**International Journal of Emerging Technology and Advanced Engineering**
**Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 7, Issue 4, April 2017)**

# A Review on Job Scheduling Algorithms for Load Balancing

Monika Joon[1], Dr. Neetu Sharma[2]

*[1]M.Tech Scholar, [2]A.P., CSE Dept, Ganga Institute of Technology & Management, Kablana, Jhajjar, Haryana, Maharashi Dayanand University, Rohtak, Haryana, India*

***Abstract: -*** **With the increase in the time factor, as per service providers are also increasing but the pipelines are still narrow through which the communication takes place from one client to the server machine. Now if discuss about the increasing load it happens often that one system which is using the resources is free and another is loaded with the heavy task. Need to have taken into the consideration; have applied a solid queue technique which solves the problem of scheduling of tasks or jobs. But the problem is that the current scenario does not specify anything about the jobs priority. If a job is of high priority it will have to wait until and unless a job which is getting executed gets over. This may lead to a delay which is definitely going to affect the final result. In this paper we study existing technique of for scheduling and carry out objective for proposed work.**

***Keyword:*** **scheduling algorithm, processor, load balancing, FCFS, Priority**

## I. Introduction

The FCFS-based algorithms are restricted in selecting jobs based on their general arrival order. In order for a job allocation scheme to efficiently utilize the independently locatable resources of the K-resource system, it must be free to select any job based on matching all of the jobs' resource requirements with the available system resources. As an example, consider the JMS state depicted. The job allocation scheme must map the six jobs in the job queue to a two-resource system with 16 CPUs and 32 Bytes of memory. The CPU and memory requirements of each job are specified. Assume that the order in the job queue represents the order of arrival and that each job requires the same amount of execution time t. Under these assumptions, a job allocation scheme would select a set of jobs for execution during scheduling epoch eye. The number of epochs required to schedule all jobs in the job queue is used to compare different job allocation schemes. The jobs allocated to each scheduling epoch for FCFS, FCFS/FF, and an unconstrained job allocation scheme (UNC). The UNC scheme is free to select any job in the job queue for allocation during the current epoch. Although this is a contrived example, it illustrates the basic awes of FCFS-based job allocation schemes and the potential of less restrictive job allocation schemes.

The FCFS allocation scheme allocates jobs 0 and 1 in the first scheduling epoch but then cannot allocate job 2, due to the total CPU requirement of the three jobs being greater than the system provides. FCFS/FF overcomes this aw by skipping job 2 and scheduling jobs 4 and 5 in the first epoch. However, it then must schedule jobs 2 and 3 in separate epochs as there are no other jobs available to backfill in each of these epochs. Finally, the optimal UNC algorithm was smart enough to not schedule jobs 0 and 1 in the same epoch. Instead it finds two job subsets which exactly match the machine configuration. As a result, the unrestricted job allocation scheme requires fewer scheduling epochs to complete all jobs.

## II. General Goals

*Fairness:* Fairness is important under all circumstances. A scheduler makes sure that each process gets its fair share of the CPU and no process can suffer indefinite postponement. Note that giving equivalent or equal time is not fair. Think of safety control and payroll at a nuclear plant.

*Policy Enforcement:* The scheduler has to make sure that system's policy is enforced. For example, if the local policy is safety then the safety control processes must be able to run whenever they want to, even if it means delay in payroll processes.

*Efficiency:* scheduler should keep the system (or in particular CPU) busy cent percent of the time when possible. If the CPU and all the Input/output devices can be kept running all the time, more work gets done per second than if some components are idle.

*Response Time:* A scheduler should minimize the response time for interactive user.

*Turnaround:* A scheduler should minimize the time batch users must wait for an output.

*Throughput:* A scheduler should maximize the number of jobs processed per unit time.

## III. Literature Survey

There are various research papers on load balancing that shows different technique for balancing load in a simple network or distributed network.

Load balancing is the most important issue. To study about load balancing some research papers are given below.

Masoud Nosrati Ronak Karimi Mehdi Hariri [1] proposed in this paper to get into task scheduling in operating systems. Main methods and techniques of scheduling are presented and described in brief. So, with a short introduction containing the description of long-term, medium-term, short term and dispatcher scheduling, main concepts are illustrated. Investigated methods are: FIFO, shortest Job First, fixed-priority pre-emptive scheduling, round-robin scheduling and multilevel feedback queue. These methods are compared in conclusion.

According to this [2] research Grid computing enlarge with computing platform which is collection of heterogeneous computing resources connected by a network across dynamic and geographically dispersed organization to form a distributed high performance computing infrastructure. Grid computing solves the complex computing problems amongst multiple machines. Grid computing solves the large scale computational demands in a high performance computing environment. The main emphasis in the grid computing is given to the resource management and the job scheduler .The goal of the job scheduler is to maximize the resource utilization and minimize the processing time of the jobs.

The researchers explain a new approach for CPU scheduling algorithms which can be used to improve the performance of CPU in real time operating system [3]. The new approach of CPU Scheduling algorithm is based on the integration of round-robin, SJF scheduling algorithm. Priority is calculated on the basis SJF and time quantum. It retains the advantage of simple round robin in reducing starvation and also integrates the advantage of priority scheduling. The proposed algorithm also implements the concept of SJF and time quantum by assigning new priorities to the processes.

