Different Job Scheduling Methodologies for Web Application and Web Server in a Cloud Computing Environment

Praveen K. Gupta,
Department of Computer Science & Engineering
Jaypee University of Information Technology,Waknaghat, India
E–mail: gupta88praveen@gmail.com

Nitin Rakesh, *Member, IEEE*
Department of Computer Science & Engineering and Information Technology
Jaypee University of Information Technology, Waknaghat, India
E–mail: nitin.rakesh@gmail.com

***Abstract*—Cloud computing has come out to be an interesting and beneficial way of changing the whole computing world. In this paper, we deal with the various methodologies adopted to handle all the processes and jobs concurrently executing and waiting into the web application and web server housed into the same system or different systems. Also, these different methods will be compared taking into account the same number of jobs, but varied environmental conditions and hence, the result would be formulated. Various issues like virtual resources, queuing strategies, resource managers etc. has been discussed here apart from the main coverage points. All these aspects will be closely studied, observed and proved with proper explanations.**

***Keywords*—Cloud Computing; Job Scheduling; Resource Managers; Web Application.**

I. INTRODUCTION AND MOTIVATION

Cloud comprises of many virtual storage systems, virtual data management systems, memory [1] etc. Any cloud system can have any number of web applications [3] and web servers [3] into a single system or onto different systems, like web application on system A and web server on system. Both can be on different operating environments and networks, or on the same. Behind each working application, a lot of tasks come into the picture. The web application [3] can itself manage all the jobs [2, 4] or it can be duty of web server [3] to take care of all the required procedures. The structure can have a number of components to make the whole process work flawlessly. In the coming sections, all the various job scheduling [2] techniques along with execution strategies will be discussed thoroughly.

Job scheduling[2,4, 7, 8] system will handle the various jobs in the cloud to boost up the job completion rate within the least possible time and resources' usage and thus, increasing the computational power[1-4].

Web application or any other application function normally whether it is on the same system or different ones barring the fact that the level of performance and efficiency level differs by a value, not so large, but, significant enough. Each microsecond of resource' usage counts a lot in terms of CPU performance and efficiency.

In a cloud, all the nodes need to be managed in a proper and careful manner so as to avoid any kind of unwanted anomalies. Jobs are handled by the schedulers [2, 4] and the resources by resource handlers which are they itself, embedded with the application and they demand for the subsequent number of resources as needed by them and make sure that they are used in the most appropriate manner. Web application remains in contact with the user to work in real-time with the user's requests to avoid any kind of delay. Web application access the web server to execute all the operations. Web application is made intelligent enough to take some decisions by itself with the help of schedulers and handlers. Web server can be on the same system on which application is installed or can be on any other system completely alien to this application, still working in a dignified and expected manner. In figure 1, all of these features are clubbed together to make a complete working structure of a web application and web server.

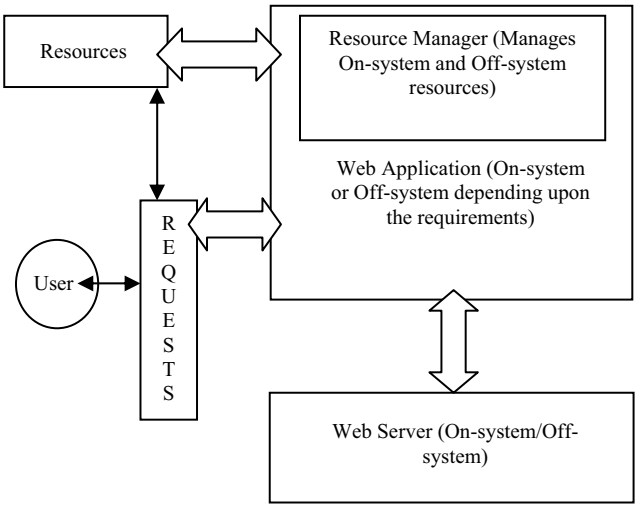

Figure 1. Basic Working Structure of Web Application and Web Server.

User' requests can be of any kind like in a form process, job or packet [3, 4] depending upon the kind of operation which needs to be done. They can be creation, abortion, deletion or modification type of interrupt [4]. Jolicloud [5] and iCloud [6] are two examples of cloud computing [1] OS.

The structure of the rest of the paper is like: section 1 is introduction part, section 2 will describe various jobscheduling methods [2, 4, 7, 8] sequentially, section 3 will put light on the comparison results of all the methods

978-0-7695-4246-1/10 $26.00 © 2010 IEEE
DOI 10.1109/ICETET.2010.24

and section 4 will conclude the paper and will also give way for further future work in this field and its related sub-fields.

