ISSN 1512-1232

HYBRID LOTTERY MULTI-LEVEL QUEUE SCHEDULING WITH A MARKOVIAN MODEL

Shweta Ojha* & Saurabh Jain**, Diwakar Shukla***

*Department of Computer Science & Applications, Sagar University, Sagar M.P. 470003, India.
e-mail: meshwetaojha@gmail.com
**Department of MCA, Shri Vaishnav Institute of Technology and Science, Indore M.P. 454331, India.
e-mail: iamsaurabh_4@yahoo.co.in
***Department of Mathematics and Statistics, Sagar University, Sagar M.P. 470003, India.
e-mail: diwakarshukla@rediffmail.com

Abstract

Scheduling computations in multi-environment systems is a complex and challenging problem. Scarce resources must be multiplexed to service requests of varying importance. Lottery scheduling is a randomized mechanism that provides responsive control over the relative execution rates of computations. But, once the tickets are allotted to the competing jobs, they are forced to move in the defined pattern. This paper introduces the concept of Multileveled queues with lottery scheduling which will provide options for further upgradation to the processes which are in execution. A Markov Chain model based study has been done to deal with transitions of processes into processor under this proposed scheme. A data model based simulation study is performed to analyze the result.

Keywords: *Markov Chain Model, Hybrid Lottery Multilevel Queue Scheduling, Transition Probability Matrix.*

1.0 INTRODUCTION

A scheduling problem exists whenever there is a shared resource that needs to be arbitrated among multiple requesting entities. However, even though the problem is general, its solution may be very different depending on the specific context it is tailored on. The day by day increasing demand on web applications has created a lot of opening for further development of new scheduling techniques and also enhancements in the pre-existing schemes. The policy chosen to manage this multi-user, multitasking environment can have an enormous impact on throughput and response time. Accurate control over the quality of service provided to users and applications requires support for specifying relative computation rates. Such control is desirable across a wide spectrum of systems.

Few general-purpose schemes even come close to supporting flexible, responsive control over service rates. They rely upon a simple notion of priority that does not provide the encapsulation and modularity properties required for the engineering of large software systems. There are several scheduling algorithms in literature that can be classified into conservative and non-conservative. The first ones transmit packets when there are any waiting in the queue, the server is never idle, and they have the lowest delays. The second ones are less efficient, since they cannot transmit packets, even if there are several waiting in the queue, and therefore they do not fully exploit the link capacity, and they also have real time restrictions in their implementation. One of the parameter to take into consideration in the fairness in which bandwidth is allocated to sessions, ideally being this allocating homogeneous between sessions, avoiding in this bursts and providing a better network performance. However, lottery scheduling is a randomized mechanism that provides responsive control over the relative execution rates of computations and efficiently implements proportional-share resource management, that means, the resource consumption rates of active computations are proportional to the relative shares that they are allocated. In spite of all these features, a lack in performance arises when we find a long calculative job like scientific applications and simulations

ISSN 1512-1232

being hitched in between many less important small jobs are due to Hellerstein [1]. It is experienced that Resource management mechanism employed by a server-operating system should have several desirable properties like throughput and response time and many others. Existing fair share schedulers by Henry [2], Kay and Lauder [3] and microeconomic schedulers by Ferguson et al. [4] and Waldspurger et al [5], successfully address some of the problems with absolute priority schemes. Shukla et al. [6] has also worked on lottery scheduling for the estimation of ready queue estimation time.

Some more work has been done in the area relating various scheduling operations and the stochastic modeling technique i.e. Markov chain model. Shukla and Jain [7] has developed a Markov chain model for the multilevel queue scheduler. Shukla, et al. [8,9,10] has highlighted the analogue between supportive divisible load scheduling and Markov chain model in linear daisy chain network and single level tree network. Markov chain based analysis has also been harvested in the field of networking by Shukla et al. [11], Shukla and Thakur [12] and Shukla and Tiwari [13]. Some other contributions are due to Sullivan et al. [14], Evans et al. [15] and Petro und Milford [16].

However, the assumptions and overheads associated with these systems limit them to relatively coarse control over long-running computations. For those systems, we have proposed this concept of introducing multilevel queue with lottery scheduling scheme. The arrival of new jobs happens in waiting queue. The scheduler allots lottery tickets to the waiting jobs as per lottery scheduling. In our scheme, these tickets are related with the multileveled queues that are followed after it, highlighting the fact that to which queue which of the packet will move. A priority scheme has also been set in between the different queues of Multi level scheme. In the special case, if any of the specific queue with higher priority is empty, then the jobs in the less priority has the option to get upgraded. With this, they get a higher chance for being processed.

2.0 2.0 GENERAL CLASS OF MULTI-LEVEL QUEUE SCHEDULING

A multilevel queue scheduling algorithm partitions the ready queue in several separate queues. Scheduling processes are permanently assigned to one queue, based on some property of the process, such as memory size, process priority or process type etc fig 2.1.

