International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 5 | ISSN : 2456-3307

A Comparative Study of Single-Queue Multiprocessor Scheduling (SQMS) and Multiple-Queue Multiprocessor Scheduling (MQMS) Based on Stochastic Modeling

Shweta Jain[1], Saurabh Jain[2]
[1]Research Scholar, Faculty of Computer Science, Pacific Academy of Higher Education and Research University, Udaipur,Rajasthan, India
[2]Professor, Shri Vaishnav Institute of Computer Applications, Shri Vaishnav Vidyapeeth Vishwavidyalaya, Indore, Madhya Pradesh, India

ABSTRACT

Multiprocessor systems have multiple processors which are basically challenging for managing resources and processor time. There is no system which can be completely deterministic whether the system has enough capacity to handle all requests and loads within the requisite time. Multiprocessor system scheduling are having two types of queue based scheduling: Single-Queue Multiprocessor Scheduling (SQMS) and Multiple Queue Multiprocessor Scheduling (MQMS). This paper analyzes both these multiprocessor scheduling schemes and their transition behaviour of processes over queues for balancing the load of a system. We have also applied stochastic modeling for simulation study and compared their performance.

Keywords : Multiprocessor systems, MQMS, Markov Chain model, Process migration, SQMS, Scheduling, Stochastic process, Transition Probability Matrix, Affinity.

I. INTRODUCTION

Today's biggest challenge of operating system designers is to design efficient scheduling policies due to the increasing growth of computation power and ability to solve different problems in operating systems. Poor performance of the operating system can have considerable impact on application's performance. Scheduling is the procedure of assigning processor time and resources to the processes for execution. Multiprocessor systems are those systems that need multiple processors and multiple memory units (optional) to provide greater capacity. These systems are increasingly at the high-end of the computing spectrum in which data is shared across multiple processors. [1], [3], [4], [8] [9] [10] developed genetic algorithms to solve the multiprocessor scheduling environment problems with fitness value and the percentage of success for scheduling real time tasks using GA parameters result in terms of fitness value and the percentage of success for scheduling real time tasks.

The key objective is to find well-organized scheduling algorithms that can efficiently exploit the accessible processors with small run-time overhead. To fully utilize the system and decrease the completion time of parallel applications by accurately distributing the processes to the processors. [2] reviewed multiprocessor operating systems that is relevant to maximize performance, focusing on the exploitation of locality in large scale systems. [4], [5] proposed the heuristics and meta-heuristic approach to find the optimal solution for wide spectrum of techniques and general problems. [6] presented Group Ratio Round-Robin (GR3), the first proportional share scheduler that combines accurate proportional fairness scheduling behaviour with O(1) scheduling overhead on both uniprocessor and multiprocessor systems.

As hardware evolved and workload increased, operating system's designers forced to pursue procedures which would ensure correctness but also achieve higher performance to facing the challenges of concurrent workload demands. [14], [15], [16] characterized the multi-queue multi-server (MQMS) queuing systems and model in addressing resource allocation problems in various networks in a time-slotted with stochastic ordering that the results are still valid for more general connectivity and arrival processes with simultaneously maximizes the instantaneous throughput and balances the queues. [7],

[11], [12], [13] worked on different task duplication-based (TDB) scheduling algorithms and discussed different types of performance metrics in terms of their priority attributes, time complexity, greater efficiency, minimum makespan time, speedup, efficiency, load balance and normalized scheduling length. [17][18][19] [20] [21] [22] [23] used a Markov chain model for the study of transition probabilities with random jumps of scheduler to analyze different scheduling schemes like Multi-level Queue(MLQ), Multi-level Feedback Queue(MLFQ) and Round Robin(RR).

A multiprocessor scheduler should consider affinity when making its scheduling decisions, perhaps preferring to keep a process on the same CPU if possible and to run it on the same processor, as it will run faster if some of its state is already present in the caches on that CPU. If, instead, one runs a process on a different CPU each time, the performance of the process will be worse, as it will have to reload the state each time it runs. [24], [28] studied brief of stochastic processes and Markov chain Model and [25], [26], [27] helped to understand different CPU scheduling scheme in various kind of operating systems.

II. MULTIPROCESSOR SYSTEMS SCHEDULING MODEL

In Multiprocessor systems, the arriving processes are kept in the process queue, moved on to the scheduler for process scheduling and then allocates these processes to other processors in the system. Every processor has a dispatch queue connected with it. The processor carry out processes in the order they appear in the dispatch queue. These dispatch queues is responsible for the communication between the scheduler and the processors. The scheduler works simultaneously with the processors. The scheduler schedules the recently arriving processes and informs the dispatch queue while the processors execute the processes assigned to them. The scheduler makes sure that the dispatch queues of the processors are filled with a minimum number of processes so that the processors will always have a few processes to execute after executing the currently running processes.

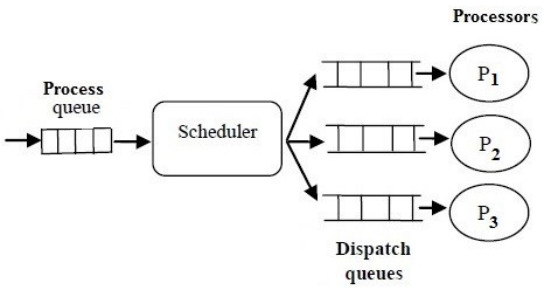

Figure 1: The Multiprocessor System Scheduling Model

There are two types of multiprocessor scheduling:
1. Single Queue Multiprocessor scheduling (SQMS) 2. Multi Queue Multiprocessor scheduling (MQMS)

Single queue scheduling in which by putting all processes that need to be scheduled into a single queue that is referred by single-queue multiprocessor scheduling or SQMS. It is the simplest approach that picks the best job to run next and adjust it to work on more than one CPU. But this approach is having lack of scalability and cache affinity. The scheduler works correctly so inserted some form of locking to accesses the single queue but locks can greatly reduce performance as the number of CPUs in systems grows. To handle this affinity problem, most SQMS schedulers having processes will continue to run on the same CPU if possible to balance load whereas implementing some kind of affinity fairness could be complex.

