Future Generation Supercomputers II: A Paradigm for Cluster Architecture

N.Venkateswaran[§], Deepak Srinivasan[†], Madhavan Manivannan[†], TP Ramnath Sai Sagar[†]
Shyamsundar Gopalakrishnan[†] VinothKrishnan Elangovan[†] Arvind M[†] Prem Kumar Ramesh[♭]
Karthik Ganesan[♭] Viswanath Krishnamurthy[♭]
Sivaramakrishnan[♭]

Abstract

In part-I, a novel multi-core node architecture was proposed which when employed in a cluster environment would be capable of tackling computational complexity associated with wide class of applications. Furthermore, it was discussed that by appropriately scaling the architectural specifications, Teraops computing power could be achieved at the node level. In order to harness the computational power of such a node, we have developed an efficient application execution model with a competent cluster architectural backbone. In this paper we present the novel cluster paradigm, dealing with operating system design, parallel programming model and cluster interconnection network. Our approach in developing the competent cluster design revolves around an execution model to aid the execution of multiple applications simultaneously on all partitions of the cluster, leading to cost sharing across applications. This would be a major initiative towards achieving Cost-Effective Supercomputing.

1. Introduction

High performance monolithic clusters, having good performance and scalability are becoming increasingly popular in the research community for their ability to cater to specific application requirements. The level of performance is characterized by the node architecture, network topology, compiler, parallel programming paradigm and operating system. Making better design choices would improve the execution time of large scale applications, which are currently predicted to be in Teraflop years. In this paper, we discuss the impact of these design choices on the application's performance and provide insights into a supercomputing model which would cater to the demands of the next generation grand challenge applications.

Performance Analysis carried out by Adolfy et al.[1] on Blue Gene/L, Red Storm, and ASC Purple clearly marks that these machines although significantly diverse along the afore-mentioned design parameters, offer good performance during "Grand Challenge Application" execution. But future generation applications might require close coupling of previously independent application models, as highlighted in NASA's report on Earth Science Vision 2030[2], which involves simulations on coupled climate models, such as ocean, atmosphere, biosphere and solid earth. These kind of hybrid applications call for simultaneous execution of the component applications, since the execution of different applications on separate clusters may not be prudent in the context of constraints laid down by cluster performance and operational cost.

There is hence a need, to develop an execution model for cost effective supercomputing which will envisage simultaneous execution of multiple applications on all partitions of a single cluster(without sacrificing the performance of individual application) unlike the current models in which different applications are executed in independent partitions of the cluster. Future supercomputing models should also address critical design aspects like reliability, fault tolerance and low power issues which are increasingly becoming important design criterions. This paper along with part-I [3] conjointly proposes a supercomputing model which is expected to offer superior performance/cost ratio and deal with the rigors posed by computational requirements of the hybrid applications (composed of interdependent applications).

This execution model introduces new challenge in the cluster architecture and operating system design for handling the increased mapping complexity and tracking mechanisms during the execution of multiple applications. This execution model introduces new challenges in the cluster architecture and operating system to enable it to handle the increased mapping complexity and tracking, during the execution of multiple applications. Also, the programming paradigm adopted should help exploit both node and cluster architecture characteristics and ease the programming complexities involved in application development. However, the support for execution of such diverse workloads encountered during simultaneous multiple application execution lies in the efficient design of the node architecture. In paper-I [3], we discuss the capability of MIP-based (Memory In Processor) heterogeneous multi-core node architectures to handle SMAG (Simultaneous Multiple AlGorithms) execution aiding the proposed execution model by running traces of mul-

§N.Venkateswaran Founder Director, WAran Research FoundaTion (WARFT), Chennai, India. Email:- waran@warftindia.org
†- WARFT Research Trainee, 2005-2007
♭−Former WARFT Research Trainee current status given at end of this paper

tiple applications in the same node.

The paper is organized into 4 sections. Section 2 discusses the scope for improvement in the design features of current generation clusters in order to meet the requirements of performance greedy hybrid applications, also taking into consideration the operating cost factor. Section 3 highlights a cluster model that incorporates all the architectural concepts proposed in section2 and investigates its potential for cost effective execution of multiple applications. Section 4 addresses the ramification of this model on performance, resource utilization profile and their influence on the performance/cost relation.

2. Design Characteristics of High Performance Clusters

Performance modeling has come a long way in helping researchers characterize cluster design to achieve expected performance. Different methodologies have been evolved to accurately compare, analyze and predict the performance of various designs and features such as the node and cluster architecture, operating system, and programming paradigm that have been identified to play dominant roles[4]. We discuss these design issues in high performance clusters and propose new directions for evolving a cluster model to meet the requirements of future generation applications.

2.1 Cluster Interconnection Network

The type of interconnects and the topology adopted affects the overall performance of the communication network. Conventional networks use wired network topologies supported by different technologies for implementing large scale HPC (High Performance Cluster) designs.The most popular choices for network interconnects are Fast Ethernet, Gigabit Ethernet, Myricom Myrinet, and InfiniBand.

The communication pattern of massive applications vary dynamically during execution time and each pattern can be served better by employing a particular interconnection topology. If it is possible to dynamically reconfigure the cluster topology to suit the communication requirements of the instant, it would greatly boost the performance of the application execution on the cluster. Although many of the currently employed networks have been successful in satisfying the high bandwidth requirements, they are unable to meet the overwhelming degree of scalability required by hybrid application execution and the concept of 'reconfigurability' can not be accomplished.

Considering the execution of hybrid applications, communication behavior across the nodes varies, in that different communication links are needed along the execution span and hence to effectively support this dynamic reconfiguration of the network becomes a necessity. Such reconfigurability cannot be easily achieved in wired interconnection network.

