CHAPTER 5

SMARTER MULTI QUEUE JOB SCHEDULING (SMQS): PRIORITY BASED ENERGY EFFICIENT SCHEDULING TECHNIQUE FOR OPTIMAL JOB PROCESSING

5.1 OVERVIEW

This Chapter discusses the development of proposed priority based energy efficient scheduling technique for optimal job processing for energy saving in cloud environment. A heuristic job scheduling which is named as smarter multi queue job scheduling algorithm (SMQS) developed to provide the effective job scheduling and to provide optimal cloud user job processing.

5.2 INTRODUCTION

The architecture, core components, scheduling strategy and strengths of proposed smarter multi queue scheduling algorithm (SMQS) are analyzed in detail. The proposed algorithm provides an intelligent computational technique in which the scheduler itself adapts the optimized job scheduling scheme from the existing one according to the scenario. The algorithm works by diving user jobs in two queues and then doing select user jobs from both queues simultaneously. The proposed technique will achieve a high degree of job scheduling by reducing energy computation and limiting execution time in cloud computing environment.

5.3 PROPOSED JOB SCHEDULING APPROACH FOR CLOUD ENVIRONMENT

A Smarter Multi queue Job Scheduling algorithm is proposed to schedule user jobs and to recognize the status of all virtual machines in real time environment in cloud environment. Based on received user jobs, the algorithm actively separate the unscheduled jobs to multiple queues then formulate merge pattern by merging the one job from the small job queue with the one job contained in large job queue to avoid a long idle time by jobs in the queues. Further scheduler dispatches the merge task set for execution.

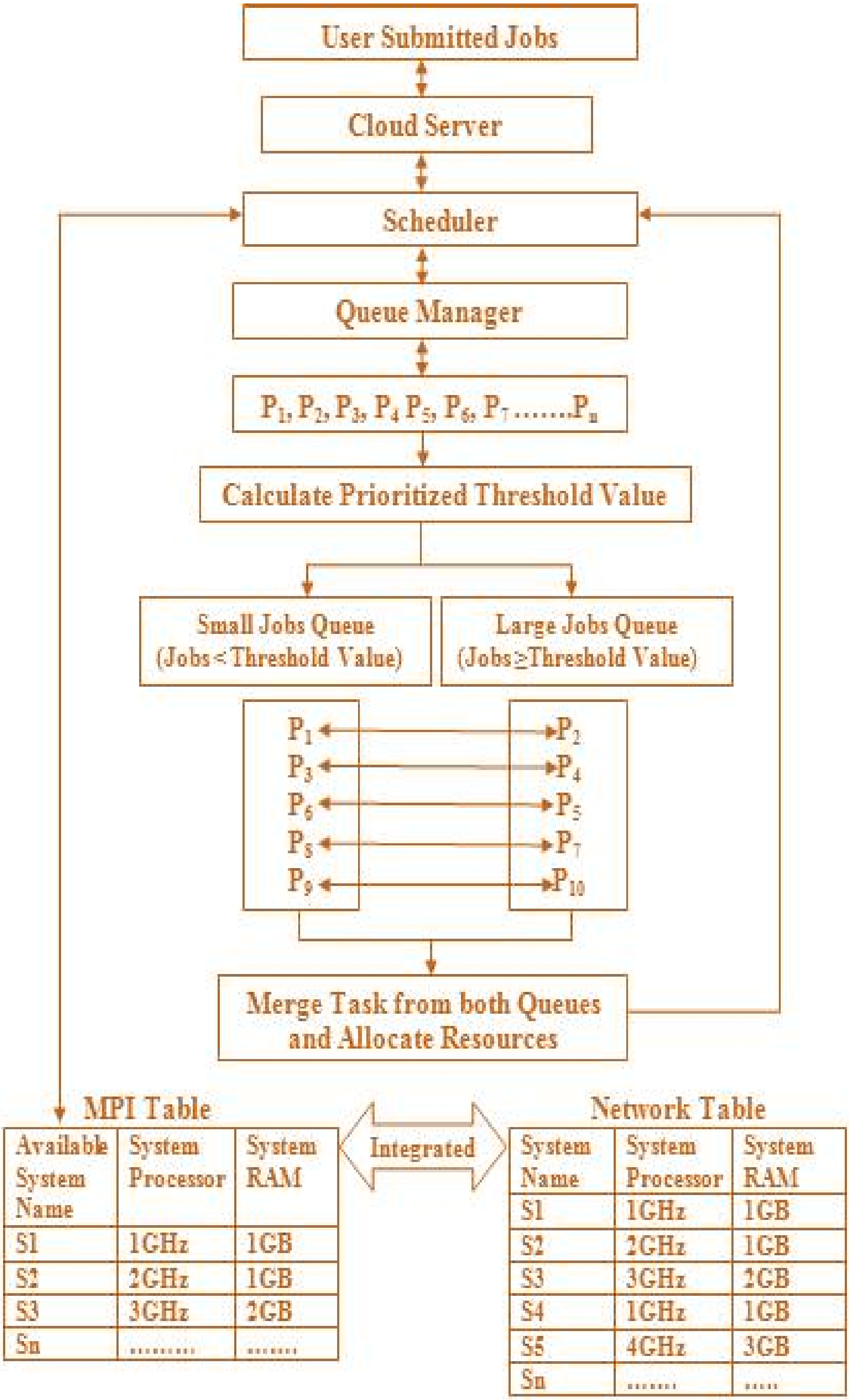
The core components which are used in the architecture are discussed below:

