

計算機システム概論 II

第3回（平成15年度9月19日） 並列処理とグリッドコンピューティング

担当 佐藤

計算機システム概論 II

もくじ

- ◆ “ CPUは何をしているのか ”
 - トランジスタ、論理回路からマイコンまで
- ◆ CPUを速くする方法
 - パイプライン
 - スーパースカラ
 - ...
- ◆ 並列処理とは
 - 並列プログラミング
- ◆ クラスタコンピューティング
 - PCクラスタ
- ◆ グリッドコンピューティング

プロセッサ研究開発の動向

- ◆ さらなる高速化、高性能化へ
 - クロックの高速化、製造プロセスの微細化
 - いまでは3GHz, 数年のうちに10GHzか
 - プロセスは0.13 μm
 - 量子的な限界?
 - アーキテクチャの改良
 - スーパーパイプライン、スーパースカラ、VLIW...
 - キャッシュの多段化、マイクロプロセッサでもL3キャッシュ
 - マルチスレッド化、Intel Hyperthreading、複数のプログラムを同時に処理
 - Intel IA32: Xeon, P4
 - クロックの高速化でトレンドを維持
 - PC市場を堅持する戦略
 - Intel IA64: Itanium2
 - 64ビットは避けられない(サーバ市場)
 - 低消費電力Itanium2 (Deerfield)、低価格化
 - AMD x86-64: Opteron
 - x86アーキテクチャの64ビット版
 - x86互換、低価格
 - Sun SPARC, IBM Power, ... Alpha, MIPS

プロセッサ研究開発の動向

- ◆ 低消費電力化へ
 - ユビキタスコンピューティング&組み込みシステム
 - どこにでもコンピュータ, バッテリ駆動
 - ノートPCやPDA、モバイル機器からのニーズ
 - ゲーム機
 - 市場としてはこちらの方が大きい
 - 電力とは
 - $P = CV^2f$ (P: 電力、C: 回路容量、V: 電圧、f: 周波数)
 - 容量 微細化、オフチップのアクセスの低減
 - 低電圧化、電圧可変 (DVS: Dynamic Voltage Scaling)
 - 周波数可変 (Intel SpeedStep,...)
 - いろいろなプロセッサがある
 - ARM: Strong ARM, Xscale (インテル)
 - Mobile Pentium, Centrino
 - Crusoe (TransMeta)
 - SH (日立): SH-1, SH-2, SH-3, SH-4, SuperSH
 - DragonBall (モトローラ)
 - MIPS: VR? (NEC)
 - PowerPC (IBM)
 - 各社いろいろなプロセッサ、M32R (三菱、ルネサス), ...

並列処理と分散処理

- ◆ 並列処理(parallel processing)とは、複数のプロセッサを用いて、処理を高速化する技術
 - HPC(High Performance Computing)
 - 数値シミュレーションなど
 - HTC (High Throughput Computing)
 - 大量のデータ処理
- ◆ 分散処理(distributed processing)は、もちろん、複数のプロセッサを用いるため処理を高速化することもあるが、本質的にはいろいろな場所で行う様々な処理、あるいは機能を結合し、機能分担させることが目的であり、必ずしも高速化だけが目的ではない
 - 分散オブジェクト技術
 - RMI , J2EE, Jini...

