Resource Management in Operating Systems- A Survey of Scheduling Algorithms

Khizar Hameed[1], Aitizaz Ali[1], Mustahsan Hammad Naqvi[1], Muhammad Jabbar[2], Muhammad Junaid[1], Aun Haider[1]
[1]Department of Computer Science and Electrical Engineering, University of Management and Technology, Sialkot, Pakistan
[2]Department of Computer Science, University of Gujrat, Gujrat, Pakistan
[khizer, aitizaz.ali, mustahsan.naqvi, muhammad.junaid, aun.haider]@skt.umt.edu.pk, m.jabbar@uog.edu.pk

***Abstract*— Resource management is an important process in all operating systems in which we have scarce resources to manage all process running on that system. There are designed, many algorithms for resource management and every algorithm provide resource allocation to processes. The basic purpose of this paper is to describe the working of these scheduling algorithms, advantages/disadvantages and which algorithm provides us high throughput and fairness? In the future, we plan to extend this survey paper for different types of operating systems such as mobile phones and embedded systems.**

Keywords— Fair-share scheduler; Lottery scheduling; Round robin;

I. INTRODUCTION

Resource management is the process in all operating systems in which particularly system resources (e.g. Central Processing Unit (CPU), random access memory, secondary storage devices, external devices, etc.) is assigned to particularly processes, threads and applications. This is usually done to achieve high throughput, quality of service, fairness and balance between all processes[1][2][3] . To perform this managing level task, we need scheduling algorithms to ensure that all the processes share the system resources equally according to need[3]. This scheduling level task is the basic requirement for those systems that are performed multitasking and multiplexing[4][5].

Scheduling policy is used in those systems where we have scarce resources for many processes. Scheduling is basically performed to reduce the waiting time and the time taken from context switching[6]. All processes that competing for resources are important and although scheduling scheme in distributed systems is very complex. For example, an important process that is in the job queue and waiting for resources for completing the task but has no resources or even not complete resources that are required to complete the task[7]. So, this poor management may lead to decreases the performance of the system. In the environment, it is the responsibility of the system to schedule the resources in fairly manner[8].

There are much kind of systems like batch system, Interactive system, real time system and embedded system and each system is performing scheduling according to requirement of the processes and available resources. When system resources are shared among processes, threads and applications, there can be lots of conflicts during sharing. To avoid these conflicts, we use different scheduling algorithms for different kind of systems. In batch system, we mainly focus on throughput, turnaround time and CPU utilization. Interactive and real time system, we require response time and predictability. For batch systems, we use First Come First Served, Shortest Job First and Shortest Remaining Time Next algorithms for managing system resources. For interactive system, we use Fair Share scheduling Round Robin (RR) scheduling, Lottery scheduling, and Priority scheduling. For real time system, we use Rate Monotonic scheduling and Earliest Deadline First scheduling [9][10][2].

In fair-share scheduling, we manage operating system performance to assigning particular system resources to competing processes by dynamically[11]. The responsibility of fair-share scheduler to assigning resources of the system in a fairly manner that each user or process running on the system gets the resources. Basically the problem arise in such systems where virtual environment runs many operating systems so all processes running on the operating systems required resources[12].so, we use fair-scheduler to handle this managerial task. Fig. 1. shows the fair resources allocation according the size of the processes.

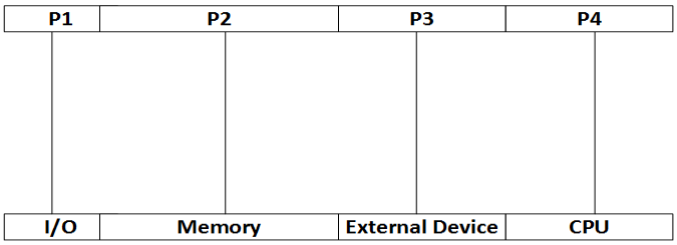

Fig. 1. Fair-Share scheduler

One of the other interactive system resource share algorithm is a lottery scheduler that is also called a flexible proportional share algorithm[13]. In a lottery scheduling the processes access the system resources on the lottery ticket base system and who draws the more lotteries for the resources automatically the winner of the system resources for required process.

The RR algorithm is also one of the scheduling approaches to determine the turnaround time. In RR, a fixed quantum time is assigned to each process and each process share the resources of the system until the time quantum expired [2].

The main objective of this paper is to provide the working of interactive system resources scheduling algorithms. How these resource scheduling algorithms are different from each other, their advantages & disadvantages? Which algorithm provides us high throughput, fairness and equality?

