Cluster Computing: The Commodity Supercomputing

By Mark Baker and Rajkumar Buyya
Software - Practice and Experience
vol. 1, no. 1, Jan 1988

Clusters
- Commodity supercomputers
- Built using commodity HW and SW components
- Playing a major role in redefining the concept of supercomputing

Important factor making the use of workstations practical:
- Standardization of tools and utilities
  - MPI and HPF
  - Allows applications to be developed and tested in a cluster and ported to a parallel platform when ready

Why are clusters preferred over MPPs?
- Workstations are becoming powerful
- Bandwidth between workstations is increasing
- Workstation clusters are easier to integrate into existing networks
- Typical lower user utilization
- Development tools are more mature
- Workstation clusters are cheap and readily available
- Clusters can be enlarged and individual nodes can have their capabilities extended

Basic definition
- A cluster is a collection of workstations or PCs that are interconnected via some network technology

Likely scenario
- Computers will be state-of-the-art
- Network will be high-bandwidth and low-latency
- Such a cluster can provide fast and reliable services to computationally intensive applications
Clusters

Topics to be discussed
- Components
- Tools
- Techniques
- Methodologies

Clusters can be classified as
- Dedicated cluster
- Non-dedicated clusters

Dedicated clusters
- A particular individual does not own a workstation
- Resources are shared
- Parallel computing can be performed across the entire cluster

Non-Dedicated clusters
- Individuals own workstations
- Applications are executed by stealing cycles
- Tension between owner and remote user
- Important issues: migration and load balance

Clusters

Suitable for applications that are not communication intensive
- Typically low bandwidth and high latency

Workstations
- Some sort of Unix platform
- Lately, PCs running Linux or NT

Commodity Components

- Processors and Memory
- Disk I/O
- Cluster Interconnects
- Operating Systems
- Windows of Opportunity
Processors

Today, single-chip CPUs are almost as powerful as processors used in supercomputers of the recent past

Related projects

- Digital Alpha 21364 processor
  - Integrate processing, memory controller, and network interface into a single chip

- The Berkeley Intelligent Ram project
  - Exploring the entire spectrum of issues involved in a general-purpose computer system that integrates a processor and DRAM onto a single chip

Processors used in Clusters

- Digital Alpha – Alpha Farm
- IBM Power PC – IBM SP
- Sun SPARC – Berkeley NOW
- SGI MIPS
- HP PA

Memory

- Amount of memory required varies with the target application
- Parallel programs distribute data throughout the nodes
- There should be enough memory to avoid constant swapping
- Caches are key: 8KB to 2MB

The System Bus

- Needs to match the system's clock speed
- Intel PCI Bus: 133MBps
  - Used in Pentium based PCs
  - Digital AlphaServers
- Decreasing the distinction between PCs and workstations
Commodity Components
- Processors and Memory
- Disk I/O
- Cluster Interconnects
- Operating Systems
- Windows of Opportunity

Disk I/O
- Disk density is increasing 60-80% every year
- Disk access time have not kept pace with microprocessor performance
- Parallel/Grand Challenge applications need to process large amount of data
- Necessary to improve I/O performance

Disk I/O
- Way of improving
  - Carry out I/O operations concurrently with the support of a parallel file system
  - Can be constructed by using the disks associated with each workstation in the cluster

Commodity Components
- Processors and Memory
- Disk I/O
- Cluster Interconnects
- Operating Systems
- Windows of Opportunity

Cluster Interconnects
- Individual nodes in a cluster are usually connected with a high-speed low-latency high-bandwidth network
- Communication uses
  - Standard Network protocol: TCP/IP
  - Low level protocol: Active Messages or Fast Messages

Cluster Interconnects
- Requirements to balance the computational power of the workstations available
  - Bandwidth: more than 10MBps
  - Latency: at most 100us
Cluster Interconnects

- **Network Technologies**
  - Fast Ethernet
  - ATM
  - Myrinet

Cluster Interconnects

- **Ethernet**
  - Cheap and widely used to form clusters
- **Standard Ethernet**
  - 10Mbps - not enough
  - **Fast Ethernet**
    - 100Mbps - meets the requirement