2.2 Operating Systems

In the current execution model, workload of a single application is mapped on to a set of nodes, which does the work of load balancing across the node of a cluster. The node is usually empowered with a stripped kernel, which performs the core OS functionalities such as memory management, Process scheduling, I/O handling and Interrupt handling. But in the context of the proposed execution model, a new OS paradigm is required for handling the complexities associated with parallel mapping and data tracking of the huge amount of data associated with the different applications. In this scenario, the reliability of the operating system is of paramount importance as the integrity of IO data sequencing is critical, particularly when dealing with million node clusters. Thus the capability of the cluster to stomach the complexities involved in multiple applications' execution lies in an efficient OS design.

2.3 Parallel Programming Paradigm and Compiler

The current Parallel Programming Languages are categorized into Data parallel languages, explicit communication model and functional languages. These parallel languages either stress on data parallelism as in NESL and C* [5, 6] or the communication model where the user is completely responsible for the creation and management of processes and their interactions with one another, as in Emerald or COOL [7]. No single language has been developed which can handle both data parallelism and communication model efficiently.

With increasing complexity of the application, the programming model needs to be simple and expressible and also allow programmers to represent complex logic efficiently. For this, the Parallel Programming Language (PPL) model should be simple and portable form of object-based so that it can be easily understand, modify and debug than its sequential counterpart. These PPL should have constructs which must be capable of exploit the level of parallelism inherently present in the application matching the underlying architecture (ISA of the node architecture). A new PPL model of the MIP SCOC cluster incorporating all the above features will be discussed in section 3.6.

3. MODEL FOR NEXT GENERATION SUPERCOMPUTERS

In order to create a design space for supercomputers, the focus should also be on aspects like power, performance, cost and their related tradeoffs. In this section we present a conceptual model (fig. 1) for cluster design taking into consideration all the design issues discussed in section 2. The cluster model comprises of MIP-paradigm based heterogeneous multi-core nodes which are capable of handling 'Simultaneous Multiple AlGorithm' (SMAG) execution. In our discussion, we primarily attempt to give a conceptual overview of the proposed cluster model.

3.1 Simultaneous Multiple APPlication (SMAPP) Execution & Cluster/ Host Architecture

Due to the ever increasing demand posed by scientific and engineering applications, it becomes mandatory to evolve supercomputing clusters whose node architecture is highly tuned towards these applications. This to a large extent alleviates the time complexity of the application execution. Such a node architecture capable of achieving SMAG, encompassing varied functional units and covering wide range of applications is explained in [3]. The quasi-ASIC design of the node, while improving the capabilities to handle increased complexity of the application, also enables the node

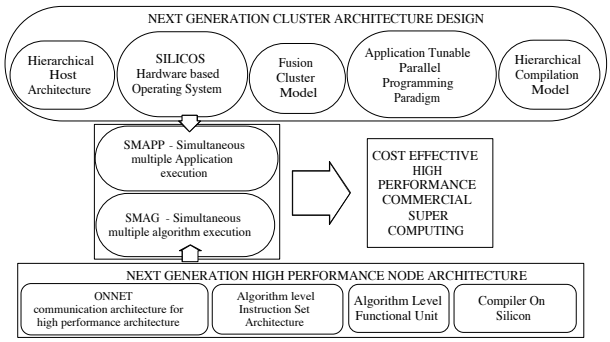


Figure 1: Model for next generation Supercomputers

to support the intended multiple application execution at the cluster level. The details of this cluster architecture and the execution flow of SMAPP are discussed in section 3.2 and section 3.6 respectively.

The fig. 2 is an abstraction of the SMAPP flow in the

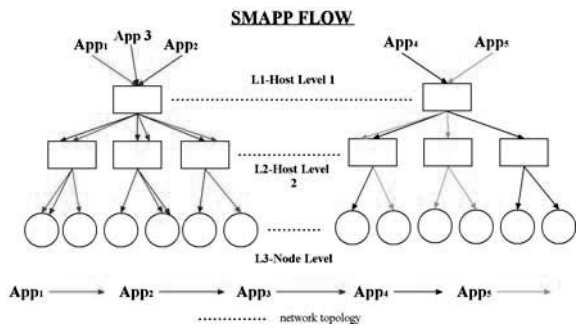


Figure 2: SMAPP concept

multiple host hierarchial cluster. A major challenge for the host system lies in tackling the mapping complexities and sequencing the terabytes of data resulting from the simultaneous execution of several applications, which otherwise might lead to a potential 'virtual I/O bottleneck'. Moreover, considering the computational strength of the MIP-paradigm based node architecture, the host system should be efficient in meeting the feed rate required by the nodes. Preprocessing at the cluster reduces the compiling complexity required in the node thereby aiding efficient execution, by reducing the compiling load on the node, leaving only the scheduling. Especially when the cluster size becomes very large (million nodes), these issues have a huge impact. Investigation on simplifying these issues lead us to the idea of developing a hierarchical based host system. The architecture and functionalities of the host system will be presented in 3.3 & 3.4

In accordance with the above mentioned school of thought, we have developed a hybrid, pyramid structured cluster design as depicted in fig. 3, wherein two stages of problem decomposition occurs at the primary and secondary hosts, to efficiently partition and map the applications onto the MIP cluster[8,9].

