Run-Time Management for Heterogeneous Shared Clusters

A. Paulo Santos

Departamento de Informática, Universidade do Minho
4710 – 057 Braga, Portugal
cei5134@mestrado.di.uminho.pt

Abstract. The requirements for High Performance Computing (HPC) and/or High Throughput Computing (HTC) are increasing worldwide. To achieve high performance standards in a shared cluster (SC), based on COTS (commodity off the shelf) platforms, the following requirements were identified: (i) heterogeneity, which will provide capabilities to operate with different operating systems, computing nodes, storage technologies and communication topologies and standards; (ii) on-line management, which will provide run-time analysis and administration of CPU load, memory usage, disk space and communication bandwidth, for an efficient decision on job allocation for the right computing node. This presentation discusses some of the needed features, for the heterogeneous and run-time management, with a review on available non-proprietary cluster computing software.

1 Introduction

By far, the most important and needed features for a SC, made whit COTS PC´s is to be Heterogeneous (Section 2), because if so, we can make very significant savings in SO and architecture nodes. Another important feature for the effective efficient of a SC is to be possible to made a Run-Time workload management, because the meaning of clustering, is making several PC´s work together, and for this goes well, we need to manage them in run-time (Section 3 and 4). Some important resources monitoring will be focused to accomplish this.

2 Heterogeneous Shared Cluster – HSC

In the simplest way, a cluster is a group of cheap Commodity Off The Shelf (COTS) computers and/or workstations - nodes, tied together with a high-speed network or switch, in order to achieve processing power – High Performance Computing (HPC) and/or High Throughput Computing (HTC), of a magnitude, impossible to otherwise obtain without the use of purpose-built supercomputers

In a Shared Cluster we have two layers of machines. Firstly are the workstations nodes, terminals that can be used to log on and execute programs from. Then there are the dictated nodes whose only role are to be remote assistants to the workstations machines in running parallel and sequential applications. The goal to implement a Shared Cluster is to offers to users an image of a single node [2], which will provide a High Performance Computing (Fig. 1.).

There is two types of SC, one gives time-sharing, where jobs must compete with others to have the best node, the other gives space-sharing in witch every job have all the SC a specific time to accomplish. We will consider only the time-sharing approach in SC.

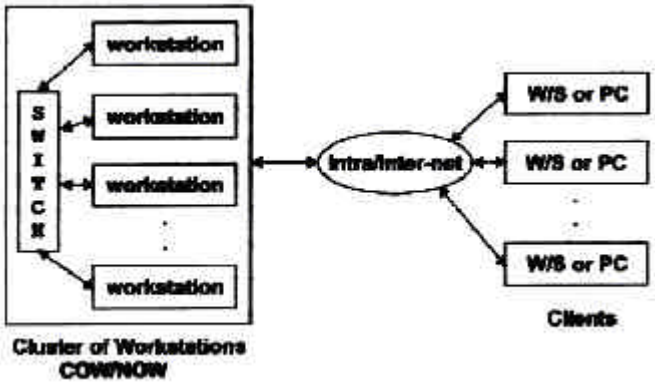

Fig. 1. Shared Cluster

An HSC must provide capabilities to operate with different operating systems, computing nodes [3], storage technologies and communication topologies [4].

4.1 Operating Systems

The more important feature for heterogeneity is the one who provide different operating systems. For this there are many possible choices of operating system for a PC cluster. NT and Linux comprise the majority of operating systems installed on PC clusters, although Linux is usually the operating system of choice for PC clusters because of its reliability, scalability and Open Source. Software tools are widely available for both NT and Linux clusters. Both have several usable implementations of the two most commonly used message passing interfaces: MPI and PVM. Actually there is no non-proprietary management software tool to provide different OS in Clusters, this was the main difficulty for achieve some acknowledgments about this mater.
The management tool must consider that applications designed for Linux and Windows NT behave differently on each platform, and because so, we must have a Hardware Description Language – HDL, to run on same way, on each one. Still, there are significant differences between Windows NT and Linux, the most notable of which is Windows NT's limited multi-user capabilities, While Linux inherent multi-user functionality provides exceptional file sharing, Windows NT is relatively constrained, leaving users with the problem of facilitating file sharing in mixed environments. Another way is to implement a client-server architecture, where the clients platforms machines supports Java runtime system, Java Virtual Machine – JVM, which will provide a common interface between the clients and server.

