AN APPROACH TO PROCESSOR ALLOCATION SCHEDULING FOR HETEROGENEOUS MULTI-CLUSTER SYSTEM

Jyoti Bala¹, Mohit Angurala²
¹M.Tech Scholar, Golden College of Engineering & Technology, Gurdaspur(PB)
²Assistant Professor, Dept of Computer Science & Engineering, Golden College of Engineering & Technology, Gurdaspur(PB)

Abstract
The requirement of high computational tasks which requires large job sizes has raised, due to this the demand of supercomputer increases which could handle large computational requirements. The fact-finding is made on evolving an Adaptive Scheduling Policy for Non-dedicated Heterogeneous multi-cluster system. This adaptive scheduling policy which will raise the performance of multi-cluster compared to existing policies Multi-cluster are systems are used now a days to get high performance and higher system utilizations. Two major factors for effective scheduling namely, Job scheduling and Processor allocation have been discussed and accordingly the algorithm is designed. The basis of the main issues of algorithm in heterogeneous clusters such as Resource Fragmentation and Speed Heterogeneity. The scheduling policies are there to allocate the processor to each job and adaptive scheduling is one of the scheduling which deals with these jobs that require suitable processor. In this research is base on Adaptive Scheduling Policy for Heterogeneous Multi-cluster system. Furthermore, the performance of the proposed algorithm approach dealing with Multi-site execution is compared with the case of Multi-pool configuration in which different clusters in the computing grid are viewed as different processor pools and each job must be allocated to a set of processors belong to exactly one of these pools. No jobs can continuously use processors from various pools. So, if the first job in queue could not fit into any single site in the grid, it would have to wait.

Indexing terms/Keywords
Multi-Cluster System, Processor Allocation, Job Scheduling

SUBJECT CLASSIFICATION
Parallel Processing

TYPE (METHOD/APPROACH)
Research Paper

INTRODUCTION
A Parallel Processing in build up more than one processing units or Processor to execute a program by some interconnection network and software require to build processing units work with each other. In Parallel computer program are more difficult as compared to sequence, main reason that concurrency that means many calculations done continuously which solved complex problem into smaller ones by parallel policy.

Advantages of Parallel Processing-
- Program execution time fast.
- Get highly throughput means how many number of processors completed per unit time.
- It’s have a to handle large amount of data.

Disadvantages of Parallel Processing-
- Need more hardware requirements.
- More heat and power consumption
- It is not good for low power and mobile devices.

2. MULTI-CLUSTER
A combination of two or more independent but interconnected server is called a cluster. Multi-cluster means more than one cluster, we will receive job under Parallel Processing in heterogeneous (means do not similar cluster) Multi-cluster System. Multi-cluster systems are used now days to design to be very fast and raised system utilizations and make up large job sizes (in terms of processing unit) able to permit jobs to use processors in multiple clusters simultaneously that is to employ processor co-allocation. Two main parts for effective scheduling is Job scheduling and Processor allocation. The job scheduling strategy are there to assign the processor to each job and processor scheduling is one of the scheduling that assign with these jobs which need suitable processor. This adaptive scheduling strategy will raise the performance of Multi-cluster
Types of clusters

1. **High availability clusters**: They are used to provide the services every time to the end-user even when the system component stops working. They improve the availability of the cluster approach. They are used for the databases, mail, file, print and web or application servers.

2. **Load balancing clusters**: Load balancing clusters works via getting all workload to arrive by mono or more load balancing front ends, which are then shared to a group of back end servers.

3. **High performance clusters**: They are most commonly used in scientific computing. They are hired to give an maximum performance via dividing a computational task over lots of different nodes in the cluster.

4. **Grid clusters**: They are hired for workloads that contains large amount of packets of jobs or independent jobs which don’t require the sharing of data between the jobs in the computation process. Grids focus on managing the distribution of jobs to computer that will do the job in parallel with the left of the grid cluster.

It may be categorized as two types one is homogeneous and other is Heterogeneous:

- **Homogeneous Multi-cluster**: A homogeneous multi-cluster is the one in which the configuration of all the participating nodes are same whether in terms of speed, memory etc.