並列と並行

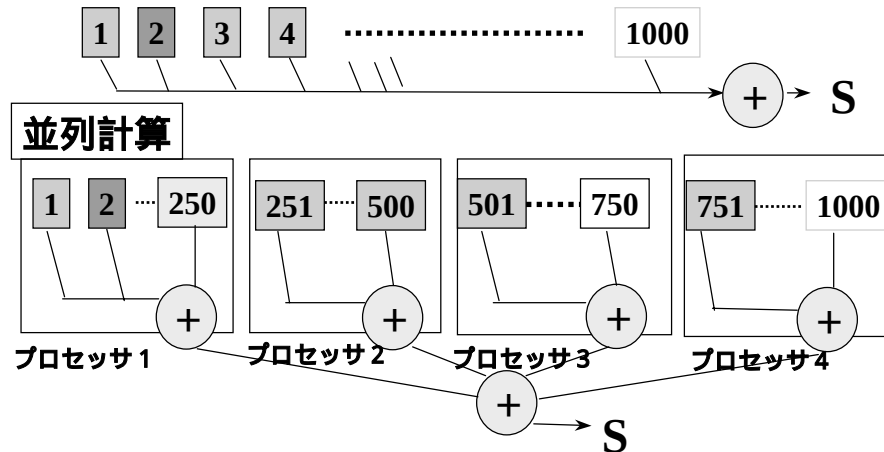
- ◆ 並列処理の「並列」(parallel) という用語は、「物理的に同時に動作する」ことを意味する
 - 複数のCPUで同時に処理されているものを制御する場合には「並列」
- ◆ 「並行」(concurrent) は「論理的に同時に動作する」ことを扱う場合に用いる
 - 一つのCPUの中でいろいろな処理に対応するサービスを行うオペレーティングシステムは「並行」処理の典型的な例
- ◆ スレッド(thread)：一連の命令実行の流れのことをいう
 - たとえば、1つのプログラムの実行のこと
 - 2つのプログラムを同時に実行することをマルチスレッド(multi-thread)実行という。

並列処理の簡単な例

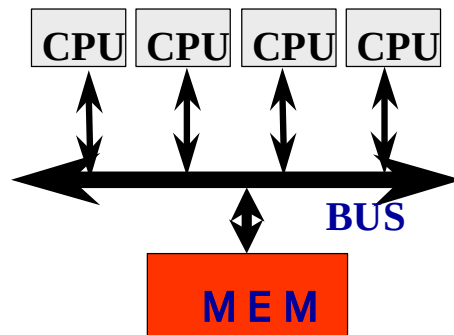
逐次計算

```
for(i=0; i<1000; i++)  
  S += A[i]
```

並列計算



共有メモリ型計算機

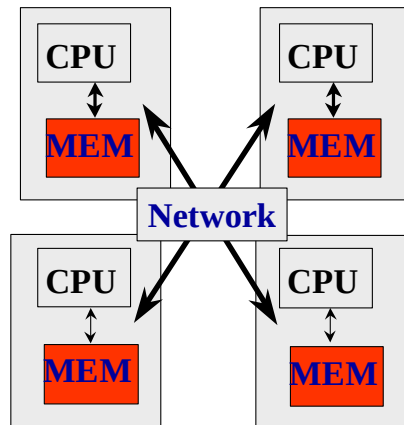


◆複数のCPUが一つのメモリにアクセスするシステム。

◆それぞれのCPUで実行されているプログラム(スレッド)は、メモリ上のデータに互いにアクセスすることで、データを交換し、動作する。

◆大規模サーバ

分散メモリ型計算機



◆CPUとメモリという一つの計算機システムが、ネットワークで結合されているシステム

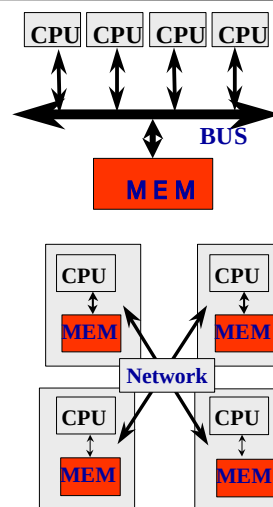
◆それぞれの計算機で実行されているプログラムはネットワークを通じて、データ(メッセージ)を交換し、動作する

◆超並列 (MPP : Massively Parallel Processing) コンピュータ

◆クラスタ計算機

並列処理の利点

- ◆ 計算能力が増える。
 - 1つのCPUよりも多数のCPU。
- ◆ メモリの読み出し能力(バンド幅)が増える。
 - それぞれのCPUがこのメモリを読み出すことができる。
- ◆ ディスク等、入出力のバンド幅が増える。
 - それぞれのCPUが並列にディスクを読み出すことができる。
- ◆ キャッシュメモリが効果的に利用できる。
 - 単一のプロセッサではキャッシュに載らないデータでも、処理単位が小さくなることによって、キャッシュを効果的に使うことができる。
- ◆ 低コスト
 - マイクロプロセッサをつかえば。