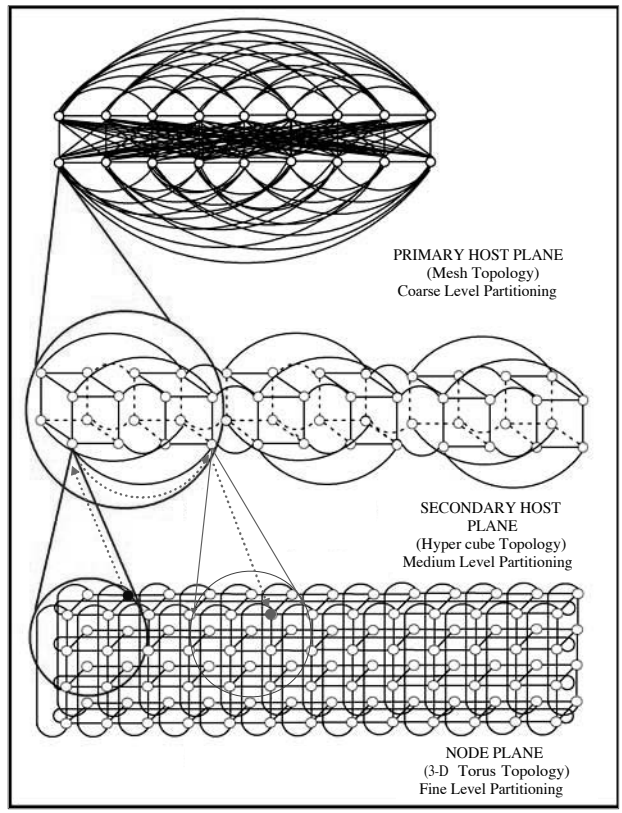


Figure 3: MIP Cluster architecture

3.2 SILICon Operating System (SILICOS)

In order to handle the complexity involved in simultaneous execution of multiple applications and to manage scheduling, memory, preprocessing of multiple application and I/O operations, we had highlighted the need for an efficient Operating System (OS) design. A software OS may not be proficient enough to exploit the power of the underlying MIP-paradigm based nodes as well as to meet the computational speeds of such nodes. In view, we have resorted to a hardware based operating system termed as SILICOS [10], the functionality of which is distributed across the primary and the secondary host planes. By designing suitable architectures for the primary and secondary host plane's processors to incorporate the functionalities of the cluster operating system. While the above mentioned core OS functionalities are implemented on a hardware platform in SILICOS, the rest are formulated as software libraries residing in primary and secondary hosts.

Besides managing the complexities associated with SMAPP, the primary and secondary host architectures will make the hardware based operating system more reliable and immune to software aging. Fault tolerance in SILICOS is realizable through on-demand network reconfiguration among the primary and secondary host planes. In the event that a specific resource of a particular host system fails, the load is transferred to the healthy sections of the host plane. System maintenance can also be undertaken with great ease when such a hierarchical host system is adopted.

When multiple applications are executed simultaneously, one of the main overheads on the OS is the ability to keep track of the traces of every application executed across all partitions of the cluster. To facilitate tracking, the Primary Host generates a generic tracking format which is interpreted and translated at various levels of the cluster. The control, data and instruction flow in the MIP cluster is portrayed in [fig. 10] along with the various format definitions, based on which the multiple applications are tracked and the data consistencies maintained and the corresponding field definitions can be found in [fig. 4]. This way the node resident OS tasks are relieved from the nodes to a great extent thereby improving their processing capability.

TABLE DEFINING THE VARIOUS FIELDS OF THE LIBRARY/DATA/INSTRUCTION PACKET

FIELD NAME	FIELD DEFINITION
APPLICATION ID	IDENTIFIES THE APPLICATION
LIBRARY ID	IDENTIFIES THE LIBRARY
SUB LIBRARY ID	IDENTIFIES THE SUB LIBRARY
COMPUTATION INVOLVED	NUMBER OF ALFU OPERATIONS
COMMUNICATION COMPLEXITY	NUMBER OF DEPENDENCIES ACROSS LIBRARIES WITH THE LIBRARY IDS
LIBRARY TYPE	DEFINES THE TYPE (MATRIX / GRAPH / SCALAR / VECTOR/ USER /SYSTEM /GLUE)
NODE ID	IDENTIFIES THE NODE
SECONDARY HOST	IDENTIFIES THE SECONDARY HOST
PRIMARY HOST	IDENTIFIES THE PRIMARY HOST
LOGICAL ADDRESS 1	GIVES THE (LOGICAL) STARTING ADDRESS FOR THE LIBRARY DATA IN THE GLOBAL HARD DISK
LOGICAL ADDRESS 2	GIVES THE (LOGICAL) ENDING ADDRESS FOR THE LIBRARY DATA IN THE GLOBAL HARD DISK
DATA PACKET ID	IDENTIFIES THE PACKET POSITION FROM THE DATA PACKETS OF THE LIBRARY
DATA/INSTRUCTION FIELD	FIELD WERE THE DATA / INSTRUCTION ARE EMBEDDED
TOTAL INSTRUCTION	NUMBER OF INSTRUCTIONS IN THE INSTRUCTION PACKET
PARALLEL	NUMBER OF PARALLEL INSTRUCTIONS IN THE INSTRUCTION PACKET
DEPENDENT	NUMBER OF DEPENDENT INSTRUCTIONS IN THE INSTRUCTION PACKET
PACKET SIZE	GIVES THE SIZE OF THE PACKET
FOOTER	IDENTIFIES THE END OF THE PACKET

Figure 4: Field definition for Library/Data/Instruction packet

3.3 PRIMARY HOST FUNCTIONAL ARCHITECTURE

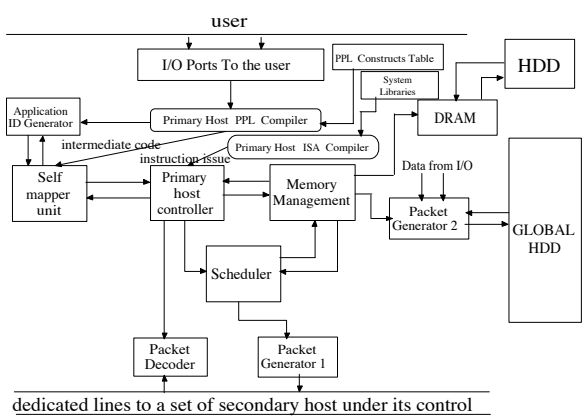


Figure 5: Functional Architecture of Primary host

The primary host has several hardware modules that forms the core kernel functionalities, thereby keeping the software complexity low. The primary host performs parallel decomposition of the multiple applications into their constituent sub-problems. This coarse-grain partitioning of multiple applications at the primary host plane requires close interaction among the host processors for uniform exchange of workload (sub-problems belonging to multiple applications). The interaction will facilitate distribution of multiple application load (application mix) across a set of secondary host processors under a primary host. The exchange of workloads across the host processors is achieved through Self Mapping process among the primary hosts. This necessitates that the primary hosts be completely connected. The primary host architecture is shown in (fig. 5).