- **Heterogeneous Multi-cluster**: A heterogeneous cluster is the one in which the configuration of all the participating nodes are different whether in terms of speed, memory etc. In practice the clusters are heterogeneous in nature. The Processors that use different features are called heterogeneous. It is feasible only if it work well improvement in all working nodes. Performance can be expressed in terms of improved their response time, waiting time, execution time and co-allocation time.

3. **PARALLEL JOB SCHEDULING POLICIES**

These can be categorized broadly into two parts Space Sharing and Time Sharing.

1. **Space Sharing**: Its policy gives the requested resources to the job until the job is completely executed. It has low overhead and high parallel efficiencies. It is further of three types Static scheduling, dynamic scheduling and Adaptive scheduling.

1.1 **Static scheduling**: Scheduling is performed at compile time, and then jobs are allocated to individual processors before execution. The allocation remains same during the execution. Most static scheduling methods needs to gather information needed for the scheduler to make further decisions, according to what kind of resources are necessary for job.

1.2 **Dynamic scheduling**: Scheduling is performed at run time and it can support dynamic load balancing and fault tolerance. Even when job is in its execution state, Load balancing could share the load between nodes

1.3 **Adaptive Scheduling**: It combines static and dynamic scheduling. The part of static scheduling is to collect information required for the scheduler to make decisions on what kinds of resources are compulsory. And the other part of dynamic scheduling is load balancing of job’s loop.

2. **Time Sharing**: This technique is employed to ensure that the time on a processor is divided into many discrete intervals or time slots which are assigned to unique jobs. The size of time slots depends on the cost of scheduling.

3.1 **Job Request Types**

A job request that needs co-allocation consists of multiple components, namely sub-job requests. Each sub-job request is associated with a single cluster; thereby sometimes such a job is also called as multi-cluster job.

In classifies a job request into one of the following types:

- **Ordered**: Each sub-job defines the number of processors it requires and identify the cluster from which those processors will be allocated.

- **Unordered**: Each sub-job specifies the number of processors it requires but does not identify the cluster from which those processors will be allocated.
**Flexible**: The job specifies the total number of processors required collectively from multiple clusters, instead of the number of processors for each sub-job.

**Total**: This job specifies the total number of processors required from a large single cluster, instead of multi clusters. On this way, a job of this type is also called a single-cluster job.

### 4. PROCESSOR ALLOCATION

Processor allocation is concerned with assignment of required number of processors for incoming job from available sites. Site or Cluster is used exchange in this thesis. The processor allocation problem can be treated as how the system handles the first job in the waiting queue. When the first job in queue cannot fit into any single site in a computing grid, different schedulers handle the situation in different ways. The following are two possibilities which are commonly used.

#### 4.1 Single Site selection: This is the simplest allocation policy. Each job must be allocated to a set of processors belonged to exactly one cluster site.

**4.1.1 First-fit**: Allocate the first site found in the searching process that has enough available processors for the waiting job.

**4.1.2 Best-fit**: Allocate the site with the minimum available processors which is enough for the waiting job.

**4.1.3 Worst-fit**: Allocate the site with the maximum available processors that can fulfil the requirement of the waiting job.

**4.1.4 Median-fit**: Among all the sites that have enough processors for the waiting job, allocate the site with the median number of available processors.

**4.1.5 Random-fit**: Randomly allocate a site among all the sites that have enough available processors for the waiting job. The system first search the computing grid to find all those sites that have enough available processors, and then pick up one randomly.

**4.1.6 Most-fit**: This policy tries to allocate the job to a cluster which produces a leftover processor distribution, leading to the almost all number of instant subsequent job allocations. It requires a more complex searching process, involving simulated allocation activities, to determine the target cluster. For every cluster which has sufficient processors for the waiting job, the system performs a series of simulated activities, based on the best-fit policy, to compute how many immediate subsequent allocations can follow the allocation decision. After each cluster has been checked, the system selects the cluster with the largest number of immediate subsequent allocations to perform current job allocation.