クラスタ技術

並列プログラミング

- ◆ **メッセージ通信 (Message Passing)**
 - 分散メモリシステム (共有メモリでも、可)
 - プログラミングが面倒、難しい
 - プログラマがデータの移動を制御
 - プロセッサ数に対してスケーラブル
- ◆ **共有メモリ (shared memory)**
 - 共有メモリシステム (DSMシステムon分散メモリ)
 - プログラミングしやすい (逐次プログラムから)
 - システムがデータの移動を行ってくれる
 - プロセッサ数に対してスケーラブルではないことが多い。

並列プログラミング

- ◆ **メッセージ通信プログラミング**
 - MPI, PVM
- ◆ **共有メモリプログラミング**
 - マルチスレッドプログラミング
 - pthread, solaris thread, NT thread
 - **OpenMP**
 - 指示文によるannotation
 - thread制御など共有メモリ向け
 - HPF
 - 指示文によるannotation,
 - distributionなど分散メモリ向け
- ◆ **自動並列化**
 - 逐次プログラムをコンパイラで並列化
 - コンパイラによる解析には制限がある。指示文によるhint
- ◆ **Fancy parallel programming languages**

メッセージ通信プログラミング

◆ sendとreceiveでデータ交換をする

```
int a[250]; /* それぞれ、250個づつデータを持つ */

main(){ /* それぞれのプロセッサで実行される */
    int i,s,ss;
    s=0;
    for(i=0; i<250;i++) s+= a[i]; /*各プロセッサで計算*/
    if(myid == 0){ /* プロセッサ0の場合 */
        for(proc=1;proc<4; proc++){
            recv(&s,proc); /*各プロセッサからデータを受け取る*/
            s+=ss; /*集計する*/
        }
    } else { /* 0以外のプロセッサの場合 */
        send(s,0); /* プロセッサ0にデータを送る */
    }
}
```

POSIXスレッドによるプログラミング

◆ スレッドの生成

Pthread, Solaris thread

```
int s; /* global */
int n_thd; /* number of threads */
int thd_main(int id)
{ int c,b,e,i,ss;
  c=1000/n_thd;
  b=c*id;
  e=s+c;
  ss=0;
  for(i=b; i<e; i++) ss += a[i];
  pthread_lock();
  s += ss;
  pthread_unlock();
  return s;
}
```

```
for(t=1;t<n_thd;t++){
    r=pthread_create(thd_main,t)
}
thd_main(0);
for(t=1; t<n_thd;t++)
    pthread_join();
```

- ◆ ループの担当部分の分割
- ◆ 足し合わせの同期

OpenMPによるプログラミング

これだけで、OK!

```
#pragma omp parallel for reduction(+:s)
for(i=0; i<1000;i++) s+= a[i];
```

OpenMPとは

- ◆ 共有メモリマルチプロセッサの並列プログラミングのためのプログラミングモデル
 - ベース言語(Fortran/C/C++)をdirective (指示文) で並列プログラミングできるように拡張
 - OpenMPの指示文は並列実行モデルへのAPIを提供
 - 従来の指示文は並列化コンパイラのためのヒントを与えるもの
 - 科学技術計算向け (5%のコードが95%の実行時間を実行)
- ◆ 共有メモリマルチプロセッサシステムの普及
 - SGI Origin /ASCI Blue Mountain System
 - SUN Enterprise
 - PC-based SMPシステム
- ◆ 米国コンパイラ関係のISVを中心に仕様を決定
 - Oct. 1997 Fortran ver.1.0 API、 Nov 2000 ver.2.0 API
 - Oct. 1998 C/C++ ver.1.0 API
 - <http://www.openmp.org/>

クラスタコンピューティング

- ◆ PCやネットワークなど、汎用（コモデティ）の部品を組み合わせ、並列システムを構成する技術
 - PCが急激に進歩した！
 - ネットワークも早くなっている！
 - 安価に、個人レベルでも並列システムを作れる！