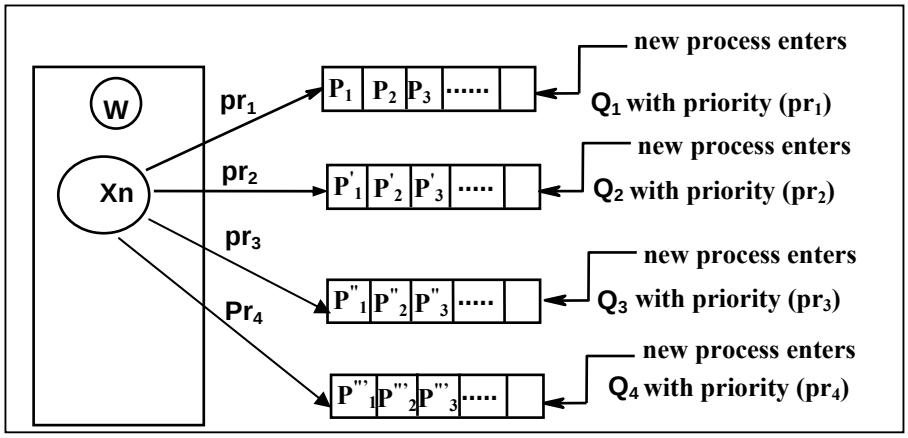

Fig. 2.1 (A General Multi-level Queue System Diagram)

Shukla and Jain [7] has vital contribution in the field of modelling of multi-level queue scheduling algorithm and also for other scheduling algorithms. They have taken into account a class of multilevel queue scheduling schemes with the assumption of random jumps of scheduler and a rest state along with keen interest on comparative study of different schemes.

The basic idea behind multilevel queue is that in the corresponding system there are several queues, each has a level which corresponds to a level of priority, each level has its own scheduler,

ISSN 1512-1232

there is one main scheduler which handles the scheduler for each queue. So, suppose:

Queue 1: High Priority, Shortest Job First
Queue 2: Medium Priority, Round-Robin
Queue 3: Low Priority, Round Robin

Then overall queues could be a priority queue so the problem with this is that once a job is put on a queue, it must remain there until it completes. This could create some problems with CPU hogging and starvation. If a very long job is put on Q1, it could hog all of the time given to Queue 1, preventing the other jobs from running. This is bad because jobs in Q1 are high priority.

3.0 3.0 LOTTERY SCHEDULING SCHEME

Lottery scheduling is a randomized resource allocation mechanism. Resource rights are represented by lottery tickets. Each allocation is determined by holding a lottery; the resource is granted to the client with the winning ticket (Waldspurger and Weihl).

This effectively allocates resources to competing clients in proportion to the number of tickets that they hold. Since any client with a non-zero number of tickets will eventually win a lottery, the conventional problem of starvation does not exist. The lottery mechanism also operates fairly when the number of clients or tickets varies dynamically. For each allocation, every client is given a fair chance of winning proportional to its share of the total number of tickets. Since any changes to relative ticket allocations are immediately reflected in the next allocation decision, lottery scheduling is extremely responsive. Scheduling by lottery is probabilistically fair. Meaning by that, all the processes in the waiting queue have the equal chance for getting being processed. The expected allocation of resources to clients is proportional to the number of tickets that they hold. Since the scheduling algorithm is randomized, the actual allocated proportions are not guaranteed to match the expected proportions exactly. However, the disparity between them decreases as the number of allocations increases. David Petrou and John W. Milford has studied dynamic ticket adjustments that influence scheduling order while preserving CPU utilization targets and they have achieve throughput and responsiveness nearly equal to the other specific scheduler like FreeBSD schedulers [18]. Recent work from Sullivan et al. introduces application-specific “negotiators” that enable automatic ticket exchanges between processes desiring different resource allocations [19].

A straightforward way to implement a centralized lottery scheduler is to randomly select a winning ticket, and then search a list of clients to locate the client holding that ticket. This requires a random number generation to traverse a client list of length n, accumulating a running ticket sum until it reaches the winning value. An example list-based lottery is presented in Figure 3.1.

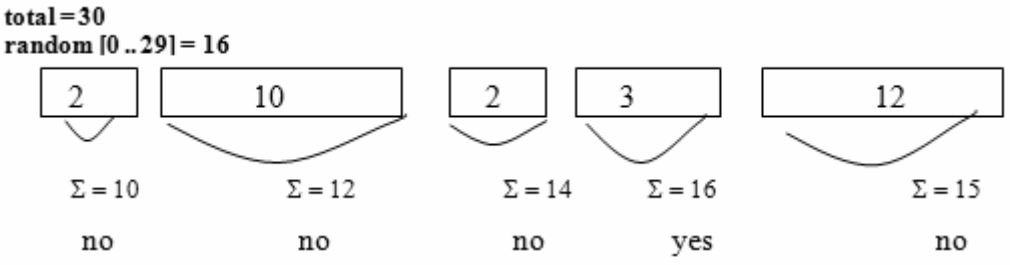

Figure 3.1: **Example Lottery.** Five clients compete in a list-based lottery with a total of 30 tickets. The sixteenth ticket is randomly selected, and the client list is searched for the winner. A running ticket sum is accumulated until the winning ticket value is reached.

ISSN 1512-1232

However, Hybrid lottery scheduling heuristically identifies and rewards interactive processes by how much of their allocated CPU time they consume. However, some interactive processes, such as those that render graphics, also consume moderate amounts of CPU. Evans et al. suggest several methods based on past user action and window manager cooperation for an operating system to recognize interactive [17]. Working on the same line, we are proposing a hybrid of lottery with Multilevel queue scheduling scheme so that we could incorporate the randomization of lottery with the assured performance of other. Here, we are developing this model with the help of some assumptions that is being discussed.