Cluster Interconnects

- **ATM (Asynchronous Transfer Mode)**
  - Switched virtual circuit technology
  - Developed for telecommunication
  - Intended to be used for LAN and WAN
    - Presents an unified approach to both
  - Based around small-fixed size packets
  - Designed for a number of media
    - Example: copper wire and optic fiber
    - Performance varies with the hardware

Cluster Interconnects

- **ATM (Asynchronous Transfer Mode)**
  - Usually, no optical fiber in desktops
  - ATM on CAT-5
    - 15.5 Mbps
    - Allows upgrades of existing networks without replacing cabling

Cluster Interconnects

- **SCI – Scalable Coherent Interface**
  - IEEE 1596 Standard
  - Low-latency high-bandwidth distributed shared memory access across a network
  - Provides a scalable architecture that allows large systems to be built out of inexpensive mass produced components

Cluster Interconnects

- **SCI – Scalable Coherent Interface**
  - Point-to-point architecture
  - Directory-based cache coherent
  - Faster than any network technology available
  - Scalability depends on switches
  - Expensive
  - Produced for SPARC Sbus and PCI Based systems
### Cluster Interconnects
- **Myrinet**
  - 1.28 Gbps full duplex LAN supplied by Myricom
  - Based on cut-through switches
  - Proprietary and high-performance
    - Low latency and high bandwidth
  - Used in expensive clusters

### Commodity Components
- **Processors and Memory**
- **Disk I/O**
- **Cluster Interconnects**
- **Operating Systems**
- **Windows of Opportunity**

### Operating Systems
- **Modern operating systems**
  - Multitask
  - Multithreading at kernel level
  - User-level high-performance multithreading without kernel intervention
  - Network support
  - Most popular
  - Solaris, Linux, and Windows NT

### Solaris
- From Sun
- Unix-based multithreaded and multi-user system
- Supports Intel x86 and SPARC platforms
- Network support includes TCP/IP stack, RPC and NFS

### Linux
- Unix-like operating system
- Developed by Linus Torvalds, a Finnish undergraduate student in 91-92
- Open source and free
  - Lots of contribution from other programmers
  - Wide range of SW tools, libraries and utilities
  - Robust, reliable, POSIX compliant
Operating Systems

- Linux
  - Pre-emptive multi-tasking
  - Demand-paged virtual memory
  - Multi-user support
  - Multi-processor support

Linux: Why is it so popular?
- FREE! Available from the Internet and can be downloaded without cost
- Runs on cheap x86 platforms, yet offers the power and flexibility of Unix
- Easy to fix bugs and improve system performance
- Users can develop or fine-tune HW drivers and these can be made easily available to other users
- Applications and system software are freely available (for example: GNU software)

- NT
  - Microsoft Corporation is the dominant provider of SW in the personal computing market place
  - IN 1996, NT and Windows 95 had together 66% of the desktop OS market share

Windows of Opportunity

The resources available in the average NOW offer a number of research opportunities:
- Parallel processing
- Network RAM for virtual memory
- Software RAID
- Multi-path communication

Commodity Components

- Processors and Memory
- Disk I/O
- Cluster Interconnects
- Operating Systems
- Windows of Opportunity
Programming Tools

- For HPC on Clusters
  - Message Passing Systems
    - PVM, MPI
  - Distributed Shared Memory Systems
  - Parallel Debuggers and Profilers
  - Performance Analysis Tools
  - Cluster Monitoring

Message Passing Systems

- Message Passing Libraries allow efficient parallel programs to be written for distributed memory systems
- Provide routines to initiate and configure the messaging environment
- Provide functions for sending and receiving data

Message Passing Systems

- Two most popular high-level message-passing systems for scientific and engineering applications
  - PVM
  - MPI defined by the MPI forum

Message Passing Systems

- MPI
  - Specification for message-passing
  - Designed to be standard for distributed memory parallel computing using explicit message passing
  - Attempts to establish a practical, portable, efficient, and flexible standard for message passing
  - Available on most of the HPC systems including SMP machines