クラスタコンピューティング

- ◆ クラスタシステム：既存のワークステーションやPCを（既存の）ネットワークで結合して、並列計算を行うシステム
- ◆ 第1世代：既存のLANで並列計算するシステム
 - Poorman's supercomputer
 - 遊休のワークステーションを利用
 - お茶大 Sun IPC cluster（計算センタのワークステーションを利用）
 - イーサネット
- ◆ 第2世代：クラスタ専用の計算機システム
 - etlwiz Alpha cluster, 東大喜連川研クラスタ
 - 100 BASE-TX SWITCH, ATM
 - beowulf class クラスタとも呼ばれる。
- ◆ 第3世代：高速のネットワークによるクラスタ
 - 高並列計算機（MPP）なみの性能
 - RWCP PC cluster
 - Myrinet, Gigabit Ethernet, Fiber Channel, DEC Memory Channel, IBM SP2 network
- ◆ その他
 - SMPクラスタ（UCB CLUMPS, RWC COMPaS）

クラスタコンピューティングを支える技術

- ◆ ハード、ソフトのコモデティ化、高性能化、標準化
- ◆ ハードウェア
 - プロセッサテクノロジーロードマップの恩恵
 - 急激な高性能化、価格性能比の向上
 - ネットワークの高性能化
 - ethernet : 10Mbps から 100Mbps そしてGigabit ether
 - MyrinetなどのSAN Network
 - 高性能 I/Oインタフェースの標準化
 - PCI など
- ◆ ソフトウェア
 - 並列通信ライブラリの発展・標準化
 - PVM, P4, TCGMSG, MPI, MPI2
 - 標準ライブラリ上に並列ソフトウェアが開発できる。
 - フリーなオペレーティングシステムの普及

クラスタの“レシピ”

- ◆ まず、ノードプロセッサの選択
 - PC ならば、20万円、DEC alphaならば、50万円、SMP ?
 - メモリサイズ、デスクサイズは？
- ◆ 何台？
 - (8 台で、PC ならば160万円)
- ◆ ネットワークは何にする？
 - 100 Base Ethernetならば、ノードあたり2万円、スイッチ150万円、ハブ？
 - myrinetならば、ノードあたり30万円、スイッチ80万円
 - gigabit ether ?
- ◆ ノードのオペレーティングシステムは？
 - FreeなOS (netBSD, linux)ならば、ただ
 - Solaris/x86, Digital Unix
 - Windows NT
- ◆ 通信ライブラリ
 - TCP/IPならば、MPICH (だだ！)
 - 専用のネットならば、ドライバが必要
- ◆ その他
 - ラック、ケーブル・ハブ、コンソールスイッチボックス (50万)
- ◆ あとは、組み立て！

