MULTI PRIORITY BASED SCHEDULING ALGORITHM FOR QUEUING SYSTEMS

Md. Imran. R. N[1], Ms. R. AKILA[2]
Department of computer science &Engineering
B.S.Abdur Rahman University, Vandalur
Email: rnmdimran@gmail.com[1], niceakila@bsauniv.ac.in[2]

Abstract

Mean waiting time is of a greater concern wherever queuing system is involved. Priority Scheduling is the general method used to reduce the mean waiting time of higher priority processes. However this method just focuses on favouring higher priority processes leaving the lower priority processes out of consideration and forcing them to starve for a longer period resulting in higher average mean time. The proposed system focuses on reducing the average mean time by favouring the higher priority processes and also making sure that the lower priority processes are not being made to starve. This is done using a dynamic multi priority based Scheduling Algorithm having Earliest Deadline First (EDF) algorithm as a base. The reason EDF is taken as a base as it achieves a better balance among priority classes by considering the timing constraints of the processes as well compared to other algorithms. Cloud is developing day by day and faces many challenges. One of them is scheduling. Scheduling refers to a set of policies to control the order of work to be performed by a computer system. This presents a Generalized Priority algorithm for efficient execution of task and comparison with FCFS and Round Robin Scheduling .Algorithm is tested in and result shows that it gives better performance compared to other traditional scheduling algorithm.

1. INTRODUCTION

1.1.1 Priority Scheduling

Processes are scheduled based on certain priorities decided. Some of them are "Shortest job First", "Earliest Deadline First", "First Come First Serve" etc. Processes to be scheduled are placed in some queue before starting to execute. Based on the priorities decided, Processes in the queue are sent to the processor to be processed and once the queue is completed with no processes left in its bin the execution is completed.

1.1.2 Multi-Priority Scheduling

There are situations where single priority assigned may be of lesser impact .In these situation, multiple priorities come of a greater use. Giving more than one priority to a single waiting queue makes it a multiple priority scheduling as more than one priority is given to a single waiting queue.

1.2 CLOUD SCHEDULING

Cloud computing comes in focus development of grid computing, virtualization and web technologies. Cloud computing is an internet based computing that delivers Infrastructure as a service (IaaS), Platform as a service (PaaS) and Software as services (SaaS).

The main goal of the proposed protocol is to improve the utilization of servers allocated to the jobs along with the following other concepts.

- To process the job having higher priority.
- Improve the resource utilization.

- Minimizes the completion time (make span) of MapReduce jobs
- Minimizing the waiting time
- Minimizing the switching time
- All these properties are proposed by very few scheduling algorithms and mechanisms of cloud computing. These regions are always irregular and that is the reason it is important to schedule jobs in cloud computing.

1.3 ROUND ROBIN SCHEDULING

Each process gets a small unit of CPU time (time quantum).
Usually 10-100 ms. After quantum expires, the process is pre-empted and added to the end of the ready queue. Suppose N processes in ready queue and time quantum is Q ms. Each process gets 1/N of the CPU time, in chunks of at most Q ms.

1.4 SHORTEST JOB FIRST

Response time suffers infinite, performance is the same as FIFO too small, throughput suffers and percentage overhead grows. If there were 3 compilations to echo each keystroke! In practice, need to balance short-job performance and long-job throughput. Typical time slice is10ms-100ms.

1.5 EARLY DEADLINE FIRST

EDF achieves best waiting times for higher priorities in lower to moderate loads and while only being number of times more than static priority algorithms in high loads However, for the lowest priority classes, it achieves comparable waiting times to Round-Robin and First-Come-First-Served in low to moderate loads and achieves waiting times only twice the amount of Round-Robin in high system loads. With EDF scheduling, the priority of a request is decided at runtime by its absolute deadline, such that the request with the highest priority is the one with the earliest deadline, at any given time

2. LITERATURE OVERVIEW

Zahir Tari et al., [1] have proposed an algorithm for scheduling and compared the performance with other basic scheduling algorithms .The main motto behind scheduling algorithms is to reduce the mean waiting time .Mean waiting time is the major issue wherever queuing the processes comes into play.This algorithm works with the technique of using a dynamic multi priority based Scheduling Algorithm having Earliest Deadline First (EDF) algorithm as a base. The reason EDF is taken as a base as it achieves a better balance among priority classes by considering the timing constraints of the processes as well compared to other algorithms.