4.2 Computing Nodes - PC´s and Workstations

Probably the most commonly used node architecture in cluster construction is the Intel or Intel-clone PC processor, due to the price factor. PC machines are significantly cheaper on a one-to-one basis than Unix workstations, although workstations still have a slight edge in sheer speed. Since one of the main goals of clustering systems in the first place is to reduce the cost of HPC systems, the lower unit price makes PCs very attractive. Again, a way of leading with the heterogeneity of nodes computing, is to implement a client-server architecture, where the clients platforms machines supports Java runtime system, Java Virtual Machine – JVM, which will provide a common interface between the clients and server.

4.3 Communications Topology

Vital to the cluster systems in operation today (and most likely those in the future) is the

existence of a network connecting the nodes and allowing them to communicate information with each other - or a master server controlling the cluster. Current Popular interconnection network topologies include Tree, Crossbar, Hypercube and Mesh Topologies, which are represented diagrammatically in Fig 2.

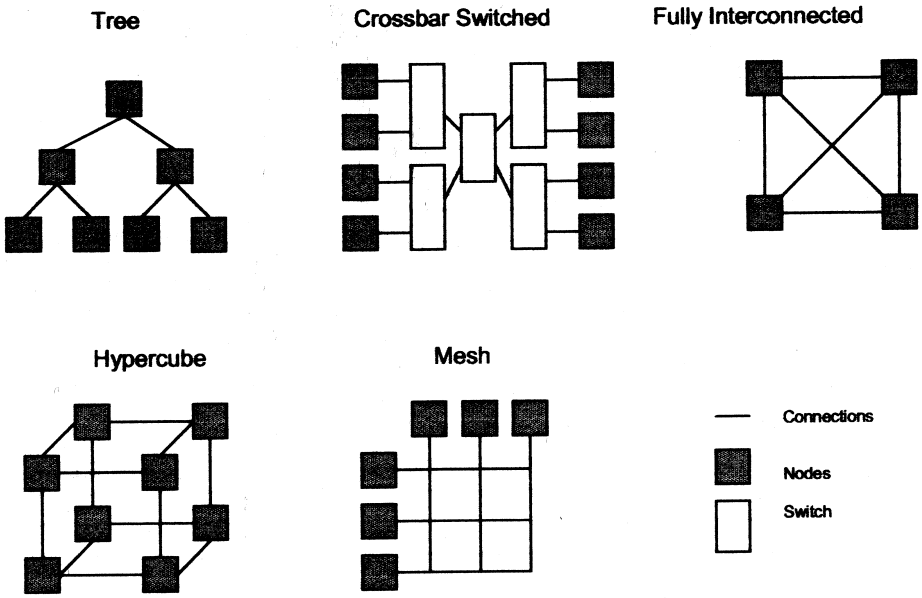

Fig. 2. Network Topologies

While once 1OMbps Ethernet, was the most popular interconnection protocol, new 100Mbps and 1000Mbps Ethernet systems are now being used in order to provide extra speed without much extra overhead required to change network protocols.

3 Management for HSC

The operating system is responsible for managing host resources, so system software must assume many of the traditional operating-system responsibilities across the network. That means it must support a variety of heterogeneous hardware and software platforms and various types of system components such as servers, networks, and PCs. System software must act as the glue [5] that fuses network and host resources to present a single-system environment to end-users (Fig. 3).

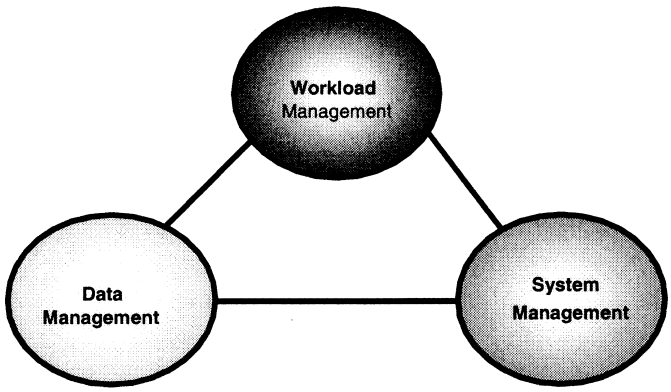

Figure 3. System Software "Triangle".