II. JOB SCHEDULING PROCEDURE

On the client side or receiving end, there will be cloud computing users' [2] and on the other end, service providers [2] of the cloud environment making the virtual instances[2] of services available whenever demand arises. The Main Goal here of job scheduler [2, 7, 8]is to do its best to allocate resources efficiently to the user's jobs so that they can get executed. The computation regarding the user's jobs will be done with the help of various parameters and conditions to be encountered in the simulation of jobs. The performance of jobs over a stipulated time will be gauged to chalk out the efficiency of scheduler in handling the web application's jobs.

An application can have any number of processes [4] under it which also depends on the ongoing processes. Different processes are kept in different process groups [4] to maintain a certain order of uniformity in the scheduling [4] process and therefore, are allotted their respective ProcessGroupID. Every process can contribute to any number of jobs depending upon the type and number of operations to be performed by that particular instance of process. Jobs are assigned a particular ID, JobID [4] by their respective processes which also bear a unique ProcessID [4]. Both JobID and ProcessID are useful in the sense that they can be used to find and use anyone of them by just referring to their particular IDs. If a particular job and process has already been executed earlier, and another instance of that process comes again for operation, then, the scheduler will look if the previous instance of it is present in the buffer of the resource manager or not. If it is still there, the scheduler will load that particular process or job quickly without wasting time and resources through a more direct get-and-execute[4] cycle. This will ultimately add to the server time and save waiting time of jobs and idle server time which will be discussed later in this section. Further, Jobs can be sub-divided into many categories like in terms of arrival time for execution, service time taken for the job, priority and some more factors.

In Table 1, eight jobs with their respective JobIDs ranging from A-H can be found in 1st column. 2nd column contains the priority of all of these jobs which will be used in case of priority scheduling. Lastly, 3rd and 4th column has arrival time [4] and service time[4] of JobID A-H.

TABLE 1. JOB SPECIFICATION TABLE

Job ID	Priority	Service Time	Arrival Time
A	1	4	3
B	2	2	1
C	3	1	6
D	4	5	9
E	2	7	2
F	4	1	4
G	1	2	5
H	3	3	7

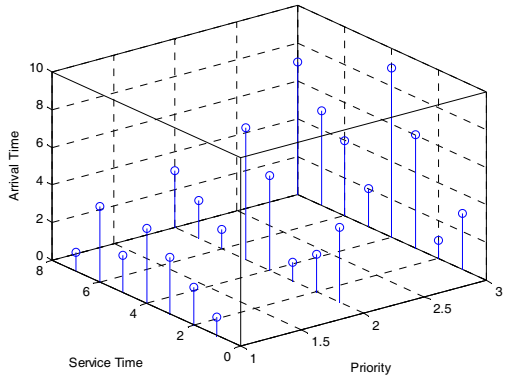

Figure 2. Graphical Representation of Table 1

In figure 2, Table 1 has been graphically interpreted and represented. Discussing about different job scheduling methods in connection-oriented system first, let's discuss FIFO (First in First Out) [2, 4] type of scheduling. This job execution procedure takes into account the arrival time of the jobs. Any job which arrives first will get executed first, regardless of their service time. No pre-emption [4] is allowed here. It's a simple first-cum-first-serve concept based scheduling procedure.

For queue representation, Gantt charts have been used. The job execution queue will look like the queue shown below in which the order of execution runs from left side to the right side.

B	E	A	F	G	C	H	D

So, job B will get executed first, then E and so on. They will not be pre-empted and will execute in this order for their full service time. The server time [4] distribution queue is as shown below:

B	E	A	F	G	C	H	D
1	3	10	14	15	17	18	21	26

From this,
Total service time [2, 4] = 26 ms. average service time [2, 4] = 26/8 = 3.25 ms. average waiting time [2, 4] = (10+1+17+21+3+14+15+18) / 8 = 99 / 8 = 12.375 ms.

The waiting time [4] for each job would be different with respect to each other when the jobs arrive at different instants of time. In priority-based [2, 4] type of scheduling, arrival time and service time are completely ignored. Only priority of a particular job is the main criteria in job queuing [2] and execution. The job execution queue is as below:

A	G	B	E	H	D	F	C

When two or more jobs of the same priority arrive, they will be allotted resources to execute depending on their respective arrival times. This is a special case. The server time distribution queue is as shown below:

A	G	B	E	H	D	F	C
1	5	7	9	16	19	24	25	26

Total service time= 26 ms, average service time= 26 / 8 = 3.25 ms, average waiting time = (1+7+25+19+9+24+5+16) / 8 = 106 / 8 = 13.25 ms.