3.1 ASSUMPTIONS OF THE MODEL

Considering a Multilevel queue scheduling scheme with three queues Q_2, Q_3 and Q_4 as active queue for processing the jobs. These queues are differentiated from each other on the basis of priority, preference, timestamps weight or any of such basic characteristic features which somehow or the other , effect the overall performance of the system. All these queues are getting input from proposed weighting queue denoted as W. this is the queue where new arrival of the processes takes place. This weighting queue allots a randomized lottery tickets are allotted to each of the upcoming processes. These lottery tickets decide the allocation of that particular job to a specific queue for execution. Here, the concept of quantum is also taken into consideration which is specified as a small pre-defined slot of time given for processing to the expecting processes in the various queues. Symbol n denotes the n^{th} quantum allotted by the scheduler to a process for execution (n = 1, 2, 3……). The assumptions of the model are:

1. All the new processes will arrive only at the queue W, which is denoted as waiting queue.
2. The scheduler has a random movement over the queues.
3. The swapping between the queues takes place on the basis of their specified quantum.
4. The scheduler starts processing of any queue Q_i with probability Pr_i (i=1,2,3….), then picks up the first process of that queue and allots a quantum for processing.
5. Processes remains with the processor until the quantum is over. If it completes within that, then gets out of that Q_i.
6. Within quantum, if a process did not completes, it moves to the waiting queue where scheduler assigns a new lottery ticket to the remaining process as a fresh packet which will further guide it to its completion.
7. State Q_1 and Q_5 are denoted as waiting state and end state.
8. Q_5 that is specified as end state is an absorbing state.
9. Similar to the behavior of the Multilevel queue, quantum allotment procedure, within Q_i , by the scheduler continue until Q_i is empty. The scheduler jumps from any state to any other state at the end of a quantum.

3.2 MARKOV CHAIN MODEL

Let $\{X^{(n)}, n \geq 1\}$ be a Markov chain where $X^{(n)}$ denotes the state of the scheduler at the n^{th} quantum of time. The state space for $X^{(n)}$ is $\{Q_1, Q_2, Q_3, Q_4, Q_5\}$ where scheduler X moves stochastically over these in different quantum. Predefined initial selection probabilities of states are:

$$\left.\begin{aligned} P[X^{(0)} = Q_1] &= pr_1 \\ P[X^{(0)} = Q_2] &= pr_2 \\ P[X^{(0)} = Q_3] &= pr_3 \\ P[X^{(0)} = Q_4] &= pr_4 \\ P[X^{(0)} = Q_5] &= pr_5 \end{aligned}\right] \quad \text{.....(3.2.1)}$$

ISSN 1512-1232

with $pr_1 + pr_2 + pr_3 + pr_4 + pr_5 = \sum_{i=1}^{5} pr_i = 1$.

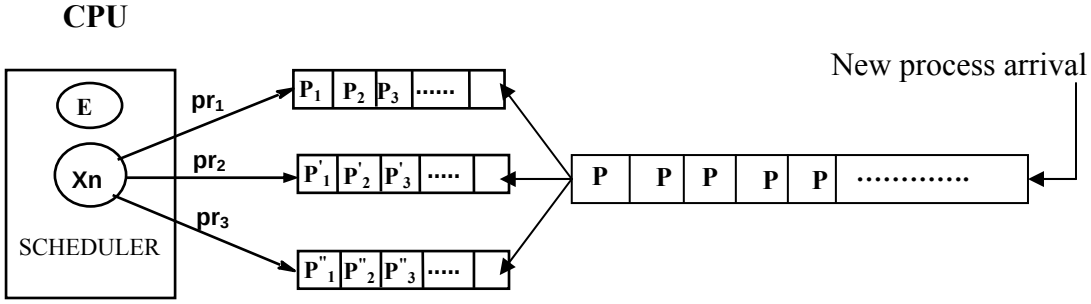

Fig. 3.2.1 (Hybrid Multi-level lottery Queue System Diagram)

Let s_{ij}(i,j=1,2,3,4,5) be the transition probabilities of scheduler over five proposed states then unit-step transition probability matrix for $X^{(n)}$ is

$s_{ij} = P\left[X^{(n)} = Q_i / X^{(n-1)} = Q_j\right], i + j$;

$X^{(n)}$

$X^{(n-1)}$	Q_1	Q_2	Q_3	Q_4	Q_5
Q_1	s_{11}	s_{12}	s_{13}	s_{14}	s_{15}
Q_2	s_{21}	s_{22}	s_{23}	s_{24}	s_{25}
Q_3	s_{31}	s_{32}	s_{33}	s_{34}	s_{35}
Q_4	s_{41}	s_{42}	s_{43}	s_{44}	s_{45}
Q_5	s_{51}	s_{52}	s_{53}	s_{54}	s_{55}

…….(3.2.2)

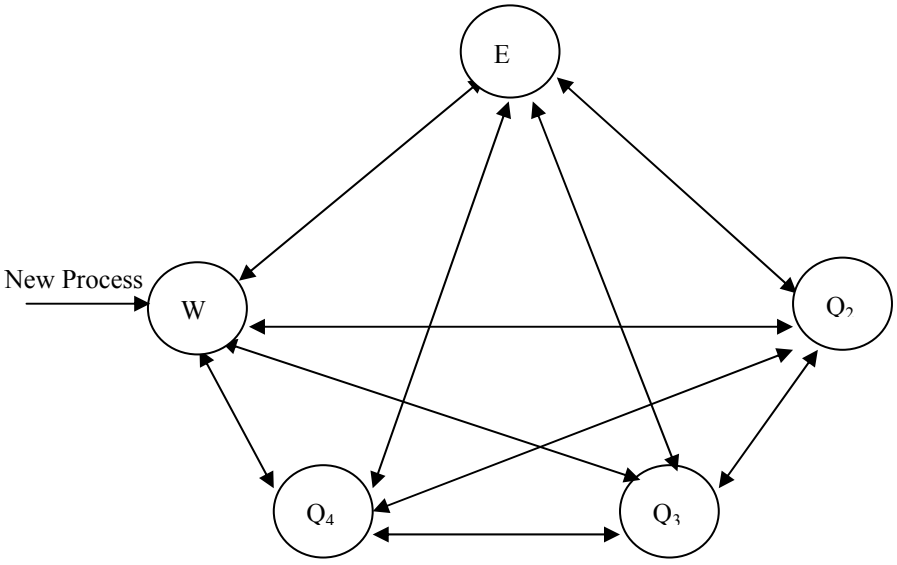

Fig. 3.2.2 (Unrestricted Transition Diagram)

ISSN 1512-1232

3.3 SPECIFICATIONS FOR THE PROPOSED MODEL

a Up gradation of the processes in the lower order queues if thee upper order queues are empty. This will further provide a way to harness the availability of a resource that is rarely available.

b A new process only enters at W.

c Transition may take place from W to D but not the other way. That will actually signify the condition when the provided timestamp is exhausted or, in case of system failure or sudden shut down.