4.4 Data Management

Data is the lifeblood of system software. Because of the multitude of applications, data management must be separated from the applications and made transparently accessible to them across the network.

4.5 System Management

System management ensures that all components of the system are operating effectively to support the users. System management in a heterogeneous computing environment encompasses a variety of functionality - such as network management, user administration, data backup and storage, system and network security, software delivery, database tools, event management, help desks, and Internet and intranet administration. For maximum effectiveness, these tools are often integrated into a uniform framework.

4.6 Workload Management – Functions and Requirements

Supporting the applications that are critical, workload management is concerned with the effective management of all types of processing tasks and transactions, as well as with the maximum utilization of hardware and software resources. In functional terms, workload management software schedules, analyses, and monitors the processing of the application workload. Workload management assumes many of the traditional operating system responsibilities at the network level. It dynamically schedules user activities to fully harness shared heterogeneous computing resources. Whereas data management provides a *shared data platform* for all applications, workload management provides a *shared computing platform* for all applications. Whereas system management focuses on hardware and software resources and is used by the administrators, workload management supports the computing workload, and it is used, directly or indirectly, by the end users who are entrusted with running their works. In this sense, workload management picks up where system management leaves off. System management ensures that HSC resources function properly, whereas workload management ensures that those computing resources are effectively matched with the requirements of the works/tasks. Traditionally, workload management was not considered distinct of the system software. With the strong shift toward HSC environments, however, workload management has become increasingly important. What makes workload management so important is its ability to harness the full power of heterogeneous hardware and software in increasingly complex SC environments. To deliver the complete benefits of a virtual mainframe in a SC environment, it is essential to integrate data management, system management, and workload management.

Workload Management – Functions. As mentioned previously, workload management is the effective management of the computing workload to deliver compelling HPC with significant cost savings. From a functional perspective, it involves the scheduling, analyzing, monitoring, and administration of distributed application workload. Let's look at each of these three functional areas, as shown in Figure 4.

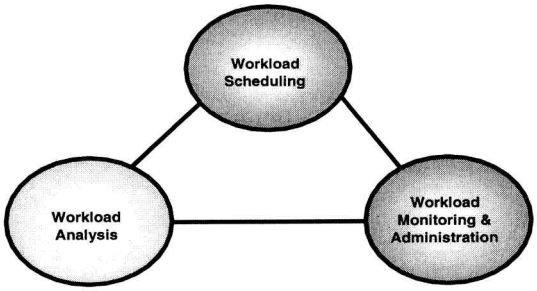

Figure 4. Three Primary Functions of Workload Management.

Workload Scheduling

The most important and fundamental function of workload management is the dynamic scheduling of jobs using the best available computing resources. Depending upon job requirements, scheduling can be based on one or more of the following criteria: resource availability, priority and policy, calendar, and workflow and events.

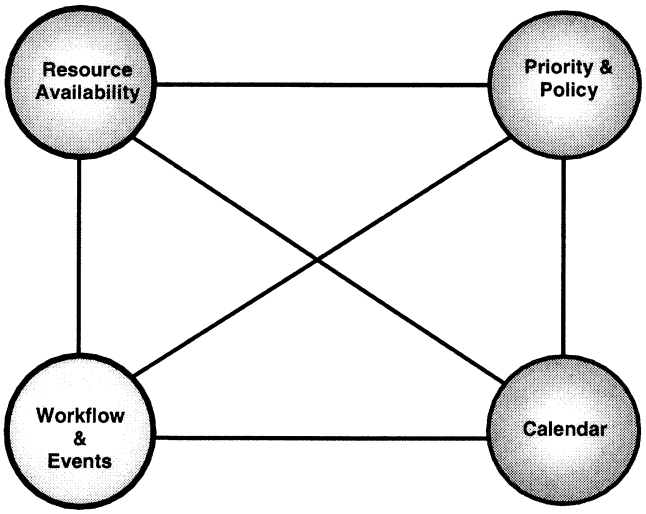

Figure 5. Workload Scheduling.

