DOI: http://dx.doi.org/10.26483/ijarcs.v8i8.4776

ISSN No. 0976-5697

Volume 8, No. 8, September-October 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

IMPLEMENTATION OF PROBABILITY-BASED PRIORITY-DRIVEN STRIDE SCHEDULING IN DISTRIBUTED SYSTEM

Shweta Jain
Research Scholar, Faculty of Computer Science
Pacific Academy of Higher Education
and Research University, Udaipur.

Saurabh Jain
Professor,
Shri Vaishnav Institute of Computer Applications
Shri Vaishnav Vidyapeeth Vishwavidyalaya, Indore.

Abstract: Generally fair share scheduling methods is not suitable for I/O bound and interactive applications whereas priority scheduling supports to execute variety of processes and applications. Stride scheduling is a deterministically fair share scheduling scheme that defines the share of processor time allocating to a client. This paper proposes priority-driven stride scheduling using stochastic modeling in distributed system that improves throughput and reduces response time by applying probabilistic study with the concept of fairness in resource distribution. A model-based simulation study is performed to analyze the transitions of clients onto the processor of this proposed scheme.

Keywords: Client, Distributed system, Fair-share, Markov Chain Model, Scheduler, Scheduling, Server, Stochastic modeling, Stride.

1. INTRODUCTION

Distributed system of independent computers seems as a single coherent system [1] in which computing systems put together composed of large number of computing units connected by high speed network. They have broad spectrum of systems including databases, media-based applications, networks, web-servers etc., which can be accessed by concurrent clients. Schedulers for distributed system and multithreaded system usually face insufficiency of resources in which they are ordered to control and service wide variety of requests for users and applications. At run time, multiple clients can request the service at the same time or quickly one after the other. These clients can come from the same or different location in the system. Due to rapid demand of new and efficient applications with multi-user and multitasking environment are requires developing new scheduling schemes or enhancing exiting ones that effects on throughput and response time to provide quality of service across a wide spectrum of systems.

A distributed scheduler is a resource management component of a distributed operating system that focuses on distributing the load of the system among the individual computing units to enhance overall performance[2]. [3] introduced load distribution algorithms with a taxonomy of approaches to the resource management problem providing a common terminology and classification mechanism necessary in addressing this problem. This load distribution environment arises because of the random arrival of clients and requires their random processing. So that there is a possibility of several computing units are likely to be idle or lightly loaded and some others can be heavily loaded, which would degrade the overall performance. There is always another possibility that one server or system is idle while a client is being waited upon at another server [2]. [4] presented three different load balancing algorithms for distributed systems that consist of a number of identical processors and a CSMA communication system by an analytic model.

[5] Firstly suggested and introduced **Stride scheduling** which is a deterministic proportional-share scheme to allocate resources for client processes and efficiently supports the flexible resource management concept to schedule other resources as per processor time. They also introduced novel hierarchical stride scheduling algorithm that achieves better throughput accuracy and lower response time. This fair-share allocation scheduling deals such type of workload and executes time-consuming or parallel applications in background. This scheduling is performed well enough for compute-bound applications whereas degraded the response time of interactive and I/O-bound applications. [6] presented a novel implementation of fair-share stride scheduling using priority scheduler to support compute-bound and interactive application. [7] suggested Virtual-Time Round-Robin (VTRR), a proportional share scheduler that can provide good proportional sharing accuracy with scheduling overhead and also discussed about the problems of achieving perfect fairness due to large numbers of clients with whatever resources they are allocated among various servers in distributed system.

[8] presented an event-based stride scheduling method for time critical collision detection that meets real-time constraints by balancing and prioritizing computation spent on intersection tests without starvation. [9] was adopted symmetrically initiated algorithm in distributed system. [10] based on the performance trends of load sharing algorithms and the recommendations for the selection of a load distribution algorithm. [11], [12 a], [13 b] also have studied scheduling policies in distributed systems to find the ways to distribute client processes among various processors in order to achieve fairness and performance like minimizing execution time, minimizing communication, maximizing resource utilization, minimizing response time, average completion time per application and multiple independent tasks with load balancing with other overheads.

[14] and [15] used a Markov chain model for the study of transition probabilities with random movement of scheduler to analyze performance of scheduling schemes like Round Robin(RR) and Multi-level feedback

Queue(MLFQ). The priority-driven stride scheduling system proposes smaller waiting and response times to give controls the resource utilization rates of clients that are proportional to the relative fair shares allocated by server using markov chain model in distributed system. [16] and [17] studied brief of stochastic processes and Markov chain Model.

2. STRIDE SCHEDULING

Stride scheduling is a deterministic fair-share allocation mechanism of resources in which tickets are used to define the share of processor time allocating to a client [6]. Allocation of resources is in discrete time slices referred as quantum [5]. Resource rights are considered by ticket that can be subjected in different amounts and conceded among clients. Throughput rates for lively clients are directly proportional to their ticket allocations whereas response times are inversely proportional to ticket allocation[5]. This scheduling calculates the time interval that is referred by stride in which a client must remain among consecutive allocations that means stride field is inversely proportional to tickets [5][6]. Let stride be $\mathbf{St_i}$ and two consecutive allocations be $\mathbf{ta_1}$ and $\mathbf{ta_2}$ in a time quantum so that $\mathbf{St_i = ta_2 - ta_1}$.

Clients are scheduled most frequently those having shortest stride, twice as quickly those having half stride and twice slowly those having double stride. These strides are characterized in virtual time units called passes [6]. So each client is associated with three state variables tickets, stride, and pass. Since this scheduling is deterministic that clients have guaranteed to be scheduled at least once every complete cycle time or stride [19]. Resource allocation is performed for the client selection having the least pass and its pass value is incremented by its stride, so that new value of pass can be $\mathbf{Pass_1 = Pass + Stride}$. If more than one client has the same minimum pass value, then anyone can be selected.

Overall this is a fair-share scheduling scheme for resource management and obtaining responsiveness for probabilities to converge. But to some extent, perfect fairness and responsiveness is ideal condition that cannot achieve due to weak service time, poor processing time and poor response time in distributed system over the world. Moreover, this scheduling doesn't support preemption and having a high overhead.

3. PRIORITY SCHEDULING

Priority Scheduling features a priority that characterizes an order of each client. Generally highest priority clients schedule first rather than depending upon run time behaviour of client. Scheduler can use several methods to determine priorities according to different parameters and different classes of priorities can also be implemented depending on the type of client processes. This scheme provides the required controlling for client process scheduling in order to execute interactive client processes efficiently and able to improve the overall fairness. [18] used priorities to control the execution of systems to meet given requirements for optimal use of resources and incorporated scheduling policies in distributed system although implementation may incur considerable overhead and complicated executing transitions according to a priority policy for each client process because situation view is limited to the rest of the system.