Gamini Abhaya et al., [2] have stated a set of guidelines, algorithms and techniques that enable web services middleware to achieve predictable execution times. Existing web service middleware execute requests in a best-effort manner. While this allows them to achieve a higher throughput, it results in highly unpredictable execution times, rendering them unsuitable for applications that require predictability in execution. The guidelines, algorithms and techniques presented are generic in nature and can be used, to enhance existing engines and application servers, or when newly being built. The proposed algorithms schedules requests for execution explicitly based on their deadlines and select requests for execution based on laxity. This ensures a high variance in laxities of the requests selected, and enables requests to be scheduled together by phasing out execution.

Bertok et al., [3] have discussed a model and an admission control algorithm for achieving predictability in web services by means of service differentiation. We use real-time scheduling principles typically used offline, adapt them to web services to work online. The proposed model and algorithm is empirically evaluated by implementing it Apache Axis2. The implementation is benchmarked against the unmodified version of Axis2 for various types of workloads and arrival rates, given different deadlines.

3. EXISTING SYSTEM

General priority algorithms are used to schedule processes. Higher priority processes are favored. Lower priority processes are made to starve. The existing Scheduling algorithms used in cloud computing are generally Round Robin Algorithm and the First come First Serve algorithms. No priority is considered when it comes to scheduling and higher priority processes are made to wait for indefinite amount of time. The pre-defined classification in static priority scheduling makes them unsuitable for real-time tasks as cannot adjust their scheduling strategy to achieve task time-constraints at any given time. It achieves differentiated service among tasks in a pre-defined priority order rather than considering timing constraints of tasks. While favoring higher priority tasks, static priority scheduling algorithms tend to penalize lower priority tasks with dramatically increasing waiting times.

Event task scheduling mechanism doesn't meet users' requirements. Round Robin, an additional load on the scheduler to decide the size of quantum and it has longer average waiting time; higher context switches higher turnaround time and low throughput. The number of context switches is very high. It selects the load on random basis and leads to the situation where some nodes are heavily loaded and some are lightly loaded. Service providers negotiate with the users on the requirements of tasks including network bandwidth, complete time, task costs, and reliability of task.

4. PROPOSED SYSTEM

Multi-priority algorithm is used to schedule processes. Changes in scheduling process could be made at runtime as well. A study says that follow to moderate loads, Earliest Deadline First algorithm gives best performance and for medium to high loads, Round robin algorithm gives higher performance. So, based on that study, the load that is got is split into minimum job pool and maximum job pool. EDF and RR algorithms are applied simultaneously .This concept will definitely reduce the average mean waiting time of the processes waiting in the queue.

There are certain processes that need to be scheduled based on their requirements. If a process needs to be scheduled based on their timing constraints, the general way or method we use is the shortest job first algorithm. If a processes needs to be completed anytime but before it reaches its deadline, the general way use is the Earliest Deadline First algorithm. There are other methods where in scheduled processes based on a particular timing interval just like the concept of Round-Robin algorithm. All these algorithms focus on providing priorities wherein a single parameter is considered while giving priorities. When we want to give multiple priorities to a single processing queue, for example "time and deadline" or time and interval" or interval and deadline "this concept will be of a greater use.

Earliest Deadline First (EDF) is such a dynamic priority real-time scheduling algorithm that also considers the timing constraints of a task in scheduling them for execution. EDF considers the processing deadlines of tasks and executes them in the increasing order of their deadlines. EDF is a better choice for systems with aperiodic and sporadic tasks, and as a result it is considered as an optimal dynamic algorithm, in the sense that no other dynamic priority algorithm can schedule a task set that cannot be scheduled by it. Compared to a static schemes, the EDF based model attempts to reduce the performance degradation of lower priority request streams, with its dynamic priority enforcements. With the proposed model, show that by using preemptive scheduling the higher priority requests can achieve better waiting times and improve on the deadline loss rates compared to using non preemptive scheduling, whilst not over penalizing lower priority request.

5. DESIGN AND IMPLEMENTATION

5.1 SYSTEM ARCHITECTURE

Multi-Priority Scheduling is the concept have been discussing so far. Scheduling is of a major concern when it comes to calculating the performance result of a system. In the traditional system used, the general way use is some priority scheduling based on the system requirements like time, cost or quality. Based on that, the way in which the processes are scheduled are ,initially the higher priority cases will be given preference and once that queue is completed, it will come down to the medium priority cases and then finally to the lower priority cases. In this method, the higher priority cases are given at most priority and the lower

priority cases are made to starve. To overcome this, go for Multi Priority Algorithm (MPA).Each process has state that includes its text and data, procedure call stack, etc. This state resides in memory. The OS also stores process metadata for each process. This state is called the Process Control Block (PCB), and it includes the PC, SP, register states, execution state, etc. All of the processes that the OS is currently managing reside in one and only one of these states.