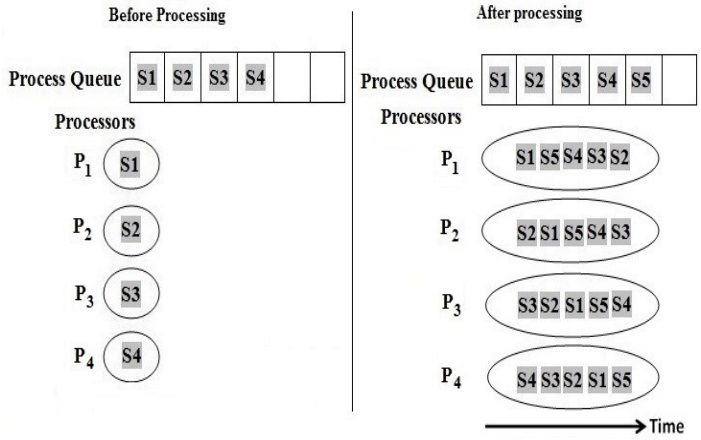

Figure 2: The General Single Queue Multiprocessor Scheduling Model

Figure 2 depicts general SQMS model in which all processes S1 to S4 go into a single queue (process queue) and processors P_1 to P_4 pull these processes from the queue as needed or on demand. SQMS can be modified to preserve affinity.

Because of the synchronization overheads caused in single-queue schedulers, some systems choose for

multiple queues, e.g., one queue per CPU so this approach is referred as multi-queue multiprocessor scheduling (MQMS) which consists of multiple scheduling queues in which as a process enters the system placing on exactly one scheduling queue that follows some heuristic or randomly picking one of them. So that independently scheduled and eliminating the problems of sharing information and synchronization occurring in the single-queue approach. Probably each queue follow a particular scheduling scheme like round robin and any other algorithm can be used.

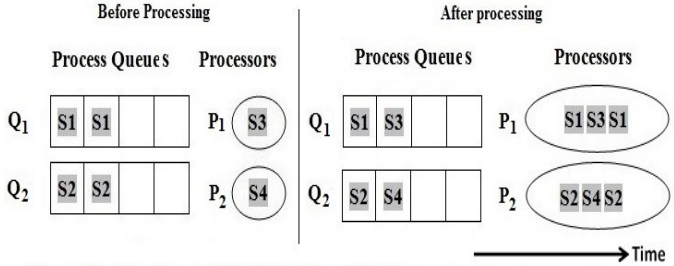

Figure 3: The General Multiple Queue Multiprocessor Scheduling Model

The figure 3 depicts MQMS model in which each processor P_1 and P_2 maintains its own queue Q_1 and Q_2 respectively. Processors schedule their processes independently. MQMS have multiple queues as per processor. So MQMS scheduling consists of multiple scheduling queues in which processes migrate into queues.

MQMS should be inherently more scalable than SQMS. As the number of processor increases then simultaneously increasing the number of queues so that no longer exits lock and cache contention problems and also provides affinity. The biggest challenge in multi-queue based approach having load imbalance. By solving this problem is to move processes by process migration. By migrating a process from one CPU to another, true load balance can be achieved. This paper applies stochastic model to these multiprocessor scheduling schemes to improve the migration of processes and load balance and compare their performance.

III. THE STOCHASTIC MODEL

A Markov process is a stochastic or random process in which the probability distribution of the current state is conditionally independent path of past states, a characteristic called the Markov property. So Markov chain is a discrete-time stochastic process with the Markov property. This is a stochastic model that generates sequences in which the probability of a symbol depends on the previous symbol.

The probability that a sequence x is generated by a Markov chain model

$$P(x) = P(x_1, x_2, \ldots\ldots, x_n)$$
$$= P(x_1, x_2, \ldots\ldots, x_{n-1}).\ P(x_n \mid x_1, x_2, \ldots\ldots, x_{n-1})$$
$$= P(x_1, x_2, \ldots\ldots, x_{n-2}).\ P(x_{n-1} \mid x_1, x_2, \ldots\ldots, x_{n-2}).\ P(x_n \mid x_1, x_2, \ldots\ldots, x_{n-1})$$
$$= P(x_1).\ P(x_2 \mid x_1) \ldots\ldots\ P(x_n \mid x_1, x_2, \ldots\ldots, x_{n-1})$$

One assumption of Markov chain is that the probability of x_i only depend on the previous symbol x_{i-1} i.e.

$$P(x_n \mid x_1, x_2, \ldots\ldots, x_{n-1}) = P(x_n \mid x_{n-1})$$

Thus,

$$P(x) = P(x_1).\ P(x_2 \mid x_1) \ldots\ldots\ P(x_n \mid x_{n-1})$$

We apply this stochastic Markov Chain model to queue-based multiprocessor scheduling including SQML and MQML schemes using some assumptions and restrictions.

