Process Scheduling:A Mechanism for Job Scheduling

B. Chaitanya, Tulasi Ananda

Dept of CSE,
R.V College of Engineering
Bangalore, India

Abstract

In computing scheduling can be defined as a mechanism of assigning tasks or jobs to resources based on certain constraints. These constraints are categorized to various mechanisms based on the requirements of the system. Scheduling is the process of running jobs more effectively through efficient hardware utilization, maximizing the throughput and minimizing the waiting and response times. This self-executing method automatically schedules jobs on the systems and assigns resources in the network to gain maximum hardware usage thereby utilizing the resources efficiently. The main goal of this methodology is to make maximum use of the hardware resources and achieve better performance. Schedulers act as load balancing mechanisms when dealing with multiple resources. Our proposed algorithm deals with efficient launching of jobs on hardware resources based on hardware configuration with respective job type.

Keywords—self-executing, Job scheduling, load balancing and Throughput.

I. INTRODUCTION

In today's world Distributed computing [1] can be defined as a system whose components are located at various networks, where communication and coordination can be happened by message passing among the components. With effective interaction the components achieve better throughput [2] and performance. But it does not suffice to barely meet the goals today rather achieve efficient computation with optimal resource utilization. This can be achieved through advantageous resource allocation to the various jobs. Job scheduling [3],[4] is the process of scheduling the jobs on the system based on the resource availability, resource requirements and other constraints. There are several ways to perform the scheduling of jobs. The major objective is to achieve performance enhancement and improving efficiency. A few of the ways of job scheduling are described below.

II. EASE OF USE

A. First Come First Serve(FCFS)

Here In [6], the jobs are assigned to resources based on first come first server basis. It doesn't follow any hardware or job-based scheduling. It blindly assigns task without any optimal utilization of resources. It leads to "convoy effect" which means if a huge job is assigned first, the rest of the queued processes must wait until the first job finishes executing. Thus, reducing the efficiency of the system and increased waiting time of the rest of the processes.

B. Shortest Job Scheduling Mechansim

SJF [7] is a type of job scheduling mechanisms which executes jobs based on minimum execution time scenario. It minimizes the average turnaround time and gives averaged waiting time. If two jobs have same execution time/priority, it then executes based on shortest job time.

C. Heuristic Scheduling Algorithm(HSA)

In [2] Yang proposes a method focused on allocation of virtualized network. Here they have developed an algorithm which mainly focus to achieve desired performance. They have developed V-heuristics scheduling algorithm for allocation of virtualized network and computing resources under user's constraint which is applied into a service-oriented resource broker for jobs scheduling. Finally, the algorithm achieves performance in utilizing the resources highest throughput, minimizing processing time.

D. Completely Fair Scheduler(CFS)

The main idea of CFS [8] is to provide Fair Scheduling by allocating processor time to jobs/tasks. That is, each Processor is given a fair amount of processing time to tasks. The advantage of CFS is Group Scheduling which helps in allocating equal amount of CPU resources to all the groups in the system.