The primary host PPL compiler generates the intermediate code, which is used by the self mapper unit to exchange the workloads associated with the different applications. Application ID generator in the host processor embeds the application ID onto the intermediate code format during load balancing and exchange across the primary hosts, which helps to keep track of the various application workloads assigned to each primary host.

The primary host scheduler balances the workload across its underlying secondary host processors and assigns them to the process scheduler, forwarded in the form of packets through the packet generator unit1. The data corresponding to the application mixes is formed into packets by the packet generator unit2 that embeds a header information which includes application ids, library and sub-library ids. These packets are stored in the global hard disk, and can be accessed later by the secondary host processor. The local compiler generates and issues the instructions to activate the primary host controller which triggers that corresponding functional modules. The design of memory management unit and packet generator are along conventional lines and the design details on self mapper and scheduler are available in [10].

The hardware aspects of SILICOS would contribute its fraction to increasing the power consumption of the cluster on the whole. This fraction of power consumption can be reduced by selectively shutting down certain functional modules after the completion of associated tasks, such as the scheduler and self-mapper in the primary host processors.

3.4 SECONDARY HOST FUNCTIONAL ARCHITECTURE

This forms the intermediate plane of the MIP cluster and majority of the Operating System functionalities are attributed to these hosts. The sub-problems issued from the primary host are further divided into libraries that the node can handle by the secondary host compiler. The functionalities of the secondary hosts include scheduling of libraries uniformly across the underlying MIP nodes and the corresponding data transfer, I/O handling, Interrupts and Exception handling, Data Packet Generation, Performance and Resources monitoring, facilitating data communication across the nodes when necessary and most importantly gathering the outputs and reforming data packets for issuing if necessary. The secondary plane performs the majority of the operating system functionalities for the cluster concerning the SMAPP execution[9]. Considering multiple applications being executed simultaneously across the nodes, the amount of interactions between the nodes might be expected to be higher. For such interactions between distant nodes, latency

becomes a huge issue and hence data from the remote node is routed through the secondary hosts. This way the number of hops drastically reduces thereby reducing latency. Also, employing a hypercube topology which accommodates more number of nearest neighbors for connecting the secondary hosts would further improve the latency issue. The secondary host architecture is shown in (fig. 6).

The process scheduler present in the secondary host processor receives format2, that contains the details of the application mix (assigned to it by the primary host scheduler) and gives the association between the instruction packet (application mix) and the data packet. The secondary host process scheduler now schedules these application mixes across the nodes in the form of packets (generated by packet generator) addressed by format1 to the PCOS (Primary Compiler On Silicon). While selective shutdown of the scheduler module is undertaken at appropriate instants of time in order to save on the power consumption, it can be re-instantiated as and when a process requirement occurs. The physical packet transfer from the global hard disk to respective on-board DRAM is assisted by memory management unit through a DMA (Direct Memory Access) transfer mechanism via dedicated set of lines.

The secondary host also governs the maintenance aspects of the cluster through the performance monitoring unit. The IO exception/interrupt module handles the interrupts and exceptions received from the distributed controller of the respective node[3]. Communication between the node and secondary host is facilitated by dedicated lines spawning from the PCOS/IO processor[3] of the node, to support high volumes of data transfer and maintenance information between the secondary host processor and the nodes. The local compiler is responsible for generating the instructions to trigger the various modules discussed above.

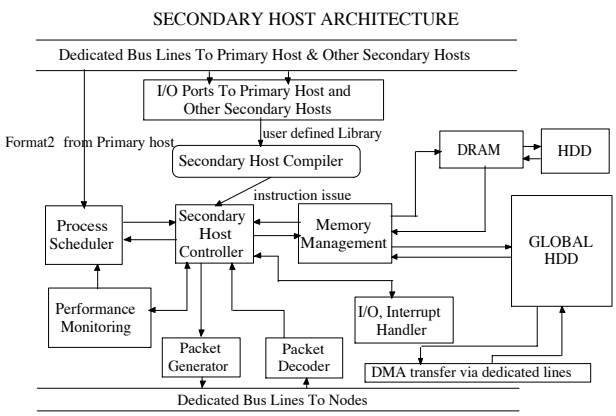


Figure 6: Functional Architecture of secondary host

3.5 FUSION CLUSTER

We had highlighted the detrimental aspects of wired cluster in the context of SMAPP execution. In order to tackle the afore mentioned intricacies involved in resorting to completely wired cluster, we incorporate a wireless element in the cluster interconnection network. While the wired interconnection provides us with increased bandwidth with lesser latency, wireless interconnection offers good power tradeoff.

The approach here is to exploit the advantages of either of them and evolve robust cluster architecture comprising of wired and wireless interconnections. The cluster is designed to have wireless links between set of hardwire interconnected nodes. Small clusters of nodes with hardwired interconnection form islands, which are grouped into hierarchical levels forming a single wired-wireless cluster and is extended to many levels. A hierarchical architecture combining the wired and the wireless network is shown in fig.7 forming the 'Fusion Cluster' [11,12]. There are two types of nodes present viz., Computing Nodes that form the Computing Population and the Input/Output Nodes that form the Transmitting/Recieving Population. The Wired Network Connection forms the basic block for the entire cluster. A single island is made of a cluster of sub-computing populations. This way, reconfiguration across hundreds of such islands will be effective to tackle the communication complexities encountered during simultaneous execution of multiple application. In a multiple level cluster environment Wired network connection is at the lowest level L0. Any island y of the wired network connection is identified by L0y. All the interconnection is hardwired at this level. A typical example of a L01 wired network is shown in fig. 7. The Wireless network Connection is wrapped over the Wired network Clusters or the Lower level wireless Clusters. All the interconnections across the islands are wireless. Any population y of the wireless network connection at level x is identified by Lxy.

