



THE AUSTRALIAN NATIONAL UNIVERSITY

TR-CS-07-01

# **A Survey of how Virtual Machine and Intelligent Runtime Environments can Support Cluster Computing**

**Samuel Chang, Peter Strazdins**

February 2007

Joint Computer Science Technical Report Series

Department of Computer Science  
Faculty of Engineering and Information Technology

Computer Sciences Laboratory  
Research School of Information Sciences and Engineering

This technical report series is published jointly by the Department of Computer Science, Faculty of Engineering and Information Technology, and the Computer Sciences Laboratory, Research School of Information Sciences and Engineering, The Australian National University.

Please direct correspondence regarding this series to:

Technical Reports  
Department of Computer Science  
Faculty of Engineering and Information Technology  
The Australian National University  
Canberra ACT 0200  
Australia

or send email to:

`Technical-DOT-Reports-AT-cs-DOT-anu.edu.au`

A list of technical reports, including some abstracts and copies of some full reports may be found at:

<http://cs.anu.edu.au/techreports/>

**Recent reports in this series:**

- TR-CS-06-02 Peter Christen. *A comparison of personal name matching: Techniques and practical issues*. September 2006.
- TR-CS-06-01 Stephen M Blackburn, Robin Garner, Chris Hoffmann, Asjad M Khan, Kathryn S McKinley, Rotem Bentzur, Amer Diwan, Daniel Feinberg, Daniel Frampton, Samuel Z Guyer, Martin Hirzel, Antony Hosking, Maria Jump, Han Lee, J Eliot B Moss, Aashish Phansalkar, Darko Stefanović, Thomas VanDrunen, Daniel von Dincklage, and Ben Wiedermann. *The DaCapo benchmarks: Java benchmarking development and analysis (Extended Version)*. September 2006.
- TR-CS-05-01 Peter Strazdins. *CycleCounter: an efficient and accurate UltraSPARC III CPU simulation module*. May 2005.
- TR-CS-04-04 C. W. Johnson and Ian Barnes. *Redesigning the intermediate course in software design*. November 2004.
- TR-CS-04-03 Alonso Marquez. *Efficient implementation of design patterns in Java programs*. February 2004.
- TR-CS-04-02 Bill Clarke. *SoLemn: Solaris emulation mode for Sparc Sulima*. February 2004.

# A Survey of How Virtual Machine and Intelligent Runtime Environments Can Support Cluster Computing

Samuel Chang  
*Department of Computer Science*  
*Australian National University*  
*sychang11@gmail.com*

Peter Strazdins  
*Department of Computer Science*  
*Australian National University*  
*peter@cs.anu.edu.au*

## Abstract

The use of cluster computers in high performance computing environments has recently been on the rise and at present, it presents a fertile source of unsolved research problems. With the Australian National University's (ANU) High Performance Computing (HPC) research group planning to undertake two projects in the field of cluster computing very shortly, the aim of this survey is to lay the groundwork for both projects by providing comprehensive, up to date information on current research related to virtual machines and intelligent runtime environments for use in the context of a large cluster or grid system. This survey covers a broad range of information sources, namely the proceedings of top conferences, journals, and technical reports published by notable research groups and researchers. We summarize key papers and make listings of research groups which are undertaking projects that we feel are worth taking note of. Subsequently, the information gathered is synthesized to identify unsolved problems and potential areas of future work. Notable findings presented in this survey are that paravirtualization of operating systems is no longer a necessary requirement for use with the Xen hypervisor as processor vendors Intel [10] and AMD [1] are building virtualization support into their processors, migration of entire operating systems now incurs as little as 100ms down time by iteratively pre copying pages [15]. We have also come to the conclusion that an important first step in the successful development of an intelligent runtime environment is to develop a system which can accurately profile the hardware that a cluster system is made up of as well as applications that would be run on it.

## 1. Introduction

Supercomputers provide a significant increase in computing power that allows an otherwise unsolvable computationally intensive problem, such as weather forecasting or the simulation of airplanes in wind tunnels, to be solved in a reasonable amount of time. Cluster<sup>1</sup> computers, which are systems made up of commodity off the shelf (COTS) components, are increasingly becoming the tool of choice when it comes to building a supercomputer.

Currently, more than 70% of the top 500 supercomputers are cluster systems [84]. This trend does not come as a surprise as cluster computers

provide a substantial increase in computing power at a relatively inexpensive cost compared to other supercomputing architectures. Furthermore, by using COTS components to build a cluster computer, should hardware in the system fail, parts are easily and cheaply replaced without the need for expensive custom made and proprietary components.

The purpose of this survey is to lay the groundwork for two projects related to cluster computing that would be undertaken in the very near future by the ANU's HPC research group. These projects are briefly summarized below:

Project number one [83], "Virtualization Support for Heterogeneous Clusters", explores how the Xen virtual machine monitor can be used in a heterogeneous cluster computing environment. This project includes evaluating and reducing the performance overhead of

---

<sup>1</sup> This survey solely concentrates on high performance clusters as opposed to availability or throughput clusters.

Xen, the configuration and selection of an operating system (OS) tuned to the needs of an application over the cluster and migrating an application over the cluster by migrating the virtual machines running on it.

The second project [83], “Next Generation Grid Enabled Cluster Computers: Performance Optimization for e-Science” explores the development of intelligent runtime environments for large cluster and/or grid systems that are capable of selecting where best to execute a user’s application code based on cost/benefit considerations. This project would build on the first by utilizing live migration technologies to allow live migration of running application processes between nodes in the cluster and as a means of tailoring the operating system to particular applications.

This paragraph gives a brief summary of how we conducted this survey. Firstly, we looked through the proceedings of top conferences<sup>2</sup> in architecture, operating systems, networks, cluster computing and grid computing before moving on to look at literature found in the major IEEE and ACM journals related to the above mentioned areas. Next, we browsed through technical reports published by relevant research groups, both in industry and academia. Finally, we went through the publication lists of notable academics so to pick up any potentially interesting papers that may have slipped through the cracks. The reader is directed to Appendix A, which lists all our sources of information.

The paper is organized as follows: Section 2 contains information on virtual machines, with Subsection 2.1 covering virtual machine technology and Subsection 2.2 covering the migration of virtual machines. Next, Section 3 introduces the concept of intelligent runtime environments and the components of which it comprises of. These components, which are scheduling and load balancing and fault tolerance mechanisms, are covered in Subsection 3.1 and 3.2 respectively. The Open Source Cluster Application Resources (OSCAR) suite used in cluster management is covered Subsection 3.3. Section 4 contains unsolved problems and suggestions for potential areas of future work. Some literature which does not fall into any of the above categories that the reader is highly recommended to read can be found in Section 5. Finally, Section 6 concludes the paper. Section 7 serves a twofold purpose

---

<sup>2</sup> The reader is advised that some of the papers mentioned in this survey require membership with the ACM Digital Library and/or IEEE Explore in order to view.

as our bibliography as well as a collection of relevant literature, grouped by topic. Appendix A lists all the sources from which we gathered our information and Appendix B contains a list of research groups and notable projects which we feel are worth keeping track of.

## 2. Virtual Machines

A virtual machine is an environment whereby a guest operating system is presented with a set of virtualized hardware devices and is managed by a virtual machine monitor (VMM), or hypervisor. The hypervisor is commonly run on a host operating system or, in some cases, directly on bare hardware. Multiple guest operating systems can be run simultaneously on a single physical machine, in which case the hypervisor provides multiplexing capabilities of the underlying hardware.

### 2.1 Virtual Machine Technology

Since 2005, virtual machines are again becoming a hot topic in academia and industry. Venture capital firms are competing to fund startup companies touting virtual machine based technologies while major hardware manufacturers Intel, AMD, Sun Microsystems and IBM are developing virtualization strategies that target markets with revenues in billions and growing. In research labs and universities worldwide, researchers are developing approaches based on virtual machines to solve mobility, security and manageability problems [12].

Why this sudden resurgence in virtual machine technology? Some of the many benefits of using virtual machines are: simplicity of service management, application scheduling flexibility, ease of testing and debugging complex systems, hardware multiplexing, enhanced system management capabilities, the ability to run conflicting processes in isolation from each other, heightened system security, and live migration, dynamic content distribution and secure trusted computing.

The main benefits of virtual machine use in the context of HPC cluster systems are:

- The ability to multiplex hardware, whereby granting the user the power to select, change and

run multiple operating systems catered for specific applications on a single node.

- The ability to present a similar, virtualized, generic hardware interface to guest operating systems regardless of the underlying hardware.
- Intelligent runtime environments that are used to manage the cluster are granted the flexibility to migrate entire operating systems by migrating the virtual machine in which the OS is encapsulated from node to node across the cluster computer. The topic of virtual machine migration is thoroughly covered in Subsection 2.2.