In short term scheduling, the kernel runs the scheduler at least when a process switches from running to waiting (blocks), a process is created or terminated, an interrupt occurs (e.g., timer chip).In Non-preemptive system, the Scheduler runs when process blocks or is created, not on hardware interrupts, Preemptive system, OS makes scheduling decisions during interrupts, mostly timer, but also system calls and other hardware device interrupts In Priority Scheduling, the Processes are scheduled based on certain priorities decided. Some of them are "Shortest ob First", "Earliest Deadline First", "First Come First Serve" etc.Processes to be scheduled are placed in some queue before starting to execute. Based on the priorities decided, Processes in the queue are sent to the processor to be processed and once the queue is completed with no processes left in its bin the execution is completed. In Multi-Priority Scheduling there are situations where single priority assigned may be of lesser impact .In these situations; multiple priorities come of a greater use. Giving more than one priority to a single waiting queue makes it a multiple priority scheduling as more than one priority is given to a single waiting queue.

In Cloud Scheduling internet based computing that delivers Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Services (SaaS). In SaaS, software application is made available by the cloud provider. In PaaS an application development platform is provided as a service to the developer to create a web based application. In IaaS computing infrastructure is provided as a service to the requester in the form of Virtual Machine (VM).These services are made available on a subscription basis using payas-you-use model to customers, regardless of their location. Cloud Computing still under in its development stage and has many issues and challenges out of the various issues in cloud scheduling plays very important role in determining the effective execution. Scheduling refers to the set of policies to control the order of work to be performed by a computer system. There have been various types of scheduling algorithm existing in distributed computing system, and job scheduling is one of them.

The main advantage of job scheduling algorithm is to achieve a high performance computing and the best system throughput. Scheduling manages availability of CPU memory and good scheduling policy gives maximum utilization of resource. In Round Robin Scheduling each process gets a small unit of CPU time (time quantum) usually 10-100 ms After quantum expires, the process is preempted and added to the end of the ready queue Suppose N processes in ready queue and time quantum is Q ms: each process gets 1/N of the CPU time, in chunks of at most Q ms. Early Deadline First EDF achieves best waiting times for higher priorities in lower to moderate loads and while only being number of times more than static priority algorithms in high loads. However, for the lowest priority classes, it achieves comparable waiting times to Round-Robin and First-Come-First-Served in low to moderate loads and achieves waiting times only twice the amount of Round-Robin in high system loads as shown in Figure 4.1.

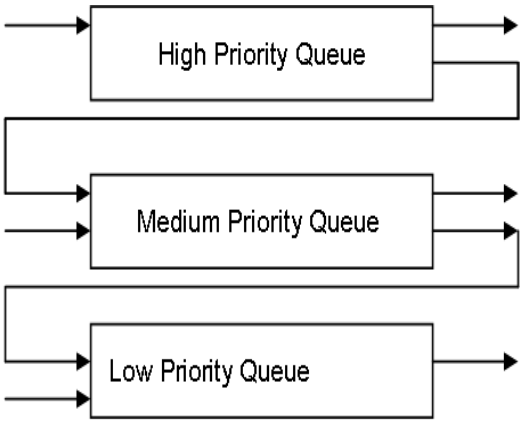


Figure 5.1 Traditional Priority Algorithm.

The given queue of processes is queued initially in the way arrive. Initially apply the SJF algorithm to the given queue and the processes are scheduled in the order of their time taken. Once it's done,

calculate the mean waiting time of that process applying SJF. Then apply EDF to the queue and schedule the processes based on their deadlines. Once that is done, split the queue into two. One is the minimum job pool and the next one is that maximum job pool. The processes with the minimum deadlines are placed in the minimum queue and the processes with maximum deadlines are placed in the maximum queue based on the aggregate values.

The jobs are split into minimum and maximum, action each queue individually as shown in Figure 4.2. EDF is applied to the minimum queue and then Round Robin algorithm is applied to the maximum queue. This is because EDF gives best performance from low to moderate loads and RR gives best performance for moderate to high loads. The respective mean waiting times are calculated and the performance comparison will be given in a chart of their reduced mean waiting times.