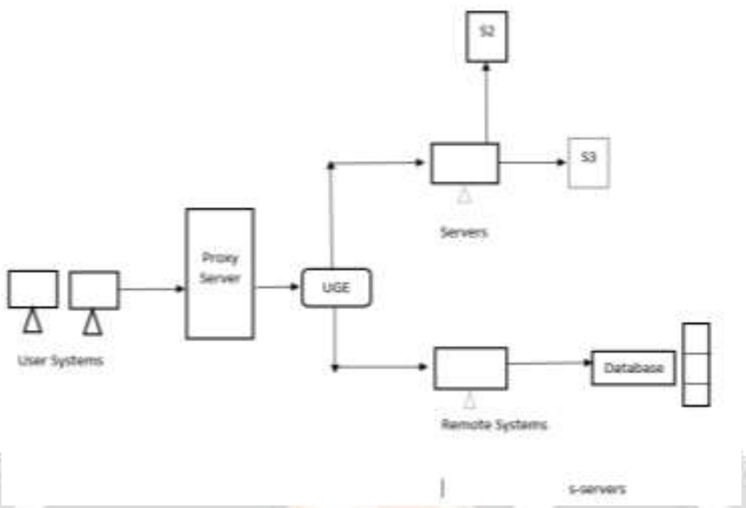

Fig. 1. System Architecture

III. Methodology

The above figure depicts the Architecture of the system in detail. The user systems are generally connected to the remote system by using proxy. The remote systems can be accessed using SSH. Once the connection is established, the user can gain access to remote machines. The remote machines store all the data in its database. The stored data can be viewed in logs. Each server will be of varied size and capacity. Whenever the algorithm executes, it launches jobs on all the servers based on limits and constraints of the algorithm. The jobs will be relaunched until they get executed on the servers and achieve desired results. The framework contains various components like application, connector, Linux operating systems in a network, cluster. The application layer communicates with the underlying hardware resources using ports. Each job will be launched on the underlying hardware based on hardware requirement. The application layer contains different component's which interact with various applications and will be helpful in displaying various parameters from the database. The jobs will be launched on the respective clusters and results are stored and displayed. A cluster can be defined as a set of individual systems connected together.

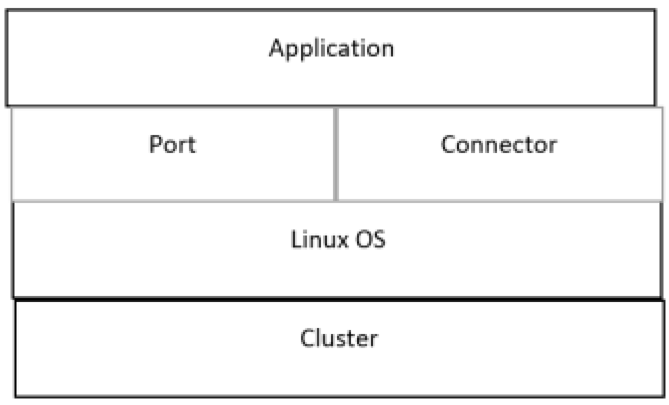

Fig. 2. Framework

IV. Implementation

The below figure shows the detailed process of how the jobs are scheduled

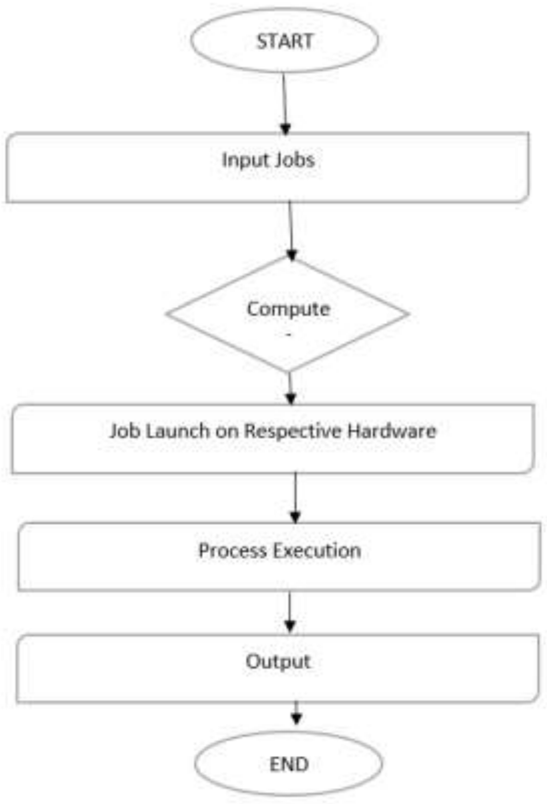

Fig. 3. Flow Chart

A. *Steps*

Step 1: Run the algorithm in the environment by passing required input parameters.
Step 2: The parameters like input path and type of data
Step 3: The algorithm retrieves parameters based on input data from the location in the network.
Step 4: Jobs are scheduled to respective machines based on obtained data
Step 5: Respective jobs are scheduled, and results are stored in to the Database.
Step 6: The data can be displayed in a browser.

RESULTS

The below graph shows the performance of the system as processors increase with respective to time.