These values are same as of FIFO scheduling. The difference would be only in terms of waiting time of all the jobs. Shortest Job First (SJF) [4] type of scheduling, priority and service time is completely ignored. Only service time of a particular job is the main criteria in job queuing and execution. The job with lesser service time will get more preference to execute in comparison to other jobs with higher values. Here, we have considered it to be non-preemptive [4]. The job execution queue is as below:

F	C	B	G	H	A	D	E

In case of two or more jobs with same service time, job with more priority or arrival time will be preferred, if available and any one of these or both. The server time distribution queue is as shown below:

F	C	B	G	H	A	D	E
1	2	3	5	7	10	14	19	26

Total service time= 26 ms, average service time = 26 / 8 = 3.25 ms, average waiting time = (10+3+2+14+19+1+5+7) / 8 = 61 / 8 = 7.625 ms.

These values are same as of FIFO and priority scheduling. There would be variation in the waiting time of all the jobs.The comparison of the arrival times of various jobs in different job scheduling methods is tabulated below:

TABLE 2. SERVER TIME DISTRIBUTION TABLE FOR FIFO, SJF AND PRIORITY

Job	FIFO	SJF	Priority
A	10	10	1
B	1	3	7
C	17	2	25
D	21	14	19
E	3	19	9
F	14	1	24
G	15	5	5
H	18	7	16

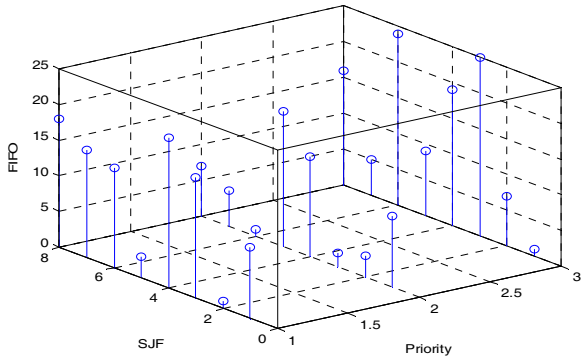

Figure 3. Graphical Representation of Table 2

So, in all the above execution algorithms, the total time taken by the jobs to fulfill their objectives is same. Only the sequence of execution varies in each type depending upon the arrival time, service time and priority. The wait time is different in each case.

Now, we will describe some special cases which can occur in these types of scheduling:

- Job gets aborted: In this case, all of the scheduling methods will move onto the next available job. The service time and wait time will change because of this.
- Job Starves: If job waits for a long time, it can starve[4] which can lead to job abortion or steep increase in waiting time.
- Job Overriding: Let us assume that resource manager overrides any job, may be because of lesser number of resources or to complete it first due to limited resource demand. This can be a very complex and lengthy procedure.

Suppose, the Job Scheduler acts like a server consisting of only one main job handler and executer. But, if it were to be a double or triple-server executing more than one job simultaneously, then, the whole job scheduling mechanism goes under ramification according to the type of Simulation the Jobs follows. This can make the Job Scheduling more complex and lead to a bit delayed completion.

Thinking of a Lined manner queuing of jobs for execution regardless of priority makes it clear enough for the Scheduler Server to implement the Execution algorithm but when it comes to Priority-wise Execution, more than one server will find it ambiguous to timely execute all the Jobs before they starve and that too, in an optimum way.

In connection-less system in unguided media, i.e., when web application is on one system and the web server to be used is on another system as a different location. In this situation, the communication takes place wirelessly, or in atmosphere. The jobs are sent through wireless signals using packets [3]. Packet can contain various different jobs of different processes.

Each job is transmitted along with the relevant packetID. If all the packets reach the destination system as per the given correct sequence of transmission, the execution remains same as like in case of connection-oriented, but changes if any packet gets corrupted or lost during transmission. Each and every attribute of system changes like arrival time, service time etc.

All of the scheduling [2, 4, 7, 8] methods work like that of connection-oriented systems. Time is lost during transmission of packets, leading to increase in waiting time of jobs and the server time increases too. Jobs can be distributed into different packets and they can be sent to other system at the same time or different instants of time. So, some packets of a particular job can arrive much before other or in any random order, thereby, delaying the execution of it.

In figure 4, the modular structure of on-system and off-system application operations is shown. Web application can be on-system or off-system. Both of them have a number of processes under them which can be further grouped in ProcessGroup or simply, jobs. Jobs, in case of connection-less transmission, are sent in form of packets. As there are many packets to be sent, they are packed into different packet groups and transmitted along the wireless medium. They are received by the web server and arranged in the correct sequential executable manner as needed.

III. RESULTS AND ANALYSIS

The average service time and average waiting time values are calculated from the different job methodologies. In Table 3, they are shown for better understanding and comparison analysis purposes.