The rest of the paper is organized as follows. Section II discussed existing state of the art resource management algorithms. Performance evaluation of scheduling algorithms is presented in Section III. Section IV concludes the paper and future work.

II. Existing State of the Art Resource Management Algorithms

Resource management in operating systems is a very important process to ensure that all users or processes get resources accurate and in a fairly manner. All processes running on operating system require the resources to complete its execution. Many CPU scheduling algorithms have been designed to allocate the system resources to processes for ensure fairness among processes[11][14].

A. Fair Share scheduler

The Fair share scheduler is one of the CPU scheduling algorithms that are designed to share the available resources to the processes by dynamically. Users share the system shares that are proportional to the available resources of the system[15]. It is also seen that the fair share scheduling algorithm perform fairness among users as well as process. Fair share scheduler is usually done to achieve fairness between processes that no processes waiting for a long time to get resources. Fair share scheduling allocates the system resources to process by priority decision. The user has highest priority get the more system shares as compare to low priority users. When the same priority users compete for systems resources so there are many approaches to allocate resources. Firstly, user who has importance allocates the system resources and secondly the same level of system resources is assigned to same priority users. It is also seen that multimedia applications that running on the operating systems require more system shares as compare to other database applications. Basically two terms Shares and usage mostly common use in fair share scheduler. User have more shares in the system can do more work on system as compare to other user that have low shares in system can do less work. On the other side, the term usage defines the quantity of work that has user done by using the system[16][11][15].

It is the not the responsibility of the scheduling algorithms to just divide the available resources to processes or users but divide the available resources according to the requirement of the processes for execution. As clear from Fig. 2. that user 1 has more shares on system so it can do more work on system as compared to user 2 that have fewer shares.

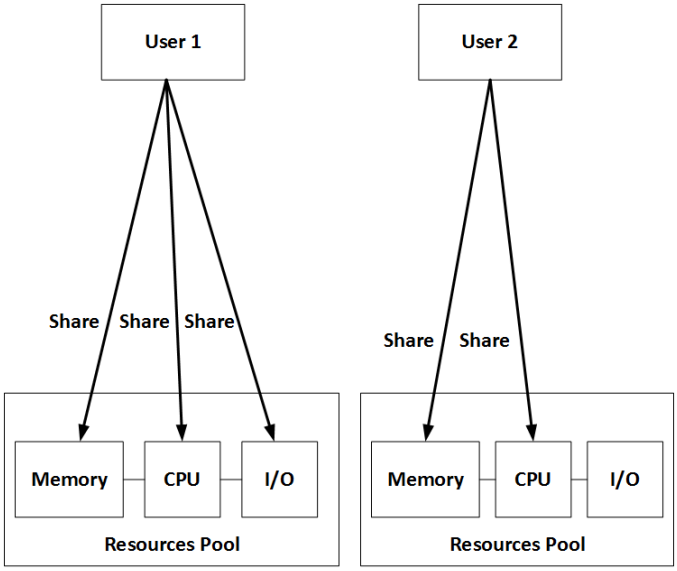

Fig. 2. Shares

There are two views to describe the fair share scheduler. One is the user point of view and the second is the program. In user

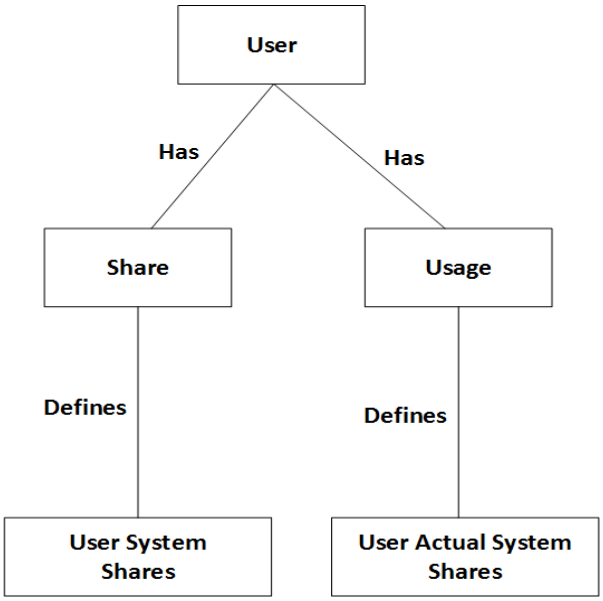

Fig. 3. User's View of Share [14]

view, a usage of user is maintained by charges gained by each process and also maintains the record of resources usage, but in process view of the fair share scheduler; maintain the cost of running processes and also maintain the priorities of processes [11]. As Clearly from Fig. 3.