IV. THE PROPOSED MARKOV CHAIN MODEL FOR SQMS AND MQMS SCHEMES

In multiprocessor system model, we assign P processors to a set of N parallel queues in a time-slot. The connectivity of each queue to each processor is assumed to be transition process. In MQMS, each processor can process at most one queue and each queue can be processed by at most one processor during each time slot whereas in SQMS each processor can process only single queue at each time slot. So optimal assignment policy over queues and processors for a system in stochastic ordering to balance load and concurrency. Considering two types of scheduling schemes SQML and MQML with Markov Chain model as follows:

4.1. SQMS Schemes

Figure 5 and 6 depict that single queue multiprocessor scheduling (SQMS) scheme defines three processors P_1, P_2, P_3 and a queue Q which accepts a new process. In this scheme, putting all processes that need to be scheduled into a single queue Q and each processor simply picks the next process from this shared queue Q globally. A quantum is a little pre-defined time slot that

is specified for executing processes in a queue Q. So there are 4 states in this scheduling systems in which scheduler has a random movement over these states Q, P_1, P_2 and P_3 as per variation in quantum. Scheduler moves from one state to other state at the end of a quantum. Process remains with processors until the quantum is completed. If a process didn't complete in specified quantum then scheduler allots next quantum to the next process of the queue Q. Symbol n assigns the n^{th} quantum that is allocated by the scheduler to a process for execution (n=1,2,3,4.). Suppose $\{X^{(n)}, n\geq 0\}$ be a Markov chain where X is a state of scheduler at the n^{th} quantum of time that jumps randomly over these 4 states in different quantum of time. The state space for the random variable X is $\{Q, P_1, P_2, P_3\}$.

a) Unrestricted SQMS Scheme

Figure 4 depicts unrestricted transition model of SQMS scheme in which each queue get processing from at most one processor during each time slot and transition among processors. Consider the initial probabilities of states are as follows:

$$P[X^{(0)} = Q] = pb_1, \quad P[X^{(0)} = P_1] = pb_2$$
$$P[X^{(0)} = P_2] = pb_3, \quad P[X^{(0)} = P_3] = pb_4$$

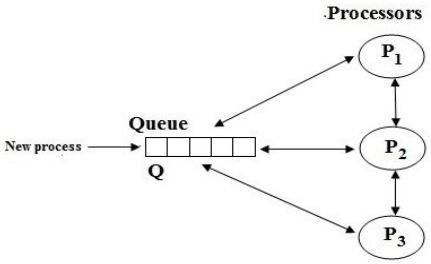

Figure 4: The Unrestricted Transition Model of SQMS Scheme

Let T_{ij} (i, j=1,2,3,4) be transition probabilities of scheduler over four states then transition probability matrix are shown in figure 5.

$X^{(n-1)}$ \ $X^{(n)}$	Q	P_1	P_2	P_3
Q	T_{11}	T_{12}	T_{13}	T_{14}
P_1	T_{21}	T_{22}	T_{23}	T_{24}
P_2	T_{31}	T_{32}	T_{33}	T_{34}
P_3	T_{41}	T_{42}	T_{43}	T_{44}

Figure 5: Trasition Probability Matrix

The state probabilities can be determined for the n^{th} quantum by the following expressions:

$$P[X^{(n)}=Q] = \left[\sum_{m=1}^{4} \cdots \sum_{l=1}^{4} \sum_{k=1}^{4} \left\{ \sum_{j=1}^{4} \left(\sum_{i=1}^{4} pb_i T_{ij} \right) T_{jk} \right\} T_{kl} \cdots T_{ml} \right]$$

$$P[X^{(n)}=P_1] = \left[\sum_{m=1}^{4} \cdots \sum_{l=1}^{4} \sum_{k=1}^{4} \left\{ \sum_{j=1}^{4} \left(\sum_{i=1}^{4} pb_i T_{ij} \right) T_{jk} \right\} T_{kl} \cdots T_{m2} \right]$$

$$P[X^{(n)}=P_2] = \left[\sum_{m=1}^{4} \cdots \sum_{l=1}^{4} \sum_{k=1}^{4} \left\{ \sum_{j=1}^{4} \left(\sum_{i=1}^{4} pb_i T_{ij} \right) T_{jk} \right\} T_{kl} \cdots T_{m3} \right]$$

$$P[X^{(n)}=P_3] = \left[\sum_{m=1}^{4} \cdots \sum_{l=1}^{4} \sum_{k=1}^{4} \left\{ \sum_{j=1}^{4} \left(\sum_{i=1}^{4} pb_i T_{ij} \right) T_{jk} \right\} T_{kl} \cdots T_{m4} \right]$$

b) Restricted SQMS Scheme

In figure 6**,** some transition among queue and processors are restricted where a new process can enter to the queue Q then move process queue to/from any of parallel processors P_1, P_2 and P_3. Some transitions P_2 to P_1, P_3 to P_2, P_1 to P_3 and P_3 to P_1 are restricted as a special scheme.

