International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 3, Issue 6, December 2014)

Load Balancing in Distributed System Using FCFS Algorithm with RBAC Concept and Priority Scheduling

Geeta[1], Charanjit Singh[2]
[1]*M.Tech Scholar, CSE, MMU, Sadopur, Ambala, India*
[2]*CSE, MMU, Sadopur, Ambala, India*

***Abstract*-- Now a days CPU workload, hardware technology and multiprocessor services are developing rapidly. For availability, scalability and higher performance more and more server are required Load balancing is key issue in these type of situation. To avoid overload and for maximum throughput load balancing is required. In distributed system computers are not same type means not same configuration so that some computer finish their work earlier and sit ideal which degrade the performance of multicomputer system. For proper load balancing a new algorithm is developed which use first come first serve and priority scheduling with RBAC An efficient system has three element – collection of device, network connect to these computer and software that enable to share data between these computer. This paper contains a scheduling algorithm for proper load balancing in distributed environment.**

***Keywords*-- distributed system, load balancing, priority, FCFS.**

I. Introduction of Distributed System

A distributed system is a collection of independent computers that appears to its users as a coherent system. It consists of a collection of autonomous computers linked by a computer network and equipped with distributed system software. They do not share memory or clock; computer communicates with each other by exchanging message each other over communication network. In distributed system each computer has its own memory and run its own operating system [1]. Two types of resources are used in distributed system.

- Local resources
- Global resources

Local resources are owned and controlled by same system. While the resources owned and controlled by other system are said to be remote resources.

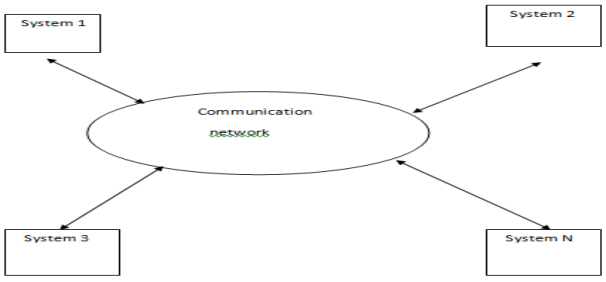

Figure 1: Distributed system network

The processer in a distributed system may vary in size and function. Distributed system includes small microcomputer, work stations, minicomputer and large general purpose computer system. They appears to its user as a centralized system but distributed system are differ because in centralized system data resides in one single location while in distributed system data resides in several location.

II. Load Balancing

Load Balancing is the approach of distributing the load of the server when the execution of jobs has to be done. Load balancing is a computer networking method to distribute workload across multiple computers or a computer cluster, network links, central processing units, disk drives, or other resources, to achieve optimal resource utilization, maximize throughput, minimize response time, and avoid overload [2]. Using multiple components with load balancing, instead of a single component, may increase reliability through redundancy. The load balancing service is usually provided by dedicated software or hardware, such as a multilayer switch or a Domain Name System server shown in figure 2.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 3, Issue 6, December 2014)

Load Balancing is a method to distribute workload across one or more servers, network interfaces, hard drives, or other computing resources. Typical data Centre implementations rely on large, powerful (and expensive) computing hardware and network infrastructure, which are subject to the usual risks associated with any physical device, including hardware failure, power and/or network interruptions, and resource limitations in times of high demand.

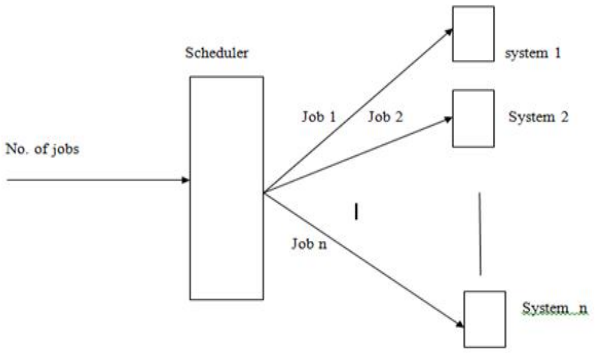

Figure 2: Load balancing in distributed system

Load balancing in the cloud differs from classical thinking on load-balancing architecture and implementation by using commodity servers to perform the load balancing. This provides for new opportunities and economies-of-scale, as well as presenting its own unique set of challenges [3]. Load balancing is used to make sure that none of your existing resources are idle while others are being utilized. To balance load distribution, you can migrate the load from the source nodes (which have surplus workload) to the comparatively lightly loaded destination nodes. When you apply load balancing during runtime, it is called dynamic load balancing — this can be realized both in a direct or iterative manner according to the execution node selection:

- In the iterative methods, the final destination node is determined through several iteration steps.
- In the direct methods, the final destination node is selected in one step.