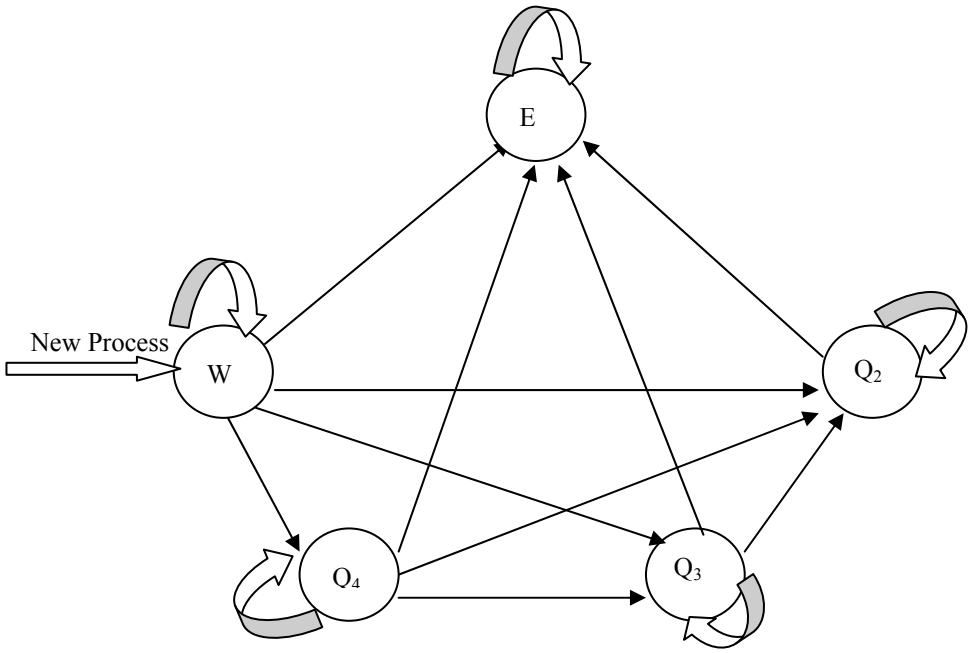

Fig. 3.3.1 (Proposed Restricted Transition Diagram)

Now, again for this restricted state, Let s_{ij}(i,j=1,2,3,4,5) be the transition probabilities of scheduler over five proposed states then unit-step transition probability matrix for $X^{(n)}$ is $s_{ij} = P\left[X^{(n)} = Q_i / X^{(n-1)} = Q_j\right], i + j$; considering the case as Q_1 as the waiting state and Q_5 as the end state. The transition probability matrix will be:-

$X^{(n-1)}$ \ $X^{(n)}$	Q_1	Q_2	Q_3	Q_4	Q_5
Q_1	s_{11}	s_{12}	s_{13}	s_{14}	s_{15}
Q_2	0	s_{22}	0	0	s_{25}
Q_3	0	s_{32}	s_{33}	0	s_{35}
Q_4	0	s_{42}	s_{43}	s_{44}	s_{45}
Q_5	0	0	0	0	s_{55}

(3.3.1)

$$\text{subject to condition } s_{15} = \left(1 - \sum_{i=1}^{4} s_{1i}\right),\ s_{25} = \left(1 - \sum_{i=1}^{4} s_{2i}\right),$$
$$s_{35} = \left(1 - \sum_{i=1}^{4} s_{3i}\right),\ s_{45} = \left(1 - \sum_{i=1}^{4} s_{4i}\right), s_{55} = \left(1 - \sum_{i=1}^{4} s_{4i}\right) \quad \text{(3.3.2)}$$

ISSN 1512-1232

and $0 \le s_{ij} \le 1$.

The state probabilities, after first quantum can be obtained by a simple relationship:

$$\begin{aligned} P\left[X^{(1)} = Q_1\right] = {} & p\left[X^{(0)} = Q_1\right] p\left[X^{(1)} = Q_1 / X^{(0)} = Q_1\right] + p\left[X^{(0)} = Q_2\right] p\left[X^{(1)} = Q_1 / X^{(0)} = Q_2\right] \\ & + p\left[X^{(0)} = Q_3\right] p\left[X^{(1)} = Q_1 / X^{(0)} = Q_3\right] + p\left[X^{(0)} = Q_4\right] p\left[X^{(1)} = Q_1 / X^{(0)} = Q_4\right] \\ & + p\left[X^{(0)} = Q_5\right] p\left[X^{(1)} = Q_1 / X^{(0)} = Q_5\right] \end{aligned}$$

$$\left.\begin{aligned} &= \sum_{i=1}^{5} pr_i s_{i1} \\ P\left[X^{(1)} = Q_2\right] &= \sum_{i=1}^{5} pr_i s_{i2} \\ P\left[X^{(1)} = Q_3\right] &= \sum_{i=1}^{5} pr_i s_{i3} \\ P\left[X^{(1)} = Q_4\right] &= \sum_{i=1}^{5} pr_i s_{i4} \\ P\left[X^{(1)} = Q_5\right] &= \sum_{i=1}^{5} pr_i s_{i5} \end{aligned}\right] \qquad (3.3.3)$$