4. PRIORITY-DRIVEN STRIDE SCHEDULING

This paper proposes to combine stride scheduling with priority scheme in which priority scheme allows controlling the preemption on awaken of clients and high load in distributed system. It also provides scheduling a client with high priority until it is allocated within its defined share of tickets. But stride scheduling allocates the fair processor time and allows a strict control of the client. Taking advantage of priority scheduling and also report for starvation, we think of a scheme stride scheduling featuring priority aging wherein the longer waits of clients in the queue, the higher its priority becomes. Priority aging reduces the starvation in scheduling of clients. Waiting time and response time depend on the priority of the client. Stride scheduling also supports of dynamic client participation, dynamic modifications to ticket allocations, and non-uniform quanta [5].

A distributed system may have a combination of heavily and lightly loaded systems. So that migrating a client process to share or balance load can help. Multiple clients access resources on a random manner in distributed system assigning by unique client identification. The outcomes of accuracy in a proportional-share scheduler can be considered by measuring the difference between the specified and actual number of allocations that a client receives during a series of allocations of tickets [5]. So that it is naturally impractical to attain this ideal condition exactly due to quantization. Each client is assigned a time quantum equal to its ticket share. A client with a larger share of tickets finds a larger quantum than a client with a small share of tickets. Running all clients with the same frequency provides proportional sharing but adjusting the size of their time quantum [7]. So this scheduler assigns a short time quantum for high priority clients. This paper gives solution for implementing priorities based on an analysis of the system using stochastic model study.

5. THE STOCHASTIC MODEL

A stochastic process has the Markov property if the probability distribution of X^n is determined by knowing the probability distribution of X^{n-I} [17]. A Markov process is a stochastic model that has the Markov property. It can be used to model a random system that changes states according to a transition rule that only depends on the current state. The state space S is a discrete space then the Markov process is called a Markov chain [16][17]. Markov chain model is applied on proposed scheduling using state transition probabilities obtaining from the clients movement over states. A stochastic process $\{X^n\}$ talking values in a discrete state space S is called a Markov chain or Markov property if:

$$P[X^{n+1} = i_{n+1}; X^n = i_n; X^{n-1} = i_{n-1}; \ldots\ldots; X^1 = i_1; X^0 = i_0] = P[X^{n+1} = i_{n+1}; X^n = i_n];$$

$$\text{for all } n \quad \text{and}$$

$$P[X^n = i_n; \ldots\ldots; X^1 = i_1; X^0 = i_0] > 0.$$

6. THE PROPOSED SYSTEM

Let P be the probability that the system is in a state in which at least one client is waiting for service and server is idle according to the utilization of each server. Suppose the distributed system has m clients at particular time which is C_1, C_2, C_3, C_4... C_m in queue and $\{X^{(n)}, n \geq 1\}$ b e significance of a Markov chain among the state space C_1, C_2, C_3, C_4... C_m and a server R where R is a server state that serves to clients for various conditions like normal, idle, blocking, any kind of failures, disturbances or errors occur which depicts in figure 1 where for each allocation, every client is specified a fair possibility of getting proportional to its share of the total number of tickets [5]. Since any changes to relative ticket allocations are instantly replicated in the next allocation decision [20]. The queue allocates a random ticket to each of next coming clients. When a client do not complete within specified quantum, it shifts to the queue where clients wait for their turn then scheduler assigns a new ticket to the remaining client process for its completion.

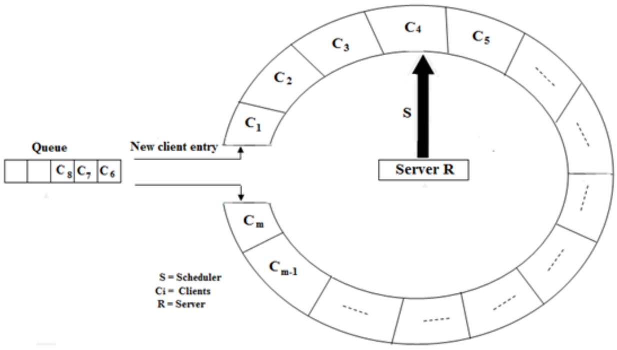


Figure 1: Priority-driven Sride Scheduling for Client Processing

Assuming S be a scheduler so that $X^{(n)}$ be the state of scheduler of the system at the end of nth quantum [15] where n = 1,2,3… and the transition of scheduler S is randomized with m+1 states in nth quantum which is shown in figure 2. The scheduler S provides stochastically a quantum of time to each client and next quantum is resoluted by a random assessment. The S begins with the client C_i and then goes to C_j where $i \neq j = 1, 2, 3...m$.

Suppose S is at any client C_i (i = 1, 2, 3…m) having least pass/stride at the end of a quantum, then it is on C_{i+1} or C_{i-1} (where i >1) in the next quantum with some assigning priority. The X happens to inactive when there is no other client in the queue. System may have a long waiting queue of clients outside from the server R where clients get various kinds of services and if one client is over inside, then a new client is waiting outside or moving to other computing units or server to enter as to retain remaining m clients. The server fulfils various services and requirements of clients according to priority assigned. So that the transition diagram for any three clients C_{i-1}, C_i, C_{i+1} and server R is given in figure 2.

Each client is set to have a share of tickets equal to the required proportional share of the scheduled resource [21]. Still a stride value St_i is determined as the inverse of the number of tickets which is used to perform a lottery method. Stride is measured in passes to represent the interval between times quantum the client executes. So the current pass value for a particular client is p_i where passes is a virtual time unit. The probability P that a client C_i holding t_i tickets in a lottery with in total T tickets will win the lottery $P = t_i / T$. The algorithm uses a constant stride1 which represents the stride for a client with one ticket [5]. Few assumptions are as follows:

- Clients are initiated with $St_i = stride_1 / t_i$ and $C_i = St_i$
- The scheduler selects the client with lowest S_i (ties come arbitrarily).

After executing it, Sti is updated to $St_i + f\ St_i$, where f is the fraction of the allocated time quanta that the client actually used (f > 0).