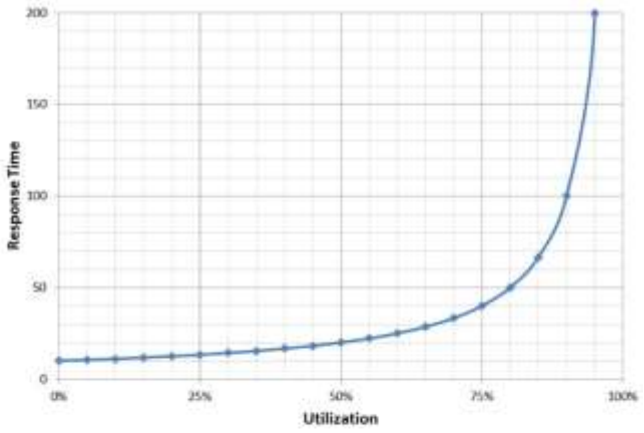

Fig. 4. Performance of Utilization vs Response Time

As the number of processors increase in the network the time of execution decreases

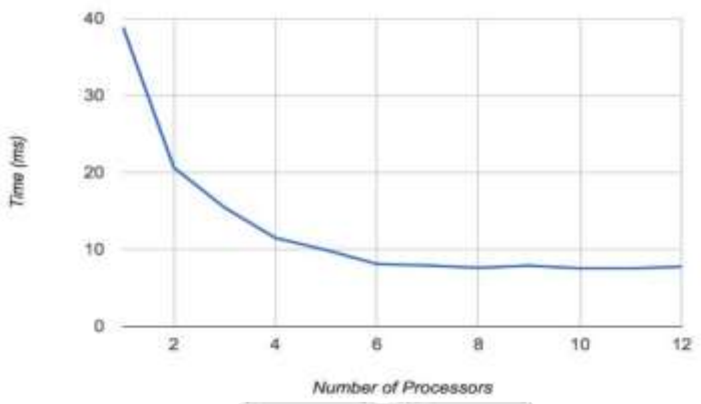

Fig. 5. Performace of Processors vs Time.

The above graph shows the utilization and response time. The response time increases as the utilization of resources increases.t is because each process will be handling various tasks and queuing of resources at hardware increases which leads to lower response time.

CONCLUSION

Our proposed self-executing procedure performs operations by utilizing the resources efficiently. It launches various jobs on different servers. It achieves superior throughput and efficiency when compared to other systems. Development of this kind of methodology where the code ensures job scheduling under resource constraints for enhanced utilization and performance.

FUTURE WORK

By making the algorithm more dynamic will be helpful in scheduling jobs with dynamic constraints. This can be made possible by integrating the machine learning and intelligence into the system through which the user can input the requirements as needed into the job scheduling process using bots. This will be the future scope of our work

REFERENCES

[1] Moin Hasan ; Major Singh Goraya "A framework for priority based task execution in the distributed computing environment" International Conference on Signal Processing, Computing and Control December 2016..

[2] Yichao Yang, Yanbo Zhou, Zhili Sun, Haitham Cruickshank" Heuristic Scheduling Algorithms for Allocation of Virtualized Network and Computing Resources" Journal of Software Engineering and Applications, vol.6, No.1-13 January 2013.

[3] A Study On Linux Kernel Scheduler Version 2.6.32 Author: Thang Minh Le

[4] Http://www.inf.ed.ac.uk/teaching/courses/os/coursework/lcpusched- fullpage-2x1.pdf

[5] Zhang Qi et al., "Workload-aware load balancing for clustered web servers", IEEE Transactions on Parallel and Distributed Systems, vol. 16.3, pp. 219-233, Febraury 2005.

[6] Tanenbaum Je., Vudhall A.Operacionnye sistemy. Razrabotka I realizacija. Saint-Peterburg, Piter, 2007.

[7] Rohan R. Kabugade1 , S. S Dhotre2 Improvement of Queue Management for Real Time Task in Operating System International Journal of Research in Advent Technology, Vol.2, No.12, December2014 E-ISSN: 2321-9637.

[8] Hap Cao.Liqiang He Performance evaluation of completely fair schedule for multicore processor system. International Conference on Electrical and Control Engineering doi: 10.1109/ICECENG.2011.6057478 February2011.