An n another kind of Load Balancing method can be used i.e. the equally spread current execution load balancing method, max-min load balancing, load balance min-min algorithm, load balance max-min-max algorithm, a hybrid method etc. All algorithm are based on basic scheduling algorithm like FCFS (first come first serve), Round robin scheduling, Priority scheduling. Explanations of these algorithms are given below.

III. Load Balancing Algorithms

3.1 FCFS (First Come First Serve)

FCFS stands for "First Come First Serve ".In this algorithm the first data which reaches to the queue first gets executed first. This algorithm is time consuming and does not perform quite efficiently when there is a case of priority in the segmentation. Other names of this algorithm are

- First-In-First-Out (FIFO)
- Run-to-Completion
- Run-Until-Done

First-Come-First-Served algorithm is the simplest scheduling algorithm. Processes are dispatched according to their arrival time on the ready queue. Being a no preemptive discipline, once a process has a CPU, it runs to completion [4]. The FCFS scheduling is fair in the formal sense or human sense of fairness but it is unfair in the sense that long jobs execute short jobs wait and unimportant jobs execute important jobs wait.

FCFS is more predictable than most of other schemes since it offers time. FCFS scheme is not useful in scheduling interactive users because it cannot guarantee good response time. The code for FCFS scheduling is simple to write and understand. One of the major drawbacks of this scheme is that the average time is often quite long.

The First-Come-First-Served algorithm is rarely used as a master scheme in modern operating systems but it is often embedded within other schemes.

3.2 Priority Scheduling Algorithm

This algorithm discards the disadvantages of FCFS and round robin algorithm. In this algorithm a priority of each job is decided on the basis of the properties of the tasks. The priority of the task may be judged on the basis of the time consumption of the tasks or CPU burst time of the tasks

- The SJF algorithm is a special case of the general priority scheduling algorithm. A priority is associated with each process, and the CPU is allocated to the process with the highest priority. Equal-priority processes are scheduled in FCFS order.
- An SJF algorithm is simply a priority algorithm where the priority (p) is the inverse of the (predicted) next CPU burst. The larger the CPU burst, the lower the priority, and vice versa.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 3, Issue 6, December 2014)

- Priorities are generally indicated by some fixed range of numbers, such as 0 to 7 or 0 to 4095. However, there is no general agreement on whether 0 is the highest or lowest priority. Some systems use low numbers to represent low priority; others use low numbers for high priority. We use as low numbers represent high priority.
- As an example, consider the following set of processes, assumed to have arrived at time 0, in the order P1, P2, P3----P5 with the length of the CPU burst given in milliseconds:

Table 1:
Table shows no. of process with priority

Process	Burst Time	Priority	Waiting Time	Turnaround Time
P_1	10	3	6	16
P_2	1	1	0	1
P_3	2	4	16	18
P_4	1	5	18	19
P_5	5	2	1	6
Average	-	-	8.2	12

Using priority scheduling, we would schedule these processes according to the following chart.

Figure 3: Schedule of process according to priority scheduling

Priorities can be defined either internally or externally.

I. Internally defined priorities use some measurable quantity or quantities to compute the priority of a process. For example, time limits, memory requirements, the number of open files, and the ratio of average I/O burst to average CPU burst have been used in computing priorities.

II. External priorities are set by criteria outside the OS, such as the importance of the process, the type and amount of funds being paid for computer use, the department sponsoring the work, and other, often political, factors [5].

Priority scheduling can be either pre-emptive or non-pre-emptive. When a process arrives at the ready queue, its priority is compared with the priority of the currently running process.

I. A pre-emptive priority scheduling algorithm will pre-empt the CPU if the priority of the newly arrived process is higher than the priority of the currently running process.

II. A non-pre-emptive priority scheduling algorithm will simply put the new process at the head of the ready queue.

A major problem with priority scheduling algorithms is indefinite blocking, or starvation. A process that is ready to run but waiting for the CPU can be considered blocked.

I. A priority scheduling algorithm can leave some low priority processes waiting indefinitely.

II. In a heavily loaded computer system, a steady stream of higher-priority processes can prevent a low-priority process from ever getting the CPU.