Clusters in RWCP TRC



RWC Workstation Cluster I
(5 SS20s 75 MHz)
1995



RWC Workstation Cluster II
(36 SS20s 75 MHz)
1996



RWC PC Cluster I
(32 Pentiums 166 MHz)
1996



RWC PC Cluster II
(64 Pentium
Pros 200 MHz)
1997



Expanded RWC PC Cluster II
(128 Pentium Pros 200 MHz)
1998



RWC Alpha Cluster I
(32 Alpha 21164s 500 MHz)
1998



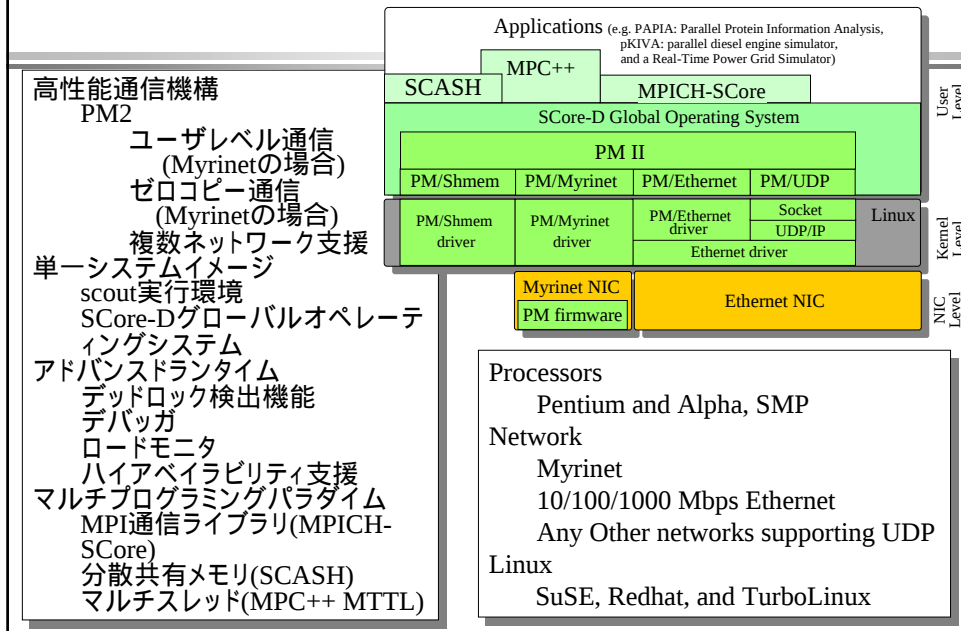
RWC SCore Cluster I
(16 dual Pentium III 500 MHz
16 Compaq XP-1000 500MHz)
1999

クラスタのソフトウェア

◆ SCore

- クラスタ向けの基本ソフトウェア
 - 通常のソフトウェアを組み合わせただけでは不十分
- 高速通信ライブラリPM
 - TPC/IPをつかわない
 - Myrinetなどの高速の専用ネットワークが使える
- シングルシステムイメージ
 - ギャングスケジューリング：並列ジョブのマルチタスキング
- 大規模クラスタのサポート

SCore Version 3 Software Architecture



クラスタテクノロジーの課題

- ◆ 並列ソフトウェアは、MPPとそれほど違いはない。ソフトは依然として問題。
 - MPIはやはり、面倒！
 - OpenMP！HPF？
- ◆ 通信インタフェースの高性能化
 - 既存のプロトコルでは通信性能を使いこなせない
 - Gigabit Ether 100MB/s以上だが、TCP/IP/Gigabitでは30MB/s
 - リモートメモリ通信
 - たとえば、VIA
 - 標準化、コモデティ化は依然として重要
- ◆ 並列プログラムのジョブ管理
 - マルチユーザ並列プログラム環境
 - たとえば、Score-D/gang scheduling(RWCP), implicit co-scheduler (UCB)
- ◆ スケーラビリティは重要か？中規模までならクラスタで十分
 - RWCP 1000CPU/512ノードのクラスタを構築
 - ネットワークが重要、ほとんどMyrinet
 - NCSAやSandia-Lab等で、大規模クラスタができている

グリッドとは

- ◆ 世界中のPCをつないで計算 seti@home,...
 - これだけではない！
- ◆ スーパーコンピュータをつないで大規模計算
 - これだけではない！

グリッドとは

- ◆ グリッド技術とは広域の高速ネットワーク上において大量のデータ、計算資源、貴重な装置等を共有し、協調作業、資源の有効活用するネットワーク基盤技術（ソフトウェア、ネットワーク、ハードウェア）と、これを活用する応用技術
 - 大量のPC（クラスター）などの計算資源の相互に共有し、大量かつ大規模な計算を行う。
 - 大規模なデータの処理。例えば、加速器の観測データ、電波望遠鏡の観測データなどを大量のPCなどの計算資源で、処理する技術が注目されている。
 - スーパーコンピュータ等の高速な計算機を結合し大規模な計算を行う（meta-computing）。
 - 遠隔の計算資源と手元のPCなどの計算機をシームレスに結合する技術（computing portal）
 - 電子顕微鏡、衛星などの高価な装置の遠隔共有
 - 電子会議システムなどの共同作業のサポート
 - 研究室などにある遊休の計算機の利用

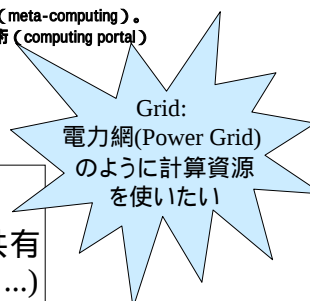
従来のインターネット技術

E-mail
www,
データの交換

グリッド

計算資源の
シームレスな共有
(CPU, Storage, ...)