Workload Analysis

Workload analysis supports comprehensive investigation of the workload data to assess overall system performance. Workload analysis uses data pertaining to 1) the loading and availability of the computing resources, 2) the jobs processed and their resource usage, and 3) the system configuration to provide insights into workload characteristics. Workload analysis can be used for capacity planning, system bottleneck removal, system performance tuning, system upgrade planning, and future workload performance and requirement forecasting. As part of workload analysis, charge-back accounting calculates appropriate charges for resource usage. Workload analysis is a powerful means of implementing service-level agreements (SLAs), so that IT and the users it supports can establish a consensus on how technology supports users processes.

Workload Monitoring and Administration

As workload flows through the HSC, and as the installed base of available resources evolves and expands, dynamic monitoring capabilities are needed to track workload processing. That way, adjustments and corrective actions can be taken immediately, before a potential problem causes a major disruption. Similarly, workload management policies must be configurable and flexible, and the use of system resources through a set of administrative tools. Often, such workload monitoring and administration tools can be integrated into system management.

With the rise of Web-based user interfaces, system managers and administrators will be able to remotely monitor and administer their entire distributed workload from virtually anywhere within the LAN. Taken collectively, the above three core functions of workload management provide applications with the availability, performance, reliability, and scalability.

Workload Management – Requirements. To accommodate the complexity of HSC sys-

tems and the heft of application demands, effective workload management must meet the following requirements:

Dynamic scheduling

Dynamic scheduling exploits the full potential of all resources, scheduling jobs as quickly as possible to deliver optimal performance. As such, it reacts to incoming higher-priority jobs by suspending or migrating other jobs to free up resources

Scalability

Workload management should be equally applicable to small, medium, and large HSC environments, and must be able to deliver value on a single server, a cluster of servers, across divisions and departments, and throughout the LAN.

Support all architectures and operating systems

Different types of systems are suitable for different types of workload. Workload management is the glue that transparently integrates heterogeneous systems and fully harnesses their respective strengths.

Robust and fault tolerant

Since a major failure in the execution of application workload can paralyze the SC, workload management must be available at all times. Fault tolerance should ensure that partial failures are addressed resiliently, with no meaningful impact on critical processes and or application.

Open and standards-based

Must be integrated with other system software and all types of applications, a general-purpose framework with well-defined APIs is needed to assured that all products would work together effectively.

General purpose

Must provide interactive transactions, batch processing, parallel processing, real-time process control, and routine production jobs

Other Requirements

Automated, Transparent, Flexible and Policy-based

4 Run-Time Management for HSC

The management of a HSC will be focus on workload management capabilities. For this, an efficient analysis, administration and real-time monitoring tools, are required, which are the most important features to get an effective and efficient dynamic workload scheduling [6].

4.1 Requirements for an Effective RTMT

The Run-Time Monitoring Tool (RTMT) must allow the user to monitor system activities and resource utilization of various components of Pc´s/Workstation clusters. It also must permit, to monitors the machine at various levels: component, node and the entire system level exhibiting a single system image (SSI). RTMT, must allows the system administrator to monitor several activities.

Aggregation in Visualization of Resource Utilization. The use of aggregation in visualization allows one to scale the visualization to a whole cluster. The administrator can create user-groups containing a set of nodes based on a resource allocation policy and monitor them. The same statistics for different nodes are combined to obtain a single statistic. This technique is called group/machine utilization.

Process Activities. The utilization of a CPU resource can be measured by monitoring

process activities, which helps in identifying CPU/memory-intensive processes. The parameters that can be monitored: pid, command, uname, uid, nice, status, user (%), sys (%), total (%), total CPU (%), and start up time. RTMT must continuously updates the process and system data at a user specified sampling interval. This information must be sorted based on selected parameters such as process-id, user name, etc.

System Logs. Monitoring of the system logs maintained by the operating system. It allows one to process system messages and syslog files for entries that occur at a specific time, or for entries that contain a specific keyword or word pattern.

Kernel Activities. RTMT must supports software instrumentation of system resources (such as CPU, memory, disk and network) and their activities. When a particular resource has more than one instance, it must provide monitoring of each resources instance individually. The invocation of the kernel-data-catalog option allows instrumentation of kernel activities related to resources as CPU, memory, disk and network.