**4.2 Cross-site execution (co-allocation)**: A scheduler with this policy would try to run a parallel job across several sites if there is no single site having enough free processors. However, a parallel job might take much longer execution time when running across site boundaries. This is because the speed and bandwidth of inter-site network is usually much slower and less than those of intra-site network. The overhead caused by communication and data migration is considered as an increase of the job's runtime. The enlarged runtime for multi-site parallel execution is calculated by multiplying the original single-site runtime with a specified system parameter, slowdown. The following evaluates three different heuristic methods for cross site allocation:

**4.2.1 Fixed order**: The system maintains a list of all the sites in the computing grid. The order of that list is fixed. Every time the system pick up a site from the list according to the stable order until the total number of available processors on all selected sites is larger than or equal to the requirement of the waiting parallel job.

**4.2.2 Larger first**: The system first sorts the sites in the computing grid into decreasing order according to the number of available processors. Then, the system repeatedly picks up a site according to the sorted order until the total number of available processors on all selected sites is larger than or equal to the requirement of the waiting parallel job.

**4.2.3 Smaller first**: In contrast to larger-first, the system first sorts the sites in the computing grid into increasing order according to the number of available processors. Then, the system repeatedly picks up a site according to the sorted order until the total number of available processors on all selected sites is larger than or equal to the requirement of the waiting parallel job.

### 3. LITERATURE SURVEY

In this we make a short overview of most popular Scheduling in Multi-cluster methods. But we mainly focus on the functional aspects of the methods. A brief summary of the literature is given below:

**E. Deelman et al (2014) [1]** research on explain the design, build out and advancement of the Pegasus Workflow Management System, which depict abstract workflow descriptions onto distributed computing architecture in order to perform its intended functions and scalable workflow execution. Huang et al (2013,2010) [3,4,5] research regarding adaptive processor allocation based on moldable property of parallel jobs for heterogeneous system and studied about speed of heterogeneous system at inter cluster level in which nodes on different cluster may have different compute speed. But nodes on same site have same speed. Harpreet kaur et al (2013) [2] work is based on speed heterogeneity but only at inter cluster level that is computing speed among the cluster different but within cluster the speed of processors is same. The study related heterogeneous multi-cluster system in which co-allocation is used. A scheduling algorithm is feasible if average response time of all incoming jobs improves. Pinal et al (2013) [7] study regarding cloud computing each task requires to be executed by available resource to get less waiting time and increase utilization of resource. Simulation under various conditions and get better results with less of waiting time, processing time. Po-chi Shih et al (2013) [8] research regarding performance of TLA by simulations with different workloads and system configurations in terms of average turnaround time. Simulations results indicate have up to 87% performance improvement. Nancy et al (2013) [6] research...
regarding different sites (cluster) of processor have different computing speed and number of processor within and among the clusters. Basic processing Unit (BPU) is used to compute the speed of processor. The scheduling process used techniques of job selection, sit selection and processor selection into single algorithm with improving mean response time and utilization in a heterogeneous multi-cluster system. R.S. Chang et al (2012) [9] the author proposed the adaptive scoring job scheduling algorithm to schedule independent set of tasks composing of compute intensive and data intensive means short time jobs. When new jobs obtain by job scheduler, it gives them to the most suitable resources according to their cluster score. Local and global update are used to acquire the currently status (cluster score) of resources in Grid environment. R Garg et al (2015) [10] work is based on Grid computing enables large scale resource sharing and collaboration of solving science and engineering applications. Different methods have been used with static and dynamic method. A tasks schedule to resources before execution time and later schedule them at time of execution. It deals with heterogeneous dynamic grid environment, major approach involves initial static scheduling, resources monitoring and rescheduling with getting minimum execution time for workflow application.

5. CONCLUSION AND FUTURE SCOPE

In this paper, a survey on various scheduling methods has been done. From the survey, it has been found problems in existing approaches and how these problems will be improved in heterogeneous multi-cluster. Job scheduling decides the job sequence for processor allocation whereas processor allocation is concerned with assignment of processors for incoming job. Speed heterogeneity is a major factor that affects the overall performance of the system.

It has also been give evidence by the simulations that the cross-site allocation results in lower Average Waiting times and Mean Response time in almost all the cases than allowing the jobs to wait for the processor as in case of Multi-pool configuration.

6. REFERENCES