Message Passing Systems

- MPICH
  - Most popular of the current free implementations of MPI
  - Developed at Argonne National Laboratory and Mississippi State
Message Passing Systems

- MPICH
  - Portable, built on top of a restricted number of HW-independent low-level functions, which form the ADI
  - ADI, Abstract Device Interface contains 25 functions
  - The rest of MPI contains 125 functions and is implemented on top of ADI

Programming Tools

- For HPC on Clusters
  - Message Passing Systems
    - PVM, MPI
  - Distributed Shared Memory Systems
  - Parallel Debuggers and Profilers
  - Performance Analysis Tools
  - Cluster Monitoring

Distributed Shared Memory

- SW Solution
  - Built as a separate layer on top of the message-passing interface
  - Virtual memory pages, objects, and language types are units of sharing
  - Implementation achieved by
    - Compiler
    - User-level runtime package
  - Examples
    - Munin, TreadMarks, Linda, Clouds

Distributed Shared Memory

- Implemented using HW or SW solutions

Message Passing Systems

- MPICH
  - ADI, basic point-to-point message passing
  - Remaining MPICH, management of communicators, derived data types, collective operations
  - Has been ported to most computing platforms, including NT
## Distributed Shared Memory

- **HW Solution**
  - Better performance
  - No burden on user and SW layers
    - Transparency
  - Finer granularity of sharing
    - Extensions of the cache coherence schemes
  - Increased HW complexity

## Programming Tools

- **For HPC on Clusters**
  - Message Passing Systems
    - PVM, MPI
  - Distributed Shared Memory Systems
  - Parallel Debuggers and Profilers
  - Performance Analysis Tools
  - Cluster Monitoring

## Parallel Debuggers and Profilers

- **Debuggers**
  - A small number of debuggers can be used in a cross-platform heterogeneous environment
  - 1996: Forum to establish a standard
    - Define functionality, semantics, syntax
    - For a command-line parallel debugger

## Parallel Debuggers and Profilers

- **Highly desirable to have some form of easy-to-use parallel debuggers and profiling tools**
- **Most vendors of HPC systems provide some form of debugger and performance analyzer for their platforms**
- **Ideally, these tools should be able to work in a heterogeneous environment**
  - Develop on NOWs, run on dedicated HPC system

## Distributed Shared Memory

- **HW Solution**
  - Typical classes of HW DSM systems
    - CC-NUMA (DASH)
      - Cache-Coherent Non-Uniform Memory Access
    - COMA (KSR1)
      - Cache-Only Memory Architecture
    - Reflective memory systems (Merlin)
      - Automatically transmits the data to all connected computers local memory, transparently, with zero overhead and at extremely high speeds
Parallel Debuggers and Profilers

A parallel debugger should
- Set both source-level and machine-level breakpoints
- Sharing breakpoints between groups of processes
- Define watch and evaluation points
- Display arrays and array slices
- Manipulate code variables and constants

Total View
- Commercial product from Dolphin Interconnect Solutions
- Currently, the only widely available parallel debugger for multiple HPC platforms
- However, only for homogeneous environments

Programming Tools

For HPC on Clusters
- Message Passing Systems
  - PVM, MPI
- Distributed Shared Memory Systems
- Parallel Debuggers and Profilers
- Performance Analysis Tools
- Cluster Monitoring

Performance Analysis Tools

Used to help programmers understand the performance characteristics of a particular application
- Analyze and locate parts of an application that exhibit poor performance and create bottlenecks
- Useful for sequential applications
- Enormously helpful for parallel applications

Performance Analysis Tools

Create performance information
- Produce performance data during execution
- Provide a post-mortem analysis and display of the performance information
- A few tools can do runtime analysis
  - Either in addition to or instead of the most-mortem analysis