Fair share scheduler includes fixed budget model, virtual time round robin, lottery scheduling, RR, max-min fair scheduling, stride scheduling, proportional share scheduling, and hierarchical share scheduling[17][18][19][20].

B. Fixed Budget

Another way of addressing the fair policy is fixed budget model in which budget associated with each user. When the

user uses the resources they have to charge for using resources for execution of processes and their budget will be reduced and when the user has no budget, automatically it cannot use the system resources further. It means that the user must have a fixed budget to use the system resources. A user that has more budget can use the system shares more and who have less budget obviously less shares of the system. There are many approaches have been designed in the fixed budget model to define limits and bounds of systems resources like disc capacity and printer.

C. VM-aware fair scheduler

Fairness in also most critical issue in Symmetric multiprocessors (SMP) where many processes run on heterogeneous systems with different computational competence. To perform fairness in a virtual machines environment, VM-aware fair scheduler has been proposed to overcome preemptive process issues that compete for resources in a virtual machine environment. The performance of VM-aware fair scheduler for preemptive processes is greatly as compare to other virtual machine credit base algorithms[21].

Fair share scheduling is also used in UNIX operating systems for resource management. Different terminologies of resource management are used with different names in many systems like a process resource manager in Hewlett Packard, workload manager in IBM and system resource manager in the SUN. Workload term in these systems is used as an interchangeably as process and same as other systems we use fair share scheduling to allocate system resources to workloads to achieve high performance and throughput. Response time of workloads is also an important factor when we are describing about fair share scheduling and we aggregate that this scheduling algorithm works fine if it gives precise and effective resource sharing to workloads[11].

Let us consider an example of fair share scheduling in which numbers of workloads running on the systems whose arrive time and service time is known in advance. We know that processes or workloads that are assigned to processers are fractionally sharing among the processer of system and conformation that resources assigned to each processes fairly. For example, workloads share the 10 and 20 resources of systems so these works load divide the time slice of the system in the ratio of 1:2.As we already discussed thae response time of workloads in fair share scheduling for UNIX system is also important .There are also assigned the priorities to workloads[11].

D. Practical Fair Share scheduler

There have been also made extension in kernel scheduler of Linux titled as Practical Fair Share scheduler that is designed to meet the requirements of real-time processes. Fair share algorithm scheme in most common in real-time process with soft timelines. Practical fair scheduler is basically used for migration and placement techniques in symmetric multiprocessor or multicore systems. Migration technique is used in processes to balance the resources of the system among the processes that is automatically leads to better performance of system[22].

One describes the practical implementation of fair share scheduling on two different systems to check the performance by changing parameters of the fair share algorithm. One system is an industrial system and the other is university systems. Basically, first they developed the works loads model of these systems, analyzed these work model and applied fair share scheduling to these works models to get the experimental results. These Experimental results tell that whose system use the system resources “fairly” and performance of the fair share scheduling under these systems[15].

They described implementation through Moab fair share that includes one window of time and fair share scheduling parameters that assigns the dynamic priority to processes are Fair share interval, fair share depth and fair share decay shortly FS_INTERVAL, FS_DEPTH and FS_DECAY respectively. Windows of time describes the usage of systems shares by processes, FS_INTERVAL describes the each fair share window duration, FS_DEPTH describes the total numbers of past fair shares windows that are used in current fair share window calculations and FS_DECAY describes the weightage of each past fair share windows into current fair share windows[15][23].We can calculate the usage of each usage if we know the parameters of fair share scheduling[15].

E. Virtual Time Round Robin

One algorithm that is describing the fair share scheduling scheme is Virtual Time RR that is combination of fair queuing and RR algorithm. By using fair queuing technique, the process residing in the running queue according to their resources and fixes time quantum that is the main characteristic of RR is associated with each process. If the processes exceeds the limit of proportional shares resources then it put back to the start of the queue and its execution will start again[10].