Similarly, the state probabilities after the second quantum could be obtained by simple relationship:

$$\left.\begin{aligned} P\left[X^{(2)} = Q_1\right] &= \sum_{j=1}^{5} \left(\sum_{i=1}^{5} pr_i s_{ij}\right) s_{j1} \\ P\left[X^{(2)} = Q_2\right] &= \sum_{j=1}^{5} \left(\sum_{i=1}^{5} pr_i s_{ij}\right) s_{j2} \\ P\left[X^{(2)} = Q_3\right] &= \sum_{j=1}^{5} \left(\sum_{i=1}^{5} pr_i s_{ij}\right) s_{j3} \\ P\left[X^{(2)} = Q_4\right] &= \sum_{j=1}^{5} \left(\sum_{i=1}^{5} pr_i s_{ij}\right) s_{j4} \\ P\left[X^{(2)} = Q_5\right] &= \sum_{j=1}^{5} \left(\sum_{i=1}^{5} pr_i s_{ij}\right) s_{j5} \end{aligned}\right] \qquad (3.3.4)$$

Remark 3.3.1 In the similar fashion, for n quantum, the generalized expression:

$$\left.\begin{aligned} P\left[X^{(n)} = Q_1\right] &= \sum_{m=1}^{5} \ldots\ldots \sum_{t=1}^{5} \sum_{k=1}^{5} \left\{ \sum_{j=1}^{5} \left(\sum_{i=1}^{5} pr_i s_{ij} \right) s_{jk} \right\} s_{kt} \ldots\ldots s_{m1} \\ P\left[X^{(n)} = Q_2\right] &= \sum_{m=1}^{5} \ldots\ldots \sum_{t=1}^{5} \sum_{k=1}^{5} \left\{ \sum_{j=1}^{5} \left(\sum_{i=1}^{5} pr_i s_{ij} \right) s_{jk} \right\} s_{kt} \ldots\ldots s_{m2} \\ P\left[X^{(n)} = Q_3\right] &= \sum_{m=1}^{5} \ldots\ldots \sum_{t=1}^{5} \sum_{k=1}^{5} \left\{ \sum_{j=1}^{5} \left(\sum_{i=1}^{5} pr_i s_{ij} \right) s_{jk} \right\} s_{kt} \ldots\ldots s_{m3} \\ P\left[X^{(n)} = Q_4\right] &= \sum_{m=1}^{5} \ldots\ldots \sum_{t=1}^{5} \sum_{k=1}^{5} \left\{ \sum_{j=1}^{5} \left(\sum_{i=1}^{5} pr_i s_{ij} \right) s_{jk} \right\} s_{kt} \ldots\ldots s_{m4} \\ P\left[X^{(n)} = Q_5\right] &= \sum_{m=1}^{5} \ldots\ldots \sum_{t=1}^{5} \sum_{k=1}^{5} \left\{ \sum_{j=1}^{5} \left(\sum_{i=1}^{5} pr_i s_{ij} \right) s_{jk} \right\} s_{kt} \ldots\ldots s_{m5} \end{aligned}\right] \qquad (3.3.5)$$

ISSN 1512-1232

4.0 NUMERICAL ILLUSTRATIONS USING DATA SET CREATED WITH THE HELP OF MATHEMATICAL MODEL

The basic and scientific approach for data analysis related to state transition probabilities is managed by a data model with two parameters α and d. The i stands for number of queues.

$X^{(n)}$

$X^{(n-1)}$	Q_1	Q_2	Q_3	Q_4	Q_5
Q_1	α	α +d.i	α +2d.i	α +3d.i	1-(4 α +6d.i)
Q_2	α+d.i	α +2d.i	α +3d.i	α +4d.i	1-(4 α +10d.i)
Q_3	α+2d.i	α +3d.i	α +4d.i	α +5d.i	1-(4 α +14d.i)
Q_4	α+3d.i	α +4d.i	α +5d.i	α +6d.i	1-(4 α +18d.i)
Q_5	α+4d.i	α +5d.i	α +6d.i	α +7d.i	1-(4 α +22d.i)

5.0 GRAPHS

5.1 Case I with α=0.1

Graph for α=0.1 and d=0.002

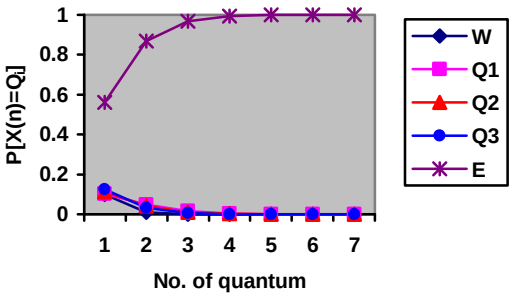

Fig5.1.1 (α=0.1, d=0.002)

Graph for α=0.1 and d=0.004

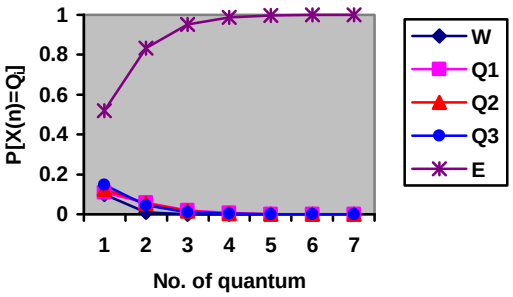

Fig5.1.2 (α=0.1, d=0.004)