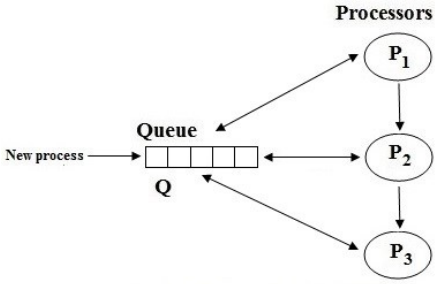

Figure 6: The Restricted Transition Model of SQMS Scheme

So the transition probability matrix under restricted scheme are

$X^{(n-1)}$ \ $X^{(n)}$	Q	P_1	P_2	P_3
Q	T_{11}	T_{12}	T_{13}	T_{14}
P_1	T_{21}	T_{22}	T_{23}	0
P_2	T_{31}	0	T_{33}	T_{34}
P_3	T_{41}	0	0	T_{44}

Figure 7: Trasition Probability Matrix for Restricted SQMS Scheme

Define Indicator function R_{ij}, where i, j = 1, 2, 3, 4 such that

$$R_{ij} = \begin{cases} 0 \text{ when } (i=2, j=4)(i=3, j=2)(i=4, j=2)(i=4, j=3) \\ 1 \text{ otherwise} \end{cases}$$

The generalized equation of SQSM restricted scheme for n quantum is as follows:

$$P[X^{(n)}=Q] = \Big[\sum_{m=1}^{4} \cdots\cdots \sum_{k=1}^{4} \sum_{j=1}^{4} \sum_{i=1}^{4} (R_{1i}T_{1i})(R_{ij}T_{ij})(R_{jk}T_{jk}) \cdots\cdots (R_{m1}T_{m1}) \Big]$$

$$P[X^{(n)}=P_1] = \Big[\sum_{m=1}^{4} \cdots\cdots \sum_{k=1}^{4} \sum_{j=1}^{4} \sum_{i=1}^{4} (R_{1i}T_{1i})(R_{ij}T_{ij})(R_{jk}T_{jk}) \cdots\cdots (R_{m2}T_{m2}) \Big]$$

$$P[X^{(n)}=P_2] = \Big[\sum_{m=1}^{4} \cdots\cdots \sum_{k=1}^{4} \sum_{j=1}^{4} \sum_{i=1}^{4} (R_{1i}T_{1i})(R_{ij}T_{ij})(R_{jk}T_{jk}) \cdots\cdots (R_{m3}T_{m3}) \Big]$$

$$P[X^{(n)}=P_3] = \Big[\sum_{m=1}^{4} \cdots\cdots \sum_{k=1}^{4} \sum_{j=1}^{4} \sum_{i=1}^{4} (R_{1i}T_{1i})(R_{ij}T_{ij})(R_{jk}T_{jk}) \cdots\cdots (R_{m4}T_{m4}) \Big]$$

4.2. MQMS schemes

Assume that three processors P_1, P_2 and P_3 are assigned to a set of queues Q_1, Q_2, and Q_3 in a time slot manner in MQMS scheme that are shown in figure 8 and 10 where queues Q_1, Q_2, and Q_3 accept a new process. In this scheme, placing all processes that need to be scheduled into queues Q_1, Q_2, and Q_3 and each processor chooses the next process from these queues. A quantum is a tiny pre-defined time period that is specified for executing processes in these queues. So there are six states in this scheduling systems in which scheduler has a stochastically moving over these states Q_1, Q_2, Q_3, P_1, P_2 and P_3 that varied according to quantum. Scheduler moves from one state to other state at the end of a quantum. Process remains with processors until the quantum is completed. If a process didn't complete in specified quantum then scheduler allots next quantum to the next process of the queues. Symbol n assigns the n^{th} quantum that is allocated by the scheduler to a process for execution (n=1,2,3,4,5,6). Suppose $\{X^{(n)}, n\geq 0\}$ be a Markov chain where X is a state of scheduler at the n^{th} quantum of time that goes randomly over these six states in different quantum of time. The state space for the random variable X is { Q_1, Q_2, Q_3, P_1, P_2, P_3}.

a) Unrestricted MQMS Scheme

The connectivity of each queue to each processor is randomly changing with time; each processor can serve at most one queue and each queue can be served by at most one processor during each time slot as stochastic ordering which is mentioned in figure 9.

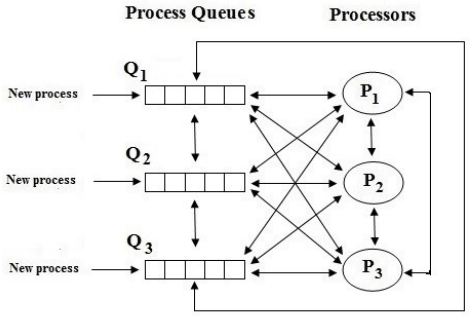

Figure 9 : The Unrestricted Transition Model of MQMS Scheme

Consider the initial probabilities of states are as follows:

$$P[X_{(0)}=P_1] = p_{11} \quad P[X_{(0)}=P_2] = p_{12} \quad P[X_{(0)}=P_3] = p_{13}$$
$$P[X_{(0)}=Q_1] = p_{21} \quad P[X_{(0)}=Q_2] = p_{22} \quad P[X_{(0)}=Q_3] = p_{23}$$

Now defining Transition Probability Matrix for this scheme is in figure 10

$X^{(n-1)}$ \ $X^{(n)}$	Q_1	Q_2	Q_3	P_1	P_2	P_3
Q_1	T_{11}	T_{12}	T_{13}	T_{14}	T_{15}	T_{16}
Q_2	T_{21}	T_{22}	T_{23}	T_{24}	T_{25}	T_{26}
Q_3	T_{31}	T_{32}	T_{33}	T_{34}	T_{35}	T_{36}
P_1	T_{41}	T_{42}	T_{43}	T_{44}	T_{45}	T_{46}
P_2	T_{51}	T_{52}	T_{53}	T_{54}	T_{55}	T_{56}
P_3	T_{61}	T_{62}	T_{63}	T_{64}	T_{65}	T_{66}