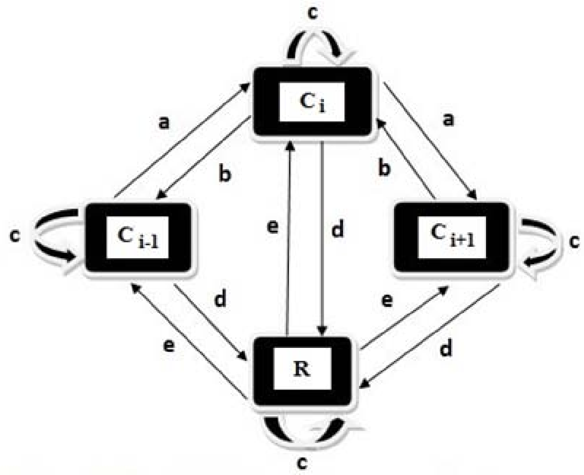


Figure 2: Transition Diagram for proposed system

Let us defining the transition among the clients by unit-step transition probabilities for X^n with its respective priorities a, b, c, d and e are as follows:

$$P\,[\,X^{(n+1)} = C_{i+1}\ /\ X^{(n)} = C_i\,] = a$$

$$P\,[\,X^{(n+1)} = C_{i-1}\ /\ X^{(n)} = C_i\,] = b$$

$$P\,[\,X^{(n+1)} = C_i\ /\ X^{(n)} = C_i] = c$$

$$P\,[\,X^{(n+1)}) = C_i\ /\ X^{(n)} = R] = d$$

$$P\,[\,X^{(n+1)} = R\ /\ X^{(n)} = C_i] = e$$

Where i = 1, 2, 3…….m and i >= 1

So the unit step transition probability matrix (m+1) × (m+1) to assign priority is:

$$\begin{array}{cc|cccccccc} & & \multicolumn{8}{c}{\longleftarrow X^{(n)} \longrightarrow} \\ & & C_1 & C_2 & C_3 & C_4 & & C_{m-1} & C_m & R \\ \hline & C_1 & c & a & 0 & 0 & \text{---------} & 0 & b & d \\ & C_2 & b & c & a & 0 & \text{---------} & 0 & 0 & d \\ & C_3 & 0 & b & c & a & \text{---------} & 0 & 0 & d \\ X^{(n-1)} & C_4 & 0 & 0 & b & c & \text{---------} & 0 & 0 & d \\ & & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ & C_{m-1} & 0 & 0 & 0 & 0 & \text{---------} & c & a & d \\ & C_m & a & 0 & 0 & 0 & \text{---------} & b & c & d \\ & R & e & e & e & e & \text{---------} & e & e & 1 \end{array}$$

with $a + b + d + c + e = 1$ and $(m + 1)(d + e) =1$

Let us assume **pb** be the probability among transition between server R and clients C_{i-1}, C_i, C_{i+1} .So The state probabilities for n = 0 are as follows:

$$P[X^{(0)} = Ci] = pb_i \; ; \quad \text{where } i = 1,2,3,4\ldots m$$

$$P[X^{(0)} = R] = 0$$

The state probabilities after the first quantum are as follows:

$$\begin{aligned} P[X^{(1)} = Ci] = {} & P[X^{(0)} = Ci_{-1}]. P[X^{(1)} = Ci / P[X^{(0)} = Ci_{-1}] + \\ & P[X^{(0)} = Ci]. P[X^{(1)} = Ci / P[X^{(0)} = Ci] + \\ & P[X^{(0)} = Ci_{+1}]. P[X^{(1)} = Ci / P[X^{(0)} = Ci_{+1}] \\ = {} & (pb_{i-1})a + (pb_i)c + (pb_{i+1})b \; ; \\ & (\text{if } i\text{-}1\text{=}0 \text{ then } i\text{-}1\text{=}1 \;\&\; \text{if } i\text{+}1\text{>}m \text{ then } i\text{+}1\text{=}m) \end{aligned}$$

$$P[X^{(1)} = R] = (d + e). \sum_{i=1}^{m} pb_i = d + e$$

Similarly, state probabilities after second quantum are expressed as follows :

$$\begin{aligned} P[X^{(2)} = Ci] = {} & (pb_{i-2})a + (pb_{i-1})c + (pb_i)b]a + \\ & (pb_{i-1})a + (pb_i)c + (pb_{i+1})b]c + \\ & (pb_i)a + (pb_{i+1})c + (pb_{i+2})b]b \\ = {} & P[X^{(1)} = Ci_{-1}].a + P[X^{(1)} = Ci].c + P[X^{(1)} = Ci_{+1}].b \end{aligned}$$

$$P[X^{(2)} = R] = \sum_{i=1}^{m} P[X^{(1)} = Ci] .d + P[X^{(2)} = R]. e$$

The general expression for n quantum are as follows:

$$P[X^{(n)} = Ci] = P[X^{(n-1)} = Ci_{-1}].a + P[X^{(n-1)} = Ci].c + P[X^{(n-1)} = Ci_{+1}].$$

$$P[X^{(n)} = R] = \sum_{i=1}^{m} P[X^{(n-1)} = Ci].d + P[X^{(n-1)} = R].e$$

7. SOME RESTRICTED SCHEDULING SCHEMES

Some restrictions and conditions can be applied of this proposed scheme as mentioned above that can produce different scheduling schemes from the general class of expression in previous section.

SCHEME-A: When a = 0, b = 0, c = 1, d =0, e = 0

Unit step transition probability matrix (m+1) × (m+1) for $X^{(n)}$ of scheme-A which is general class of fair share stride scheduling scheme according to priority of clients for all quantum n are as follows:

$$\begin{array}{cc|cccccccc} & & \multicolumn{8}{c}{\longleftarrow X^{(n)} \longrightarrow} \\ & & C_1 & C_2 & C_3 & C_4 & \text{---------} & C_{m-1} & C_m & R \\ \hline & C_1 & 1 & 0 & 0 & 0 & \text{---------} & 0 & 0 & 0 \\ & C_2 & 0 & 1 & 0 & 0 & \text{---------} & 0 & 0 & 0 \\ & C_3 & 0 & 0 & 1 & 0 & \text{---------} & 0 & 0 & 0 \\ & C_4 & 0 & 0 & 0 & 1 & \text{---------} & 0 & 0 & 0 \\ X^{(n-1)} & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ & C_{m-1} & 0 & 0 & 0 & 0 & \text{---------} & 1 & 0 & 0 \\ & C_m & 0 & 0 & 0 & 0 & \text{---------} & 0 & 1 & 0 \\ & R & 0 & 0 & 0 & 0 & \text{---------} & 0 & 0 & 1 \end{array}$$