CPU parameters

The monitoring of CPU parameters helps the user to understand how the CPU is being utilized. This will provide the monitoring of numbers of mutecs, interrupts, context switches, system calls, forks, execs, page in, page out, swap-in and swap-out operations performed per second, and to visualize these activities using graphically. It also allows the monitoring of the process run queue, I/O queue and swap queue.

Memory parameters

RTMT must allow continuous instrumentation of memory availability, memory in-use, free memory, percentage of memory in-use, reserved swap space, allocated swap space and available swap space.

Disk parameters

RTMT must allows the monitoring of disk operations such as reads, writes, number of jobs waiting in the queue for disk service, and disk request run-time and wait-time.

Network parameters

The software instrumentation of network parameters such as input packets, output packets, and errors in packet transmission helps to detect network bottlenecks. This instrumentation must be displayed of percentage of incoming and outgoing data packets containing packet format errors.

Device Control. RTMT must supports the control of multiple instances of the same resource, for instance, it must allows CPUs in a SMP node to be set to on-line or off-Iine mode.

Events Generation. It must allow the administrator to define events such as sending e-mail when the user crosses resource utilization limits. This helps the administrator to have effective control over system resource utilization and, therefore, change resource allocation policies.

Diagnostics. RTMT must do validation, testing, and stress testing on external devices such as disk, tape drives and network connectivity.

Data Representation. Is very important and user friendly that RTMT information, uses pie charts, bar charts and line graphs for representing resource usage such as disk and memory utilization as well as graphs for representing various kernel activity parameters of the CPU, disk, network, etc.

4.2 Requirements for an Effective Run-Time Administration Tool

Resources Monitoring. Must allow the user to build system database (nodes and groups)

comprising node name, communication interfaces, and specific area of a disk to be monitored.

Static Information. The listing of users working on selected nodes or entire cluster, system information, configuration, and packages installed on nodes.

Instrumentation. Allows instrumentation of system resources such as CPU, disk, memory and network, and their parameters, both at macro and micro level.

Monitoring Level. Supports monitoring of cluster at node level, group level or entire system level, and thus exhibits a single system image.

Distributed Parallel Execution. Allows the parallel execution of selected operations on a single or group of workstations, real-time and interactive resource monitoring (e.g. processor, memory, disk and network utilization), and normal maintenance tasks such as node or cluster shutdown.

Portability. RTMT clients must be portable across all platforms supporting the Java runtime system, JWM (Java Virtual Machine).

RTMR Client-Server Paradigm. The system model must follows the client-server paradigm with nodes to be monitored acting as servers and the monitoring systems or user-stations acting as clients. The cluster nodes must be monitored from any workstation, PC, or a node of the cluster itself.

Groups of PC´s/Workstations Monitoring. A client can either monitor all the nodes or selectively monitor a few nodes of the cluster. For effective monitoring, the concept of group must be supported. A set of nodes must forms a group and nodes must be selected based on the allocation of resources to various user groups. Such a grouping mechanism helps in monitoring and gathering usability statistics, with which the system administrator can change the resource allocation strategy.

5 Conclusion

For organization and academics community, the way to achieve HPC and/or HTC, to process critical applications, whit the unused power PC´s/Workstations, which they have on their networks, is to implement HSC. For this, workload management is essential to the creation of a virtual mainframe, allowing distributed computing to achieve the robustness and maturity of the legacy mainframe while also delivering the accessibility, cost-effectiveness, and openness of distributed environments.

References

[1] Walbrink, R. and Pallinger, M.: OS Cluster Systems. Monash University, Melbourne, Australia, (2000).
[2] Scyld Computing Corporation: Scyld Beowulf Clustering for HPC, (2001).
[3] Ng, J. and Rogers, G.: Clusters and Node Architecture. Monash University, Melbourne, Australia, (2000).
[4] Griffiths, A. and Matherall, G.: Cluster Interconnections Networks: Monash University, Melbourne, Australia, (2000)
[5] Casemore, B. and Zhou, S.: Workload Management: Platform Computing, Markham, Ontario, Canada, (1998).
[6] Buyya, R.: Portable and Scalable Monitoring System for clusters. SP&E, (2000).