Graph for α=0.1 and d=0.006

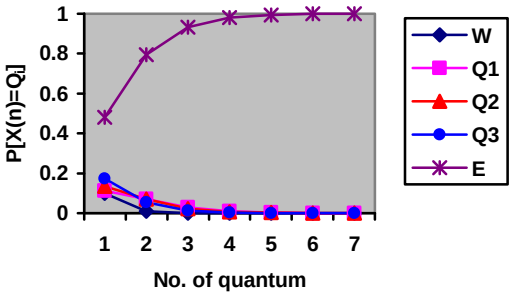

Fig5.1.3 (α=0.1, d=0.006)

Graph for α=0.1 and d=0.008

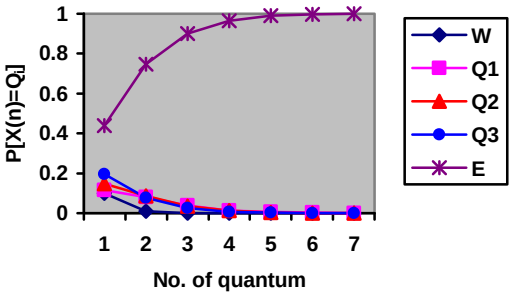

Fig5.1.4(α=0.1, d=0.008)

ISSN 1512-1232

5.2 Case II with α=0.12

Graph for α=0.12 and d=0.002

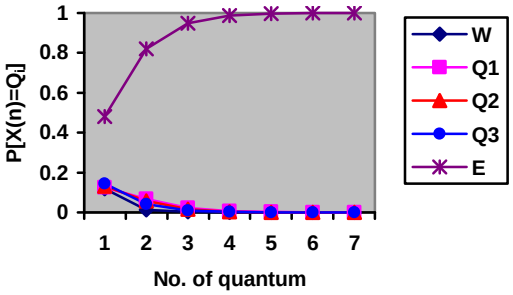

Fig5.2..1 (α=0.12, d=0.002)

Graph for α=0.12 and d=0.004

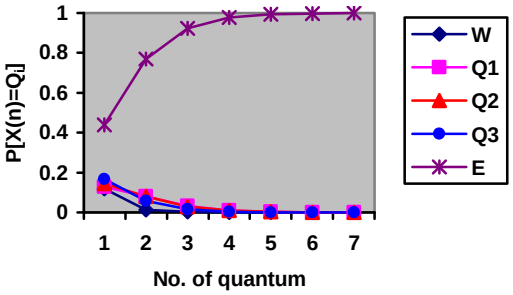

Fig5.2.2 (α=0.12, d=0.004)

Graph for α=0.12 and d=0.006

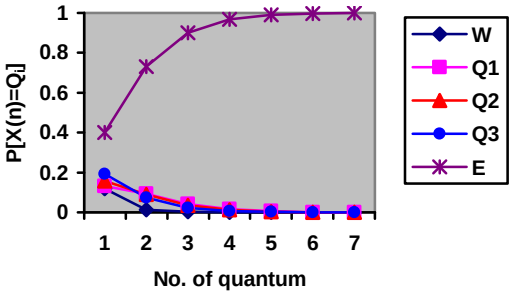

Fig5.2.3 (α=0.12, d=0.006)

Graph for α=0.12 and d=0.008

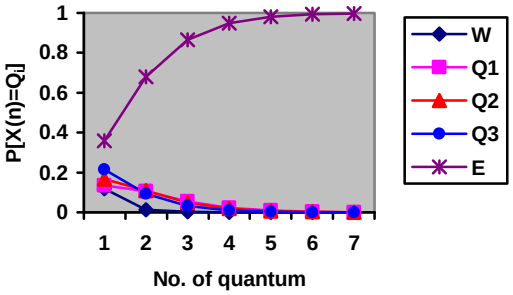

Fig5.2.4 (α=0.12, d=0.008)

5.3 Case III with α=0.14

Graph for α=0.14 and d=0.002

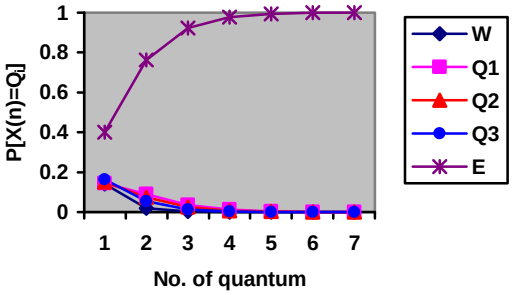

Fig5.3.1 (α=0.14, d=0.002)

Graph for α=0.14 and d=0.004

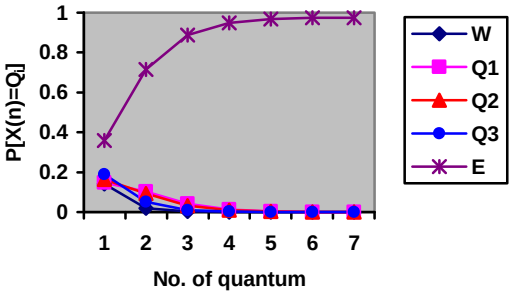

Fig5.3.2 (α=0.14, d=0.004)

Graph for α=0.14 and d=0.006

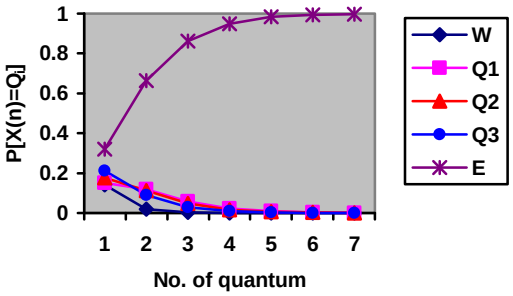

Fig5.3.3 (α=0.14, d=0.006)