With the ability to multiplex hardware, it is possible to highly customize the operating system and runtime environment for each application. For example, operating systems containing different kernel configurations, system software parameters, loaded modules as well as memory and disk space can be booted and run based on the specific needs of an application. Multiplexing of hardware also presents some interesting research scenarios. It allows the possibility of running two operating systems with differing responsibilities on a node, say, a compute OS and a file serving OS, and the ability to toggle the processing power supplied to each operating system, based on its workload.

Most cluster systems are heterogeneous systems built from COTS components. The heterogeneity arises from the fact that only as funds progressively become available, are nodes added to the cluster system. This presents a problem as a wide range of hardware and operating systems can be found in the cluster system at any point in time. Management of, let alone using, such a system becomes a serious challenge. However, when virtual machines are employed, the VMM can present to each operating system a generic set of virtualized hardware devices, allowing the appearance of a homogeneous system despite heterogeneous underlying hardware.

Also, with new versions of software and packages being frequently released, there is an increasing burden of updating software and at the same time possibly needing to reconfigure the cluster system for its use. With virtual machine technology however, there is the potential to possibly update software by redistributing VM images with the updated software and seamlessly merging it together with the current virtual machine image.

### 2.1.1 Summary of Key Papers (Virtual Machine Technology)

A must read paper on virtual machines is entitled “Xen and the Art of Virtualization” [1] by P. Barham, B. Dragovic et al. This paper describes the fundamental design and architecture of the Xen virtual machine monitor, a highly popular VMM used in virtual machine research today. Understanding how Xen was designed and works is fundamental to its utilization and building on top of it. A point to note is that the Xen hypervisor has undergone several revisions since the publication of the abovementioned paper and while significant improvements have been made to it, it has also, not surprisingly, become more complex! Recently, CPU manufacturers Intel and AMD have added virtualization support to their CPUs named VT-x [10] and AMD-V [1] respectively. This additional support allows an unmodified OS to run on Xen, thus making paravirtualization is no longer a necessary requirement.

Worth a quick glance through is the paper “Xen and the Art of Repeated Research” [6] by B. Clark, T. Deshane et al. The authors of this paper conducted multiple experiments with the Xen VMM and reported their findings, which, to a large extent, support those found in [1]. Some key findings presented in the paper are that Xen can support up to about sixteen servers on an average server class machine as well as that Xen performs surprisingly well when compared to hardware made specifically for virtualization.

The paper “Survey of Virtual Machine Research” [8] by R. Goldberg, though dated, provides core concepts still used in virtual machine research today. Two sections of this paper that are highly worth reading are both the “Principles” and “Practice” sections. Both sections provide good introductory material with the Principles section describing the fundamentals of a VMM and the Practice section describing scenarios in which a virtual monitor can be used.

A more recent paper by authors M. Rosenblum and T. Garfinkel entitled “Virtual Machine Monitors: Current Technology and Future Trends” [12], explains the reason for the revival of virtualization, describes the current scene of virtual machine technology as well as lists the challenges that would be faced during the implementation of a VMM today. Three highly applicable areas that it touches on are the challenges associated with CPU virtualization, memory virtualization and I/O virtualization.

In the paper “When Virtual Is Better than Real”, [5] authors P. Chen and B. Nobel make a case as to why virtual machines are worth while investments in any computing environment. The paper provides three case studies: secure logging, intrusion prevention and detection and environment migration. The section of most relevance to our research “Environment migration” as it provides a good overview of the challenges faced in migration of virtual machines as well as the possible methods of overcoming them. Some of these challenges include the development of a mechanism to preserve the synchronous state of an operating system during migration, and the support of migration between physical machines which do not have identical hardware. Migration between physical machines with different hardware poses a problem as despite a hypervisor providing a virtualized set of physical devices to the guest operating system, guest operating systems occasionally directly access physical hardware components for improved performance.

As a node in the cluster may contain more than one CPU or be a multi-core processor, there is a need to efficiently utilize that resource. That idea is explored in E. Bugnion, S. Devine et al.’s paper “Disco: Running Commodity Operating Systems on Scalable Multiprocessors” [4], which examines the problem of extending modern operating systems to run efficiently on large scale shared memory multiprocessors without a large implementation effort. This is achieved by inserting an additional layer of software between the hardware and the operating system, with the additional layer acting like a virtual machine monitor in that multiple copies of “commodity” operating systems can be run on a single scalable computer. The monitor also allows these commodity systems to efficiently cooperate and share resources with each other. The resulting system contains most of the features of custom scalable operating systems developed specifically for scalable multiprocessor systems at a fraction of the cost.

S. King, G. Dunlap et al.’s paper “Operating System Support for Virtual Machines” [11] provides a good introduction to the two different types of VMMs in use today, namely a Type I VMM and a Type II VMM. The paper defines a Type I VMM to be one which runs directly on bare hardware with guest virtual machines running on top of it, while a Type II VMM requires an additional host operating system layer between the bare hardware and the VMM. The paper then goes on to describe three optimizations to improve the performance of a Type II VMM, which yielded a

performance increase of about 14 to 35 percent. This finding is of interest since Xen is a Type II VMM.

The paper by W. Huang, J. Liu et al. entitled “A Case for High Performance Computing with Virtual Machines” [9] provides a strong argument for the use of VM technology in cluster computing systems. It introduces and backs up the potential benefits, including management, system security, performance isolation, checkpointing, restarting and live migration. The paper also demonstrates that despite running applications in a virtualized environment, HPC applications can achieve almost the same performance as those running in a native non-virtualized environment. Two techniques used to obtain such good performance is VMM bypass I/O and Scalable VM image management.

## 2.2 Virtual Machine Migration

There are many reasons for the migration of a virtual machine. For example, from the point of view of a system administrator, the ability to migrate an entire virtual machine across hardware simplifies the issue of server maintenance. The administrator could migrate an operating system containing a web server running on a primary machine to a secondary and take the primary machine offline for servicing purposes.

However, in the context of high performance computing, the main benefit that virtual machine migration offers is the ability to migrate entire operating systems containing running applications to different nodes of a cluster computer. Some benefits of migrating an entire operating system are increased fault tolerance, load balancing and better performance due to reduced latency. In the case of fault tolerance, a virtual machine running on a node which has failed can be migrated to an available node. Load balancing attempts to distribute the workload evenly throughout the cluster. Lastly, by migrating a virtual machine containing running applications closer to their data sources, latency is reduced, which in turn results in better performance.

A logical question to ask is why entire operating systems should be migrated as opposed to migrating an individual process or application. The justification of virtual machine migration is presented in the following section.

### 2.2.1 Migrating Processes

While it would be ideal to be able to map a process to a node, or subset of nodes, in a cluster for the entire execution duration of the process, this unfortunately is not always optimal nor is it even possible. For example, the lack of optimality arises when a subset of nodes, more suited to the particular process than the ones the process is currently executing on, becomes available. There would be inefficient use of resources unless execution of the process is migrated to the more optimal subset of nodes. Also, in a very straightforward scenario, should a node fail, processes would no longer be able to execute on it. A webpage that lists some motivations for process migration can be found at [17].

The paper “The Design and Implementation of Zap: A System for Migrating Computing Environments” [21] lists five key reasons for supporting process migration. These are

- Fault resilience by migrating processes off of faulty hosts
- Data access locality by migrating processes closer to the data
- Better system response time by migrating processes closer to users
- Dynamic load balancing by migrating processes to less loaded hosts
- Improved service availability and administration by migrating processes before host maintenance so that applications can continue to run with minimal downtime.

Despite these advantages, the migration of individual processes is not widely used today due to significant complications. For example, the necessity of similar operating systems and underlying hardware poses a challenge as a heterogeneous cluster could be running different hardware and different operating systems. Even if the conditions for the OS and hardware were met, there is still the problem of residues of the process remaining on the original machine, with complicated abstraction layers being required for such migrations to work. Also, should the original machine, from which the processes were originally migrated from fail, there is a high chance that the processes would cease to function.

### 2.2.2 Migrating Virtual Machines

With virtual machine technology, an entire guest operating system is cleanly encapsulated inside a virtual machine. A narrow interface to the virtualized hardware provided by the hypervisor allows migration of the entire guest operating system to be cleanly executed. An operating system running in a virtual machine can be migrated to execute on an arbitrary node regardless of the underlying hardware. The migration of an entire OS and all of its applications as one unit avoids many of the difficulties faced by process-level migration approaches [15]. Any application formerly running in the guest operating system can resume execution once migration is complete.