(a) **Network Table**: The cloud environments dispense heterogeneous resources with varying process or speed so a network table constitutes a set of heterogeneous resources. Resources are usually pooled to cater to several consumers with dynamic allocation and assignment on the basis of consumers demand. A multi-tenant model will be followed in the process.

(b) **MPI Table**: The scheduler uses an MPI table that's synchronized to it so that the jobs forwarded by scheduler are scheduled via this table which further works in collaboration with the network. The Message passing interface table acknowledges all the vacant system from network and maintains the array of all these available systems for processing.

(c) **Scheduler**: The main purpose of implementation of scheduler is to utilize all the network resources efficiently and cater to multiple individuals in an effective and optimized manner. Using an MPI table in the suggested smarter MQS technique, it will ensure that all the merge task set allocated to virtual system and all resources should be utilized in an optimal way.

The computational architecture of working of proposed smarter multi queue algorithm is shown below in Figure 5.1:


User Submitted Jobs
Cloud Server
Scheduler
Queue Manager
$P_1, P_2, P_3, P_4 P_5, P_6, P_7 \ldots\ldots P_n$
Calculate Prioritized Threshold Value
Small Jobs Queue (Jobs < Threshold Value)
Large Jobs Queue (Jobs ≥ Threshold Value)
P_1
P_3
P_6
P_8
P_9
P_2
P_4
P_5
P_7
P_{10}
Merge Task from both Queues and Allocate Resources
Integrated

MPI Table

Available System Name	System Processor	System RAM
S1	1GHz	1GB
S2	2GHz	1GB
S3	3GHz	2GB
Sn	....	....

Network Table

System Name	System Processor	System RAM
S1	1GHz	1GB
S2	2GHz	1GB
S3	3GHz	2GB
S4	1GHz	1GB
S5	4GHz	3GB
Sn	....	.....

Figure 5.1 Architecture of Smarter MQS for Cloud Computing

5.4 SMARTER MQS ALGORITHM WORKING

The above Figure 5.1 shows the architecture of Smarter MQS algorithm for scheduling user jobs, the proposed algorithm working consists of the following steps:

Step 1: The Scheduler in collaboration queue manager accepts the unscheduled job from multiple users for processing. It further provides resources for the processing of these jobs. The Queue Manager directs the scheduler about the output of the processed jobs. The Scheduler is directly linked to MPI table.

Step 2: According to the projected energy required by each process a prioritized threshold value is calculated. Now keeping this value as base value the jobs are categorized into small and large queue jobs.

Step 3: As the jobs has been categorized in different queues so as to reduce the idle job waiting time, merging pattern will be formed by merging of user processes from both the queues. The pattern can have multiple set of merge task. By doing this each process is allocated to resource in the desirable amount of time and very less probability that deadlines for several process completions will be missed.

Step 4: A single set of merge user task from merge pattern is picked up and will be allocated by scheduler to the available system looking at the list maintained by MPI Table.