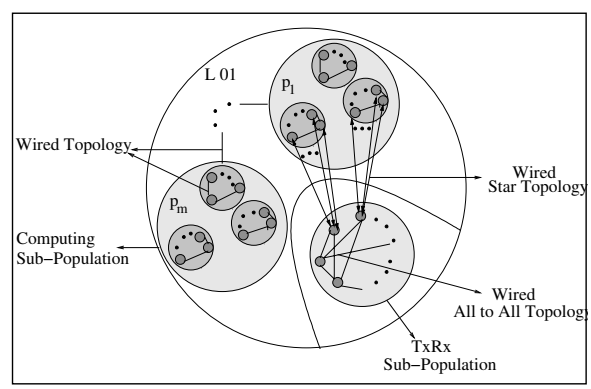


Figure 7: Fusion Cluster Model

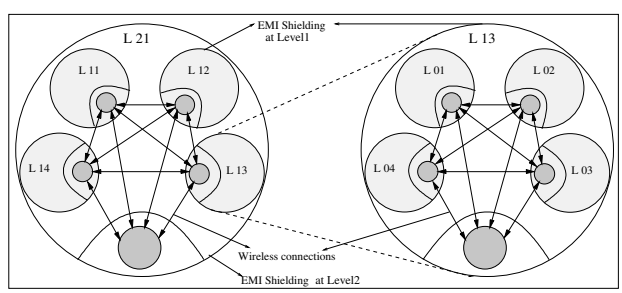


Figure 8: Fusion Cluster Architecture

The power consumed is expected to be much less in wireless transmission, when compared to that of wired networks. The field strength required for transmission across the fusion levels will be in the range of sub millivolts. Considerable amount of power is expected to be consumed by the interconnection network particularly when the cluster is scaled towards a million node while maintaining the high bandwidth requirement. For this type of wired interconnection network it will be inefficient to match the network topology with the varied communication requirements present in an application mix. On the other hand with wireless networks virtual reconfigurability is inherently achieved due to the broadcasting of data stream across the islands. This will greatly help in automatically matching the network topology with that of varied communication requirements present in an application mix. By varying the number of nodes within an island and the number of islands, performance(bandwidth)/power relation can be scaled to extreme levels. In a million node cluster the bandwidth requirement for the wireless communication across islands will become a major bottleneck, however this could overcome by using EMI shield (Electro Magnetic Shield) around each island as shown in the fig 9.This EMI shield would help achieve bandwidth reusability across any fusion levels. Considering the environment and the overall physical location and distance the noise due to external interference and hence managing higher signal to noise ratio may not be critical, provided sufficient care is taken to isolate high frequency switching activities that could affect the S/N ratio.

For million node clusters, where power will be a dominant parameter rather than performance highest reconfigurability matching with application requirements is attained by using wireless networks. In an application mix, some algorithms will tend to exploit the topology more efficiently in comparison with others making the mapping process complex. So, virtual reconfigurability in topology makes the mapping independent on the whole, to help achieve performance scalability. This may not be feasible in wired networks of large size as the power requirement for reconfiguration in wired networks is likely to be higher.

Currently wireless MIMO [13] technologies offer bandwidth in the range of gigabits and hence, are becoming attractive alternative which could be employed for wireless communication. Number of transmitters and receivers provided in each island facilitates MIMO technology based communication across the islands. Initial investigations show that MIMO technology could be a suitable match for the fusion cluster and can help achieve power and performance scalability at the cluster level.

Tracking of individual performance parameters will be complex and highly time consuming process in such a hybrid Fusion cluster model. Therefore we need to evolve new mapping strategies involving collective performance parameters. The mapping methodology to be adopted should be across node populations (a set of nodes) and not across individual nodes to simplify the mapping complexity. A population theory based stochastic Mapping is detailed in [14]. Performance to power ratio is scaled with the help of this fusion cluster and simultaneous execution can be effectively implemented integrating the hierarchically based host system.

3.6 PARALLEL PROGRAMMING PARADIGM

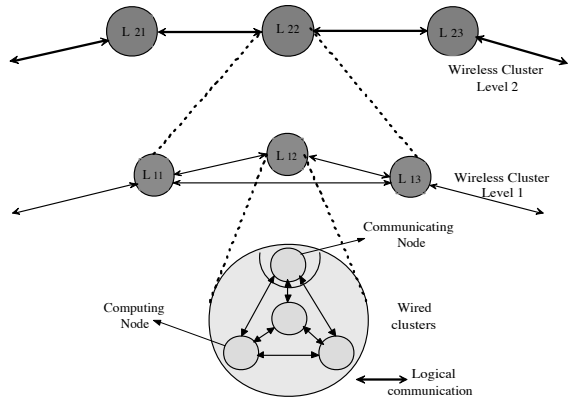


Figure 9: Fusion Cluster Architecture

With increasing complexity in programming the application, simplicity, but at the same time ,expressibility of these programming languages has become an important criterion for developing a parallel programming model. Also, the code turn around time should be very small and allow the programmer to represent any complex logic present in the application. The compilers developed for such languages must incorporate automatic parallelization of the complex applications. Hence, a design shift is required to solve all these problems.

The MIP PPL, an object oriented programming model being developed for the MIP Cluster handles all these challenges. The MIP Cluster has a set of predefined libraries written in MIP ALISA (Algorithm Level Instruction Set Architecture) [3] level assembly language. A vital aspect of the MIP Assembly language is, it does not include any addressing mode based instructions. Further, the MIP PPL allows the programmer to specify a set of general purpose operations present in the application as a set of libraries [15]. The PPL constructs represented at the ALISA level establish the link between all the libraries (based on the dependency), both user defined and application specific libraries. The language also allows the programmer to develop their own programming constructs if and when necessary. These properties of the MIP PPL eases the burden of the programmer and helps develop a smaller code space for a given set of applications and aids the user tune the language with respect to the applications. The issue of message passing across the nodes which is taken care by the SILICOS and the node IO/Network Processor, thereby allowing the programmer to represent the applications in a stand alone system is under investigation.