Components
- A means of inserting instrumentation calls to the performance monitoring routines into the users' application
- A runtime performance library
  - Set of monitoring routines that measure and record various aspects of a program's performance
- A set of tools that process and display the performance data
Performance Analysis Tools
A post-mortem performance analysis tool works by
- Adding instrumentation calls into the source code
- Compiling and linking the application with a performance analysis runtime library
- Running the application to generate a trace-file
- Processing and viewing the trace-file

Issues
- Intrusiveness of the tracing calls and their impact on the application performance
- Format of the trace-file
  - Must contain detailed and useful information but cannot be too large
  - Should conform to some standard format to enable the usage of different GUI interface to visualize the performance data

Examples
- AIMS (NASA)
- MPE (ANL)
- Pablo/SvPablo (UIUC)
- Paradyn (Wisconsin)
- VT (IBM)
- Dimemas (Pallas)

Programming Tools
For HPC on Clusters
- Message Passing Systems
  - PVM, MPI
- Distributed Shared Memory Systems
- Parallel Debuggers and Profilers
- Performance Analysis Tools
- Cluster Monitoring

Cluster Monitoring
System administration tools
- Allow clusters to be observed at different levels using a GUI
- Good management software is crucial for exploiting as a HPC platform

Cluster monitoring
- The Berkeley NOW
  - System administration tool gathers and scatters data in a database
  - Uses a Java applet to allow users to monitor a system from their browser
Cluster Monitoring

- Solstice SyMon from Sun Microsystems
  - Allows standalone workstations to be monitored
  - Uses client/server technology for monitoring
  - Node Status Reporter provides a standard mechanism for measurement and access to status information of clusters
  - Parallel applications/tools can access NSR through the NSR interface

Cluster Monitoring

- PARMON
  - Comprehensive environment for monitoring large clusters
  - Client/server technology to provide transparent access to all nodes to be monitored
    - Server - provide system resource activities and utilization information
    - Client - A Java applet capable of gathering and visualizing real-time cluster information

Representative Systems

- NOW (Network of Workstations)
- HPVM (The High Performance Virtual Machine)
- The Beowulf Project
- Solaris MC: A High Performance Operating System for Clusters

NOW

- Goal: combining distributed workstations into a single system
- Research and development
  - Network interface hardware
  - Fast communication protocols
  - Distributed file systems
  - Distributed Scheduling
  - Job Control

NOW

- Inter-processor communication
  - Basic communication primitives in the NOW
  - AM communication is essentially a simplified remote procedure that can be implemented efficiently on a wide range of hardware
  - Generalized to support a broad spectrum of applications
    - Client/server, file systems, operating systems, parallel programs

NOW

- Inter-processor communication
  - Includes a collection of low-latency, parallel communication primitives
    - Berkeley sockets
    - Fast sockets
    - Shared address space parallel C (Split-C)
    - MPI
    - HPF
**Process Management**
- **GLUNIX (Global Layer Unix)**
  - Operating system layer designed to provide
    - Transparent remote execution
    - Support for interactive parallel and sequential jobs
    - Load balancing
    - Backward compatibility for existing application binaries
  - Multi-user system implemented at the user level

**Virtual Memory**
- Utilize memory in idle machines as a paging device for busy machines
- The system is serverless
  - Any machine can be a server when it is idle or a client when it needs more memory that what is available
- Two prototypes
  - One use Solaris segment drivers to implement an external user-level pager which exchanges pages with remote page daemons
  - The other provides similar operations on similarly mapped regions using signals

**File System**
- xFS is a serverless, distributed file system
- Attempts to have low-latency, high-bandwidth access to data
- Distributes the functionality of the server among the clients
- Typical duties of the server
  - Maintaining cache coherence
  - Locating data
  - Servicing disk requests
- In xSF
  - Each client is responsible for servicing requests on a subset of files
  - File data is striped across multiple clients

**High Performance Virtual Machine (HPVM)**
- High Performance Virtual Machine
- Goals
  - Deliver supercomputer performance on low-cost COTS systems
  - Hide the complexities of distributed system behind a clean interface
- The HPVM architecture consists of
  - Number of SW components with high-level APIs such as MPI, SHMEM, Global Arrays