F. Lottery scheduling

Another technique for fair scheduling is lottery scheduling that is basically proportional share resource allocations that uses the randomized resources reservation policy for efficient response time and provides control to the execution rates of the processes. Basically proportional share resource allocations algorithms are designed to address the problems of real time applications in operating systems[10][24]. Lottery scheduling also provides the modular technique for resource allocation in which modules protect the resources from one module to another module. Basically, there are many types of processes computations and each computation requires the system resources for completion of execution. In long time running computations such as critical analysis and scientific calculations required resources that behave differently as compare to other interactive computations such as database applications that required rapid response[13][25].Simulations ,scientific calculations and critical analysis basically are processer bounds so they required processor related resources while database applications are basically related to I/O and memory bound so the requirements of resources for every applications is different[26].

In lottery scheduling, the resource allocation to the processes or users on the base of lottery ticket. The term ticket can be used in the meaning of currency in computing. The user gets the share of system resources that is proportional to the hold tickets by user[26]. One user or process that required resources for execution of process must draw a lottery ticket. Only those processes can take part in the lottery that is in the running state while the processes are in the other states are not eligible for lottery[19]. Basically the system resources will be allotted for those users who win the lottery tickets. Lottery tickets represented the resources rights. The more chances to get the system resources for those users or processes that draw the more lotteries. The probability P to win the lottery ticket is simply t / T where t represents the ticket that a user hold and T is the total available tickets so the equation becomes P=t/T. If the user draws n identical lotteries so there is a number of chances to win the ticket is E [w] = np where E [w] represents the expected numbers of wins[13].One ticket is drawn to share the resources. For example, 5th ticket is selected as a result.

Process 1 have 3 tickets ($\sum 3 >$ 5th ticket) so result is false. Process 1 not allows to accessing the resources. Process 2 has 1 ticket ($\sum 3+\sum 1=\sum 4 >$ 5th ticket) so result is false. Process 2 is also not allowing for accessing the resources. Process 3 have 3 tickets ($\sum 3+\sum 1+\sum 3=\sum 7 >$ 5th ticket) so result is true. Now process 3 is allowed for accessing the resources

There have been many changes in the lottery scheduling algorithms to ensure its best performance to resource allocation management. One of these extensions in lottery scheduling is ticket exchange method in which processes exchange their resources with each other through ticket exchanges[26].

G. Stride scheduling

One of the scheduling approach that is stride scheduling-deterministic scheduling approach that is similar to the lottery scheduling. Stride scheduling is basically designed to achieve relative high throughout rate and lower response time[17].

H. Round Robin scheduler

The RR algorithm is also one of the scheduling approaches to determine the fast turnaround time where we have no confirmation of running times of processes. RR scheduling algorithms is basically falling in the category of time quantum in which a fixed quantum time q is assigned to each process and each process wish that they request for maximum time quantum to share the resources of the system. If the process completes its execution before the given quantum time then it must leave the system resources but if the process cannot complete the execution before the allotted given time quantum then it must be cycled back to end of queue and process must wait for another time quantum to start its execution again[2][27][28].

The running time of processes under Round Robin (RR) scheduling algorithm is assigned after when it has gained the quantum time. If the process requires more system resources for completion of its execution, then it put back to end of the queue. RR basically uses the priority that is a combination of running time and arrival time of process. The RR depends on the quantum size q that is allocated to each process so there are two situations of time quantum. If the time quantum is infinite then there will be First Come First Serve policy but if the time quantum is zero then we will be in limit and the processor have no waiting line so all processes executes[2][29][30].

III. PERFORMANCE EVOLUTION

Table 1 describes the evaluation of these scheduling algorithms through different scheduling parameters such as fairness, throughput, waiting time and response time as well as evaluate which algorithms can be used to achieve fairness, high throughput and also response time.

TABLE 1. Scheduling Algorithms Criteria

Scheduling Algorithms	**Scheduling Algorithms Criteria**				
	Extended Version	**Fairness**	**Throughput**	**Waiting Time**	**Response Time**
Fair Share Scheduling [11][15]		✓	High	High	High
	Fixed Budget Model[31]	✓	High	Low	Low
	Practical Fair Scheduler [22]	✓	High	Low	Low
	VM-aware fair Scheduler [32]	✓	High	Low	Low
Lottery Scheduling [26][13] [19]		✓	High	Low	High
	Stride Scheduling [17]	✓	High	Low	Low
	Ticket Exchange [26]	✓	Low	Low	High
Round Robin[4]		✓	High	High	High
	Virtual Time Round Robin[33]	✓	High	Low	High

We evaluate our work on the bases of following parameters as described

A. Fairness

Fairness defines the equal CPU time given to each process. More generally equal CPU time given to the processes on the priority and workload bases.