The initial probabilities at n=0 for scheme-A are:

$$P[X^{(0)} = Ci] = pb_i \qquad \text{subject to the condition } \sum_{i=1}^{m} pb_i$$

The state probabilities after the first quantum are:

$$P[X^{(1)} = C_i] = pb_i$$

The generalized expressions of scheme-A for n quantum are:

$$P[X^{(n)} = C_i] = pb_i$$

SCHEME-B: When a = 0, b = 0, e = 0, c + d= 1

Unit step transition probability matrix (m+1) × (m+1) for $X^{(n)}$ of scheme-B which is fair share stride scheduling scheme according to priority of clients for all quantum n are as follows:

$$\begin{array}{cc|cccccccc} & & \multicolumn{8}{c}{\longleftarrow X^{(n)} \longrightarrow} \\ & & C_1 & C_2 & C_3 & C_4 & \text{---------} & C_{m-1} & C_m & R \\ \hline & C_1 & c & 0 & 0 & 0 & \text{---------} & 0 & 0 & d \\ & C_2 & 0 & c & 0 & 0 & \text{---------} & 0 & 0 & d \\ & C_3 & 0 & 0 & c & 0 & \text{---------} & 0 & 0 & d \\ & C_4 & 0 & 0 & 0 & c & \text{---------} & 0 & 0 & d \\ X^{(n-1)} & \vdots & \vdots & \vdots & \vdots & \vdots & \text{---------} & \vdots & \vdots & \vdots \\ & C_{m-1} & 0 & 0 & 0 & 0 & \text{---------} & c & 0 & d \\ & C_m & 0 & 0 & 0 & 0 & \text{---------} & 0 & c & d \\ & R & 0 & 0 & 0 & 0 & \text{---------} & 0 & 0 & 1 \end{array}$$

Such that c + d =1

The initial probabilities at n=0 for scheme-B are:

$$P[X^{(0)} = Ci] = pb_i$$

$$P[X^{(0)} = R] = 0$$

State probabilities after the first quantum are:

$$P[X^{(1)} = Ci] = pb_i \cdot c$$

$$P[X^{(1)} = R] = d \cdot \sum_{i=1}^{m} pb_i = d$$

General expressions of scheme-B for n^{th} quantum are:

$$P[X^{(n)} = Ci] = P[X^{(n-1)} = Ci] .c$$

$$P[X^{(n)} = R] = \sum_{i=1}^{m} P[X^{(n-1)} = Ci] .d + P[X^{(n-1)} = R] . e = \sum_{i=1}^{m} P[X^{(n-1)} = Ci] .d$$

SCHEME-C: WHEN b = 0, c = 0, d =0, e =0, a = 1

This general class scheduling scheme referred Priority-driven Stride Scheduling scheme for all quantum n.

Unit step transition probability matrix to assign priority (m+1) × (m+1) for $X^{(n)}$ of scheme-C is

$$\begin{array}{c|cccccccc} & \multicolumn{8}{c}{\longleftarrow X^{(n)} \longrightarrow} \\ & C_1 & C_2 & C_3 & C_4 & \text{---------} & C_{m-1} & C_m & R \\ \hline C_1 & 0 & 1 & 0 & 0 & \text{---------} & 0 & 0 & 0 \\ C_2 & 0 & 0 & 1 & 0 & \text{---------} & 0 & 0 & 0 \\ C_3 & 0 & 0 & 0 & 1 & \text{---------} & 0 & 0 & 0 \\ C_4 & 0 & 0 & 0 & 0 & \text{---------} & 0 & 0 & 0 \\ X^{(n-1)} \ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ C_{m-1} & 0 & 0 & 0 & 0 & \text{---------} & 0 & 1 & 0 \\ C_m & 1 & 0 & 0 & 0 & \text{---------} & 0 & 0 & 0 \\ R & 0 & 0 & 0 & 0 & \text{---------} & 0 & 0 & 1 \end{array}$$

The initial probabilities at n=0 for scheme-C are:

$$P[\, X^{(0)} = Ci \,] = pb_i \qquad \text{subject to the condition } \sum_{i=1}^{m} pb_i$$

The state probabilities after the first quantum are:

$$P[X^{(1)} = Ci \,] = pb_{i-1}$$

The general expressions of scheme-C for n quantum are:

$$P[X^{(n)} = Ci \,] = pb_{i-n}$$

SCHEME-D: When b = 0, c = 0, e = 0, a + d = 1

General class has priority-driven stride scheduling scheme for all quantum n. Unit step transition probability matrix to assign priority (m+1) × (m+1) for $X^{(n)}$ of scheme-D is

$$\begin{array}{c|cccccccc} & \multicolumn{8}{c}{\longleftarrow X^{(n)} \longrightarrow} \\ & C_1 & C_2 & C_3 & C_4 & \text{---------} & C_{m-1} & C_m & R \\ \hline C_1 & 0 & a & 0 & 0 & \text{---------} & 0 & 0 & d \\ C_2 & 0 & 0 & a & 0 & \text{---------} & 0 & 0 & d \\ C_3 & 0 & 0 & 0 & a & \text{---------} & 0 & 0 & d \\ C_4 & 0 & 0 & 0 & 0 & \text{---------} & 0 & 0 & d \\ X^{(n-1)} \ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ C_{m-1} & 0 & 0 & 0 & 0 & \text{---------} & 0 & a & d \\ C_m & a & 0 & 0 & 0 & \text{---------} & 0 & 0 & d \\ R & 0 & 0 & 0 & 0 & \text{---------} & 0 & 0 & 1 \end{array}$$

Such that a +d =1.

