A Survey of HPC Applications Appropriate for Execution on DSM and Grid Systems

Pankaj Bhambri¹, Dr. O.P. Gupta²

¹Department of Information Technology, Guru Nanak Dev Engineering College,
Ludhiana, Punjab, INDIA
pkbhambri@gndec.ac.in

²Department of Information Technology, Punjab Agriculture University,
Ludhiana, Punjab, INDIA
opgupta@pau.edu

Abstract—Distributed memory Beowulf clusters and Grids are gaining more and more interest as low cost parallel architectures are meeting high performance computing demands. With the rise of cheap commodity cluster solutions, Distributed Memory approach has inspired many research projects and has lead to significant advances in this area. Software DSM (SDSM) systems enable the use of shared memory programming approaches in loosely coupled environments. Shared memory is appealing because it isolates the programmer from the underlying communication scheme used to maintain consistency across the distributed system. On the other hand, modern organizations depend on network-attached personal desktop and mobile computer systems. A characteristic of these systems is that they are relatively resource rich (in terms of CPU power, memory, and disk capacity) but are utilized only for a fraction of the time during a day. Even during the time they are in use, their average utilization is much less than their peak capacity. Grid computing is basically coordinated resource sharing of these existing underutilized resources. These computing systems usually support embarrassingly parallel applications which consist of many instances of the same computation with their own data. The applications are usually involved with scientific problems, which need large amounts of processing capacity over long periods of time. In this paper we have surveyed various applications which can run on DSM and Grid environments individually. Based on the analysis of these applications we propose to identify applications which can be executed in an integrated model comprising of shared memory and grid-SMIG (Shared Memory Integrated with Grid). We are in the process of architecting and designing SMIG model and the applications we select will prove useful in testing this model.

Keywords—Distributed Memory, Grids, Clusters, Resource Sharing and SMIG Model.

I. INTRODUCTION

Distributed memory Beowulf clusters and Grids are gaining more and more interest as low cost parallel architectures are meeting high performance computing demands. With the rise of cheap commodity cluster solutions, Distributed Memory approach has inspired many research projects and has lead to significant advances in this area. Software DSM (SDSM) systems enable the use of shared memory programming approaches in loosely coupled environments [1,2,3,4,5,6,7]. Shared memory is appealing because it isolates the programmer from the underlying communication scheme used to maintain consistency across the distributed system.

On the other hand, modern organizations depend on network-attached personal desktop and mobile computer systems. A characteristic of these systems is that they are relatively resource rich (in terms of CPU power, memory, and disk capacity) but are utilized only for a fraction of the time during a day. Even during the time they are in use, their average utilization is much less than their peak capacity. Grid computing is basically coordinated resource sharing of these existing underutilized resources [8,9,10,11,12,13]. These computing systems usually support embarrassingly parallel applications which consist of many instances of the same computation with their own data. The applications are usually involved with scientific problems, which need large amounts of processing capacity over long periods of time.

In this paper we have surveyed various applications which can run on DSM and Grid environments individually. The paper is structured as follows. In section 2 we give a brief overview of DSM and grid computing followed by the types of applications which can be executed in both these environments individually. Based on the analysis of these applications we propose to identify applications which can be executed in an integrated
model comprising of shared memory and grid-SMIG (Shared Memory Integrated with Grid). Hence in section 4 we put forth a detailed comparison of DSM and grid computing and explain the types of applications which can be executed in a DSM based grid environment. We are in the process of architecting and designing SMIG model and the applications we select will prove useful in testing this model.

II. OVERVIEW OF DSM AND GRID COMPUTING

In this section we give a brief overview of DSM and grid computing paradigms. A DSM system, shown in figure 1 consists of a set of N interconnected processors with local memories, which communicate with each other using message passing paradigm.

The system poses a single global virtual memory image with memory accesses managed by the DSM software or hardware. Besides the memory hierarchy in each node the DSM organization adds another level to this hierarchy. The ratio of access costs to different levels in this extended memory hierarchy can be several orders of magnitude making its management performance critical. The goal of DSM system is to transparently manage the performance by either hardware or software mechanisms. Hardware DSM systems like DASH, Alewife and FLASH have tightly integrated node or memory controllers which connect processors to both memories and high speed switching networks].

