)[70] です．GTK はセキュリティやリソース管理などグリッド環境下で必要となるモジュールを用意しています．現在のところ Ver.2 が最も新しいものですが，今後は Open Grid Services Architecture(OGSA)[71] と呼ばれる規格に向かって行くようです．OGSA は Globus チームと IBM が中心になって動いている新しい規格ですが，一番のポイントはグリッドが計算だけでなく，web サービスも取り込むことができるようにしようということです．これまで，グリッドというと大規模な計算の可能性が注目されていましたが，多様な web サービスと大規模な計算の融合をねらっています．確かに各企業ともさかんに web サービスを提供しています．しかしながら，それらのサービスは，ユーザーにとってはただ提供されるだけの，一方的なものです．それらのサービスがグリッドと融合することによって，ユーザーはより自由に多様なシステムを構築することが可能になるのです．しかもその際には，基本的にはユーザは巨大な計算サーバも大規模なデータベースも手元に持つ必要はないのです．

今後，このようなシステムの中で，サービスを行うサーバーや，計算サーバーとして PC クラスタが中核を担うことになるであろうということは言うまでもありません．

7. おわりに

非常に駆け足で，小規模な PC クラスタの構築について解説しました．複数台の PC を用意し，ネットワーク接続し，各 PC に Linux をインストールして，MPI にてプログラミングを行えば，PC クラスタはすぐさま使用できます．ですから，その使用は非常に簡単であるといえます．

次のステップとしては，ノード数を増加させたり，ギガビットを利用したり，SCore を導入したりしてみると面白いかもしれません．

これまで，PC クラスタを構築されたことのない方が本稿をきっかけにされて，その第 1 歩を踏み出されれば幸いです．

参 考 文 献

[1] TOP500 Supercomputer Sites. http://www.top500.org/list/2002/06/.
[2] Helics Cluster, University of Heidelberg. http://helics.iwr.uni-heidelberg.de/.
[3] 東工大/松岡研，Prest III PC クラスタ. http://cluster-team.is.titech.ac.jp/.
[4] C Plant Cluster, Sandia National Lab. http://www.cs.sandia.gov/cplant/.
[5] Google Bets The Farm On Linux. http://www.internetwk.com/lead/lead060100.htm.
[6] ”Star Wars” effects studio shifts to Intel.