### 2.2.3 Summary of Key Papers (Virtual Machine Migration)

A key sister paper to [3] is called “Live Migration of Virtual Machines” [15], by C. Clark, K. Fraser et al. In this paper, the authors describe several mechanisms and strategies for the live migration of virtual machines across clusters, using the Xen VMM. An added bonus is that astonishingly good results were obtained, with all migrations taking under 100ms. This is achieved by using a pre-copy approach in which pages of memory are iteratively copied from the source machine to the destination host, all without ever stopping the execution of the virtual machine being migrated. Towards the end, the virtual machine gets paused for a very brief period to copy any remaining pages to the destination, before resuming execution there. However, there are two limitations to this approach. Firstly, wide area network redirection across subnets is not supported, resulting in a situation similar to that requiring the use of Mobile IP. Secondly, the authors assume that the cluster is running on a network file system. Should there be local disks which need to be migrated, the authors state that total migration times may be intolerably extended. We are reassured that our project is headed in the right direction with the authors stating that, “A key challenge is to develop cluster control software which can make informed decisions as to the placement and movement of virtual machines”.

In the paper “Internet Suspend/Resume” [19], authors M. Kozuch and M. Satyanarayanan propose an approach to allowing an operating system to be accessed from any location using a thin client device. The benefit of such a system is that a user would be able to use his individually customized operating

system from any location supporting such a system. This is achieved through the unprecedented use of layering virtual machine technology on a distributed file system. One useful idea presented in the paper is called “Proactive State Transfer” which attempts to predict the location of where the system would be next accessed. By predicting a system’s access patterns, files could then be copied across the network before actual migration occurs, resulting in lesser time spent during the actual migration of the virtual machine.

C. Sapuntzakis, R. Chandra et al.’s paper entitled “Optimizing the Migration of Virtual Computers” [22] introduces the concept of a capsule, which is the state of a running computer’s disks, memory, CPU registers and I/O devices. Capsules are then used for the migration of virtual machines across different machines. There is similarity to [15] in terms of the goal of the paper and some of the techniques used. However, the main drawback of this paper is that the entire VM would have to be suspended while migration takes place, resulting in downtime and loss of productivity. Useful migration optimization techniques such as ballooning, demand paging of capsule disks, hash based compression, hash cache design, finding similar blocks and HCP protocol can be learnt from the paper.

In the paper “Self-migration of Operating Systems” [18], authors J. Hansen and E. Jul suggest a different approach to migrating virtual machines. Instead of traditional host driven migration, the authors propose self-migration of virtual machines and claim there are additional benefits when virtual machines are migrated in this manner. For example, there would be less overhead incurred from communication between the VMM and the virtual machine as well as increased security benefits. The network and CPU cost of performing the migration is attributed to the guest OS, rather than to the host environment. Portability is another benefit of self migration. Since migration happens without hypervisor involvement, this approach is less dependent on the semantics of the hypervisor and can be ported across different hypervisors and microkernels.

“The Design and Implementation of Zap: A System for Migrating Computing Environments” [21] by S. Osman, D. Subhraveti et al. describes a novel system for transparent migration of applications. Zap introduces the novel concept of a pod, which are groups of processes that are provided a consistent, virtualized view of the underlying system. A consistent view is achieved by placing a thin virtualization layer

on top of the operating system. While Zap’s emphasis is on the migration of processes, there are concepts presented that should be taken into consideration when designing a virtual machine migration system. These concepts, found in Section 4.2 to 4.5 deal with memory virtualization and migration, file system virtualization and migration, device virtualization and migration and network virtualization and migration.

It is worth to bring to the attention of the reader a virtualization feature called Solaris Containers, found in the Sun Solaris 10 operating system designed by Sun Microsystems [26]. Solaris containers are described as “a breakthrough approach to virtualization and software partitioning by allowing many private execution environments be created within a single instance of the Solaris OS. Each environment has its own identity, separate from the underlying hardware. This causes it to behave as if it is running on its own system, making consolidation simple, safe, and secure [23].

### 3. Intelligent Runtime Environments

Optimal mapping of processes to nodes, or processors, in a cluster is crucial to obtaining the most from the cluster system. On one hand there is the goal of providing maximal computing power to a running process or application while on the other is the goal of servicing as many jobs as possible. These goals must be achieved even in the event of node failures. Compounding the problem is that subsets of nodes in a computational grid may be owned by different organizations that impose different restrictions and policies on the usage of resources. There is clearly a need for an intelligent runtime environment<sup>3</sup> to handle these issues.

#### 3.1 Scheduling and Load Balancing

Many factors can affect the performance of a cluster from the point of view of a process, including the processing power of the node(s) on which it is currently executing as well as the locality of execution of the process with respect to its required data. Generally, the higher the processing power of a node,

---

<sup>3</sup> We define an intelligent runtime environment to be a system that consists of scheduling, load balancing, fault tolerance and other mechanisms.

the faster a process executes while the closer a process is to its data sources, the lower the latency for communication, resulting in better performance. Also, depending on whether the application is computationally intensive or data intensive, it may be prudent to design a cluster with nodes that attempts to cater to each type of application.

Scheduling problems are worsened by the fact that the cluster system is usually heterogeneous, comprising of nodes with different processor speeds and resources, for example network connection speeds. Cluster systems could also possibly contain nodes that have a varying number of processors. While attempting to produce a schedule statically is hard, imagine trying to schedule workloads dynamically while multiple jobs are executing on a cluster!

### 3.1.1 Summary of Key Papers (Scheduling and Load Balancing)

The paper “Parallel Job Scheduling on Multicluster Computing Systems” [30] by J. Abawajy, S. Dandamudi et al. claims that the key to making cluster computing work well is by using middleware technologies that can manage the policies, protocols, networks and job scheduling across the interconnected set of computing resources. The authors develop an online dynamic scheduling policy for heterogeneous cluster systems that supposedly performs better than space-sharing and time-sharing policies.

Authors H. Li, D. Groep et al.’s paper entitled “Predicting job start times on clusters” [41] asserts that the job start time predictions are useful to guide resource selections and to balancing the workload distribution in computational grids. The paper introduces a system for predicting job start times on clusters based on statistical analysis of historic job traces and simulation of site schedulers.

The benefits that would be gained from splitting a job across many compute nodes either in a large cluster or grid would depend on the communication patterns between processes, workload on the communication links and the maximum bandwidth of the links. The paper “A Study on Job Co-Allocation in Multiple HPC Clusters” [45] by J. Qin and M. Bauer aims to understand the impact of communications on multi-processor jobs in order to develop scheduling strategies and job allocation algorithms for multi cluster grids which can accommodate communication factors.

“Efficient Resource description and high quality selection for virtual grids” [40] by Y. Kee, D. Logothetis et al. describe a novel resource selection and binding component that utilizes a resource description language and returns a set of selected and bound resources, otherwise known as a virtual grid. The authors claim that the resource description language introduced in the paper effectively captures application level resource abstractions.

A dynamic resource management system for cluster systems called Cluster on Demand is explained in the paper “Dynamic Virtual Clusters in a Grid Site Manager” [35] by J. Chase, D. Irwin et al. Going a step further than just running a VMM on each individual node, the Cluster on Demand system virtualizes an entire physical cluster, partitioning it into multiple virtual clusters. It allocates virtual clusters from a common server pool, with each virtual cluster being an independent entity from the next and having its own working environment. Targeted for use in a large computational grid environment, Cluster on Demand layers on top of existing grid middleware and acts as a site manager that has dynamic policy-based allocation abilities. A very interesting question that arises after reading this paper is: what are the advantages and disadvantages of running a VMM on each individual node as opposed to aggregating the entire cluster system and running a VMM on top of it.

Selecting the hardware to build a cluster system from the vast range of hardware options available while under budget constraints is not an easy task. In the paper “Robust Processor Allocation for Independent Tasks When Dollar Cost for Processors is a Constraint” [49], authors P. Sugavanam, H. Siegel et al. evaluate a set of six heuristics with respect to identifying a set of machines whose total cost falls within an organization’s budget, while at the same time meeting the computational needs of the tasks that would be run on it. The parameter which the heuristics aim to optimize is the makespan, or time taken to complete a set of tasks. The heuristics not only evaluate the system under ideal conditions but also consider situations when performance degrades due to unpredictable circumstances. Of the six heuristics evaluated, the GENITOR heuristic performs the best closely followed by the Partition/Merge Greedy Iterative Maximization heuristic. As the authors claim that the general problem of optimally mapping tasks to machines in a HPC environment has been shown to be NP-complete, developing heuristics for scheduling and

load balancing in intelligent runtime environments should also be considered.