A research team presents the problem of real-time scheduling spans a broad spectrum of algorithms from simple uniprocessor to highly sophisticated multiprocessor scheduling algorithms [4]. In this paper, they study the characteristics and constraints of real-time tasks which should be scheduled to be executed. Analysis methods and the concept of optimality criteria, which leads to design appropriate scheduling algorithms, will also be addressed.

The main objective of this paper [5] is to introduce a new CPU algorithm called A Novel CPU Scheduling Algorithm which acts as both preemptive and non-preemptive based on the arrival time.

The prosed algorithm helps to improve the CPU efficiency in real time uni-processor-multi programming operating system. CPU Scheduling is the basis of multi-programmed operating system.

Computing systems serve many purposes; they cannot imagine a world without them. So it gives us the ability to do tasks at great speeds and it opens a new world of possibilities that they are just starting to explore [6]. One of the most important factors of computing systems is the ability to run several processes at once, or at least a good scheduler gives that impression. So to achieve this, the scheduler causes the system to switch quickly between processes.

A group of some researcher [7] has research on Builders of real-time systems often use priority scheduling in their systems without considering alternatives. This paper examines one alternative, pre-run-time scheduling, and show that when it can be applied it has significant advantages when compared to priority scheduling schemes.

Some scientists [8] has described the primary objective of this paper is to develop an enhanced approach for existing Priority Scheduling algorithm. The proposed approach greatly helps in minimizing the Response time thereby reducing the waiting time for lowest priority processes. Operating system handles a variety of functions such as process management, resource allocation, memory management, networking, file management etc. In all these tasks Resource allocation is of utmost importance.

In year 2005, a research [9] has provided information regarding while it is widely believed that preempt ability is a necessary requirement for developing real-time software, there are additional costs involved with preemptive scheduling, as compared to non-preemptive scheduling. Furthermore, in the context of fixed-priority scheduling, feasibility of a task set with non-preemptive scheduling does not imply feasibility with preemptive scheduling (and vice-versa).

## IV. RESEARCH METHODOLOGY

*Step 1-* Data is loaded on Microsoft Excel Sheet that works backend and accessed by our project.

*Step 2-* Use frontend .net accesses that data for implementation of algorithm.

*Step 3-* FCFS concept applies on data known as increasing algorithm and get results.

*Step 4-* For better performance of new algorithm, FCFS concept is used with priority scheduling.

*Step 5*- First task execute using concept of increasing method and then concept of priority applied.

*Step 5*- All task exclude first task will implement on basis of priority queue.

*Step 6*- FCFS and Priority implemented and got results.

*Step 7*- The result should be better than existing FCFS algorithm.

## V. Objective

Our objective includes the following:

- Our first objective is to design a task and scheduler system
- Our second objective is to design the increasing time algorithm for scheduling
- Our third objective is to implement a hybrid structure of FCFS and priority algorithm.

## VI. Conclusion

It is expected that the time for task execution will be reduced if there will be implementing the FCFS along with the priority queue concept. Performance of systems is improved with the combination of FCFS and priority queue. In general, configuration of system is fixed but in this project creates three systems according our requirement and applied task on these for execution. FCFS and priority concept is failure to achieve best performance if implemented separately. But a combined approach will make the execution fast and improve performance.

## REFERENCES

[1] Masoud Nosrati Ronak Karimi Mehdi Hariri, “Task Scheduling Algorithms Introduction” World Applied Programming, Vol. (2), Issue (6), June 2012

[2] Pinky Rosemarry, Ravinder Singh, Payal Singhal and Dilip Sisodia, “Grouping Based Job Scheduling Algorithm Using Priority Queue and Hybrid Algorithm In Grid Computing” International Journal of Grid Computing & Applications (IJGCA) Vol.3, No.4, December 2012

[3] Neeraj Kumar, Nirvikar, “Performance Improvement Using CPU Scheduling Algorithm-SRT”, International Journal of Emerging Trends & Technology in Computer Science, Volume 2, Issue 2, March – April 2013

[4] Arezou Mohammadi and Selim G. Akl, “Scheduling Algorithms for Real-Time Systems”, School of Computing, Queen’s University, Canada

[5] Sukumar Babu Bandarupalli1, Neelima Priyanka Nutulapati, Prof. Dr. P. Suresh Varma, “A Novel CPU Scheduling Algorithm-Preemptive & Non-Preemptive” International Journal of Modern Engineering Research, Vol.2, Issue.6, Nov-Dec. 2012

[6] Deepali Maste, Leena Ragha and Nilesh Marathe, “Intelligent Dynamic Time Quantum Allocation in MLFQ Scheduling” International Journal of Information and Computation Technology, Volume 3, Number 4 2013

[7] Jia Xu, David Lorge Parnas," Priority Scheduling Versus Pre-Run-Time Scheduling" The International Journal of Time-Critical Computing Systems, 18, July 2000

[8] Siddharth Tyagi, Sudheer Choudhary, Akshant Poonia, “Enhanced Priority Scheduling Algorithm” International Journal of Emerging Technology and Advanced Engineering, Volume 2, Issue 10, October 2012

[9] Yun Wang and Manas Saksena, “Scheduling Fixed-Priority Tasks with Preemption Threshold" RTCSA’99, Hong Kong December 2005