The compilation process is delegated to various levels in the MIP SCOC cluster, from the primary host processors to the PCOS/SCOS within the node. The primary host compiler handles the front end phases, namely lexical, syntax and semantics. The symbol table generated by the primary host compiler keeps track of the various libraries along with their respective application. The compiler differs from conventional ones in the concept of linked library generation and that the compiler itself is implemented in a hardware platform rather than a software based one.

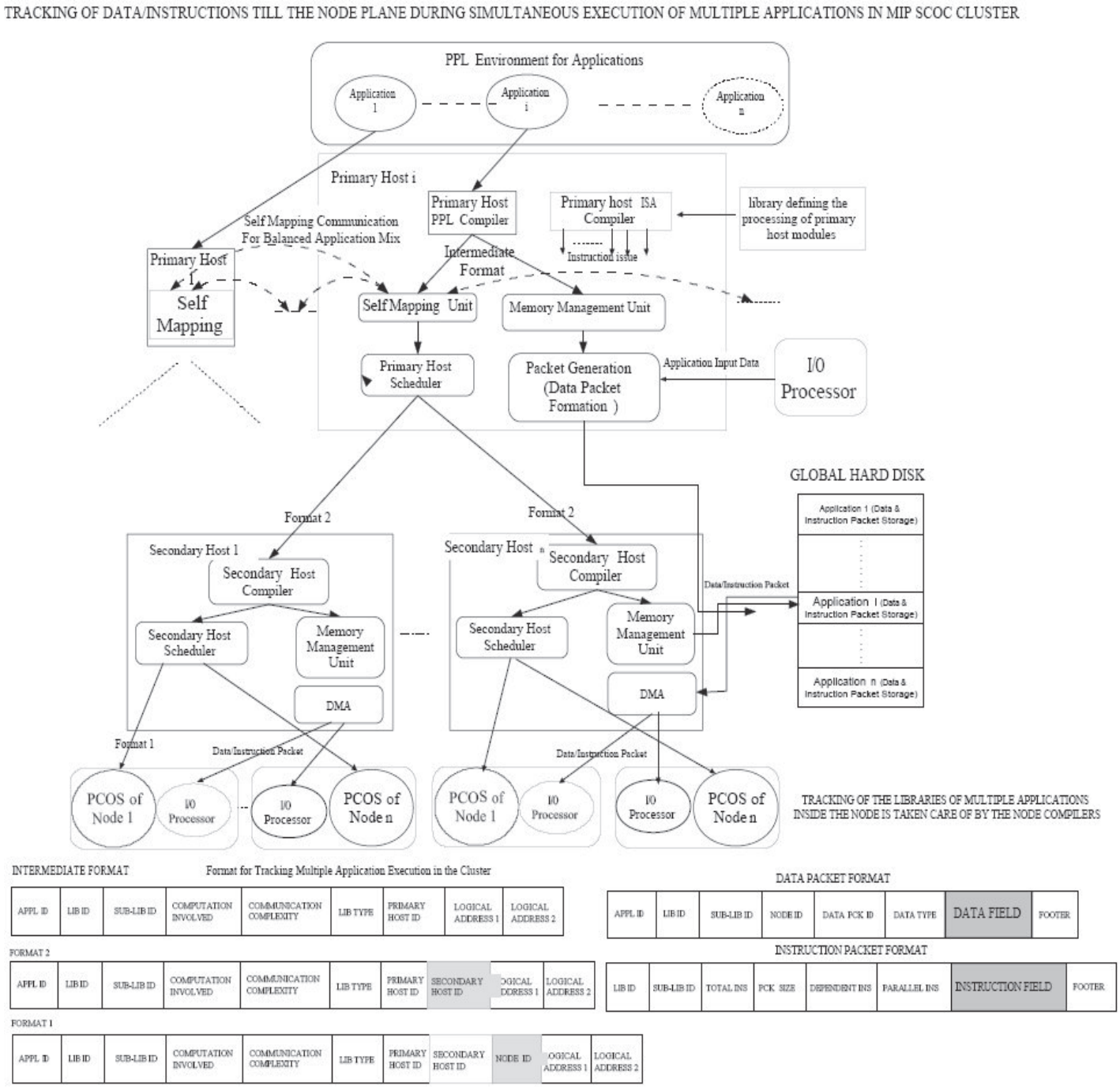
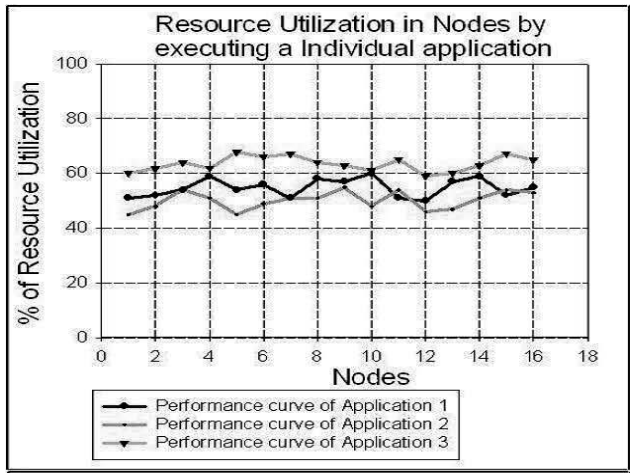
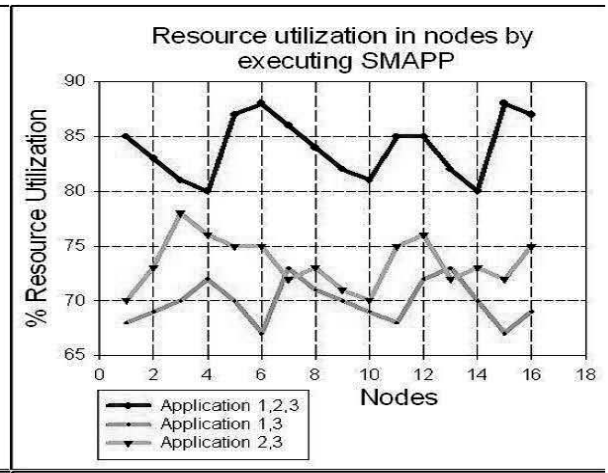
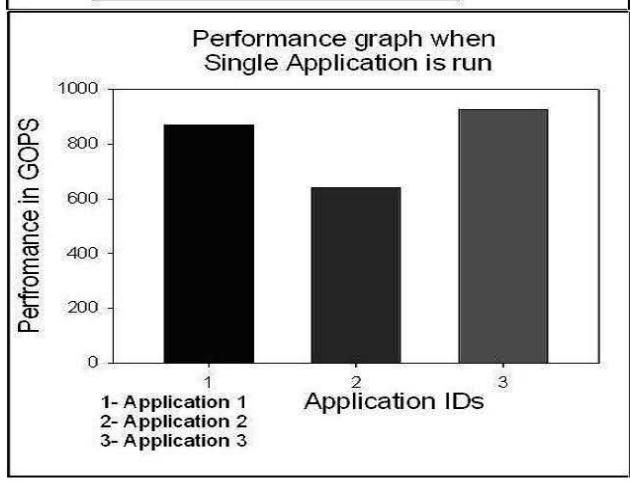
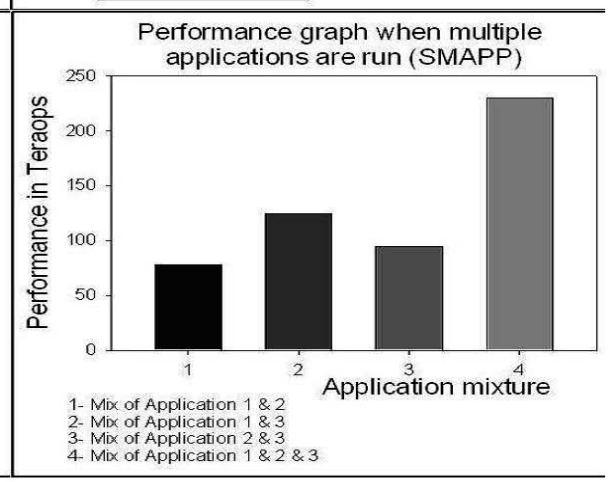