3.3 Round Robin

Round robin uses the time slicing mechanism. The name of the algorithm suggests that it works in the round manner where each node is allotted with a time slice and has to wait for their turn. The time is divided and interval is allotted to each node. Each node is allotted with a time slice in which they have to perform their task. The complicity of this algorithm is less compared to the other two algorithms. An open source simulation performed the algorithm software know as cloud analyst, this algorithm is the default algorithm used in the simulation [6]. This algorithm simply allots the job in round robin fashion which doesn't consider the load on different machines.

- The round-robin (RR) scheduling algorithm is designed especially for time-sharing systems. It is similar to FCFS scheduling, but pre-emption is added to switch between processes.
- A small unit of time, called a time quantum or time slice, is defined. A time quantum is generally from 10 to 100 milliseconds. The ready queue is treated as a circular queue.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 3, Issue 6, December 2014)

To implement RR scheduling,

I. We keep the ready queue as a FIFO queue of processes.
II. New processes are added to the tail of the ready queue.
III. The CPU scheduler picks the first process from the ready queue, sets a timer to interrupt after 1 time quantum, and dispatches the process.
IV. The process may have a CPU burst of less than 1 time quantum.
    i. In this case, the process itself will release the CPU voluntarily.
    ii. The scheduler will then proceed to the next process in the ready queue.
V. Otherwise, if the CPU burst of the currently running process is longer than 1 time quantum,
    i. The timer will go off and will cause an interrupt to the OS.
    ii. A context switch will be executed, and the process will be put at the tail of the ready queue.
    iii. The CPU scheduler will then select the next process in the ready queue.

IV. RBAC (Role Based Access Control)

A RBAC system has two phases in assigning a privilege to a user: in the first phase, the user is assigned one or more roles; and in the second phase, the roles are checked against the requested operations. In RBAC, permissions are associated with roles rather than users, thus separating the assignment of users to roles from the assignment of permissions to roles. Users acquire access rights by their roles, and they can be dynamically re-assigned or removed from roles without changing the permissions associated with roles. The number of roles is typically much smaller than the number of users. Roles may have a hierarchical structure, and it reflects the organization's lines of authority and responsibility. For example, is a sample fragment of role hierarchy from Microsoft? Different roles, such as CEO, CTO, and VIP are arranged in the diagram, where junior roles appear at the bottom and senior roles at the top. However, it is not clear how to define roles for a specific application domain. For example, if a tenant from a start-up company tries to build up its own access control model for its application, it is difficult to start from scratch and define role hierarchy and related policies.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 3, Issue 6, December 2014)

V. RESULTS

Table 3:
Comparison on basis of 5 tasks

No. of task	Task name	Existing algorithm					Proposed algorithm				
		Task execution pattern	Task execution time	Average Execution time	Task On each system	Max task	Task execution pattern	Task execution time	Average execution time	Task on each system	Max task
5	taska0, taska1, taska2 taska3, taska4	System1task2, system2-task1, system3-task1, system4-task1, system5-task0	11,20,19 21,23	18.8	2 1 1 1 0	2	system5-task1, system4-task1, system3-task1, system1-task1, system2-task1	9,14,2 0, 16,5	12.8	1 1 1 1 1	1
5	taska0, taska2, taska4 taska6, taska8	System1task3, system2-task2, system3-task0, system4-task0, system5-task0	26,28,20, 15,22	22.2	3 2 0 0 0	3	system5-task1, system4-task1, system3-task1, system1-task1, system2-task1	8,6,3,1 112	8	1 1 1 1 1	1
5	taskb0, taskb2, taskb4 taskb6, taskb8	System1task2, system2-task1, system3-task1, system4-task1, system5-task0	13,18,25, 24,22,	20.4	2 1 1 1 0	2	system5-task1, system4-task1, system3-task1, system1-task1, system2-task1	5,12,9, 15,17	11.6	1 1 1 1 1	1
5	taska1, taska3, taskb0 taskb4, taskb8	System1task3, system2-task1, system3-task1, system4-task0, system5-task0	28,20,14, 26,23	22.2	3 1 1 0 0	3	system5-task1, system4-task1, system3-task1, system1-task1, system2-task1	2,7,6,4 , 9	5.6	1 1 1 1 1	1
5	taska5, taska7, taskb1 taskb5, taskb9	System1task3, system2-task1, system3-task1, system4-task0, system5-task0	13,15,23, 22,12	17	3 1 1 0 0	3	system5-task1, system4-task1, system3-task1, system1-task1, system2-task1	9,6,11, 3,5	6.8	1 1 1 1 1	1

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 3, Issue 6, December 2014)