The initial probabilities at n=0 for scheme-D are

$$P[X^{(0)} = Ci \,] = pb_i$$

$$P[X^{(0)} = R \,] = 0$$

The state probabilities after the first quantum are:

$$P[X^{(1)} = Ci \,] = pb_{i-1} \,.a$$

$$P[X^{(1)} = R] = d \cdot \sum_{i=1}^{m} pb_i = d$$

The general expressions of scheme-D for nth quantum are:

$$P[X^{(n)} = Ci \,] = P[X^{(n-1)} = Ci_{-1} \,] \,.a$$

$$P[X^{(n)} = R \,] = \sum_{i=1}^{m} P[X^{(n-1)} = Ci_{-1} \,] \,.a.\,d + P[X^{(n-1)} = R] \,.e \;=\; \sum_{i=1}^{m} P[X^{(n-1)} = Ci_{-1} \,] \,.a.\,d$$

SCHEME-E: When b = 0, d =0, e=0, a + c = 1

Unit step transition probability matrix to assign priority (m+1) × (m+1) for $X^{(n)}$ of scheme-E is

$$\begin{array}{c|cccccccc} & \multicolumn{8}{c}{\longleftarrow X^{(n)} \longrightarrow} \\ & C_1 & C_2 & C_3 & C_4 & \text{---------} & C_{m-1} & C_m & R \\ \hline C_1 & c & a & 0 & 0 & \text{---------} & 0 & 0 & 0 \\ C_2 & 0 & c & a & 0 & \text{---------} & 0 & 0 & 0 \\ C_3 & 0 & 0 & c & a & \text{---------} & 0 & 0 & 0 \\ C_4 & 0 & 0 & 0 & c & \text{---------} & 0 & 0 & 0 \\ X^{(n-1)} \ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ C_{m-1} & 0 & 0 & 0 & 0 & \text{---------} & c & a & 0 \\ C_m & 0 & 0 & 0 & 0 & \text{---------} & 0 & c & a \\ R & a & 0 & 0 & 0 & \text{---------} & 0 & 0 & c \end{array}$$

The initial probabilities at n=0 for scheme-E are:

$$P[\, X^{(0)} = Ci \,] = pb_i \qquad \text{subject to the condition } \sum_{i=1}^{m} pb_i$$

State probabilities after the first quantum are:

$$P[X^{(1)} = Ci \,] = pb_{i-1} \,.a + pb_i \,.c$$

General expressions of scheme-E for nth quantum are:

$$P[X^{(n)} = Ci \,] = P[X^{(n-1)} = Ci_{-1} \,] \,.a + P[X^{(n-1)} = Ci \,] \,.c$$

SCHEME-F: When b = 0, e =0, a + c + d = 1 .

Unit step transition probability matrix to assign priority (m+1) × (m+1) for $X^{(n)}$ of scheme-F is

$$\begin{array}{c|cccccccc} & \multicolumn{8}{c}{\longleftarrow X^{(n)} \longrightarrow} \\ & C_1 & C_2 & C_3 & C_4 & \text{---------} & C_{m-1} & C_m & R \\ \hline C_1 & c & a & 0 & 0 & \text{---------} & 0 & 0 & d \\ C_2 & 0 & c & a & 0 & \text{---------} & 0 & 0 & d \\ C_3 & 0 & 0 & c & a & \text{---------} & 0 & 0 & d \\ C_4 & 0 & 0 & 0 & c & \text{---------} & 0 & 0 & d \\ X^{(n-1)} \ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ C_{m-1} & 0 & 0 & 0 & 0 & \text{---------} & c & a & d \\ C_m & a & 0 & 0 & 0 & \text{---------} & 0 & c & d \\ R & 0 & 0 & 0 & 0 & \text{---------} & 0 & 0 & 1 \end{array}$$

The initial probabilities at n=0 for scheme-F are

$$P[X^{(0)} = Ci \,] = pb_i$$

$$P[X^{(0)} = R \,] = 0$$

State probabilities after the first quantum are:

$$P[X^{(1)} = C_i \,] = P[X^{(0)} = C_{i-1} \,] \,.a + P[X^{(0)} = C_i \,] \,.c$$

$$P[X^{(1)} = R \,] = d \cdot \sum_{i=1}^{m} pb_i = d$$

General expressions of scheme-F for nth quantum are:

$$P[X^{(n)} = C_i \,] = P[X^{(n-1)} = C_{i-1} \,] \,.a + P[X^{(n-1)} = C_i \,] \,.c$$

$$P[X^{(n)} = R \,] = \sum_{i=1}^{m} P[X^{(n-1)} = C_i \,]$$

8. SIMULATION STUDY AND GRAPHICAL ANALYSIS

We know that P be the probability that the system is in a state in which at least one client is waiting for service and server is idle according to the utilization of each server. We can estimate P using probabilistic analysis and plot a graph against system utilization. For moderate system utilization, value of P is high, i.e., at least someone is idle. Hence, performance can be improved by fair sharing of resources to the clients. Result from imposing or restricting priorities between transitions. However for fairly demanding, the design of the system executes transitions having a set of priorities according to a priority policy using Markov chain model. So all the scheduling schemes are compared and the simulation study is performed on above mentioned various schemes as follows:

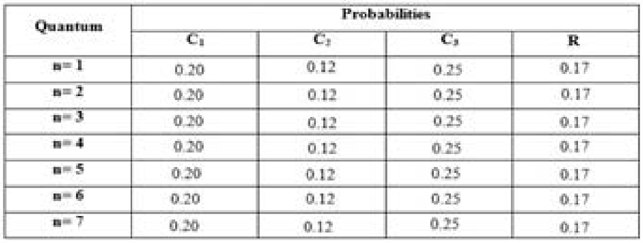
Scheme- A

Here a = b = d = e = 0, c =1, So the transition probability matrix is:

$$\begin{array}{c|c|cccc} & & \multicolumn{4}{c}{\longleftarrow X^{(n)} \longrightarrow} \\ & & C_1 & C_2 & C_3 & R \\ \hline \uparrow & C_1 & 1 & 0 & 0 & 0 \\ X^{(n-1)} & C_2 & 0 & 1 & 0 & 0 \\ & C_3 & 0 & 0 & 1 & 0 \\ \downarrow & R & 0 & 0 & 0 & 1 \end{array}$$

Let us assume initial probabilities $pb_1 = 0.2$, $pb_2 = 0.12$, $pb_3 = 0.25$ and $pb_4 = 0.17$

Table 1: Transition Probability Matrices of P[$X^{(n)} = C_i/R$] for scheme–A



Quantum	Probabilities			
	C1	C2	C3	R
n= 1	0.20	0.12	0.25	0.17
n= 2	0.20	0.12	0.25	0.17
n= 3	0.20	0.12	0.25	0.17
n= 4	0.20	0.12	0.25	0.17
n= 5	0.20	0.12	0.25	0.17
n= 6	0.20	0.12	0.25	0.17
n= 7	0.20	0.12	0.25	0.17

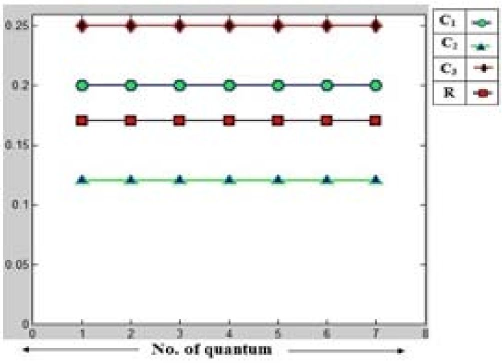


Figure 8.1: Transition graph for Scheme-A

Figure 8.1 depicts that variations in the quantum do not influence the state probabilities P_i. The scheme-A gives fair chance to process every client as assigning priority.