One of the advantages of DSM is that it facilitates the transition from sequential to parallel programs. After identifying possible sources of parallelism, most of the data structures can be retained without change but synchronization variables need to be added to achieve a correct shared memory parallel program. DSM is a better way of inter process communication than message passing which relieves the application programmer from writing programming constructs. The major advantages of DSM include a simple abstraction, provides better performance in some applications and large memory space at no extra cost. Additionally in DSM systems both sending and receiving processes need not exist at the same time and process migration is easier in DSM implementation. Because of these advantages DSM has received attention from the research community as compared to message passing systems. The choice of DSM concept to cater to high end computations is a natural solution today.

There are basically three broad categories of grids as shown in figure 2 Information grid, Resource grid and Service grid. The Information grid delivers information on any kind of topic to any place in the world. Information can be retrieved via a telephone network or a modem just like plugging into the electrical power grid.

File sharing services like Gnutella are also part of today's Information grid. This file sharing service is set up by individuals who want to exchange files of Mp3 audio tracks, video films or software. The Resource grid provides mechanism for coordinated use of resources like computers, data archives, application services etc. The core idea is to provide easy, efficient and transparent access to any available resource irrespective of location. Resources may be computing, power, data storage, network bandwidth etc.

Resource grids can be further divided into Computational and Data grids. The first type of grid- Computational grid is the natural extension of large parallel and distributed systems, comprising
of available underutilized compute systems, and individual users who use a single point of contact with the grid to execute high end computations. Such a grid aggregates the processing power from a distributed collection of systems. This type of grid is primarily comprised of low powered computers with minimal application logic awareness and minimal storage capacity. The World Wide Grid (WWG) and NSF TeraGrid are examples of this model.

The second type of Resource grid is data grid. Data grids allow large datasets to be stored in repositories across a grid and moved about with the same ease like small public files. Data Grids deal with providing services and infrastructure for distributed data-intensive applications that need to access, transfer and modify massive datasets stored in distributed storage resources. Businesses interested in data grids typically have IT initiatives to expand data-mining abilities while maximizing the utilization of an existing storage infrastructure investment, and to reduce the complexity of data management. EU Data grid project is a typical example.

The Service grid delivers services and applications independent of their location, implementation, and hardware platform. Service Grids extend the notions of computational and data grids to provide transparent access to remote libraries and applications. They can be implemented using Web services acting as facades for remote services in conjunction with UDDI (Universal Description, Discovery, and Integration), proving location transparency.

III. OVERVIEW OF APPLICATIONS FOR HPC

Searching applications need not use a shared memory, since a search operation can run independently on parallelized data [14]. If the data is available then any one parallel thread will return the result. An important key to algorithm design is to use sorting as a basic building block, because once a set of items is sorted, many other problems become easy. Typical sorting applications for large data cannot be parallelized since there would be an inherent would execute faster if the data can be replicated on all processors.

Search preprocessing is perhaps the single most important application of sorting. This search enables a user to test whether an item is in a dictionary, once all keys are sorted. Sorting provides quick and easy solution to many applications extensively used on the internet. Bucketing is a very effective idea whenever we are confident that the distribution of data will be roughly uniform. Bucket sort is a sorting algorithm that works by partitioning an array into a finite number of buckets. Each bucket is then sorted individually, either using a different sorting algorithm, or by recursively applying the bucket sorting algorithm.

Applications suitable for utilizing available DSM include the obvious ones like sorting (Sort a list of names, Organize an MP3 library, Display Google PageRank results and List a specific news items in reverse chronological order). There are a few problems which become easy when sorted like finding the median, closest pair, Binary search in a database and duplicates in a mailing list. Other non obvious applications where sorting will be useful are Data compression, Computer graphics, Computational biology, Supply chain management, Book recommendations on Amazon and Load balancing on a parallel computer.