5.5 STRENGTHS OF SMARTER MQS

The major benefits provided by this Smarter MQS Architecture are as follow:

(a) **Lesser Switching Time:** It is the time taken for switching from one process to another. Unlike popular belief, it isn't a one-step task as it includes several tasks that have to be done by administration like saving and loading of registers, memory maps and so on. The switching time is higher in usual algorithms due to long waiting queues but in our proposed smarter MQS algorithm, as it is dealing with two queues instead of three queues (as used in MQS architecture build in [65]) so the switching time, as a result, will be significantly reduced.

(b) **Reduce Job Completion Time**: It is the amount of time a job takes to execute completely. As the scheduler of our discussed technique combines the jobs from both the queues we are using, resulting in all the jobs being executed without waiting for turns, thereby it will be reducing job completion time to some extent. Also for the jobs the time that is spent waiting in queues contributes to job idle waiting time, the proposed approach discussed in this paper will fixes this problem too. Traditional scheduling methods also cause low throughput due to high response times.

(c) **Easy Jobs Allocation**: There is large number of user who submits their jobs in cloud environment so the scheduler aims to control degree of multiprogramming. It selects ready jobs from the queue and submits them to various virtual systems for processing. As in our proposed technique, scheduler is directly synchronized with the MPI table which maintain the list of available virtual systems as a result no PING command will be issued by the scheduler side to the Network(contains set of virtual systems) because it selects the system from the list maintained by MPI table. By this method, the workload on the behalf of scheduler gets reduced and so the energy.

(d) **Minimize Starvation**: As the scheduling policy framed in proposed smarter multi level queue scheduling aims to cover the indefinitely postponement of the user process so no user process will go in the freeze state. The probabilities of getting virtual systems for each type of user jobs are same so resources will be allocated to user process in a fair manner.

5.6 SCHEDULING STRATEGY FOR SMARTER MQS

In the high performance computing community scheduling of parallel jobs on cluster and supercomputer proved to be an active research topic [89]. Job Scheduling algorithm in Cloud is subdivided in two main parts: Batch Mode Heuristic Scheduling Algorithm (BMHA) other is On-Line Mode Heuristic Scheduling Algorithm. The basic difference between the two is that in BMHA as the job arrived they are queued and after fix period of time the scheduling algorithm start while in other as the jobs arrive to system they get queued for processing. FCFS, Round Robin, Min-Min, Max-Min algorithm are the example of BMHA scheduling algorithm [90]. The study of scheduling on cloud is considered as one of the quality study done by the researchers although many studies explained the scheduling on cloud in simple and clear form, there still exist diverse definitions [91].

Job scheduling algorithms is one of the most challenging hypothetical problems in the cloud computing domain area [37]. The scheduling problem aim to solve in this paper can be stated as:

In order to map all the Jobs (J) of Workflow(W) to a set of Cloud services (S). S = $\{s_1, s_2, . . . , s_x, . . . , s_z\}$ is the set of available computing cloud services with varied processing capability. The Scheduler utilizes this S in such a way, that the estimated total energy and time incurred for processing the user jobs should be minimized.

While Make span(W) ≤ Deadline ;Total Energy (W) Consumed and Cost < Previous Efficient MQS Technique (used in [65]) alongside satisfying all the constraints).

5.6 SMARTER MQS FLOWCHART

The summarized working process for proposed Smarter MQS approach described as:

Step1: After receiving the cloud user jobs the cloud scheduler calculates the priority threshold value. Now, on basis of this value the categorization of user job is done in small job and large job queues.

Step2: Further the processes are merged from both queues forming a merge pattern containing number of merge user processes set for execution.

Step3: Scheduler by referring to MPI table maps the merged user processes for the execution to the available systems in the cloud environment

The below mentioned Figure 5.2 flow chart shows the whole scheduling process of proposed smarter multi queue scheduling algorithm:

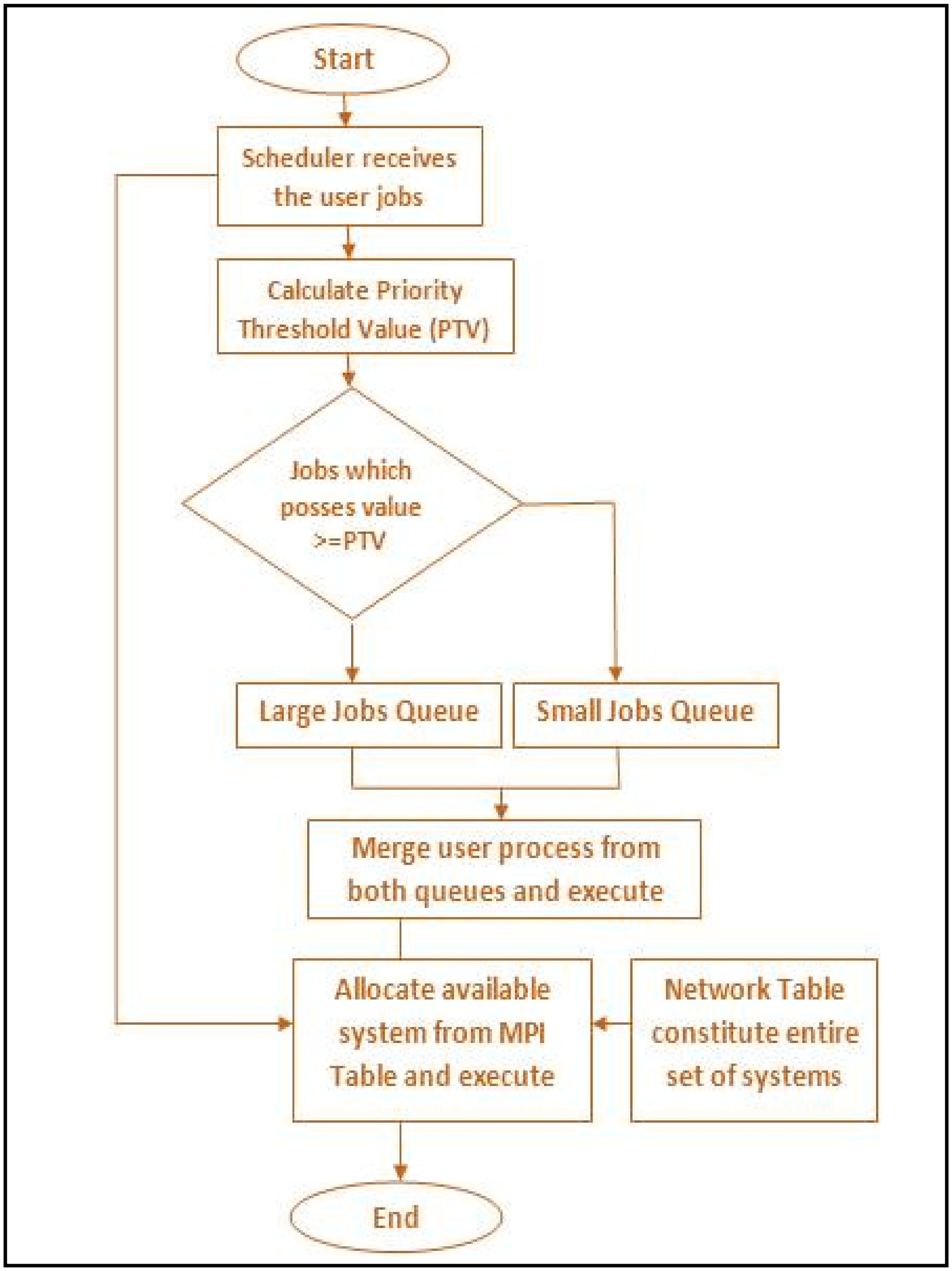

Start
Scheduler receives the user jobs
Calculate Priority Threshold Value (PTV)
Jobs which posses value >=PTV
Large Jobs Queue
Small Jobs Queue
Merge user process from both queues and execute
Allocate available system from MPI Table and execute
Network Table constitute entire set of systems
End

Figure 5.2 Flowchart of Proposed Smarter MQS Scheduling Algorithm

5.7 SUMMARY

In this Chapter the formulation our proposed heuristic algorithm SMQS is represented in detail. In order to achieve lesser energy consumption , the Smarter MQS scheduling algorithm works on the principal two queues and also reducing the job completion time by providing optimal strategy for execution and distribution of the cloud user jobs over available resources in cloud environment. The results of proposed Smarter MQS algorithm in comparison with Efficient MQS are analyzed under various parameters settings in Chapter 6.