B. Throughput

In the scheduling context, throughput is the total amount of work or task that a specific computer can perform in specific time. Different computers have different through time on the base of computational power.

.

C. Waiting Time

Waiting time is the time that a process can wait in the ready queue for accessing specified resources.

D. Response Time

Response time defines the amount of time that it takes to get a first response after the request submitted for resources

TABLE 2. Merits and Demerits of Scheduling Algorithms

Scheduling Algorithms	Merits	Demerits
Fair Share Scheduling	• Priority based Scheduling • Allocate resources to process by dynamically • Provide fairness, both users and processes • Processes got some effective time for execution • Provide flexibility among processes	• Difficult to achieve fairness in multiprocessor systems • Response time and wait time is also critical issue
Lottery Scheduling	• Schedule resources on lottery tickets • A User who draws more lottery have chances to win more resources • Easy to understand and implemented • Uses proportional share resource allocation policy • Resource allocation is randomized to manage various resources • Stride scheduling that is extension of lottery scheduling supports modular resources management	• Behave differently for different computational models • User who draws less lotteries automatically less chances to get resources
Round Robin	• Quantum time is the main characteristic in RR that is allocated to each process • Behave good for short CPU burst • Follow the scheme of First Come First Served • Easy to understand • Also used priority that is combination of running time and arrival time	• Unfairness problem arises with different processes with different length • Can occur more context switches due to short quantum time

Table 2 shows the basic merits and demerits of the fair scheduling algorithms that impact on the working of these algorithms and differentiate one from another

IV. Conclusion

We conduct a survey in which we describe the working of different scheduling algorithms and how resource management issues in operating systems solved thorough these scheduling algorithms. These scheduling algorithms work well in different conditions and scenarios. We also describe the merits and demerits of scheduling algorithms and important role of this paper is comparisons of different scheduling algorithms and conclude the result of these algorithms according to scheduling algorithm criteria that is fairness, throughput, waiting and response time. Basically, all algorithms are designed to achieve fairness among processes, but the difference in throughput, response time and waiting time. In the future, we plan to extend this survey version for smart mobile phone operating systems and embedded device operating systems.

References

[1] H. Bui, W. Emeneker, A. Apon, D. Hoffman, and L. Dowdy, “Fairshare Scheduling – A Case Study,” Linux Cluster Institute (LCI) International Conference on High Performance Cluster Computing, Pittsburch, PA, March,pp.1-10, 2010.

[2] E. G. Coffman and L. Kleinrock, “Computer scheduling methods and their countermeasures,” Proc. April 30--May 2, spring Jt. Comput. Conf. - AFIPS ’68, p. 11, 1968.

[3 F.L. Liana and M. S. Squillante,"Time-function scheduling: a general approach to controllable resource," Vol. 29. No. 5. ACM, 1995.

[4] S. T. Leutenegger and M. K. Vernon, “The Performance of iMultiprogrammed Multiprocessor Scheduling Policies,” pp. 226–236, 1990.

[5] J. Zahorjan and C. McCann, “Processor scheduling in shared memory multiprocessors,” ACM SIGMETRICS Perform. Eval. Rev., vol. 18, no. 1, pp. 214–225, April. 1990.

[6] T. Helmy and A. Dekdouk,"Burst Round Robin as a Proportional-share Scheduling Algorithm," IEEEGCC, 2007.

[7] J. Kay and P. Lauder,"A fair share scheduler," Commun. ACM 31, pp. 44-55 ,1988

[8] M. Shreedhar and G. Varghese, “Efficient fair queuing using deficit round-robin,” IEEE/ACM Trans. Netw., vol. 4, no. 3, pp. 375–385, Jun. 1996.

[9] S. Svr, “Process scheduling,” 1980.

[10] A. Silberschatz, P. B. Galvin, G. Gagne,"Operating system concepts," (Vol. 4). Reading: Addison-Wesley, 1998.

[11] S.C. Mana,"Recourse Management Using a Fair Share Scheduler," International Journal of Computer Science and Security (IJCSS),vol.6, pp. 29-33, 2012.

[12 E. Bolker and Y. Ding," On the performance impact of fair share scheduling," In Int. CMG Conference , pp.71-82, 2000.

[13] C. Waldspurger and W. Weihl, “Lottery scheduling: Flexible proportional-share resource management,” Proc. 1st USENIX Conf, vol. 19, pp. 1–11, 1994.