"Xenoservers: Accountable Execution of Untrusted Programs" [47] by D. Reed, I. Pratt et al., introduces the concept of establishing a global public infrastructure that provides data execution services while charging the user for the usage of that service. This system was designed because the authors felt that many networked applications could benefit from executing closer to the data or services with which they interact, and by doing so, avoid long communication latencies and overhead incurred by transferring data over congested or expensive network links. For our purposes, Section 2.2 of the paper, "Accounting and Charging for Resources", provides good ideas for designing an accounting mechanism used to charge a user for the computing services that is provided. Some accounting mechanisms include counting the CPU cycles spent executing an application, counting the number of context switches caused by an application and keeping track of the amount of data transmitted over network devices. It may be worth incorporating such a system into an intelligent runtime environment that manages a HPC cluster which may be used commercially.

## 3.2 Fault Tolerance

Despite scheduling processes to run on a cluster in the most optimal fashion, a failure of nodes or communication links can cause disruptions, resulting in the need for entire execution schedules to be recalculated. This can result in a significant waste of time and computing resources. Fault tolerance mechanisms aim to reduce computational wastage and downtime by attempting to deal with such failures in the most optimal fashion possible.

### 3.2.1 Summary of Key Papers (Fault Tolerance)

"SPHINX: A Fault-Tolerant System for Scheduling in Dynamic Grid Environments" [55], authors J. In, P. Avery et al. propose a framework that effectively schedules work across a large number of distributed clusters that are owned by different organizations in a fault-tolerant way. This is achieved in spite of the highly dynamic nature of the grid and complex policy issues. The novelty of the system lies in the use of effective monitoring of resources and job execution

tracking mechanisms to make scheduling and fault tolerant decisions.

The paper "A Resource Manager for Optimal Resource Selection and Fault Tolerance Service in Grids" [57] by H. Lee, S. Chin et al. describes three mechanisms for enabling optimal performance on grid systems. Firstly, they introduce a resource manager that finds the optimal set of resources. Secondly they develop a fault detector that detects the occurrence of resource failures. Finally a fault manager guarantees that the submitted jobs complete and improves the performance of job execution due to job migration even if failures happen.

Authors G. Lanfermann, G. Allen et al.'s paper "Nomadic Migration: Fault Tolerance in a Disruptive Grid Environment" presents a grid migration framework which provides an application with the ability to seek out and exploit remote computing resources by migrating tasks from site to site, dynamically adapting the application to a changing environment. A "Grid Object", which is a collection of key features of a general grid resource is used to allow scalable information transfer, while fault tolerance is achieved through the use of a peer to peer approach while

## 3.3 Open Source Cluster Application Resources (OSCAR)

The "Open Source Cluster Application Resources (OSCAR)" project is a suite of cluster management tools and best practices. It allows a user to effortlessly set up and administer a Beowulf cluster. The creators of OSCAR describe it as a snapshot of the best known methods for building, programming, and using HPC clusters [60].

A potential benefit of using OSCAR is that of simplified system administration and management of heterogeneous cluster systems. E. Focht's paper "Heterogeneous Clusters with OSCAR: Infrastructure and Administration" [58] discusses design considerations for heterogeneous clusters, sketches the installation procedure and briefly describes administration issues with complex heterogeneous configurations. Although not directly related to virtual machines or intelligent runtime environments, ease of cluster management would be also highly beneficial to any project involving a cluster system.

Due to the frequency of new versions of operating systems being released, repeatedly setting up test systems for use with OSCAR software becomes tedious. The paper “OSCAR Testing with Xen” [61] by G. Vallee and S. Scott describes the use of the Xen hypervisor to help ease the testing of new OSCAR packages over heterogeneous systems, with varying hardware and operating systems. This paper would be worth a read to learn about the strategies that the authors used and evaluate the possibility of the adaptation of similar strategies for use with our projects.

## 4. Future Areas of Research

Thus far, we have conducted a literature survey covering the main areas related to our research projects. In this section we analyze the gathered information and list some problems which we feel have not been adequately covered in the current research literature and are worth tackling in future.

### 4.1 Virtual Device Performance

Despite significant advances made with the Xen VMM and the ability to migrate virtual machines, one weakness still remaining in the VMM is that of poor performance of virtualized devices. For example, network I/O over a virtualized device is much slower than I/O over the real network device. Authors A. Menon, A. Cox et al.’s paper “Optimizing Network Virtualization in Xen” [27] claim that performance degrades by a factor of 2 to 3 times for receive workloads while transmit workloads degrade by up to a factor of five times. Fortunately, they go on to present methods to optimize virtual network interfaces by optimizing the implementation of data transfer path between guest and driver domains and modifying Xen to provide support for guest operating systems to effectively utilize advanced virtual memory features such as superpages and global page mappings.

A paper similar to the one mentioned above is “Virtualizing I/O Devices on VMware Workstation’s Hosted Virtual Machine Monitor” [28] by J. Sugerman, G. Venkitachalam et al. Although VMware Workstation instead of the Xen VMM was used, the authors also reported similar findings of poorer performance when using virtualized devices due to additional virtualization overheads. Some suggestions

made by the authors to reduce virtualization are: modifying the guest OS, optimization of the guest driver protocol, modification of the host OS and bypass of the host OS.

While ANU’s HPC research group’s main focus is not on OS research, it would be worthwhile to invest some time investigating additional optimizations which can be made to reduce the cost of using a virtualized network device with the Xen hypervisor as the efficiency of virtualized network devices significantly affects the overall performance of a cluster computer.

### 4.2 Communication Pathways

It is common for nodes in a cluster system to have multiple network interfaces. By aggregating the interfaces, an increase in network performance is achieved. A paper that deals with this topic is “Performance Enhancement of SMP Clusters with Multiple Network Interfaces using Virtualization” [79] by P. Strazdins, R. Alexander et al. A key finding made is that with one communication stream per process pair, configuring streams to run independently across interfaces generally yielded significantly better performance than did channel bonding. However, better gains for channel bonding were observed as the number of streams and communication intensity increased. The best performance gains came from configurations using Xen virtualized hosts with VMM-bypass for network I/O.

Currently, even with the latest version of the Xen VMM (3.0.3), there is still no way to allow bypass of the Xen VMM for intranode communication. Intranode bypass would be highly valuable for use in nodes containing multiple CPUs that wish to communicate with each other. Current intranode communication between Xen guests on the same node is currently an order of magnitude slower than the native shared memory transport as shown in [79].

### 4.2 Network Connectivity after Migration

Virtual machines usually share a bridged connection with the host machine, resulting in a one to one mapping of an IP address to MAC address. When a virtual machine gets migrated to a new host computer, the IP address to MAC address mapping is no longer

valid, resulting in packets destined for the virtual machine no longer being able to reach it.

The first approach that comes to mind would be that of Mobile IP [81]. However, Mobile IP is not a very viable solution as the host node from which a virtual machine had just been migrated away from would still have to be kept online to continually forward packets to the new host node at which the virtual machine now resides. Should the old host node be taken offline, the network connection is broken. Furthermore, in the event that the virtual machine is repeatedly migrated, packets would have to be forwarded through multiple nodes, resulting in a high amount of wasted bandwidth.

An alternative approach suggested by authors by A. Snoeren and H. Balakrishnan in their paper “An End-to-End Approach to Host Mobility” [82] would be to allocate each virtual machine a token that it can use to identify itself against a central DNS server. In this way, end to end connectivity is maintained even after migration has occurred. It would be worth exploring the feasibility of such a solution and researching other possible mechanisms of maintaining network connectivity.

### 4.3 Memory Management

With the possibility of multiple OS running simultaneously on a single node, efficient memory management, and in particular, reclamation of unused pages becomes an important aspect of a VMM. There are two benefits of keeping the memory used by a virtual machine to a minimum. The first and most obvious benefit is that it provides other virtual machines the option of using the free memory if required. The second benefit is that the virtual machine migration process is sped up due to a smaller working set needing to be migrated

“Memory Resource Management in VMware ESX Server” [29] by C. Waldspurger discusses techniques for the management of memory in virtual machines. The three memory management techniques that are introduced in the paper are: a ballooning technique that reclaims memory from a VM by implicitly causing the guest OS to invoke its own memory management algorithms, an idle memory tax which solves the problem related to share-based management of share-space resources and a content based page sharing and

hot I/O page remapping technique that reduces I/O copying overheads in large memory systems.