計算資源の仮想化



グリッドとは

- ◆ なにが変わったのか？
 - 以前から、分散コンピューティングの研究はあった。
 - インターネットの急激な進歩、普及
 - 全世界で共通な基盤を作ろうとしている！(Grid Forum, ..., ApGrid)
- ◆ Grid: (高速) 広域ネットワーク上で、データ、計算資源、装置を共有し協調作業をサポートするネットワーク基盤
- ◆ 何につかえるのか？
 - 計算資源(スーパーコンピュータ)を共有し、大規模計算を行う(meta-computing, MPI-G, ..., ITBL?)
 - 遠隔の計算資源と手元のPC/WSをシームレスに結合(computing portal, GridPRC)
 - 大量のデータの処理(data intensive computing, gfarm for ATLAS)
 - 高価な装置の遠隔共有(電子顕微鏡、衛星データ、加速器?)
 - 共同作業のサポート(電子会議システム、AccessGrid)
 - 遊休の計算機の利用、SETI@home, P2P, ..., XtermWeb

グリッドのソフトウェア

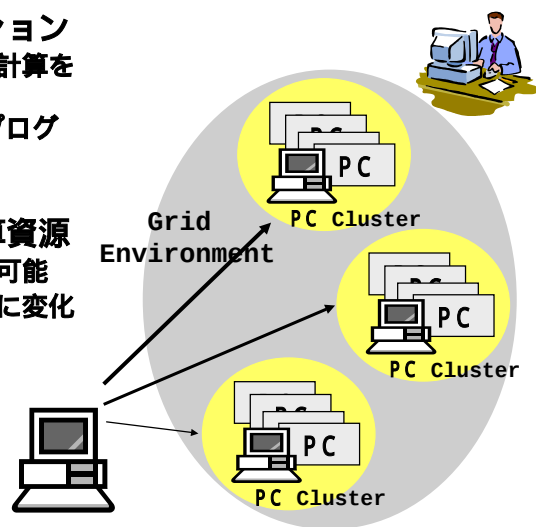
- ◆ Globus: 最もmajorになりつつあるgridコンピューティングのツールキット(ミドルウェア)
 - ジョブ起動、セキュリティ、通信のためのライブラリ
 - globusrun: globus 版のrsh
 - MPICH-G: globus版のMPICH
 - gftp: globus版のftp
- ◆ Nimrod-G(enfusion): クラスタ、Grid向けのjob dispatcher. jobレベルのパラメータサーチを行うツール(GUI付き)
- ◆ Condor-G: 遊休の計算機に対するジョブスケジューラ。プロセスの移送もサポート
- ◆ GridRPC: Grid上の遠隔手続き呼び出し
 - Ninf-G, NetSolve, OmniPRC

Globus

- ◆ Resource management ; GRAM
 - 計算資源の割り当て・ジョブ起動実行制御
- ◆ Communication Infrastructure Globus IO
 - 様々なプロトコルをサポートする通信レイヤ
- ◆ Metacomputing Directory Service MDS
 - Grid上のGlobusで利用できる計算資源の情報提供
 - LDAPを使った情報提供
- ◆ Globus Security Interface GSI
 - 認証などのセキュリティ機構
 - X509 Certificate-base の認証
 - Single-Sign-ON
- ◆ Heartbeat Monitor HBM
 - システムの状況モニタ
- ◆ Remote data access GASS
 - ファイルへのリモートアクセスサービス
- ◆ executable management GEM
 - 実行ファイルの構築、転送

グリッドと並列アプリケーション

- ◆ 典型的なアプリケーション
 - パラメータ検索：同じ計算を膨大な計算資源で実行
 - master-slave型の並列プログラム
- ◆ 典型的なグリッド計算資源
 - 複数のクラスタが利用可能
 - 計算資源の状況が動的に変化