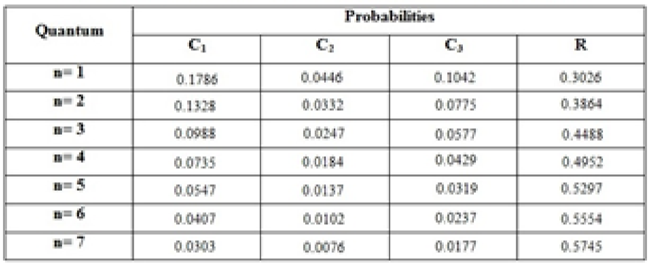
Scheme- B

Here a = b = e = 0, c + d = 1, d = 0.256 , So the transition probability matrix is:

$$\begin{array}{c|c|cccc} & & \multicolumn{4}{c}{\longleftarrow X^{(n)} \longrightarrow} \\ & & C_1 & C_2 & C_3 & R \\ \hline \uparrow & C_1 & 0.744 & 0 & 0 & 0.256 \\ X^{(n-1)} & C_2 & 0 & 0.744 & 0 & 0.256 \\ & C_3 & 0 & 0 & 0.744 & 0.256 \\ \downarrow & R & 0 & 0 & 0 & 1 \end{array}$$

Let us assume initial probabilities $pb_1 = 0.24$, $pb_2 = 0.06$, $pb_3 = 0.14$ and $pb_4 = 0.19$

Table 2: Transition Probability Matrices of P[$X^{(n)} = R$] for scheme–B



Quantum	Probabilities			
	C1	C2	C3	R
n= 1	0.1786	0.0446	0.1042	0.3026
n= 2	0.1328	0.0332	0.0775	0.3864
n= 3	0.0988	0.0247	0.0577	0.4488
n= 4	0.0735	0.0184	0.0429	0.4952
n= 5	0.0547	0.0137	0.0319	0.5297
n= 6	0.0407	0.0102	0.0237	0.5554
n= 7	0.0303	0.0076	0.0177	0.5745

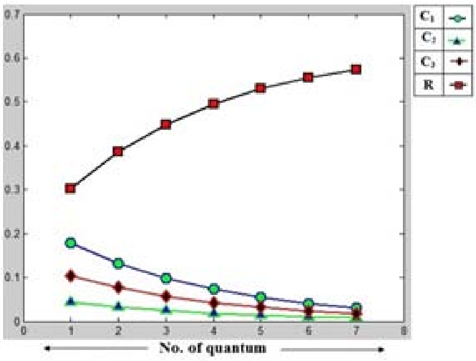


Figure 8.2: Transition graph for Scheme-B

Graph of scheme-B is representing by figure 8.2 which indicates clients C_i (i = 1, 2, 3) and server R during processing. The proposed scheduler in the system performs transition to clients and server. The state probabilities of client/ clients over a large number of quantum are dropping quickly and the probability of server is going to comparatively high. This scheme specifies that the clients and server take extra time to finish, delay in processing becomes large enough and not fair enough to use resources.

Scheme- C

Here a = 1, b = 0, c = 0, d =0, e =0, So the transition probability matrix is:

$$X^{(n)} \longleftrightarrow \qquad X^{(n-1)} \updownarrow \quad \begin{array}{c|cccc} & C_1 & C_2 & C_3 & R \\ \hline C_1 & 0 & 1 & 0 & 0 \\ C_2 & 0 & 0 & 1 & 0 \\ C_3 & 1 & 0 & 0 & 0 \\ R & 0 & 0 & 0 & 1 \end{array}$$

In this scheme, we can consider two cases having different probabilities, they are as follows:

(a) Let us assume initial probabilities $pb_1 = 0.19$, $pb_2 = 0.23$, $pb_3 = 0.06$ and $pb_4 = 0.12$.

Table 3(a): Transition Probability Matrices of P[$X^{(n)} = R$] for scheme–C(a)

Quantum	Probabilities			
	C_1	C_2	C_3	R
n= 1	0.06	0.19	0.23	0.12
n= 2	0.23	0.06	0.19	0.12
n= 3	0.19	0.23	0.06	0.12
n= 4	0.06	0.19	0.23	0.12
n= 5	0.23	0.06	0.19	0.12
n= 6	0.19	0.23	0.06	0.12
n= 7	0.06	0.19	0.23	0.12

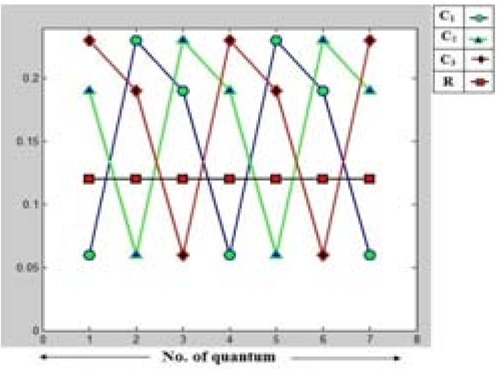


Figure 8.3(a): Transition graph for Scheme-C(a)

In figure 8.3(a) of scheme-C(a), the graph of system follows the priority-driven stride scheduling scheme where scheduler can initiate dealing out any client. The transition probability reaches a maximum at some particular quantum for an particular client whereas the probability falls down when quantum increases. So the state probability remains the identical at regular interval when scheduler transits among clients and server.

(b) Let us assume initial probabilities $pb_1 = 1$, $pb_2 = 0$, $pb_3 = 0$ and $pb_4 = 0$

Table 3(b): Transition Probability Matrices of P[$X^{(n)} = R$] for scheme–C(b)

Quantum	Probabilities			
	C_1	C_2	C_3	R
n= 1	0	1	0	0
n= 2	0	0	1	0
n= 3	1	0	0	0
n= 4	0	1	0	0
n= 5	0	0	1	0
n= 6	1	0	0	0
n= 7	0	1	0	0

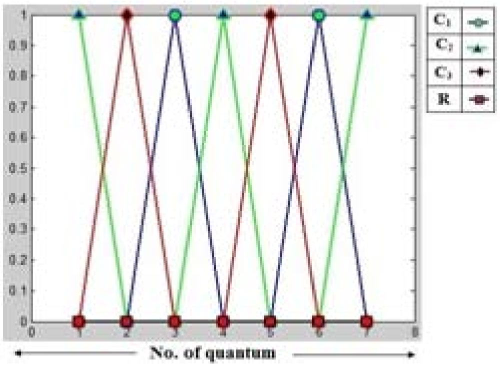


Figure 8.3(b): Transition graph for Scheme-C(b)

The scheme-C(b) is purely priority-driven stride scheduling scheme, which starts from the first process, the state probabilities are in up and down pattern. After a constant interval of quantum, each client allows processing and serving to be high probability and similarly also goes low.