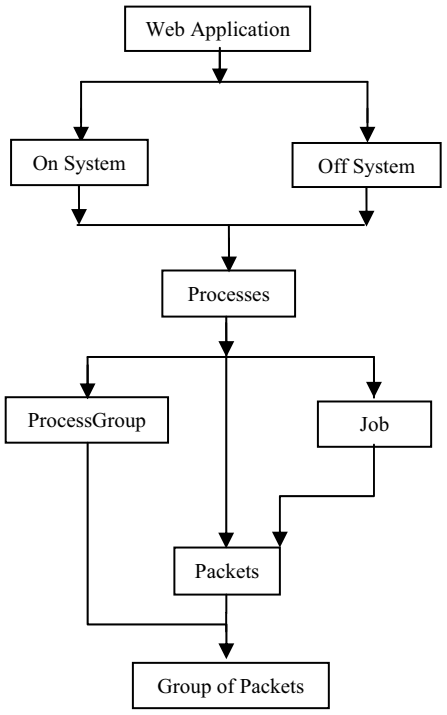

Figure 4. Modular Structure for On-System and Off-System Operations

TABLE 3.AVERAGE SERVICE TIME AND AVERAGE WAITING TIME OF FIFO, SJF AND PRIORITY TYPE OF SCHEDULING

	Avg. Service Time (ms)	Avg. Waiting Time (ms)
FIFO	3.25	12.375
SJF	3.25	7.625
Priority	3.25	13.25

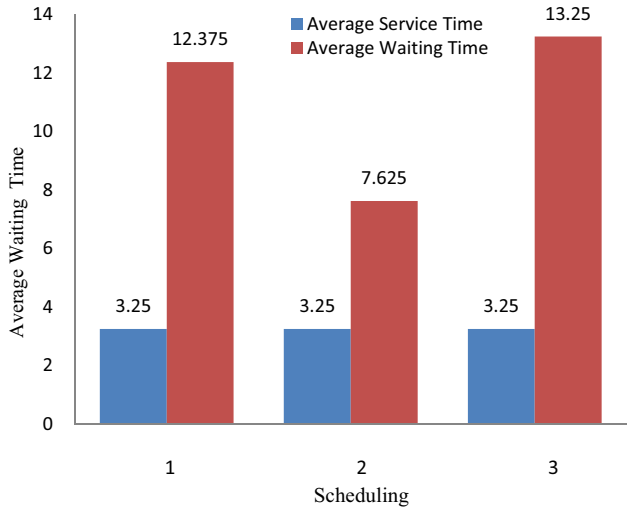

Figure 5. Graphical Representation of Comparison between Average Waiting Time of FIFO, SJF and Priority type of Scheduling

In Table 3 and figure 5, the tabular and graphical implication as well as comparison chart of average waiting time of FIFO, SJF and priority has been shown. This can be for the on-system or off-system job management strategies in a cloud environment[1] keeping in view the respective requirements of the user and the end-system.

As seen, the average waiting time for jobs in case of SJF is comparatively lesser with respect to FIFO and priority. But, the time taken by all the jobs to execute is same in all of the methods. They arrive at different point of time and wait for different time intervals before accomplishing their task. This is the main result obtained after a detailed comparison of all the techniques.

So, this will lead to increased data transfer rate and Efficiency in a cloud environment as the job execution occurs in the most optimum way. Data error and packet loss decreases due to more competent system. The Lifespan of a job in a cloud environment varies differently for priority, arrival and service time.

IV. CONCLUSION AND FUTURE WORK

We presented a detailed working of different job scheduling methods behind the web application on a web server on the same system or on different systems in a virtual cloud computing environment [1]. The various terminologies and concepts were thoroughly looked and thus explained.

This paper will give way to more future findings regarding the scheduling techniques in a cloud environment. More efficient and faster ways to schedule jobs and increase CPU throughput [2, 4] needs to be discovered. Also, this will fuel the greater knowledge and popularity of cloud environment among the peoples.

REFERENCES

[1]. B. P .Rimal, E. Choi and I. Lumb, "A Taxonomy and Survey of Cloud Computing Systens", Proceedings of 5th International Joint Conference on INC, IMS and IDC, 2009, pp. 44-51

[2]. L. Li, "An Optimistic Differentiated Service Job Scheduling System for Cloud Computing Service Users and Providers", Proceedings of 3rd International Conference on Multimedia and Ubiquitous Engineering, 2009, pp. 295-299

[3]. T. C. Chieu, A. Mohindra, A. A. Karve and A. Segal, "Dynamic Scaling of Web Applications in a Virtualized Cloud Computing Environment", Proceedings of IEEE Conference on e-Business Engineering, 2009 , pp. 281-286

[4]. Silberschatz, P. B. Galvin, G. Gagne, Operating System Concepts, 7th Edition.

[5]. Jolicloud, www.jolicloud.com.

[6]. iCloud, www.icloud.com.

[7]. N. W. Paton, M. A. T. de Aragao, K. Lee, A. A. A. Fernandes, "Optimizing Utility in Cloud Computing through Automatic Workload Execution", Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 2009, pp. 1-8.

[8]. B. Peng, B. Cui, X. Li, "Implementation Issues of a Cloud Computing Platform", Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 2009, pp. 1-8.