Graph for α=0.14 and d=0.008

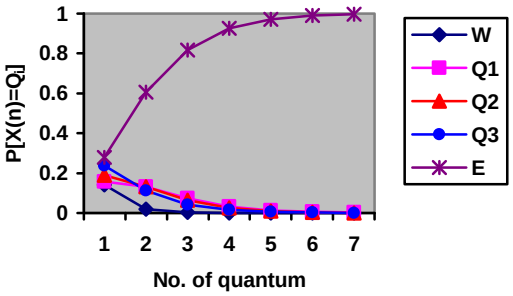

Fig5.3.4 (α=0.14, d=0.008)

ISSN 1512-1232

5.4 Case IV with α=0.16

Graph for α=0.16 and d=0.002

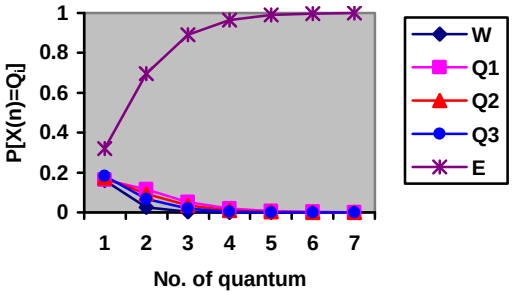

Fig5.4.1 (α=0.16, d=0.002)

Graph for α=0.16 and d=0.004

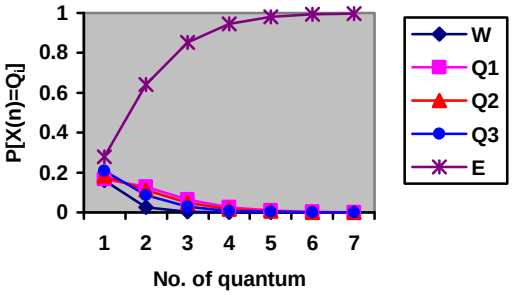

Fig5.4.2 (α=0.16, d=0.004)

Graph for α=0.16 and d=0.006

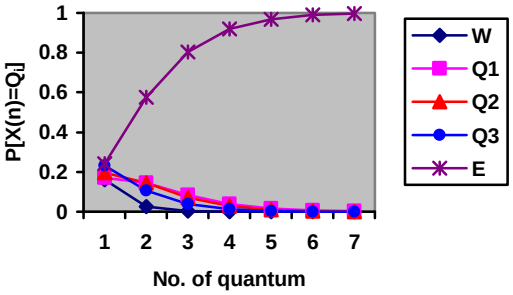

Fig5.4.3 (α=0.16, d=0.006)

Graph for α=0.16 and d=0.008

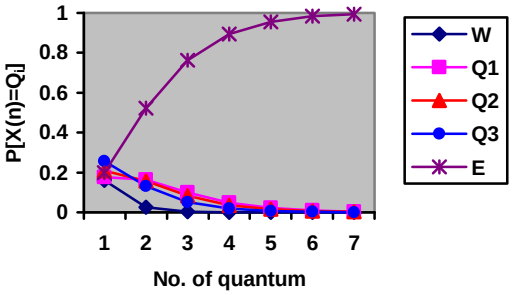

Fig5.4.4 (α=0.16, d=0.008)

5.5 Case V with α=0.18

Graph for α=0.18 and d=0.002

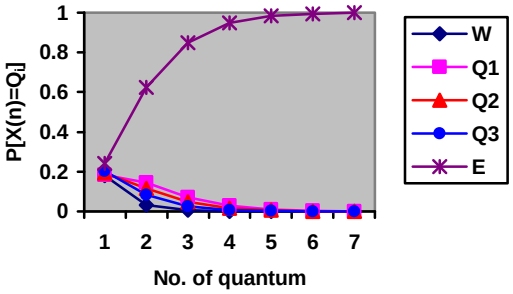

Fig5.1.1 (α=0.18, d=0.002)

Graph for α=0.18 and d=0.004

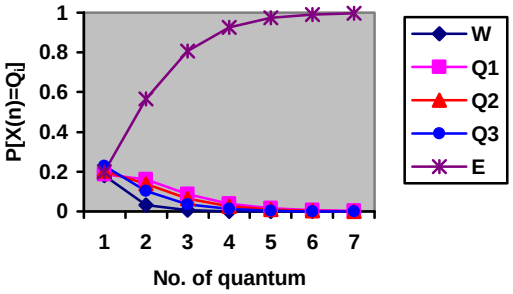

Fig5.5.2 (α=0.18, d=0.004)

Graph for α=0.18 and d=0.006

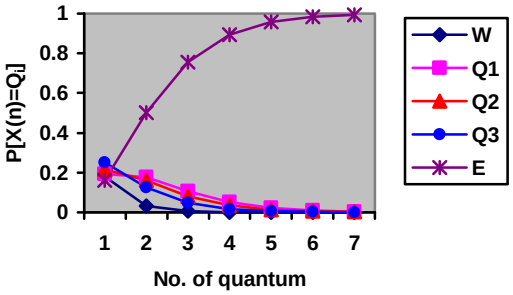

Fig5.5.3 (α=0.18, d=0.006)

Graph for α=0.18 and d=0.008

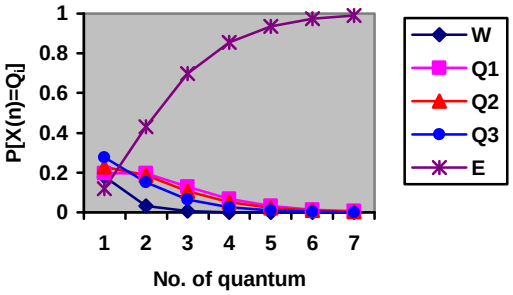

Fig5.5.4 (α=0.18, d=0.008)