Figure 10 : Trasition Probability Matrix of Unrestricted MQMS Scheme

The generalized equation of MQMS unrestricted scheme for n quantum is as follows:

$$P[X^{(n)}=Q_1] = \sum_{m=1}^{6} \cdots\cdots \sum_{l=1}^{6} \sum_{k=1}^{6} \sum_{j=1}^{6} \sum_{i=1}^{6} p_{bi} T_{ij} T_{jk} T_{kl} \cdots\cdots T_{m1}$$

$$P[X^{(n)}=Q_2] = \sum_{m=1}^{6} \cdots\cdots \sum_{l=1}^{6} \sum_{k=1}^{6} \sum_{j=1}^{6} \sum_{i=1}^{6} p_{bi} T_{ij} T_{jk} T_{kl} \cdots\cdots T_{m2}$$

$$P[X^{(n)}=Q_3] = \sum_{m=1}^{6} \cdots\cdots \sum_{l=1}^{6} \sum_{k=1}^{6} \sum_{j=1}^{6} \sum_{i=1}^{6} p_{bi} T_{ij} T_{jk} T_{kl} \cdots\cdots T_{m3}$$

$$P[X^{(n)}=P_1] = \sum_{m=1}^{6} \cdots\cdots \sum_{l=1}^{6} \sum_{k=1}^{6} \sum_{j=1}^{6} \sum_{i=1}^{6} p_{bi} T_{ij} T_{jk} T_{kl} \cdots\cdots T_{m4}$$

$$P[X^{(n)}=P_2] = \sum_{m=1}^{6} \cdots\cdots \sum_{l=1}^{6} \sum_{k=1}^{6} \sum_{j=1}^{6} \sum_{i=1}^{6} p_{bi} T_{ij} T_{jk} T_{kl} \cdots\cdots T_{m5}$$

$$P[X^{(n)}=P_3] = \sum_{m=1}^{6} \cdots\cdots \sum_{l=1}^{6} \sum_{k=1}^{6} \sum_{j=1}^{6} \sum_{i=1}^{6} p_{bi} T_{ij} T_{jk} T_{kl} \cdots\cdots T_{m6}$$

b) Restricted MQMS Scheme

Figure 11 represents restricted MQMS scheme in which each queue is restricted to get processing from at most one processor during each time slot.

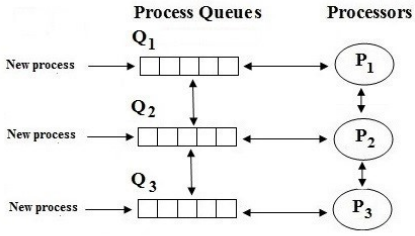

Figure 11: The Restricted Transition Model of MQMS Scheme

Defining Transition Probability Matrix for this scheme is in figure 12

$X^{(n-1)}$ \ $X^{(n)}$	Q_1	Q_2	Q_3	P_1	P_2	P_3
Q_1	T_{11}	T_{12}	0	T_{14}	0	0
Q_2	T_{21}	T_{22}	T_{23}	0	T_{25}	0
Q_3	0	T_{32}	T_{33}	0	0	T_{36}
P_1	T_{41}	0	0	T_{44}	T_{45}	T_{46}
P_2	0	T_{52}	0	T_{54}	T_{55}	T_{56}
P_3	0	0	T_{63}	T_{64}	0	T_{66}

Figure 12 : Trasition Probability Matrix of Unrestricted MQMS Scheme

Define indicator function R_{ij}, where i, j = 1, 2, 3, 4, 5, 6 such that

$$R_{ij} = \begin{cases} 0 \text{ if } (i=1, j=3)\,(i=1, j=4)\,(i=1, j=5)\,(i=2, j=6) \\ (i=3, j=1\,(i=3, j=4)\,(i=3, j=5)\,(i=4, j=2)\,(i=4, j=3) \\ (i=5, j=1)\,(i=5, j=3)\,(i=6, j=1)\,(i=6, j=2)\,(i=6, j=5) \\ 1 \text{ otherwise} \end{cases}$$

The generalized equation of MQMS restricted scheme using indicator function R_{ij} for n quantum is as follows:

$$P[X^{(n)}=Q_1] = \sum_{m=1}^{6} \cdots \sum_{l=1}^{6} \sum_{k=1}^{6} \sum_{j=1}^{6} \sum_{i=1}^{6} (R_{li}T_{li})\,(R_{lj}T_{lj})\,(R_{jk}T_{jk}) \cdots (R_{m1}T_{m1})$$

$$P[X^{(n)}=Q_2] = \sum_{m=1}^{6} \cdots \sum_{l=1}^{6} \sum_{k=1}^{6} \sum_{j=1}^{6} \sum_{i=1}^{6} (R_{li}T_{li})\,(R_{lj}T_{lj})\,(R_{jk}T_{jk}) \cdots (R_{m2}T_{m2})$$

$$P[X^{(n)}=Q_3] = \sum_{m=1}^{6} \cdots \sum_{l=1}^{6} \sum_{k=1}^{6} \sum_{j=1}^{6} \sum_{i=1}^{6} (R_{li}T_{li})\,(R_{lj}T_{lj})\,(R_{jk}T_{jk}) \cdots (R_{m3}T_{m3})$$

$$P[X^{(n)}=P_1] = \sum_{m=1}^{6} \cdots \sum_{l=1}^{6} \sum_{k=1}^{6} \sum_{j=1}^{6} \sum_{i=1}^{6} (R_{li}T_{li})\,(R_{lj}T_{lj})\,(R_{jk}T_{jk}) \cdots (R_{m4}T_{m4})$$

$$P[X^{(n)}=P_2] = \sum_{m=1}^{6} \cdots \sum_{l=1}^{6} \sum_{k=1}^{6} \sum_{j=1}^{6} \sum_{i=1}^{6} (R_{li}T_{li})\,(R_{lj}T_{lj})\,(R_{jk}T_{jk}) \cdots (R_{m5}T_{m5})$$

$$P[X^{(n)}=P_3] = \sum_{m=1}^{6} \cdots \sum_{l=1}^{6} \sum_{k=1}^{6} \sum_{j=1}^{6} \sum_{i=1}^{6} (R_{li}T_{li})\,(R_{lj}T_{lj})\,(R_{jk}T_{jk}) \cdots (R_{m6}T_{m6})$$