**Illinois Fast Messages (FM)**
- Originally developed on a Cray T3D and a cluster of SPARCstations connected by Myrinet
- Has a low-level software interface that delivers hardware communication performance
- Has a higher-level layer interface for greater functionality, application portability, and ease of use
**HPVM**
- Illinois Fast Messages (FM)
  - Based on the Berkeley AM
  - FM is not the surface API, but the underlying semantics
  - Contains functions for sending long and short messages and for extracting messages from the network
  - Guarantees reliable and ordered packet delivery and control over the communication scheduling

**The Beowulf Project**
- Initiated in the Summer of 1994
- Sponsored by NANA
- Goal
  - Investigate the potential of PC clusters for performing computational tasks
  - Beowulf refers to a Pile-of-PC (PoPC) as a loose ensemble of PCs which is similar to clusters or networks of workstations

**The Beowulf Project**
- Emphasis of PoPC on the
  - use of mass-market commodity components
  - dedicated processors
  - Usage of a private communication network
- Overall Goal
  - Achieve the 'best' overall system cost/performance ratio for the cluster

**The Beowulf Project**
- Adds to the PoPC model
  - No custom components
    - Accepted standard interfaces: PCI bus, IDE and SCSI interfaces, Ethernet
  - Incremental growth and technology tracking
  - Usage of readily available and free SW components
    - Linux

**The Beowulf Project**
- Grendel SW Architecture
  - Collection of SW tools being developed and evolving within the Beowulf project
  - Tools for resource management and to support distributed applications
  - Beowulf distribution includes several programming environments and development libraries
    - PVM, MPI, BSP, IPC, and pthreads

**The Beowulf Project**
- Grendel SW Architecture
  - Key to success
    - Inter-process communication bandwidth and system support for parallel I/O
    - TCP/IP over Ethernet
  - Uses multiple Ethernet networks in parallel to satisfy the bandwidth requirement
    - Transparent to the user
    - Implemented as an enhancer to the Linux kernel
  - Has been shown that up to 3 networks can be ganged together leading to significant throughput
The Beowulf Project

Grendel SW Architecture
- Each node runs its own copy of the Linux kernel
  - Nodes may participate in a number of global spaces (for example, process id)
  - Need a mechanism that allow unmodified versions of standard UNIX processes utilities (for example, ps) to work across cluster

Programming Model
- Supports several distributed programming paradigms
  - Message passing
    - MPI, PVM, and BSP
  - Distributed shared memory

Summary and Conclusions

Hardware and Software Trends
- Cluster Technologies Trends
- Predictions about the Future
- Final Thoughts

HW and SW Trends

Important advances that contributed to the existence of clusters
- Fast Ethernet
- Switched network circuits
- Workstations' performance
- Microprocessor performance leading to HP PCs
- Powerful and stable Unix system for PCs

Trends
- Processor speeds will keep going up
- Memory sizes will keep going up faster than memory speeds
  - To compensate organize DRAM in banks and transfer in parallel to/from the banks
  - Also, use memory hierarchy
- Disks are also becoming larger faster than they are becoming faster

Network performance is increasing
Network costs are decreasing
HP technologies (ATM, SCI, Myrinet) are promising
Linux is the main used system
NT is catching up
Cluster Technology Trends

- Network is key
- Ethernet technologies are more likely to be the main stream
  - 1 Gigabit Ethernet
  - 10 Gigabit Ethernet

Predictions about the Future

- The gap will continue to close
- Stealing-cycles systems will continue to use whatever resource they find available
- Dedicated clusters used for HPC will continue to evolve as new and more powerful technologies become available

Predictions about the Future

- More than one processor per node will become fairly common
- To reduce latency, cluster SW will bypass the OS kernel
- NIC will be more intelligent
- OS will provide a rich set of tools and utilities and will also be robust and reliable

Final Thoughts

- The need for computational power exceeds our abilities to fulfill this need
- Clusters are the most promising way by which this gap can be reduced
- COTS-based clusters has a number of advantages
  - price/performance
  - growth
  - provision of a multi-purpose system