### 4.4 File Systems

Consideration must be given to the type of file system that the cluster system uses and the tradeoffs that would occur. For example, a distributed file system would possibly ease the burden of migrating entire virtual hard disk drive (HDD) files across networks during migration operations, with only the runtime environment of an operating system needing to be migrated. An alternative design may be for each node in the cluster to have its own local storage device, which would better support the possibility of each node doubling as a computer and server node.

A classic paper that the reader is recommended to read is “The Google File System” [80] by S. Ghemawat, H. Goto, et al. The file system that is presented is a scalable distributed file system for large distributed data-intensive applications that provides fault tolerance while running on inexpensive commodity hardware. Furthermore, this file system has been shown to be practical and workable as it is widely deployed within Google as the storage platform for the generating and processing of data.

### 4.5 Profiling of Hardware and Software

Being able to accurately profile both the hardware of the cluster as well as the applications that would be run on it is crucial to being able to fully utilize an intelligent runtime environment as the more accurately hardware capabilities and software requirements are understood, the more accurate the mappings of applications to nodes would be. This section discusses in detail some papers related to the profiling of hardware and applications.

“Computational Workload Prediction for Grid Oriented Industrial Applications: The Case of 3D-Image Rendering” [66] by A. Litke, K. Tserpes et al., describes a module for predicting computational workloads of jobs assigned for execution on grids. The module aims to identify the complexity of a given job and predict the computational power that it would require from the grid infrastructure. The module is designed with the use of a trained artificial neural network. Accurately predicting the computational

workload of a given application is a significant aspect of accurately profiling software.

The characteristic of the workload on the ASCI Blue Pacific is described in the paper “The Characteristics of Workload on ASCI Blue-Pacific at Lawrence Livermore National Laboratory” [70] by A. Yoo and M. Jette. This paper presents some interesting findings, for example, the fact that there is little correlation between a job’s execution time and resource demands. Another interesting point found in the paper is that most jobs have very small node and memory requirements. By jobs having small memory requirements, multiprogramming of parallel jobs is highly favored.

In the paper “CPU Load Predictions on the Computational Grid” by Y. Zhang, W. Sun et al, a prediction strategy that forecasts the future CPU load of individual nodes in a cluster based on the previous workload history is described. The CPU workload forecast is calculated independently on each node and when passed on to a scheduler, allows the scheduler to calculate the expected future workload on a particular subset of nodes, thus allowing the scheduler to make intelligent and optimized decisions about how to schedule a given workload.

The performance of a cluster system is largely dependent on how the data on which it operates on is partitioned among its nodes. Thus, it is necessary to strive to develop an optimal partitioning scheme since a scheme that is suboptimal would result in unnecessary communication cost overhead between its nodes being incurred. The paper “Communication Cost Analysis for Parallel Networks” [62] by P. Crandall and M. Quinn mathematically characterizes the communication costs for five partitioning methods, which are: scatter, contiguous point, contiguous row, interleaved and block decomposition. These various partitioning methods are analyzed under varying problem sizes and cluster configurations. Cost equations that express the expected performance based on the various options are then established, enabling the reader to be able to make better informed data decomposition selections.

Authors L. Hu and I. Gorton’s paper “Performance Evaluation for Parallel Systems: A Survey” [64] describes and reviews the various performance evaluation techniques used in the evaluation of the performance of a parallel software system. These techniques include measurement, simulation and analytical modeling. Other issues such as the selection

of metrics most suited to a particular setup and how to construct a good model are also covered in the paper. For issues dealing with workload characterization, the reader may wish to jump straight to section 4.2 of the paper which deals with workload characterization and describes the various steps and techniques associated with it.

Benchmarking and performance evaluation of high performance computing systems is an ongoing process that not only consumes significant staff time but also generates a large amount of data that needs to be stored and analyzed. To help deal with this situation, authors S. Simon, M. Courson et al. describe an automated system that creates an extensive performance profile of a cluster system in their paper entitled “Automated Techniques for Performance Evaluation of Parallel Systems” [67]. Their system is composed of three main components, namely, a data collection and storage module, a data analysis module and a runtime execution module.

The paper “Performance Evaluation of Parallel Systems” [63] by Authors P. Cremonesi, E. Rosti et al. provides a thorough analysis of current performance evaluation methodologies that have been developed for use in the evaluation of a cluster system. The paper concentrates on a few performance case studies of individual cluster system components such as processor, memory, interconnection network, I/O and the operating system. The multiple case studies which the paper contains may be a useful aid in the design of a profiling system.

In the paper by H. Chen, W. Chen et al. entitled “MPIPP: An Automatic profile-guided parallel process placement toolset for SMP clusters and multiclusters” the authors propose a fully automatic scheme for optimized process placement in clusters systems. The authors use a profile guided approach to automatically find an optimized mapping that reduces the cost of point to point communications for arbitrary message passing applications. The implemented toolset is called the MPI Process Placement toolset, which consists of three components, which are: A tool to get the communication profile of a MPI application, a tool to get the network topology of target clusters and an algorithm to find optimized mapping. Of particular interest would be the communication profiling component as our characterization system also plans to incorporate such a component.

The characterization system that our research group plans to design could possibly employ techniques such

as hardware performance counters, instrumentation of the MPI library and to some extent, scalability prediction models. Further details of our project can be found at [69].

## 4.6 Usage of Economic Theory

Economics has been defined to be the study of how agents make choices in the face of constraints. One of the uses of economics is to understand how economies work and what the relations are between the main economic players and institutions. The branch of economics most relevant to the development of an intelligent runtime environment is Microeconomics, which is the study of how individuals, households and firms make decisions to allocate limited resources [76].

Bearing the definition of economics in mind, it is not too far fetched an idea to spend time investigating economic theory and its applicability and suitability to designing algorithms for intelligently managing a large scale cluster or grid system. Such a system has a limited amount of resources while being subject to having to allocate its resources to competing parties.

“A Framework for Measuring Supercomputer Productivity” M. Snir and D. Bader [68] describes utilizing the economic concepts of productivity and Utility Theory to measure the productivity of a HPC system. The authors describe how these concepts are able to capture essential aspects of a HPC system, such as the importance of time-to-solution and the trade off between programming time and execution time.

Additional economics related papers that the interested reader is encouraged to look at are: “Selfish grid computing: game-theoretic modeling and NAS performance results” [72], “An economic-based resource management framework in the grid context” [73], “Employing economics to achieve fairness in usage policing of cooperatively shared computing resources” [74] and “An Economic Approach to adaptive resource management” [75].

A computer science research group that attempts to use economic theory for computer science research is Harvard University’s Harvard EconCS Group [89].

## 5. Highly Recommended Resources

In this section, we cover some papers that, while not directly related to the issues of virtual machines and intelligent runtime environments, are highly relevant to the overall picture of high performance computing, parallel systems, distributed systems and cluster computing.

“Issues and Directions in Scalable Parallel Computing” [68] by Marc Snir, the author mentions an important point which the reader should bear in mind. The author states that the success in scalable parallel computing depends on our ability to develop programming interfaces that can be conveniently used on systems ranging from a few interconnected workstations to massively parallel computers. He claims that while scalable machines can be built, the question is whether we would be able to program them in a similar fashion. In short, Snir asserts that the main challenge of developing parallel computing systems is in the development of scalable software systems.

The authors also recommend the paper “An Overview of Heterogeneous High Performance and Grid Computing” [77] by J. Donarra and A. Lastovetsky. Though slightly dated, this paper still gives the reader a good introduction to hardware platforms, typical uses programming issues, programming systems and the applications related to heterogeneous parallel and distributed computing, as well as the state of research in those areas.

## 6. Conclusion

Through the process of conducting this survey, we found that there has been, to date, a reasonable amount of work published with regards to virtual machines and intelligent runtime environments.

Virtual machine technology appears to be reasonably well defined, and the current trend of virtual machine research seems to be headed in the direction of the addition of new features to and the optimization of current VMM technology. As virtual machines are becoming recognized as important components of today’s computing environment, hardware manufactures are including virtualization support in their processors. With regard to virtual machine technology for use in HPC environments, there are still areas which can be worked on. One example would be modifying the Xen VMM to allow bypass of the hypervisor for intranode communication. Another area that is lacking is to enable network connectivity to be

intact while migration occurs across subnets of a cluster environment.

On the other hand, the area of intelligent runtime environments, and in particular, its scheduling component, is still very open ended and a fertile source of research problems. While there has been a lot of work published to date, there is still no definitive or widely adopted set of procedures to scheduling workloads. One possible explanation for this is that due to the large amount of heterogeneity in the systems that can be built, one single scheduling algorithm or heuristic would not be able to cater to all possible scenarios. Similarly, profiling of hardware and software largely remains an unsolved problem and is an area that, while very challenging and difficult, once successfully tackled, would be a significant breakthrough in the area of HPC research.