V. GRAPHICAL ANALYSIS AND SIMULATION STUDY

Our analysis is based on stochastic modeling technique which is a interactive scientific approach to study the behaviour of proposed system. Simulation study is performed through Markov Chain Model to analyze the scheduler behaviour under queues and processors according to proposed multiprocessor scheduling schemes that are categorized into restricted and unrestricted SQMS and MQMS.

1) For SQMS Scheme

Let the initial probabilities for both SQMS Schemes are $Pb_1 = 1$, $Pb_2 = 0$, $Pb_3 = 0$, $Pb_4 = 0$.

a) Unrestricted SQMS scheme

Consider the following Transition probability matrices and graph according to quantum variation for unrestricted SQMS Scheme:

$X^{(n-1)}$ \ $X^{(n)}$	Q	P1	P2	P3
Q	0.4	0.25	0.15	0.2
P1	0.17	0.39	0.11	0.33
P2	0.65	0.1	0.12	0.13
P3	0.31	0.16	0.41	0.12

Table 5.1: $P[X^{(n)} = Q / X^{(n)} = P_i]$ for Transition Probability Matrices of Unrestricted SQMS Scheme

Quantum	Probabilities			
	Q	P1	P2	P3
n = 1	0.4	0.25	0.15	0.2
n = 2	0.3620	0.2445	0.1875	0.2060
n = 3	0.3721	0.2376	0.1882	0.2022
n = 4	0.3742	0.2368	0.1874	0.2015
n = 5	0.3742	0.2369	0.1873	0.2015
n = 6	0.3742	0.2369	0.1873	0.2016
n = 7	0.3742	0.2369	0.1873	0.2016

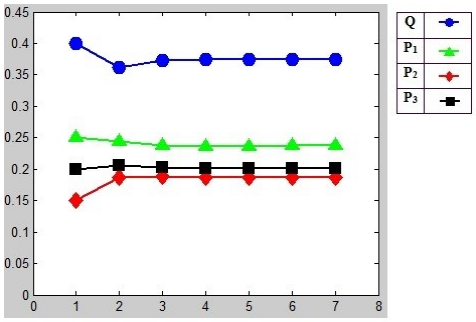

Fig 5.1

b) Restricted SQMS scheme

Consider the following Transition probability matrices and graph according to quantum variation for restricted SQMS Scheme:

$X^{(n-1)}$ \ $X^{(n)}$	Q	P1	P2	P3
Q	0.38	0.11	0.14	0.37
P1	0.13	0.27	0.60	0
P2	0.29	0	0.13	0.58
P3	0.15	0	0	0.85

Table 5.2: $P[X^{(n)} = Q / X^{(n)} = P_i]$ for Transition Probability Matrices of restricted SQMS Scheme

Quantum	Probabilities			
	Q	P_1	P_2	P_3
n = 1	0.38	0.11	0.14	0.37
n = 2	0.2548	0.0715	0.1374	0.5363
n = 3	0.2264	0.0473	0.0964	0.6298
n = 4	0.2146	0.0377	0.0726	0.6751
n = 5	0.2088	0.0338	0.0621	0.6953
n = 6	0.2060	0.0321	0.0576	0.704
n = 7	0.2048	0.0313	0.0556	0.7083

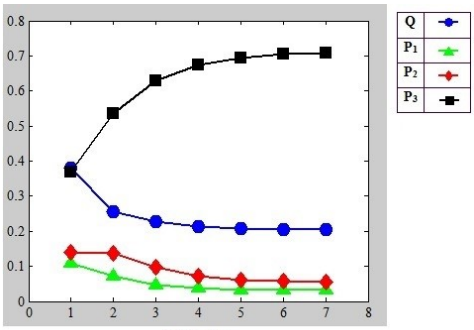

Fig: 5.2

2) For MQMS Scheme

Let the initial probabilities for both MQMS Schemes are $Pb_1 = 1$, $Pb_2 = 0$, $Pb_3 = 0$, $Pb_4 = 0$, $Pb_5 = 0$, $Pb_6 = 0$.

a) Unrestricted MQMS scheme

Consider the following Transition probability matrices and graph according to quantum variation for unrestricted MQMS Scheme:

$X^{(n-1)}$ \ $X^{(n)}$	Q_1	Q_2	Q_3	P_1	P_2	P_3
Q_1	0.27	0.03	0.06	0.09	0.14	0.41
Q_2	0.19	0.01	0.07	0.13	0.26	0.34
Q_3	0.25	0.1	0.05	0.15	0.3	0.15
P_1	0.45	0.05	0.35	0.08	0.02	0.05
P_2	0.11	0.23	0.17	0.25	0.04	0.2
P_3	0.06	0.14	0.08	0.42	0.18	0.12

Table 5.3: $P[X^{(n)} = Q_i / X^{(n)} = P_i]$ for Transition Probability Matrices of Unrestricted MQMS Scheme