Scheme- D

Here b = 0, c = 0, e = 0, a + d = 1, d = 0.256, So the transition probability matrix is:

$$X^{(n)} \longleftrightarrow \qquad X^{(n-1)} \updownarrow \quad \begin{array}{c|cccc} & C_1 & C_2 & C_3 & R \\ \hline C_1 & 0 & 0.744 & 0 & 0.256 \\ C_2 & 0 & 0 & 0.744 & 0.256 \\ C_3 & 0.744 & 0 & 0 & 0.256 \\ R & 0 & 0 & 0 & 1 \end{array}$$

Let us assume initial probabilities $pb_1 = 0.32$, $pb_2 = 0.09$, $pb_3 = 0.16$ and $pb_4 = 0.21$.

Table 4: Transition Probability Matrices of P[$X^{(n)} = R$] for scheme–D

Quantum	Probabilities			
	C_1	C_2	C_3	R
n= 1	0.1190	0.2381	0.0670	0.3559
n= 2	0.0498	0.0886	0.1771	0.4645
n= 3	0.1318	0.0371	0.0659	0.5453
n= 4	0.0490	0.0980	0.0276	0.6054
n= 5	0.0205	0.0365	0.0729	0.6501
n= 6	0.0543	0.0153	0.0271	0.6833
n= 7	0.0202	0.0404	0.0114	0.7081

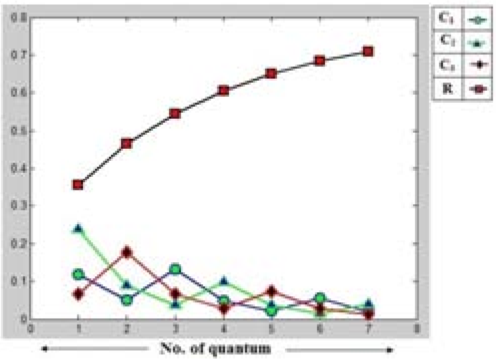


Figure 8.4: Transition graph for Scheme-D

In figure 8.4 represents graph for scheme-D where the scheme is somewhat not exactly fair scheduling because of weak interactions and services providing by server R to clients C_i (i = 1, 2, 3) due to move in opposite direction (R is going high but clients jump to be down) as per the increasing number of quantum to have indications for the system to service poor.

Scheme- E

Here b = 0, d =0, e=0, a + c = 1, a = 0.45 and c = 0.55, So the transition probability matrix is:

$$\begin{array}{cc|cccc} & & \multicolumn{4}{c}{\longleftarrow X^{(n)} \longrightarrow} \\ & & C_1 & C_2 & C_3 & R \\ \hline \uparrow & C_1 & 0.55 & 0.45 & 0 & 0 \\ & C_2 & 0 & 0.55 & 0.45 & 0 \\ X^{(n-1)} & C_3 & 0 & 0 & 0.55 & 0.45 \\ \downarrow & R & 0.45 & 0 & 0 & 0.55 \end{array}$$

Let us assume initial probabilities $pb_1 = 0.26$, $pb_2 = 0.13$, $pb_3 = 0.09$ and $pb_4 = 0.31$.

Table 5: Transition Probability Matrices of P[$X^{(n)} = R$] for scheme–E

Quantum	Probabilities			
	C_1	C_2	C_3	R
n= 1	0.2825	0.1885	0.2785	0.0405
n= 2	0.1736	0.2308	0.2603	0.1253
n= 3	0.1519	0.2051	0.3159	0.1171
n= 4	0.1362	0.1811	0.3305	0.1422
n= 5	0.1389	0.1609	0.3415	0.1487
n= 6	0.1433	0.1510	0.3420	0.1537
n= 7	0.1480	0.1476	0.3406	0.1539

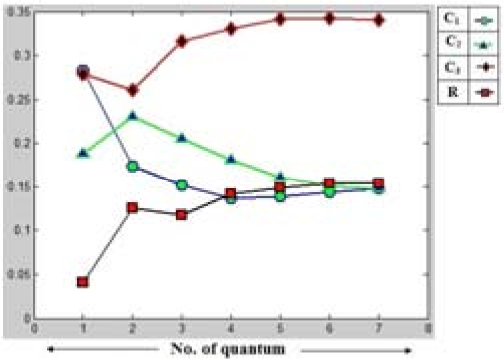


Figure 8.5: Transition graph for Scheme-E

The scheme-E graph is shown in fig 8.5 in which the quantum distribution takes over the same state or to the next state depending on the random output sampling. If the number of quantum grows, then the scheme demonstrates the state probabilities nearly a constant pattern. This means every client get almost equal and fair chance of being processed and serviced. It can be somewhat priority-driven stride scheduling.

Scheme- F

Here b = 0, e = 0, a + d + c = 1, a = 0.45 ,d = 0.256 . So the transition probability matrix is:

$$\begin{array}{cc|cccc} & & \multicolumn{4}{c}{\longleftarrow X^{(n)} \longrightarrow} \\ & & C_1 & C_2 & C_3 & R \\ \hline \uparrow & C_1 & 0.294 & 0.45 & 0 & 0.256 \\ & C_2 & 0 & 0.294 & 0.45 & 0.256 \\ X^{(n-1)} & C_3 & 0.45 & 0 & 0.294 & 0.256 \\ \downarrow & R & 0 & 0 & 0 & 1 \end{array}$$

Let us assume initial probabilities $pb_1 = 0.16$, $pb_2 = 0.07$, $pb_3 = 0.11$ and $pb_4 = 0.24$.