With the ANU's availability of funding, equipment, research history and expertise in this area, we seem to be in a good position to develop world class solutions to relevant research problems.

## 7. References and Key Papers

This section has a twofold purpose. Firstly, it serves as the bibliography section for this survey. Secondly, it doubles as a catalog of additional relevant papers that the authors feel would be useful to the reader. The papers listed are grouped according to topic to allow easy reference for the reader.

### 7.1 Virtual Machine Technology

- [1] AMD. *AMD Virtualization*. [http://www.amd.com/us-en/Processors/ProductInformation/0,,30\\_118\\_882\\_6\\_14287,00.html](http://www.amd.com/us-en/Processors/ProductInformation/0,,30_118_882_6_14287,00.html). Accessed Jan 2007.
- [2] A. Awadallah and M. Rosenblum. The vMatrix: A Network of Virtual Machine Monitors for Dynamic Content Distribution. In *Proceedings of the 10<sup>th</sup> IEEE Workshop on Future trends of Distributed Computer Systems*, Suzhou, China, 2004
- [3] P. Barham, B. Dragovic et al. Xen and the Art of Virtualization. In *Proceedings of the 19<sup>th</sup> ACM Symposium on Operating Systems Principles*, Oct. 2003, NY, USA
- [4] E. Bugnion, S. Devine et al. Disco: Running Commodity Operating Systems on Scalable Multiprocessors. *ACM Transactions on Computer Systems*, 15(4):412-447, November 1997
- [5] P. Chen and B. Nobel. When virtual is better than real. In *Proceedings of the 8th IEEE Workshop on Hot Topics on Operating Systems*, May 2001, Germany
- [6] B. Clark, T. Deshane et al. Xen and the Art of Repeated Research. *FREENIX 2004*, Nov 2004, Boston, USA.
- [7] T. Garfinkel, B. Pfaff et al. Terra: A Virtual Machine-Based Platform for Trusted Computing. In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, October 2003, Bolton Landing, NY, USA.
- [8] R. Goldberg. Survey of Virtual Machine Research. *IEEE Computer*, pages 34 - 45, June 1974
- [9] W. Huang, J. Liu et al. A Case for High Performance Computing with Virtual Machines, In *Proceedings of the 20<sup>th</sup> ICS*, 2006, Queensland, Australia
- [10] Intel Corporation. Intel Virtualization Technology: Hardware Support for Efficient Processor Virtualization. <http://www.intel.com/technology/itj/2006/v10i3/1-hardware/5-architecture.htm>. Accessed Jan 2007.
- [11] S. King, G. Dunlap et al. Operating System Support for Virtual Machines. In *Proceedings of the 2003 Annual USENIX Technical Conference*, June 2003, Texas, USA
- [12] M. Rosenblum and T. Garfinkel. Virtual Machine Monitors: Current Technology and Future Trends. *IEEE Computer*, 35(5):39-47, 2005
- [13] J. Reumann, A. Mehra et al. Virtual Services: A New Abstraction for Server Consolidation. In *Proceedings of USENIX Annual Technical Conference, General Track 2000*, San Diego, USA
- [14] Rich Uhlig, Gil Neiger, Dion Rodgers, Amy L. Santoni, Fernando C.M. Martins, Andrew V. Anderson, Steven M. Bennett, Alain Kagi, Felix

H. Leung, Larry Smith, "Intel Virtualization Technology," *Computer*, vol. 38, no. 5, pp. 48-56, May, 2005.

## 7.2 Virtual Machine Migration

- [15] C. Clark, K. Fraser et al. Live Migration of Virtual Machines. *Networked Systems Design and Implementation*, 2005
- [16] F. Douglis and J. Ousterhout. Transparent Process Migration: Design Alternatives and the Sprite Implementation. *Software Practice and Experience*, 21(7), July 1991
- [17] J. Elson. "Motivations for Process Migration". <http://www.circlemud.org/~jelson/writings/process-migration/node3.html>. Accessed Jan 2007
- [18] J. Hansen and E. Jul. Self-migration of Operating Systems. In *Proceedings of the 11<sup>th</sup> ACM SIGOPS European Workshop (EW2004)*, pages 126-130, 2004
- [19] M. Kozuch and M. Satyanarayanan. Internet Suspend/Resume. In *Proceedings of the 4<sup>th</sup> IEEE Workshop on Mobile Computing Systems and Applications, Calicoon, NY*, June 2002
- [20] Y. Li, Z. Lan, "A novel workload migration scheme for heterogeneous distributed computing," *ccgrid*, pp. 1055-1062, *Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid'05) - Volume 2*, 2005.
- [21] S. Osman, D. Subhraveti et al. The Design and Implementation of Zap: A System for Migrating Computing Environments. In *The proceedings of the 5<sup>th</sup> OSDI 2002*, Boston, USA.
- [22] C. Sapuntzakis, R. Chandra et al. Optimizing the Migration of Virtual Computers. In *Proceedings of the 5<sup>th</sup> Symposium on Operating Systems Design and Implementation (OSDI 2002)*, *ACM Operating Systems Review, Winter 2002 Special Issue*, pages 377-390, Boston, MA, USA, Dec. 2002
- [23] Sun Microsystems - Solaris 10 Datasheets. <http://www.sun.com/software/solaris/ds/utilization.jsp>. Accessed Jan 2007.

[24] Sathish S. Vadhiyar, Jack J. Dongarra, "A Performance Oriented Migration Framework For The Grid," *ccgrid*, p. 130, *3rd International Symposium on Cluster Computing and the Grid*, 2003.

[25] Geoffroy Vallée, Christine Morin, Renaud Lottiaux, Jean-Yves Berthou, Ivan Dutka Malen, "Process Migration Based on Gobelins Distributed Shared Memory," *ccgrid*, p. 325, *2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'02)*, 2002.

[26] Wikipedia, the free Encyclopedia - Solaris Containers. [http://en.wikipedia.org/wiki/Solaris\\_Containers](http://en.wikipedia.org/wiki/Solaris_Containers). Accessed Jan 2007.

## 7.3 Additional Resources for Virtual Machines

- [27] A. Menon, A. Cox et al. Optimizing Network Virtualization in Xen. In *Proceedings of the USENIX Annual Technical Conference*, 2006, Boston, USA
- [28] J. Sugerman, G. Venkitachalam et al. Virtualizing I/O Devices on VMware Workstation's Hosted Virtual Machine Monitor. In *Proceedings of the USENIX Annual Technical Conference*, 2001
- [29] C. Waldspurger. Memory Resource Management in VMware ESX Server. In *Proceedings of OSDI*, Boston, 2002

## 7.4 Scheduling and Load Balancing

- [30] J. H. Abawajy, S. P. Dandamudi, "Parallel Job Scheduling on Multicluster Computing Systems," *cluster*, p. 11, *IEEE International Conference on Cluster Computing (CLUSTER'03)*, 2003.
- [31] K. Amiri, D. Petrou et al. Dynamic Function Placement for Data-intensive Cluster Computing. In *Proceedings of the Geieral Track 2000 USENIX Annual Technical Conference*, 2000, San Diego, USA.
- [32] J. Barbosa, C. Morais et al. Static Scheduling of dependent parallel tasks on heterogeneous clusters.