Gridの並列プログラミング

- ◆ Globus shellベースのjobのsubmit

- ◆ MPICH-G

- Grid上のMPI (メッセージ通信プログラム)
- 汎用だが、すべて書かなくてはならない。
- 資源のリソースの増減に対応できない。

- ◆ Grid RPC: Ninf, NetSolve

- 直感的でわかりやすい
プログラミングインタフェース
- 並列プログラミングは 非同期なRPC
 - activeなrequestをユーザが管理

```
Ninf_call_async(A,...)
Ninf_call_async(B,...)
...
Ninf_call_wait()
```



OmniRPC

OmniRPC

- ◆ グリッド並列プログラミングミドルウェアOmniRPCの開発

- グリッドでの **Master-workers** 型の並列プログラムをサポート
- Gridのend-pointの計算資源として、**クラスタ**を対象とする
 - プライベートアドレスのクラスタについてもサポート
 - Firewall内のクラスタにもアクセス可能
- 既存のプログラムに対し、簡単な並列プログラミング環境を提供
 - 並列プログラミングの記述、制御はOpenMPで行うことができる
- クラスタでは "rsh", globus環境では "GRAM"、"ssh" 環境でも使用できる。
- 各計算資源の管理ポリシーを考慮したジョブの起動をサポート
 - サイトのスケジューラ (PBS, SGE,...)
- 大規模なグリッド環境 (upto 1000 hosts!) のサポート
- <http://www.omni.hpcc.jp/OmniRPC> で公開中

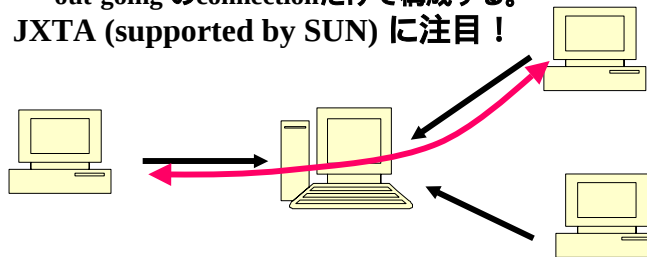
- ◆ グリッドアプリケーション

- HMCS-G (Grid-enabled Heterogeneous Multi-computer System)
 - 広域ネットワーク環境において、貴重な計算リソースである重力専用計算機GRAPE-6を共有し、汎用計算機と融合計算を行う。
 - OmniRPCを用いて、セキュリティ、認証、プログラミングインタフェースを提供
- 網羅的分子探索プログラムのグリッド並列化
 - グリッド上に分散している大規模なクラスタ計算資源を利用して、計算が可能
 - OmniRPCで複数のクラスタで実行、有効性を検証

P2Pコンピューティング

◆ P2P (pear to pear)コンピューティング

- ファイル交換ソフト(Gnudella,...)で注目
- 各PCが自立的につながる形態
 - 各PCは対等の関係(?)
- 各PCは、ファイアウォールの中でもつなぐことができることが重要
 - out-going のconnectionだけで構成する。
- JXTA (supported by SUN) に注目!



グリッドコンピューティングの動向

- ◆ 標準化はGGF (Global Grid Forum)で議論されている。
 - IETF、W3Cのような役割
- ◆ 現在のホットな話題は OGSA (Open Grid Service Architecture)
 - GridにWeb Serviceのコンセプトを導入
 - Globus Toolkit 3に試験的に導入
 - コマーシャルでの適用、利用拡大を目指している
- ◆ 各メーカーとも次世代のインターネット技術として注目している。
 - IBMがleading...
- ◆ 日本ではNational Research Grid Project (NaReGI)が進行中
- ◆ P2Pはグリッドコンピューティングとはちょっと方向が違うが、新しいアイデアとして注目
 - JXTA by SUN

終わりに

- ◆ 並列処理は必然の技術
 - プロセッサの高速化も本質は並列処理
- ◆ クラスタ技術
 - 並列システムが身近に
- ◆ グリッド技術
 - ネットがつながっていれば、計算資源を統合して利用できる

- ◆ では、何につかうか？
 - 1000台のPCでも使える時代に
 - イマジネーションが必要