Quantum	Probabilities					
	Q_1	Q_2	Q_3	P_1	P_2	P_3
n = 1	0.27	0.03	0.06	0.09	0.14	0.41
n = 2	0.1741	0.1085	0.1094	0.2516	0.1448	0.2116
n = 3	0.2368	0.0928	0.1531	0.1914	0.1343	0.1916
n = 4	0.2322	0.0906	0.1335	0.1857	0.1469	0.2110
n = 5	0.2257	0.0938	0.1338	0.1929	0.1437	0.21
n = 6	0.2274	0.0932	0.1356	0.1922	0.1436	0.2081
n = 7	0.2278	0.0931	0.1353	0.1916	0.1438	0.2086

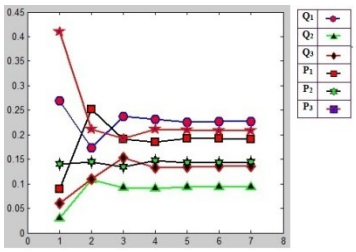

Fig 5.3

b) Restricted MQMS scheme

Consider the following Transition probability matrices and graph according to quantum variation for restricted MQMS Scheme:

$X^{(n-1)}$ \ $X^{(n)}$	Q_1	Q_2	Q_3	P_1	P_2	P_3
Q_1	0.34	0.16	0	0.5	0	0
Q_2	0.11	0.07	0.23	0	0.59	0
Q_3	0	0.44	0.16	0	0	0.4
P_1	0.32	0	0	0.14	0.06	0.48
P_2	0	0.46	0	0.04	0.16	0.34
P_3	0	0	0.76	0.11	0	0.13

Table 5.4: $P[X^{(n)} = Q_i / X^{(n)} = P_i]$ for Transition Probability Matrices of Restricted MQMS Scheme

Quantum	Probabilities					
	Q_1	Q_2	Q_3	P_1	P_2	P_3
n = 1	0.34	0.16	0	0.5	0	0
n = 2	0.2932	0.0656	0.0368	0.24	0.1244	0.24
n = 3	0.1837	0.1249	0.2034	0.2116	0.0730	0.2034
n = 4	0.1439	0.1612	0.2159	0.1468	0.0981	0.2342
n = 5	0.1136	0.1744	0.2496	0.1222	0.1196	0.2206
n = 6	0.0969	0.1952	0.2477	0.1030	0.1294	0.2278
n = 7	0.0874	0.1977	0.2577	0.0931	0.1421	0.2221

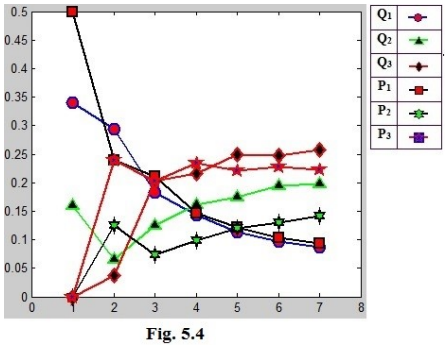

Fig. 5.4

VI. EVALUATION AND COMPARITION ON GRAPHS OF PROPOSED SCHEMES

Figure 5.1 represents graph of SQMS unrestricted scheme in which analysing state probabilities of single queue Q is little higher than processors P_1, P_2 and P_3. As quantum increases specially for n >= 2, values of

state probabilities are almost stable. Basically graph patterns are changing for the value of n=1 and n=2. This is to observe that this scheme has transiting processes in simple way among queue Q and processors P_1, P_2 and P_3 as the concept of SQMS.

Figure 5.2 signifies SQMS restricted scheme graph where probability of P_3 is going to upward direction whereas for single queue Q and for processors P_1, P_2 are moving downward as per quantum variation due to perform some restrictions. It shows that processes migrate among Q and P_1, P_2 to balance load in efficient way and better utilization of resources such as processor time.

Figure 5.3 symbolizes MQMS unrestricted scheme graph that shows three queues Q_1, Q_2, and Q_3 assigning to processors P_1, P_2 and P_3 where scheduler jumps randomly over all queues and processors without any restrictions. Here probabilities are altering upto the quantum n = 1 to n = 3 whereas from n = 4, probabilities comes almost stable or small variations. It proves that MQMS is more suitable and efficient than SQMS for handling multiple processes to maintain balance load among queues and processors.

Figure 5.4 depicts MQMS restricted scheme graph in which process migrations over queues and processors are restricted so that state probabilities of queues Q_1, Q_2, and Q_3 and processors P_1, P_2 and P_3 have lots of variations as the quantum grows. So we can say that the process scheduling aspect in this scheme is stronger than previous one.

VII. CONCLUSION

Proposed schemes are designed for single-queue and multi-queue multiprocessor scheduling using stochastic modeling. This paper evaluates and compares SQMS and MQMS under Markov Chain model. Simulation study is performed using graphical analysis applied on the proposed model in which restricted schemes is more useful, reliable and efficient than unrestricted schemes of SQMS and MQMS. It is remarkably clear that restricted MQMS scheme is better balancing of load during processing and more productively utilize the resources measures and metrics among queues and processors.

Observing one thing that if initial probabilities become change which does not affect graphical patterns of all these schemes that means minor changes occur. So that we can say MQMS provides overall better performance than SQMS to improve load balancing because increasing the number of queues as per processor in MQMS. We found that they needed to restructure the Multiprocessor OS to utilize the hardware more efficiently in order to improve performance, to reduce memory contention, to balance load and to migrate processes among queues and processors.