- In *Proceedings of Cluster 2005*, 2005, Boston Massachusetts
- [33] M. Beltran, J.L. Bosque, "Information policies for load balancing on heterogeneous systems," *ccgrid*, pp. 970-976, *Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid'05)* - Volume 2, 2005.
- [34] A. Chandra, M. Adler et al. Surplus Fair Scheduling: A Proportional-Share CPU Scheduling Algorithm for Symmetric Multiprocessors. In *Proceedings of the 4<sup>th</sup> Symposium on Operating Systems Design and Implementation*, 2000, San Diego, California.
- [35] J. Chase, D. Irwin et al. Dynamic Virtual Clusters in a Grid Site Manager. In *Proceedings of International Symposium on High-Performance Distributed Computing*, 2003, Washington.
- [36] H. Chen, W. Chen et al. MPIPP: An automatic profile-guided parallel process placement toolset for SMP clusters and multiclusters. In *Proceedings of the 20<sup>th</sup> International Conference on Supercomputing*, 2006, Cairns, Australia.
- [37] M. Dobber, G. Koole, R. van der Mei, "Dynamic load balancing experiments in a grid," *ccgrid*, pp. 1063-1070, *Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid'05)* - Volume 2, 2005.
- [38] Ligang He, Stephen A. Jarvis, Daniel P. Spooner, Graham R. Nudd, "Dynamic Scheduling of Parallel Real-Time Jobs by Modelling Spare Capabilities in Heterogeneous Clusters," *cluster*, p. 2, *IEEE International Conference on Cluster Computing (CLUSTER'03)*, 2003.
- [39] Dimitrios Katramatos, Marty Humphrey, Cheol-Min Hwang, Steve J. Chapin, "Developing a Cost/Benefit Estimating Service for Dynamic Resource Sharing in Heterogeneous Clusters: Experience with SNL Clusters," *ccgrid*, p. 355, *1st International Symposium on Cluster Computing and the Grid*, 2001.
- [40] Yang-Suk Kee, D. Logothetis, R. Huang, H. Casanova, A.A. Chien, "Efficient resource description and high quality selection for virtual grids," *ccgrid*, pp. 598-606, *Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid'05)* - Volume 1, 2005.
- [41] Hui Li, D. Groep, J. Templon, L. Wolters, "Predicting job start times on clusters," *ccgrid*, pp. 301-308, *2004 IEEE International Symposium on Cluster Computing and the Grid (CCGrid'04)*, 2004.
- [42] Jiadao Li, Ramin Yahyapour, "Learning-Based Negotiation Strategies for Grid Scheduling," *ccgrid*, pp. 576-583, *Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)*, 2006.
- [43] S. McClure and R. Wheeler. MOSIX: How Linux Clusters Solve Real World Problems. In *Proceedings of the USENIX Annual Technical Conference*, 2000, San Diego, USA
- [44] D. Petrou, J. Milford et al. Implementing Lottery Scheduling: Matching the Specializations in Traditional Schedulers. In *Proceedings of the 1999 USENIX Annual Technical Conference*, June 1999, Monterey, California.
- [45] Jinhui Qin, Michael Bauer, "A Study on Job Co-Allocation in Multiple HPC Clusters," *hpcs*, p. 3, *20th International Symposium on High-Performance Computing in an Advanced Collaborative Environment (HPCS'06)*, 2006
- [46] Xiao Qin, Hong Jiang, Yifeng Zhu, David R. Swanson, "Towards Load Balancing Support for I/O-Intensive Parallel Jobs in a Cluster of Workstations," *cluster*, p. 100, *IEEE International Conference on Cluster Computing (CLUSTER'03)*, 2003.
- [47] D. Reed, I. Pratt. Xenoservers: Accountable Execution of Untrusted Programs. In *Proceedings of HotOS*, 1999
- [48] Hermes Senger, Liria Matsumoto Sato, "Load Distribution for Heterogeneous and Non-Dedicated Clusters Based on Dynamic Monitoring and Differentiated Services," *cluster*, p. 199, *IEEE International Conference on Cluster Computing (CLUSTER'03)*, 2003.
- [49] H. Siegel, A. Maciejewski et al. Robust Processor Allocation for Independent Tasks When Dollar Cost for Processors is a Constraint. In *Proceedings of Cluster 2005*, 2005, Boston, Massachusetts

- [50]D. Sullivan, R. Hass et al. Tickets and Currencies Revisited: Extensions to Multi-Resource Lottery Scheduling. In *Proceedings of the 27<sup>th</sup> Workshop on Hot Topics in Operating Systems*, 1999, Rio Rico, USA.
- [51]B. Urgaonkar, P. Shenoy et al. Resource Overbooking and Application Profiling in Shared Hosting Platforms. In *Proceedings of the 5<sup>th</sup> Operating System Design and Implementation*, 2002, Boston, Massachusetts.
- [52]Ming Wu, Xian-He Sun, "A General Self-Adaptive Task Scheduling System for Non-Dedicated Heterogeneous Computing," cluster, p. 354, *IEEE International Conference on Cluster Computing (CLUSTER'03)*, 2003.
- [53]Yang Zhang, Anirban Mandal, Henri Casanova, Andrew A. Chien, Yang-Suk Kee, Ken Kennedy, Charles Koelbel, "Scalable Grid Application Scheduling via Decoupled Resource Selection and Scheduling," ccgrid, pp. 568-575, *Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)*, 2006.

## 7.5 Fault Tolerance

- [54]M. Mat Deris, M. Rabiei, A. Noraziah, H. M. Suzuri, "High Service Reliability for Cluster Server Systems," cluster, p. 281, *IEEE International Conference on Cluster Computing (CLUSTER'03)*, 2003.
- [55]Jang-uk In, Paul Avery, Richard Cavanaugh, Laukik Chitnis, Mandar Kulkarni, Sanjay Ranka, "SPHINX: A Fault-Tolerant System for Scheduling in Dynamic Grid Environments," ipdps, p. 12b, *19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05) - Papers*, 2005.
- [56]Gerd Lanfermann, Gabrielle Allen, Thomas Radke, Edward Seidel, "Nomadic Migration: Fault Tolerance in a Disruptive Grid Environment," ccgrid, p. 280, *2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'02)*, 2002.
- [57]H. Lee, S. Chin et al., "A resource manager for optimal resource selection and fault tolerance service in Grids," ccgrid, pp. 572-579, 2004 *IEEE*

*International Symposium on Cluster Computing and the Grid (CCGrid'04)*, 2004.

## 7.6 Open Source Cluster Application Resources (OSCAR)

- [58]Erich Focht, "Heterogeneous Clusters with OSCAR: Infrastructure and Administration," hpcs, p. 37, *20th International Symposium on High-Performance Computing in an Advanced Collaborative Environment (HPCS'06)*, 2006.
- [59]Chokchai Leangsuksun, Lixin Shen, Tong Liu, Hertong Song, Stephen L. Scott, "Availability Prediction and Modeling of High Availability OSCAR Cluster," cluster, p. 380, *IEEE International Conference on Cluster Computing (CLUSTER'03)*, 2003.
- [60]OSCAR - Open Source Cluster Application Resources. <http://oscar.openclustergroup.org>. Accessed Jan 2007.
- [61]Geoffroy Vallee, Stephen L. Scott, "OSCAR Testing with Xen," hpcs, p. 43, *20th International Symposium on High-Performance Computing in an Advanced Collaborative Environment (HPCS'06)*, 2006.

## 7.7 Profiling of Applications

- [62]P. Crandall and M. Quinn. "Communication Cost Analysis for Parallel Networks". *Technical Report 94-80-04*, 1994, Oregon State University, OR, USA.
- [63]P. Cremonesi, E. Rosti et al. "Performance Evaluation of Parallel Systems". *Elsevier Preprint, Jan 1999*, 1999
- [64]L. Hu and I. Gorton. "Performance Evaluation for Parallel Systems: A Survey". *UNSW-CSE-TR-9709*, October 1997
- [65]B. Huang, M. Bayer et al. Hpcbench - a Linux-based network benchmark for high performance networks. In *The 19<sup>th</sup> Symposium on High Performance Computing Systems and Applications*, 2005, HPCS 2005

- [66]A. Litke, K. Tserpes, T. Varvarigou, "Computational workload prediction for grid oriented industrial applications: the case of 3D-image rendering," *ccgrid*, pp. 962-969, *Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid'05)* - Volume 2, 2005.
- [67]S. Simon, M. Courson et al. "Automated Techniques for Performance Evaluation of Parallel Systems". *EURESCO99*, Spain, June 1999
- [68]M. Snir and D. Bader. "A Framework for Measuring Supercomputer Productivity". *International Journal for High Performance Computing Applications 2004*, 18: 399-416, 2004.
- [69]Peter Strazdins - Characterization of Application Performance of Cluster Computers. <http://cs.anu.edu.au/~Peter.Strazdins/postgrad/PerfEvalMethod-Apps.html>. Accessed Feb 2007.
- [70]Andy B. Yoo, Morris A. Jette, "The Characteristics of Workload on ASCI Blue-Pacific at Lawrence Livermore National Laboratory," *ccgrid*, p. 295, *1st International Symposium on Cluster Computing and the Grid*, 2001.
- [71]Yuanyuan Zhang, Wei Sun, Yasushi Inoguchi, "CPU Load Predictions on the Computational Grid \*," *ccgrid*, pp. 321-326, *Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)*, 2006.