Table 6: Transition Probability Matrices of P[$X^{(n)} = R$] for scheme–F

Quantum	Probabilities			
	C_1	C_2	C_3	R
n= 1	0.0965	0.0926	0.0638	0.3270
n= 2	0.0571	0.0707	0.0604	0.3918
n= 3	0.0440	0.0465	0.0496	0.4400
n= 4	0.0352	0.0335	0.0355	0.4758
n= 5	0.0263	0.0257	0.0255	0.5025
n= 6	0.0192	0.0194	0.0191	0.5223
n= 7	0.0142	0.0143	0.0143	0.5371

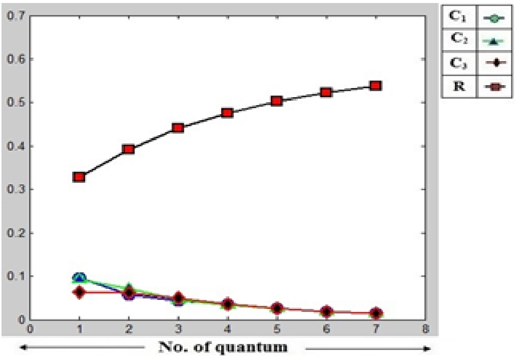


Figure 8.6: Transition graph for Scheme-F

Scheme-F graph is shown in figure 8.6 in which we found that with the increasing number of attempts, the state probabilities of clients C_i (i = 1, 2, 3) are dropping drastically whereas the state probability of server R is moving high of system. So it shows poor responsiveness among clients and server.

9. CONCLUSION

This study integrates stride scheduling schemes with priority scheme to analysis and simulate with stochastic Markov chain model. Firstly observes on graphs of schemes that scheme-A, scheme-C(a) and (b) and scheme-E looks different than scheme-B, scheme-D and scheme-F. Perhaps Scheme-B, scheme-D and scheme-F are performed almost same where poor responsiveness, weak interaction and not much fair among clients and server whereas scheme-E performs better than these mentioned scheme. Although scheme-A, scheme-C(a) and (b) are extremely well response, fairness and less overhead to clients with server as compared to other schemes. The overall performance and analysis of proposed scheduling scheme indicates that scheme-C(a) and scheme-C(b) follow purely priority-driven stride scheduling.

REFERENCES

[1]. Tanenbaum, Andrew S., Steen, Maarten van, Distributed systems: principles and paradigms, Pearson Prentice Hall Inc. and Dorling Kindersley Publishing(India) Inc., 2002.

[2]. Shounak Chakraborty and Ajoy Kumar Khan, A Study Of Load Distribution Algorithms In Distributed Scheduling, IJRET: International Journal of Research in Engineering and Technology , Volume: 02 Special Issue: 02, Dec-2011.

[3]. Casavant T.L. and Kuhl J.G., "A taxonomy of scheduling in general-purpose distributed computing systems", IEEE Transactions on Software Engineering, pp. 141-154, 1988.

[4]. Livny M. and Melman M., "Load balancing in homogeneous broadcast distributed systems", Proceedings of the ACM Computer Network Performance Symposium, Vol. 11, Issue 1, pp. 47-55, 1982.

[5]. Waldspurger, C.A. and Weihl, W.E., "Stride scheduling: deterministic proportional-share resource management", Technical Report MIT/LCS/TM-528, Massachusetts Institute of Technology, MIT Laboratory for Computer Science.,1995.

[6]. Moal D. L., Ikumo M., Tsumura T., Goshima M., Mori S., Nakashima Y., Kitamura T. and Tomita S., "Priority enhanced stride scheduling", IPSJ Transactions on High Performance Computing Systems, Vol. 43, No. SIG 6(HPS 5), pp. 99-111, Sept. 2002.

[7]. Nieh J., Vaill C. and Zhong H., "Virtual-time round-robin: an o(1) proportional share scheduler", Proceedings of the General Track: 2001 USENIX Annual Technical Conference (The USENIX Association), pp. 245-259, June 25 - 30 2001.

[8]. Coming D. S. and Staadt O. G., "Stride scheduling for time-critical collision detection", Proceedings of the ACM Symposium on Virtual Reality Software and Technology(VRST-2007), pp. 5-7, Nov. 2007.

[9]. Krueger A. and Livny M., "The diverse objectives of distributed scheduling policies", proceedings of 7th international conference of distributed computing systems, IEEE CS, pp. 242-249, 1987.

[10].N.G. Shivaratri and M. Singhal. “Advanced Concepts in Operating Systems”. Tata McGraw Hill Education Pvt. Ltd., ed. 30, 2012.

[11].Karatza, H.D., Simulation Study of Task Scheduling and Resequencing in a Multiprocessing System. Simulation Journal, Special Issue: Modelling and Simulation of Computer Systems and Networks: Part -Two, SCS, Vol. 4, Issue 68, pp. 241-247, 1997.

[12].Karatza, H.D., Scheduling Strategies for Multitasking in a Distributed System., Proceedings of the 33rd Annual Simulation Symposium, IEEE Computer Society, pp. 83-90 , 16-20 April 2000.

[13].Karatza, H.D., A Comparative Analysis of Scheduling Policies in a Distributed System using Simulation, International Journal of Simulation Systems, Science & Technology, UK Simulation Society, Vol. 1(1-2), pp. 12-20, 2000.

[14].Shukla, D., Jain, S. and Singhai, R., "A Markov Chain Model for the analysis of Round Robin scheduling scheme", International Journal of Advanced Networking and Applications, Vol.1, No.1, pp. 1-7, 2009.

[15].Jain, S. and Jain, S., "Probability-Based Analysis to Determine the Performance of Multilevel Feedback Queue Scheduling", International Journal Advanced Networking and Applications(IJANA), Vol. 08, Issue 03, pp. 3044-3069, 2016.

[16].Ross S. M., Stochastic Processes, 2nd ed. New York: J. Wiley and Sons, 1996.

[17].J. Medhi, Stochastic processes, Ed. 4, Wiley Limited (Fourth Reprint), New Delhi, 1991.

[18].Basu A., Bensalem S., Peled D., and Sifakis J., "Priority Scheduling of Distributed Systems Based on Model Checking", CAV 2009, pp 79-93, 2009.

[19].Andrew Wang, Lottery and stride scheduling, July 17, 2011, available on “http://www.umbrant.com/blog/2011/lottery_stride_scheduling.html”.

[20].Waldspurger, C.A. and Weihl, W.E., Lottery Scheduling: Flexible Proportional-Share Resource Management, Proceedings of the First Symposium on Operating System Design and Implementation, MIT Laboratory for Computer Science, November 1994.

[21].Lindberg M., "A survey of reservation-based scheduling", Department of Automatic Control, Lund Institute of Technology, pp. 1-33, October 2007.