Figure 10: Control, Data & Instruction Flow in the MIP Cluster









BENSIM APPLICATION 1

LIBRARY NAME	SIZE	TYPE
FFT	24576	SCALAR
CONVEX HULL	1500	SCALAR
DFT	24480	VECTOR
QRD	256X256 & 3X(64X64)	MATRIX & VECTOR

BENSIM APPLICATION 2

LIBRARY NAME	SIZE	TYPE
DFS	30000	GRAPH THEORETIC
BITONIC	3000	VECTOR
MATRIX MULTIPLICATION	3X(256X256)	MATRIX
GLUE	3000	SCALAR

BENSIM APPLICATION 3

LIBRARY NAME	SIZE	TYPE
LUD	512X512	MATRIX & SCALAR
SVD	300X300	MATRIX
FFT	12288	SCALAR

MULTIPLE APPLICATION WORKLOAD DISTRIBUTED ACROSS 16 NODE MIP CLUSTER.
SNAPSHOT OF EXECUTION TRACE DURING SMAPP EXECUTION

NODE ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
BENSIM 1 (LIB IDs)	FFT 4096	FFT 4096 DFT 2048 QRD 256	CONVEX HULL 1000	DFT 2048	FFT 4096	CONVEX HULL 1000	FFT 4096	DFT 2048	DFT 2048	FFT(4096) CONVEXHUL(1000)	QRD 256	----------	FFT 4096 DFT 2048	DFT 2048 QRD 256	DFT 2048	DFT2048 QRD 256
BENSIM 2 (LIB IDs)	DFS 3000 GLUE 1000	DFS 3000	BITONIC 1000 GLUE 1000	GLUE 1000	DFS 3000	----	BITONIC 1000 GLUE 1000	------	DFS 3000	DFS 3000 BITONIC 1000	------	----------	DFS 3000	BITONIC 1000	DFS 3000	GLUE 1000
BENSIM 3 (LIB IDs)	SVD 52*52	-------	SVD 52*52	QRD 64*64	GLUE 2000 MATADD 64*64	MATMUL 64*64 SVD 52*52 MATSUB 64*64	------	MATINV 64*64 CHAINMATADD 64*64 QRD 64*64	-------	----------	-------	MATMUL(64*64) MATINVER64*64 CHAINMATADD (64*64) CROUT(64*64)	-------	-----	MATMUL 64*64) MATINVER 64*64	SVD 52*52

Figure 11: Simulation Graphs & BENSIM Workload Characterization

4. SIMULATION ANALYSIS

This section discusses the cluster level simulation done for analyzing the effectiveness of simultaneous multiple application execution in terms of cost effectiveness, cluster utilization and performance gain. BENSIM (BENchmark SIMulator for High Performance Computing Architecture) [16] models the workload characteristics of real-time high end applications. This pseudo-application model is developed taking into consideration both the computational and communication complexity associated with applications, to generate different workload traits of the applications intensive in terms of communication complexity or computational complexity or even strike a balance in terms of both the above mentioned parameters. All the simulation results presented in this section are extracted by running multiple pseudo-applications comprising of workload from three distinct application trace, on a 16 node emulated MIP cluster. The performance, resource utilization and the mapping spread of the pseud-applications across the cluster have been measured and a detailed analysis is presented to show the effectiveness of executing SMAPP in the MIP cluster.