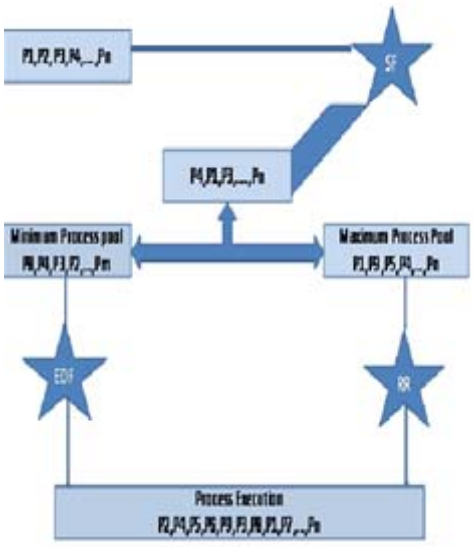
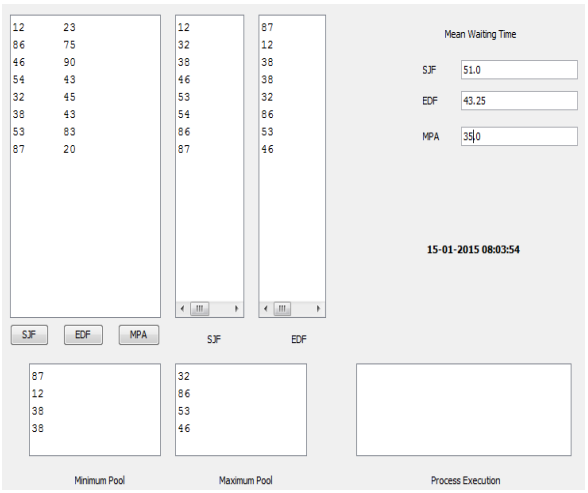


Figure 4.2 The New Scheduling System.

6. RESULT AND IMPLEMENTATION

Cloud simulator 3.0.3 integrated with net beans 7.0.1 was used to implement the project. The results showed that Multi Priority algorithm gave a waiting time much lesser than the other two algorithms namely SJF and EDF.



7. CONCLUSION

The performance metric of concern was the mean waiting time experienced by requests belonging to each priority class. The goal of using Multi-Priority Scheduling in a preemptive scheduling system is to ensure that higher priority requests are favored and experience lower mean waiting times, without leading lower priority requests to over starvation. Whenever a scheduling event occurs the queue will be searched for the process closest to its deadline. This process is the next to be scheduled for execution. Results obtained for the model will confirm that such a balance between the priority classes can indeed be achieved with the use of EDF with preemptive scheduling. Therefore, we can conclude that MPA would be a better choice for systems where differentiated request processing between multiple classes is required, yet a balanced approach where lower priority requests are prevented from over starvation, is considered important. Moreover, being a solution that is valid for any service time distribution, the model is valid for any system that uses EDF scheduling.

REFERENCES

[1] V. Gamini Abhaya, Z. Tari "Performance Analysis of EDF Scheduling in a Multi-Priority Pre-emptive M/G/1 Queue" IEEE Transactions on Parallel and Distributed systems, vol. 25, Aug 2014, pp. 2149-2159.

[2] V. Gamini Abhaya, Z. Tari, and P. Bertok, ''Building Web Services Middleware with Predictable Service Execution,'' in Proc.11th Int'l Conf. WISE, Hong Kong, China, Springer-Verlag: New York, NY, USA Dec. 2010, vol. 6488, pp. 264-270.

[3] D. Liu and Y. Lee, ‘‘An Efficient Scheduling Discipline for Packet Switching Networks Using Earliest Deadline First Round Robin,’’Telecommun. Syst.,Mar. 2005 , vol. 28, no. 3,pp. 453-474.

[4] G. S. Kumar, M. V. V. Paul, K. P. Jacob, “Mobility metric based leach-mobile protocol”, Proceedings on the 16th International Conference on Advanced Computing and Communications, **(2008)** , pp. 248-253.

[5]Vivek Chandran. K and Nikesh, “Elimination of Data Redundancy and Latency Improving in Wireless Sensor Networks”, International Journal of Engineering Research & Technology (IJERT), Vol. 3, pp. 8, August 2014.

[6]W. R. Heinzelman, A. Chandrakasan and H. Balakrishnan, “Energy Efficient Communication Protocol for Wireless Microsensor Networks”, Proceedings of the 3rd Annual Hawaii International Conference on System Sciences, **(2000)**, pp. 3005-3014.

[7] Victoria Bueno, Pablo Pavon Marino And Maurizio Rebaudengo, "A Geometric Distribution Reader Anti-Collision Protocol For RFID Dense Reader Environments", IEEE Transactions On Automation Science And Engineering, Vol. 10, No. 2, pp. 636-652, April 2013.

[8]Hiren Thakkar, Sushruta Mishra and Alok Chakrabarty, “A Power Efficient Cluster-Based Data Aggregation Protocol For WSN (MHML)”, International Journal of Engineering and Innovative Technology (IJEIT), Vol.1, pp. 4- 16, April 2012.

[9]Younis O, Fahmy S. , “ HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks” , IEEE Trans on Mobile Computing, 2004, 3(4), p. 366-379.