[14] G.J. Henry,"The unix system: the fair share scheduler," AT&T Bell Laboratories Technical Journal, 63(8), pp. 1845-1857, 1984.

[15] T. Kimbrel, B. Schieber, and M. Sviridenko, “Minimizing migrations in fair multiprocessor scheduling of persistent tasks,” J. Sched., vol. 9, no. 4, pp. 365–379, Aug. 2006.

[16] C. A. Waldspurger and W. E. Weihl, “Stride Scheduling : Deterministic Proportional-Share Resource Management,” 1995.

[17] K. Jeffay, F.D. Smith, A. Moorthyand J. Anderson,"Proportional share scheduling of operating system services for real-time applications," The 19th IEEE Proceedings In Real-Time Systems Symposium,pp. 480-491,1998.

[18] D. Petrou, J. W. Milford, and G. A. Gibson, “Implementing Lottery Scheduling : Matching the Specializations in Traditional Schedulers Implementing Lottery Scheduling : Matching the Specializations in Traditional Schedulers,” In USENIX Annual Technical Conference, pp.1-14, 1999.

[19] D. H. J. Epema and J. F. C. M. de Jongh, “Proportional-share scheduling in single-server and multiple-server computing systems,” ACM SIGMETRICS Perform. Eval. Rev., vol. 27, no. 3, pp. 7–10, Dec. 1999.

[20] C.-S. Shih, J.-W. Wei, S.-H. Hung, J. Chen, and N. Chang, “Fairness scheduler for virtual machines on heterogonous multi-core platforms,” ACM SIGAPP Appl. Comput. Rev., vol. 13, no. 1, pp. 28–40, Mar. 2013.

[21] D. Choffnes, M. Astley, and M. J. Ward, “Migration policies for multi-core fair-share scheduling,” ACM SIGOPS Oper. Syst. Rev., vol. 42, no. 1, p. 92, Jan. 2008.

[22] M. cluster Suite., Http://www.clusterresources.com/pages/products/moabcluster-, and M. website. Suite.php, “No Title.”,2008

[23] E. Hoque and T. Dey, “Comparing Lottery and EEVDF Scheduling Algorithm for Real-time Applications,” Hoque, E., & Dey, T. (2009). Comparing Lottery and EEVDF Scheduling Algorithm for Real-time Applications. Charlottesville, VA: University of Virginia, Department of Computer Science School of Engineering and Applied Sciences.2009.

[24] D. Zepp, "Lottery scheduling in the Linux kernel: A closer look." PhD diss., California Polytechnic State University, San Luis Obispo, 2012.

[25] D. G. Sullivan, R. Haas, and M. I. Seltzer, “Tickets and currencies revisited: extensions to multi-resource lottery scheduling,” Proc. Seventh Work. Hot Top. Oper. Syst., pp. 148–152, 1999.

[26] H. Zhong, K. Tao, and X. Zhang, “An Approach to Optimized Resource Scheduling Algorithm for Open-Source Cloud Systems,” 2010 Fifth Annu. ChinaGrid Conf., pp. 124–129, Jul. 2010.

[27] M. Kawser, “Performance Comparison between Round Robin and Proportional Fair Scheduling Methods for LTE,” Int. J. Inf. Electron. Eng., vol. 2, no. 5, 2012.

[28] P. R. Mohanty, P. H. S. Behera, K. Patwari, M. Dash, and M. L. Prasanna, “Priority Based Dynamic Round Robin (PBDRR) Algorithm with Intelligent Time Slice for Soft Real Time Systems,” vol. 2, no. 2, pp. 46–50, 2011.

[29] S. Hiranwal and D.K. Roy,"Adaptive round robin scheduling using shortest burst approach based on smart time slice," International Journal of Computer Science and Communication, pp. 319-323, 2011.

[30] M. Isard, V. Prabhakaran, J. Currey, U. Wieder, K. Talwar, and A. Goldberg, “Quincy : Fair Scheduling for Distributed Computing Clusters,” pp. 261–276, 2009.

[31] J. Nieh, C. Vaill and H. Zhong," Virtual-Time Round-Robin: An O (1) Proportional Share Scheduler," In USENIX Annual Technical Conference on General Track , pp. 245-259, 2011.

[32] C.S. Shih, J.W. Wei, S.H. Hung, N. Chang and J.Chen," A VM-aware fairness scheduler on heterogenous multi-core platforms," In Proceedings of the 2012 ACM Research in Applied Computation Symposium, pp. 409-415.2012.