Penguin Power:
Linux and Commodity Parallel Computing

William Emmanuel S. Yu
Department of Information Systems and Computer Science
Ateneo de Manila University
30/05/04

The INTERNET allows us to let computers TALK to each other. The GRID allows us to let computers WORK with each other.

The next logical step in the evolution of computing.

Contents

Introduction

Parallel Computing

- Dividing a single computing task into smaller tasks to be executed in multiple processing elements

- Why do we need parallel computing?
  - Make things run faster
  - Add the capability for dealing with a lot more data
  - Because its cool!

- Do I need parallel computing?
  - In most cases, No. commodity computing systems are sufficiently capable of dealing with day to day tasks
  - Depends on the architecture! (not all problems are parallelizable)

Benefits

- Cost effective way of bringing in Supercomputing Power – computing power >5 generations ahead!
- Enables current commodity computing systems to solve high performance computing problems
  - Grand Challenge Problems (Weather Modeling, Molecular Modeling and others)
  - Code Breaking - (#1 Supercomputer User – US NSA!)
- Provides parallel computing infrastructure to local institutions for educational purposes
  - Training on a low cost cluster computer can also apply to MPP systems
- Institutional Pogi Points – for having a supercomputer

Characteristics of Parallel Apps

- Data Parallel
  - Dividing data amongst the different processing elements
  - Most parallel computing applications are!

- Granularity
  - Refers to the dependency of one processing element to another
  - *Fine Grain*
    - Does require a lot of interaction
    - Dense data exchange
  - *Coarse Grain*
    - Does not require a lot of interaction
    - Sparse data exchange

Building Blocks

Building Blocks

- All required software and hardware are available off-the-shelf

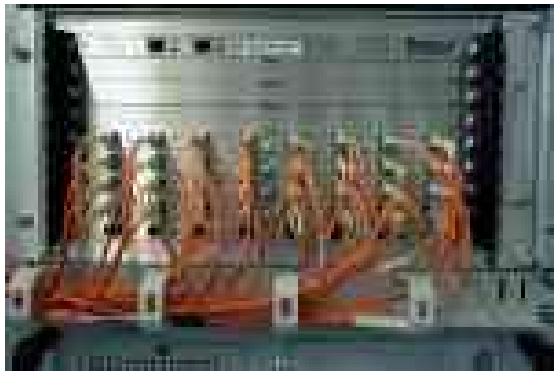
- Processing Elements
- High Speed Network
- Shared Storage
- Remote Execution Environment

Processing Elements

- Low-cost high performance x86 class processors
- Commodity high performance memory architectures
- Wide bandwidth computer architectures
- Open source operating systems such as Linux and FreeBSD

High Speed Network

- Commodity High Speed Networking Technologies
- Use regular network topologies



Myrinet

- Myricom's Myrinet – capable of 1-10 Gbps throughput
- Fast/Gigabit Ethernet – easily available

CISCO SYSTEMS

Shared Storage

- Clustering is usually about data!
- Significantly impacts performance of clusters!

- Hardware - High Performance and Cost, Low Flexibility Option
- Hardware Shared Storage Options
  - Direct Attached Storage Devices, SAN, NAS

- Software - Mediocre Performance, Low Cost, High Flexibility Option
- Software Shared Storage Options
  - Network-based File Systems (NFS)
  - Specialized Cluster File Systems (GFS, Intermezzo

Remote Execution Environment

- Simple Shell Scripts
  - Write scripts that divide data
  - Execute multiple instance of a particular application in different processing elements
  - Used for small applications that accept divided data as command line arguments
  - Common tools (RSH and SSH)

- Process Management Applications
  - Parallelization is similar to shelling script above
  - Provides higher level job and execution control
  - Only independent process can be migrated (no data dependency) – Limited uses
  - Common tools (PBS, Sun Grid Engine and others)

Remote Execution Environment

- Transparent Process Migration
  - Uses kernel level process management to migrate processes to different processing elements
  - Only independent process can be migrated (no data dependency) – Limited uses
  - Used for compile and execution farms
  - Common Tools (Mosix and BPROC)

- Parallel Computing Libraries
  - Programmers develop using these libraries to control the movement of data between processing elements
  - Provides the greatest flexibility and control
  - Used for nearly all types of parallel computing applications (ie. scientific computing)
  - Common Tools (ScALAPACK, MPI and PVM)

Remote Execution Environment

- Parallelizing Compilers
  - Use a regular programming language like C or Fortan
  - Programmer provides syntax hints to inform the compiler that sections are parallelization
  - In most cases, it is not as flexible as parallel computing libraries
  - Most common notation is *OpenMP*
  - These are commonly used for nearly all types of parallel computing applications

Parallel Computers

Types of Parallel Computers

- Massively Parallel Computing Systems
  - Has processing elements directly connected to computing buses
  - Uses proprietary technology and vendor specific
  - Computing resources dedicated to parallel computing
- Volunteer Computing Systems
  - Use of commodity hardware and software
  - Computing resources are not dedicated
- Beowulf Computing Systems
  - Also uses commodity hardware and software
  - Uses technologies that are usually free and open source
  - Computing resources are dedicated to parallel computing