http://news.com.com/2100-1001-945310.html.
[7] G. F. Pfisher: In search of CLuster. Prentice Hall 1998.
[8] T. Sterling: How to Bbuild a Beowulf. MIP Press,1999(日本語訳有り).
[9] R. Buyya: High performance Cluster Computing. Prentice Hall,1999.
[10] D. Spetor: Building LINUX CLUSTERS. O 'Reilly, 2000.
[11] Myricom: http://www.myricom.com/.
[12] Myricom: http://www.dolphinics.com/.
[13] Windows Clusters Resource Centre : http://www.windowsclusters.org/.
[14] Apple, Xserve. http://www.apple.co.jp/xserve/index.html.
[15] Debina GNU/Linux, Beowulf Project. http://www.debian.org/ports/beowulf/index.ja.html.
[16] FreeBSD Project. http://www.freebsd.org/.
[17] NetBSD Project. http://www.netbsd.org/.
[18] Red Hat. http://www.redhat.com/.
[19] Debina GNU/Linux. http://www.debian.org/.
[20] B. Ball: 標準Ｒｅｄ Ｈａｔ Ｌｉｎｕｘリファレンス. インプレス, 2001.
[21] 高原: Ｒｅｄ Ｈａｔ Ｌｉｎｕｘ ７．１サーバー構築入門. ソーテック,2001.
[22] 武藤: Ｄｅｂｉａｎ ＧＮＵ／Ｌｉｎｕｘ徹底入門. 翔泳社,2000.
[23] 芳尾: 今日からＤｅｂｉａｎ ＧＮＵ／Ｌｉｎｕｘ ２．２．オーム社,2000.
[24] GNU Compiler. http://gcc.gnu.org/.
[25] C Net Japan, SSE2. http://japan.cnet.com/Help/manual/0851.html.
[26] AMD. 3DNow!TM テクノロジの詳細. http://www3pub.amd.com/japan/products/cpg/k623d/inside3d.html.
[27] The Portland Group. http://www.pgroup.com/.
[28] The Numerical Algorithm Group. http://www.nag-j.co.jp/.
[29] absoft. http://www.absoft.com/index2.html.
[30] 富士通コンパイラ. http://www.fqs.co.jp/fort-c/.
[31] Intel Compilers. http://www.intel.com/software/products/compilers/.
[32] GDB: The GNU Project Debugger. http://sources.redhat.com/gdb/.
[33] strace homepage. http://www.wi.leidenuniv.nl/~wichert/strace/.
[34] ETNUS, TotalView. http://www.etnus.com/.
[35] Paradyn Parallel Performance Tools. http://www.cs.wisc.edu/~paradyn/.
[36] Vampire. http://www.cadence.com/datasheets/vampire.html.
[37] RWC SCore. http://www.rwcp.or.jp/topics/score/score.html.
[38] 石川他: Linuxで並列処理をしよう. 共立出版社,2002.
[39] PC Cluster Consortium. http://www.pccluster.org/.
[40] Open LDAP. http://www.openldap.org/.
[41] The Condor Project. http://www.cs.wisc.edu/condor/.
[42] PBS. http://www.openpbs.org/.
[43] Platform, LSF. http://www.platform.com/.
[44] Cluster Command and Control. http://www.csm.ornl.gov/torc/C3/.
[45] DSH, distributed shell. http://www.netfort.gr.jp/~dancer/software/dsh.html.
[46] Ganglia Cluster Toolkit. http://ganglia.sourceforge.net/.
[47] OSCAR. http://oscar.sourceforge.net/.
[48] NPACI, Rocks. http://www.rocksclusters.org/papers/rocks-documentation/book1.html.
[49] Scyld Beowulf Cluster Operating System. http://www.scyld.com/.
[50] LUCIE. http://matsu-www.is.titech.ac.jp/~takamiya/lucie/.
[51] MakeCluster. http://mikilab.doshisha.ac.jp/dia/research/person/kenzo/makecluster/index.html.
[52] The Message Passing Interface (MPI) standard. http://www-unix.mcs.anl.gov/mpi/.
[53] MPICH. http://www-unix.mcs.anl.gov/mpi/mpich/.
[54] パチェコ著, 秋葉訳: MPI 並列プログラミング. 培風館, 2001.
[55] 青山: 並列プログラミング虎の巻. http://www-6.ibm.com/jp/rs6000/resource/ufo/Whatsnew/tora3.htm.
[56] LAM/ MPI. http://www.lam-mpi.org/.
[57] WMPI. http://www.criticalsoftware.com/wmpi/.
[58] Parallel Virtual Machine. http://www.csm.ornl.gov/pvm/pvm˙home.html.
[59] MPI/PRO. http://www.mpi-softtech.com/.
[60] Open MP. http://www.openmp.org/.
[61] IEEE, TFCC. http://www.ieeetfcc.org/.
[62] 超並列計算研究会. http://www.is.doshisha.ac.jp/SMPP/.
[63] HPC メーリング・リスト. http://www.softek.co.jp/services/auto.html.
[64] Super Computing 2002. http://www.sc2002.org.
[65] IEEE, Cluster 2002. http://www-unix.mcs.anl.gov/cluster2002/.
[66] IEEE, CCGRID. http://www.ccgrid.org/.
[67] 先進的計算基盤システムシンポジウム. http://www.hpcc.jp/sacsis/.
[68] NetSolve. http://icl.cs.utk.edu/netsolve/.
[69] Ninf. http://ninf.apgrid.org/.
[70] The Globus Project. http://www.globus.org/.
[71] Open Grid Services Architecture . http://www.globus.org/ogsa/.