Table 4
Comparison on basis of 10 tasks

No. of task	Task name	Existing algorithm					Proposed algorithm				
		Task execution pattern	Task execution time	Average execution time	Task on each system	Max task	Task execution pattern	Task execution time	Average Execution time	Task on eachsystem	Max task
10	taska0,taska1,taska2 taska3,taska4,taska5 taska6,taska7,taska8 taska9	System1task4 system2task3 system3task1 system4task1 system5task1	47,35,37, 32,46	19.7	4 3 1 1 1	4	system5-task2, system4-task2, system3-task2, system1-task2, system2-task2	23,18, 11,12, 21	8.5	2 2 2 2 2	2
10	taskb0,taskb1,taskb2 taskb3,taskb4,taskb5 taskb6,taskb7,taskb8, taskb9	System1task3 system2task2 system3task2 system4task2 system5task1	40,41,38, 36,31	18.6	3 2 2 2 1	3	system5-task2, system4-task2, system3-task2, system1-task2, system2-task2	8,12, 10,17, 13	6	2 2 2 2 2	2
10	taska0,taska2,taska4 taska6,taska9,taskb1, taskb3,taskb5taskb8, taskb9	System1task4 system2task2 system3task2 system4task1 system5task1	44,31,43, 39,34	19.1	4 2 2 1 1	4	system5-task2, system4-task2, system3-task2, system1-task2, system2-task2	14,13, 27,10, 20	8.4	2 2 2 2 2	2
10	taska1,taska5,taska6 taska8,taskb8,taskb0, taskb2,taskb3taskb8, taskb9	System1task4 system2task2 system3task2 system4task1 system5task1	42,40,33, 39,33	18.7	4 2 2 1 1	4	system5-task2, system4-task2, system3-task2, system1-task2, system2-task2	12,19, 20,25 13,	8.9	2 2 2 2 2	2
10	taska3,taska4,taska7 taska8,taskb0,taskb1 ,taskb3,taskb4taskb8, taskb9	System1task4 system2task3 system3task1 system4task1 system5task1	45,33,32, 34,38	18.2	4 3 1 1 1	4	system5-task2, system4-task2, system3-task2, system1-task2, system2-task2	10,15, 13,26, 16	8	2 2 2 2 2	2

15	taska2, taska3, taska4 taska5, taska6, taska7 ,taska8 ,taska9 ,taskb0 taskb2, taskb3, taskb4 ,taskb5 Taskb6 ,taskb7	System1task4, system2-task3 system3-task3 system4-task3 system5-task2	56,53, 63,52, 57	18.7	4 3 3 3 2	4	system5-task3, system4-task3, system3-task3, system1-task3, system2-task3	11,25, 20,25, 31	7.4	3 3 3 3 3	3

VI. CONCLUSION

It is expected that the load will be reduced if we will be implementing the FCFS along with the RBAC and the priority queue concept because the RBAC will restrict the system from unauthorized access to the server whereas the priority queue will speed up the concept of execution. Performance of a system is improved with the combination of FCFS and priority queue. In general configuration of system is fixed but in this project we create system 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] Mudassar Ahmad" Prognostic Load Balancing Strategy For Latency Reduction In Mobile Cloud Computing". Middle-East Journal Of Scientific Research 16 (6): 805-813, 2013 Issn 1990-9233 © Idosi Publications, 2013 Doi: 10.5829/Idosi.Mejsr.2013.16.06.11314.

[2] Amandeep Kaur Sidhu "Analysis Of Load Balancing Techniques In Cloud Computing". International Journal Of Computers & Technology Volume 4 No. 2, March-April, 2013, Issn 2277-3061, Www.Cirworld.Com.

[3] Soebhaash Dihal "Mobile Cloud Computing: State Of The Art And Outlook". Soebhaash Dihal, Harry Bouwman, Mark De Reuver, Martijn Warnier, Christer Carlsson, (2013),"Mobile Cloud Computing: State Of The Art And Outlook", Info, Vol. 15 Iss: 1 Pp. 4 - 16.

[4] B. Subramani "A New Approach For Load Balancing In Cloud Computing". Ieee Volume 2 Issue 5 May, 2013 Page No. 1636-1640

[5] Ram Prasad Padhy "Load Balancing In Computing System". Department Of Computer Science And Engineering National Institute Of Technology, Rourkela Rourkela-769 008, Orissa, India May, 2011.

[6] Fan Yang,"Sonora: A Platform For Continuous Mobile-Load Balancing". Microsoft Research Asia University of Washington In Pods (2012).