ISSN 1512-1232

6.0 DISCUSSION ON GRAPHS

It is remarkably clear in all the graphs that the line depicting End state is above the rest of all states that indicates the smooth and continuous termination of the processes within this proposed system. Secondly, the starting point of the End state is initially showing a great deal of difference then the rest of the states but it is gradually coming down. In condition IV of the case III, it is showing unification with other lines. That means, as the system proceeds, the process completion starts with the initiation of the system too. One more difference could be observed that the gradual segregation of all the four states with increasing value of α and d in the beginning but, as the number of quantum increases, they again show a consolidation. It proves that all the queues other then the absorbing state show similar pattern at higher quantum. Even the waiting queue is showing the similar pattern. That means, this hybrid system of multi level queue and lottery queue is adding the concept of lottery packets without adversely affecting the effectiveness of the system. Other then that, however it is also proposing a chance of appraisal in case of empty processing queue there by, providing means to harness or we can say, better utilize a rare resource that is, processor time.

7.0 CONCLUDING REMARKS

With the help of data calculated on the given assumptions and the graphs obtained on them, it is coming into notice that this proposed hybrid lottery multilevel queue scheduling scheme is having all the features of the randomized nature of the lottery scheduling whereas, on the other hand, it is also harnessing the unbiased characteristic feature of the multilevel queue scheduling also. Both these features of this new system are coming into light without affecting the overall performance of the system. Principle technique used by this system is a dynamic ticket adjustments that influence scheduling order while preserving CPU utilization targets.

8.0 FUTURE WORK

All the data used here are derived according to the behavioral nature of this proposed system. It further requires the applied data values for authentication. On the basis of the proposed assumptions, a scheduler can be designed and these data are further investigated for future use.

REFERENCES

1. [Hel93] J. L. Hellerstein. “Achieving Service Rate Objectives with Decay Usage Scheduling,” *IEEE Transactions on Software Engineering*, August 1993.
2. [Hen84] G. J. Henry. “The Fair Share Scheduler,” *AT&T Bell Laboratories Technical Journal*, October 1984.
3. [Kay88] J. Kay and P. Lauder. “A Fair Share Scheduler,” *Communications of the ACM*, January 1988.
4. [Fer88] D. Ferguson, Y. Yemini, and C. Nikolaou. “Microeconomic Algorithms for Load-Balancing in Distributed Computer Systems,” *International Conference on Distributed Computer Systems*, 1988.
5. [Wal92] C. A. Waldspurger, T. Hogg, B. A. Huberman, J. O. Kephart, and W. S. Stornetta. “Spawn: A Distributed Computational Economy,” *IEEE Transactions on Software Engineering*, February 1992.
6. Shukla, D., Jain, Anjali and Chaudhary, A. : Estimation Of Ready Queue Processing Time Under Usual Group Lottery Scheduling (GLS) In Multiprocessor Environment, *International Journal of Advanced Networking and Applications,* Vol.8, No.14, pp. 39-45, 2010.
7. Shukla, D. and Jain, S. : A Markov chain model for Multi-Level queue scheduler in operating system, *Proceedings of the International Conference on Mathematics and Computer Science, ICMCS-07, pp. 522-526, 2007.*

ISSN 1512-1232

8. Shukla, D., Jain, S. and Ojha, S.: Analysis of Multilevel Queue with the Effect of Data Model Approach, *Proceedings of National Conference on Research And Development Trends in ICT (RDTICT-2010),* Lucknow University, 2010 a.
9. Shukla, D., Jain, S. and Ojha, S.: Supportive Divisible Load Scheduling and Markov Chain Model*, Presented in International Conference for Forum of Interdisciplinary Mathematics,* 2010 b.
10. Shukla, D., Jain, S. and Ojha, S.: A Comparative Study in the Behavior of Single Level Tree Network and Markov Chain Model, *Presented in International Conference for Forum of Interdisciplinary Mathematics*, 2010 c.
11. Shukla, D., Gadewar, S.K. and Pathak, R.K.: A Stochastic model for space-division switches in Computer Networks, *Applied Mathematics and Computation (Elsevier Journal), Vol. 184, No. 2, pp. 235-269,* 2007.
12. Shukla, D. and Thakur, S.: State Probability Analysis of Users in Internet between two Operators, *International Journal of Advanced Networking and Applications, Vol.1, No.1, pp. 90-95,* 2009.
13. Shukla, D., Tiwari, M., Thakur, S. and Tiwari, V.: Rest State Analysis in Internet Traffic Distribution in Multi-operator environment, Published in Research Journal of Management and Information Technology (GNIM), Vol.1 No.1, pp. 72-82, 2009.
14. David G. Sullivan, Robert Haas, and Margo I. Seltzer Tickets and currencies revisited: Extensions to multi-resource lottery scheduling. *Proceedings of the 7th Workshop on Hot Topics in Operating Systems (HotOS-VII)*, March 1999.
15. Steve Evans, Kevin Clarke, Dave Singleton, and Bart Smaalders: Optimizing Unix Resource Scheduling for User Interaction. *Proceedings of the 1993 Summer USENIX*, pages 205–218. USENIX, June 1993.
16. David Petrou and John W. Milford: Implementing Lottery Scheduling: Matching The Specialization in Traditional Scheduler, *Proceedings of annual Conference of USENIX,* Berkeley, 1992.

Article received: 2011-08-01