Beowulf Defined

- Commodity parallel computing system
- Best of both MPP and Volunteer computing systems
- Better coupling than Volunteer computing system
- More cost-effective and expandable than MPP systems

Ateneo de Manila University's Parallel Computing Initiatives

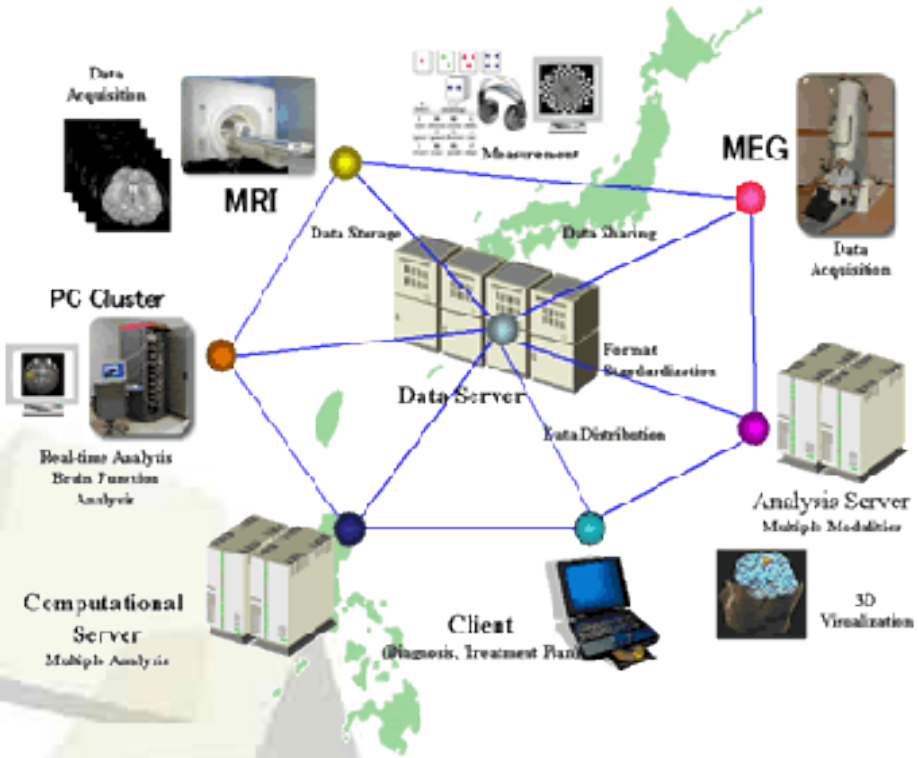
AGILA HPCS

- AGILA High Performance Computing System
- Ateneo's first production Beowulf cluster
- Completed and deployed on August 2000

- Specifications:
  - Eight (8) compute nodes composed mainly of AMD Athlon processor powered machines
  - Connected via 100 Mbps Fast Ethernet Network

- Primary Uses:
  - Climates Studies c/o Manila Observatory
  - Chemical Modelling c/o Chemistry Department
  - Parallel Computing Education c/o Mathematics Department

Medgrid Cluster



- Ateneo Medical Grid Computing Cluster
- Cluster is composed of eight (8) dual-Xeon processor nodes connected via Myrinet
- Completed and deployed on Feb 2004
- Primarily used to perform medical information technology and grid computing research

Parallel Computing Applications

Parallel Programming

- Writing applications that run on multiple processing elements
- Takes advantage of problem characteristics such as granularity and data boundedness

- Multiple ways to develop parallel programs:
  - Simple Scripting
  - Process Management Applications
  - Transparent Process Migration
  - Parallel Computing Libraries
  - Parallelizing Compilers

Parallel Computing Libraries

- Special system-level libraries that provide interface that aid in writing parallel applications
- The more application specific the easier to develop but the less the control

- Main classifications of parallel computing libraries:
  - Scientific Computing or Application Specific Libraries (PETSc)
  - Parallel Mathematical Operations Libraries (ScALAPACK, Parallel BLAS)
  - Message Passing Libraries (PVM and MPI)

Message Passing Libraries

- Most fundamental way for developing parallel applications
- Provides the greatest flexibility and the most amount of work

- Parallel Virtual Machine (PVM)
  - Former de facto standard for message passing libraries
  - Developed by Frank Dongarra and company at the University of Tennessee, Knoxville
- Message Passing Interface (MPI)
  - Developed at the industry standard for message passing libraries (MPI Forum)
  - Current de jure standard for message passing

Message Passing Libraries

- Provides a number of function calls that can be used for developing applications
- Allows programmers to use a regular programming language such as C and Fortran
- In some cases, Java is available

- Functions normally fall under one of the following categories:
  - Initiation, Identification and Termination
  - Point-to-Point Communications (SEND and RECV)
  - Collective Communications (BCAST and REDUCE)
  - Persistent Communications

References

- Beowulf HOWTO <http://www.beowulf.org>
- Logos and Pictures from their respective corporate websites:
  - Intel <http://www.intel.com>
  - AMD <http://www.amd.com>
  - Rambus <http://www.rambus.com>
  - Myrinet <http://www.myricom.net>
  - CISCO <http://www.cisco.com/>