VIII. REFERENCES

[1]. Heidari H. and Chalechale A.: "Scheduling in Multiprocessor System using Genetic Algorithm", International Journal of Advanced Science and Technology, Vol. 43, June, 2012, pp 81-94.

[2]. Dhingra S., Gupta S.B. and Biswas R.: “Genetic Algorithm Parameters Optimization for Bi-Criteria Multiprocessor Task Scheduling using Design of Experiments”, International Journal of Computer, Information, Systems and Control Engineering, Vol. 8, No. 4, 2014, pp. 600-606.

[3]. Kwong Y. and Ahmad I.: “Static Scheduling Algorithms for allocating directed task graphs to multiprocessors”, ACM Computing Surveys, Vol. 31, No. 4, December 1999, pp. 407-427.

[4]. Arya S. and Dhingra S.: "Duplication Based Simulated Annealing Algorithm For Multiprocessor Task Scheduling: An Overview", International Journal of scientific Research, Vol. 4, Issue 9 ,Sept 2015, pp 325-327.

[5]. Caprita B., Chan W. C., Nieh J., Stein C., and Zheng H.: "Group Ratio Round-Robin: O(1) Proportional Share Scheduling for Uniprocessor and Multiprocessor Systems", USENIX Association- USENIX Annual Technical Conference, 2005, pp 337-352.

[6]. Gupta S., Rajak R., Singh G. K., and Jain S.: " Review of Task Duplication Based (TDB) Scheduling Algorithms", Smart Computing Review, Vol. 5, No. 1, January 2015, pp 67-75.

[7]. Jain S. and Makkar S.: "Multiprocessor Environment using Genetic Algorithm", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 2, Issue 5, May 2012, pp 131-134.

[8]. Pradhan S. R., Sharma S., Konar D. and Sharma K.: "A Comparative Study on Dynamic Scheduling of Real-Time Tasks in Multiprocessor System using Genetic Algorithms", International Journal of Computer Applications, Vol. 120, No.20, June 2015, pp 1-6.

[9]. Edwin S.H and Ansari N.: "A Genetic Algorithm for Multiprocessor Scheduling", IEEE Transaction on parallel and Distributed System, Vol. 5, No. 2, Feb. 1994.

[10]. Ahmad I. and Ghafoor A.: "Semi-distributed load balancing for massively parallel multicomputer systems," IEEE Transactions. on Software. Engineering, Vol. 17, Issue 10, 1991, pp. 987–1004.

[11]. Kaur R. and Kaur R.: "Multiprocessor Scheduling using Task Duplication Based Scheduling Algorithms: A Review Paper", International Journal of Application or Innovation in Engineering and Management, Vol. 2, Issue 4, April. 2013, pp. 311-317.

[12]. R. Rajak: "A Novel Approach for Task Scheduling in Multiprocessor System", International Journal of Computer Application, Vol. 44, No. 11, April 2012, pp. 12-14.

[13]. Halabian H., Lambadaris I. and Lung C. H.: "On the Stability Region of Multi-Queue Multi-Server Queueing Systems with Stationary Channel Distribution", Proceedings of IEEE International Symposium on Information Theory (ISIT'11), 31 July-5 Aug. 2011, pp 1801-1805.

[14]. Halabian H., Lambadaris I. and Lung C. H.: "Optimal Server Assignment in Multi-Server Queuing Systems with Random Connectivities", IEEE International Conference on Communications (ICC 2012), 10-15 June 2012, pp. 1219-1224.

[15]. Kittipiyakul S. and Javidi T.: "Delay-optimal server allocation in multi-queue multi-server systems with time-varying connectivities," IEEE Transactions on Information Theory, Vol. 55, No. 5, pp. 2319–2333, May 2009.

[16]. Shukla, D. and Jain, S.: "A Markov Chain Model for Multilevel Queue Scheduler in Operating System", Proceedings of International Conference on Mathematics and Computer Science, ICMCS-07, 2007, pp. 522-526.

[17]. Shukla, D. and Jain, S.: "Deadlock State Study in Security Based Multilevel Queue Scheduling Scheme in Operating System", Proceedings of National Conference on Network Security and Management, NCNSM-07, 2007, pp. 166-175.

[18]. Shukla, D., Jain, S., Singhai, R. and Agarwal, R. K.: "A Markov Chain Model for the Analysis of Round-Robin Scheduling Scheme", International Journal of Advanced Networking and Applications (IJANA), Vol. 1, Issue 1, 2009, pp. 1-7.

[19]. Jain, S. and Jain, S.: "A Research Survey and Analysis for CPU Scheduling Algorithms using Probability-Based Study", International Journal of Engineering and Management Research, Vol. 6, Issue 5, December 2015, pp. 628-633.

[20]. Jain, S. and Jain, S.: "Analysis of Multi Level Feedback Queue Scheduling Using Markov Chain Model with Data Model Approach", International Journal Advanced Networking and Applications(IJANA), Vol. 07 Issue 06, 2016, pp. 2915-2924.

[21]. 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, 2016, pp. 3044-3069.

[22]. Jain, S. and Jain, S: "A Review Study on the CPU Scheduling Algorithms", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 5, Issue 8, August 2016, pp. 22-31.

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

[24]. Silberschatz A., and Galvin P.: "Operating system concept", Ed.8, John Wiley and Sons, Asia, 2010.

[25]. W. Stalling, "Operating systems", Ed.5, Pearson Eduaction, Singopore, Indian Edition, New Delhi,2004.

[26]. A. Tanenbaum, and A.S. Woodhull, "Operating System", Ed. 8, Prentice Hall of India, NewDelhi, 2000.

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