## 7.8 Economic Approaches

- [72]Yu.-K. Kwok, S. Song, K. Hwang, "Selfish grid computing: game-theoretic modeling and NAS performance results," *ccgrid*, pp. 1143-1150, *Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid'05)* - Volume 2, 2005.
- [73]Chuliang Weng, Minglu Li, Xinda Lu, Qianni Deng, "An economic-based resource management framework in the grid context," *ccgrid*, pp. 542-549, *Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid'05)* - Volume 1, 2005.
- [74]P. Xavier, Wentong Cai, Bu-Sung Lee, "Employing economics to achieve fairness in usage policing of cooperatively shared computing resources," *ccgrid*, pp. 326-333, *Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid'05)* - Volume 1, 2005.
- [75]N. Stratford and R. Mortier. An Economic Approach to Adaptive Resource Management. In *Proceedings of the 27<sup>th</sup> Workshop on Hot Topics in Operating Systems*, 1999, Rio Rico, USA.
- [76]Wikipedia, the free Encyclopedia - Economics. <http://en.wikipedia.org/wiki/Economics>. Accessed Jan 2007

## 7.9 Miscellaneous

- [77]J. Dongarra and A. Lastovetsky. An Overview of Heterogeneous High Performance and Grid Computing, 2004, *Nova Science Publishers Inc. To Appear*.
- [78]M. Snir. Issues and Directions in Scalable Parallel Computing. In *Proceedings of the 12<sup>th</sup> ACM Symposium on Principles of Distributed Computing (PODC)*, 1993, NY, USA
- [79]P. Strazdins, R. Alexander et al. "Performance Enhancement of SMP Clusters with Multiple Network Interfaces using Virtualization". *XenHPC 06*, Sorrento, 2006
- [80]S. Ghemawat, H. Gobioff et al. The Google File System. In *Proceedings of the 19<sup>th</sup> ACM Symposium on Operating System Principles*, 2003, New York, USA.
- [81]C. E. Perkins and A. Myles. Mobile IP. In *Proceedings of International Telecommunications Symposium*, pages 415-419, 1997
- [82]A. Snoeren and H. Balakrishnan. An End-to-End Approach to Host Mobility. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 155-166. ACM Press, 2000.
- [83]The Jabberwocky Project. Research Projects – Project Summaries. <http://jabberwocky.anu.edu.au>. Accessed Jan 2007.
- [84]Top 500 Supercomputing Sites. <http://www.top500.org/stats>. Accessed Feb 2007.

## 7.10 Notable Research Groups

- [85] Carnegie Mellon University Parallel Data Lab. Abacus: Dynamic Function Placement for Data-Intensive Cluster Computing. <http://www.pdl.cmu.edu/Abacus/index.html>. Accessed Jan 2007.
- [86] Carnegie Mellon University Parallel Data Lab. Batchactive Scheduling. <http://www.pdl.cmu.edu/batchactive>. Accessed Jan 2007.
- [87] Columbia University Network Computing Lab. Zap: Migration of Legacy and Network Applications. <http://www.ncl.cs.columbia.edu/research/migrate>. Accessed Jan 2007.
- [88] Duke University Internet Systems and Storage Group. Cluster on Demand. <http://issg.cs.duke.edu/cod>. Accessed Jan 2007.
- [89] Harvard University Harvard EconCS Group. <http://www.eecs.harvard.edu/econcs/>. Accessed Jan 2007
- [90] Massachusetts Institute of Technology Supercomputing Technologies Group. Adaptive Scheduling of Parallel Jobs. <http://supertech.csail.mit.edu/dynamicProcAlloc.html>. Accessed Jan 2007.
- [91] University of California – Berkeley Reliable, Adaptive and Distributed Systems Laboratory. Workload characterization and generation. [http://radlab.cs.berkeley.edu/wiki/Workload\\_characterization\\_and\\_generation](http://radlab.cs.berkeley.edu/wiki/Workload_characterization_and_generation). Accessed Jan 2007.
- [92] University of California at San Diego Concurrent Systems Architecture Group. Virtual Grids. <http://www-csag.ucsd.edu/projects/VGrADs>. Accessed Jan 2007.
- [93] University of California at San Diego Concurrent Systems Architecture Group. High Performance Virtual Machines. <http://www-csag.ucsd.edu/projects/hpvm.html>. Accessed Jan 2007.
- [94] University of Maryland-College Park Distributed Systems Software Lab. Active Harmony. <http://www.dyninst.org/harmony>. Accessed Jan 2007.
- [95] University of Southern California Computer Networks Division. The Scalable Computing Infrastructure Project. <http://gost.isi.edu/projects/scope>. Accessed Jan 2007.
- [96] University of Wisconsin-Madison Operating Systems Research Group. Condor: High Throughput Computing. <http://www.cs.wisc.edu/condor>. Accessed Jan 2007

## **Appendix A – Sources of Information**

### **Conferences (in alphabetical order)**

- CCGRID 2006, 2005, 2004, 2003, 2002, 2001
- Cluster 2004, 2003, 2002, 2001
- HotOS 2003, 2001, 1999, 1997, 1995, 1993
- HPCA 2005, 2004, 2003, 2002, 2001, 2000
- HPCS 2006, 2005, 2004, 2002
- ICDCS 2006, 2005, 2004
- ICS 2006, 2005, 2004, 2003, 2002, 2001, 2000, 1999, 1998, 1997, 1996, 1995
- Micro 2005, 2004, 2003, 2002, 2001, 2000, 1999, 1997, 1996, 1995
- OSDI 2004, 2002, 2000, 1999, 1996, 1994
- SOSOP 2005, 2003, 2001, 1999, 1997, 1995, 1993
- USENIX Technical Conference (including FREENIX Track where applicable) 2004, 2003, 2002, 2001, 2000, 1999, 1996, 1995, Summer 1994, Winter 1994

### **Journals (in alphabetical Order)**

- Communications of the ACM Volume 49 (2006), Volume 48 (2005)
- IEEE Computer Volume 39 (2006), Volume 38 (2005)
- IEEE Transactions on Computers Volume 56 (2006), Volume 55 (2006)

### **University Research Groups (in alphabetical Order)**

- Brown University
- California Institute of Technology
- Carnegie Mellon University
- Columbia University
- Cornell University
- Duke University
- Harvard University
- Massachusetts Institute of Technology
- Princeton University
- Purdue University-West Lafayette
- Rice University
- Stanford University
- University of California - Berkeley
- University of California - Los Angeles
- University of California - San Diego
- University of Illinois-Urbana-Champaign
- University of Maryland - College park
- University of Massachusetts-Amherst
- University of Michigan - Ann Arbor
- University of North Carolina-Chapel Hill
- University of Pennsylvania
- University of Southern California
- University of Texas - Austin
- University of Washington
- University of Wisconsin-Madison
- Yale University

## Appendix B - Notable Research Groups and Projects

University	Research Group	Notable Project(s)	Category
University of California-San Diego	Concurrent Systems Architecture Group <a href="http://www-csag.ucsd.edu">http://www-csag.ucsd.edu</a>	Virtual Grids	VM Migration, Cluster Scheduling and Intelligent Runtime Environments
		High Performance Virtual Machines	Virtualization, Scheduling and Intelligent Runtime Environments
University of Maryland-College Park	Distributed Systems Software Lab <a href="http://www.cs.umd.edu/projects/dssl">http://www.cs.umd.edu/projects/dssl</a>	Active Harmony	Scheduling and Intelligent Runtime Environments
Carnegie Mellon University	Parallel Data Lab <a href="http://www.pdl.cmu.edu">http://www.pdl.cmu.edu</a>	Abacus	Scheduling, Intelligent Runtime Environments and Load Balancing
		Batchactive Scheduling	Scheduling and Intelligent Runtime Environments
University of Wisconsin-Madison	UW Operating Systems <a href="http://www.cs.wisc.edu/areas/os">http://www.cs.wisc.edu/areas/os</a>	Condor	Scheduling, Intelligent Runtime Environments and High Throughput Computing
Duke University	Internet Systems and Storage Group <a href="http://issg.cs.duke.edu">http://issg.cs.duke.edu</a>	Cluster on Demand	VM Migration, Scheduling and Intelligent Runtime Environments
Columbia University	Network Computing Lab <a href="http://www.ncl.cs.columbia.edu">http://www.ncl.cs.columbia.edu</a>	Zap: Migration of Legacy and Network Applications	VM Migration
University of Southern California	Computer Networks Division <a href="http://www.isi.edu/divisions/div7">http://www.isi.edu/divisions/div7</a>	The Scalable Computing Infrastructure Project	Scheduling and Intelligent Runtime Environments
University of California-Berkeley	Reliable, Adaptive and Distributed Systems Laboratory <a href="http://radlab.cs.berkeley.edu/wiki/RAD_Lab">http://radlab.cs.berkeley.edu/wiki/RAD_Lab</a>	Workload characterization and generation	Characterization of Applications
Massachusetts Institute of Technology	Supercomputing Technologies Group <a href="http://supertech.csail.mit.edu">http://supertech.csail.mit.edu</a>	Adaptive Scheduling of Parallel Jobs	Scheduling and Intelligent Runtime Environments