Grid-aware applications are basically categorized into the following five classes:

- Distributed computing: Very large problems needing lots of CPU and memory; eg. DIS, Stellar dynamics
- High throughput: Harness existing idle resources to increase aggregate throughput; eg. Chip design, parameter studies, cryptographic problems
- On demand: Remote resources integrated with local computation, for bounded time. Eg. Medical instrumentation, cloud detection
- Data intensive: Synthesis of new information from many or large data sources. Eg. Sky survey, physics data, data assimilation
Collaborative: Support communication of collaborative work between multiple participants. E.g. Collaborative design, data exploration, education.

A new emerging class of application that can benefit from the Grid is SOA- Service-Oriented Computing (e.g. application service provider and the users' QoS- Quality of Services requirements driven access to remote software and hardware resources. Vendors such as IBM and Oracle are positioning grid computing as an important cog in their computing strategies. Grid computing is driving a new evolution in industries such as the bio-medical field, financial modeling, oil exploration, motion picture animation and many others. Open Grid Services Architecture [OGSA] is an evolving standard with significant industry support.

IV. SELECTION OF APPLICATIONS FOR DSM BASED GRID

In this section we first give a brief comparison of DSM and grid computing paradigms followed by a discussion of applications which can use an integrated DSM based grid environment. Based on this comparison it is evident that both computing paradigms are complementary to each other. So for applications which can be parallelized and independently executed, grid computing is a good option. It carries out computations using existing underutilized CPUs while users continue to do their jobs. DSM uses dedicated machines and shares memory to carry out high end computations. Hence we have selected applications suitable for DSM to run in an integrated DSM based grid environment. These are embarrassingly parallel applications like matrix multiplication, and sorting algorithms like merge sort and bucket sort. In future we propose to build the integrated environment and test the selected applications in this environment.

V. CONCLUSION

DSM and grid computing paradigms optimize resources and carry out high end computations. They are active areas of research and have been individually explored for research and commercial applications. There are very few examples of combined implementations. As a part of our research we propose to build a DSM based grid model. Hence in this paper we have focused on identifying applications which are suitable for to form our test suite. We found that applications needing large memory for computations can exploit DSM paradigm and these will also be suitable for testing our model. Currently we are in the process of designing and building this model.

REFERENCES

[7] Lu, Dwarkadas, Cox, Zwaenpoel; Message Passing Versus Distributed Shared Memory on Network of Workstations.

Table I: JUMP DSM VERSUS SLINC

<table>
<thead>
<tr>
<th>Criteria</th>
<th>DSM</th>
<th>Grid computing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motivation</td>
<td>Need for memory larger than a single memory available on a machine</td>
<td>Need for CPU power or storage more than a machine</td>
</tr>
<tr>
<td>Basic objective</td>
<td>Logical abstraction of physically distributed memory</td>
<td>Exploit underutilized/ idle resources ( CPU power and storage)</td>
</tr>
</tbody>
</table>
### How it works

<table>
<thead>
<tr>
<th>How it works</th>
<th>Single global address map managed through either hardware or software</th>
<th>Execute job on volunteer machines in background with low priority</th>
</tr>
</thead>
</table>

### Applications for which it can be used

<table>
<thead>
<tr>
<th>Applications for which it can be used</th>
<th>Caters to HPC applications which need to share memory</th>
<th>Caters to HPC applications which can be parallelized and executed independently</th>
</tr>
</thead>
</table>

### Advantages- benefits

<table>
<thead>
<tr>
<th>Advantages- benefits</th>
<th>Optimization of memory in a cluster</th>
<th>Optimization of CPU power in a cluster</th>
</tr>
</thead>
</table>

### Limitations

<table>
<thead>
<tr>
<th>Limitations</th>
<th>Needs dedicated machines for application execution; needs even number of machines including master</th>
<th>Not suitable for applications which can be parallelized but need to share data</th>
</tr>
</thead>
</table>

### Features of volunteers

<table>
<thead>
<tr>
<th>Features of volunteers</th>
<th>No concept of volunteer by dedicated hosts</th>
<th>Allows sharing of memory and CPU power on a single machine between science application and volunteer</th>
</tr>
</thead>
</table>