Graph1 shows resource utilization profile of the 16 node MIP cluster when the three pseudo-applications governing different workload sets as shown in the table 1,2 and 3 are mapped individually. Such variation in resource utilization for the three pseudo-applications is prevalent mainly due to the distinct workload characteristic that can be associated with it, mapping strategies adopted by the hosts and the architectural characteristics of the MIP paradigm based node architecture as discussed in paper-I [3]. This resource utilization directly reflects in the performance of the pseudo-application when run on the cluster which as shown in graph3.

Graph 2 depicts the resource utilization profile of the cluster when the three pseudo-application are simultaneously mapped for execution. The variation in resource utilization during SMAPP execution is mainly attributed to the simultaneous multiple algorithm execution at the MIP paradigm based node architecture as discussed in paper-I [3], the cluster operating system and the flexibility provided by the cluster interconnection. This resource utilization directly reflects in the performance of the pseudo-application when run on the cluster which is shown in graph3.

5. Conclusion

In this paper, we have proposed a novel architectural model for high performance clusters. In the beginning we had indicated the need to develop more efficient model for future high performance application execution to pave way for cost sharing among . We primarily discussed a model for multiple applications execution simultaneously on all partitions of the cluster. We had delegated the cluster operating system role to a hierarchical host system developed to tackle the complexities (ranging from mapping of the multiple applications onto the cluster to tracking the execution of these applications on the node) posed by this execution strategy. To bolster the performance of the cluster, we had proposed a hybrid interconnection network - Fusion Cluster. The element of wireless inter-connectivity introduced new dimensions to the cluster, by lessening the communication overheads and most importantly by bringing in the feature of reconfigurability. In retrospect, we have illustrated a complete perspective of the proposed Supercomputing cluster in order to meet the immense performance requirements posed by high-end cluster computing applications. We have also presented an initial version of the simulator that has been developed for node architecture simulation and are currently developing an event driven simulation framework for simulating all the components of the MIP SCOC cluster in parallel environment. Such a cluster design having a strong node architecture foundation would help realize 'Cost-Effective Supercomputing' in its entirety while delivering enhanced cluster performance.

✠✠✠✠✠✠✠✠✠✠✠✠✠✠✠✠✠✠✠✠

The following former WARFT research trainees pursuing their Graduate program elsewere:
Prem Kumar Ramesh - ECE Dept, Iowa State University
Karthik Ganesan - ECE Dept, University of Texas Austin
Sivaramakrishnan - CS Dept, University of Houston
Viswanath Krishnamurthy - CS Dept, Iowa state University

✠✠✠✠✠✠✠✠✠✠✠✠✠✠✠✠✠✠✠✠

References

[1] Adolfy Hoisie et. al, "Performance Comparison Through Benchmarking and Modeling of Three Leading Supercomputers: Blue Gene/L, Red Storm, and Purple" in *Supercomputing Conference 2006 (SC '06)*

[2] Peter Hildebrand (Chair), Warren Wiscombe (Science Lead) et. al,"Earth Science Vision 2030 Predictive Pathways for a Sustainable Future" *NASA Working Group Report*

[3] Venkateswaran et. al , "Future Generation Supercomputers I: A Paradigm for Node Architecture", *to be submitted to ACM CAN Journal 2007*

[4] David H. Bailey, Allan Snavely, " Performance Modeling: Understanding the Present and Predicting the Future" in *Lawrence Berkeley National Laboratory (University of California) Year 2005*

[5] Guy E. Blelloch.,"NESL: A nested data-parallel language (version 2.6)" in *Technical Report CMU-CS-93-129, School of Computer Science, Carnegie Mellon University, April 1993.*

[6] Wolfgang Gellerich, Michael M. Gutzmann,"Massively Parallel Programming Languages: A Classification of Design Approaches" in *ISCA, 9th Int'l Conference on Parallel and Distributed Computing Systems*

[7] Paul Lu,"COOL: Parallel Programming Using C++", *MIT Press, 1996*

[8] Niranjan Kumar Soundararajan, "Hardware Compilation concept for the MIP S.C.O.C and Hierarchically-based Multiple Host System for the MIP cluster", *A Thesis Submitted to Waran Research Foundation 2002* www.warftindia.org/thesis/niranjan.pdf

[9] Karthik Ganesan, "Hierarchical Multihost based operating System for simultaneous Multiple application Execution on MIP SCOC Cluster" *A Thesis Submitted to Waran Research Foundation 2006* www.warftindia.org/thesis/gk.pdf

[10] Vinothkrishnan Elangovan, "Architecture of SILICon Operating System (SILICOS) for MIP SCOC Cluster" *A Thesis Proposal Submitted to Waran Research Foundation 2006* www.warftindia.org/thesis/evk.pdf

[11] Prem Kumar Ramesh, "Towards Multi-million Node Fusion Cluster Mapping Schedule: Employing Population Theory and Temperature Aware Load Balancing", *A Thesis Submitted to Waran Research Foundation 2006* www.warftindia.org/thesis/prem.pdf

[12] Sivaramakrishnan Nagarajan,"SMART Communication Library for Performance Scalability of Multi Million Node Fusion Cluster" *A Thesis Proposal Submitted to Waran Research Foundation 2005* www.warftindia.org/thesis/siva.pdf

[13] A Goldsmith, SA Jafar, N Jindal, S Vishwanath,"Capacity limits of MIMO channels" *Page 1. 684 IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 21, NO. 5 2003*

[14] Sudharsan Gunanathan,"Application of Non-Stationary Process and Population Theory towards Multi Million Node Cluster mapping" *A Thesis Proposal Submitted to Waran Research Foundation 2005* www.warftindia.org/thesis/sud.pdf

[15] Viswanath Krishnamurthy,"Library Design for MIP SCOC" *A Thesis Submitted to Waran Research Foundation 2006*, www.warftindia.org/thesis/vish.pdf

[16] "BENSIM: WARFT BENchmark SIMulator for High Performance Copmuting Architecture" www.warftindia.org/generic.php?cat=tool