Time Slicing Based Multiprocessor Scheduling

Merugu Gopichand, Professor & Head, Department of IT, Vardhaman College of Engineering, Shamshabad, Hyderabad, Telengana, India. Gopi_merugu@yahoo.com

Abstract-- Scheduler is the major component in all kind of high performance computing, various types of scheduling available in the open research domain. In the existing system, three types of scheduling algorithms viz Static, Node Duplication and Online scheduling are discussed. In this paper, Time Slicing based Multiprocessor Scheduling (TSMS) is introduced to improve the job execution rate.

Key words-- Node, Online, Queue, Static, TSBS.

I. INTRODUCTION

Multiprocessor scheduling is the low level venture of distributed computing, it has various research problems still under solving, the research problems are waiting time in the queue, load on a processor, failure rate of jobs and processors, and handling dynamic situation, in this paper we are concentrating to improving the job execution rate design constraints are fixed.

The rest of the paper is organized as follows Section-I surveys related work. Section-II explaining the proposed algorithm, Section-III explains the proposed algorithm, Section –IV Illustrates the experimental setup and results, Section-V Compares the proposed work with existing work and SectionVI concludes the paper with feature scope.

II. RELATED WORK

Choudhuri and Kumar [1] stated three different scheduling algorithms which are lacking in failure rate of Job's in consideration.

Rewini and lewis [2] explained the scheduling of parallel program execution under different targets yielding better performance on job execution but lacks in resource failure management.

Gupta and Mosse [3] revealed the parallel scheduling using the data structure named as Compact Task Graphs(CTG) for generating the schedule of execution on targets complied with non-adaptive representation of failure nodes in STC.

Kumar and Sing[4] deal with duplication on heterogeneity environment through precedence constrained task graphs which illustrates the mal behavior of resource of resource when duplication exists.

Ahmed and Kwok [5] concentrated on optimal allocation using search techniques get failed in search of sensitive nodes.

III. PROPOSED WORK

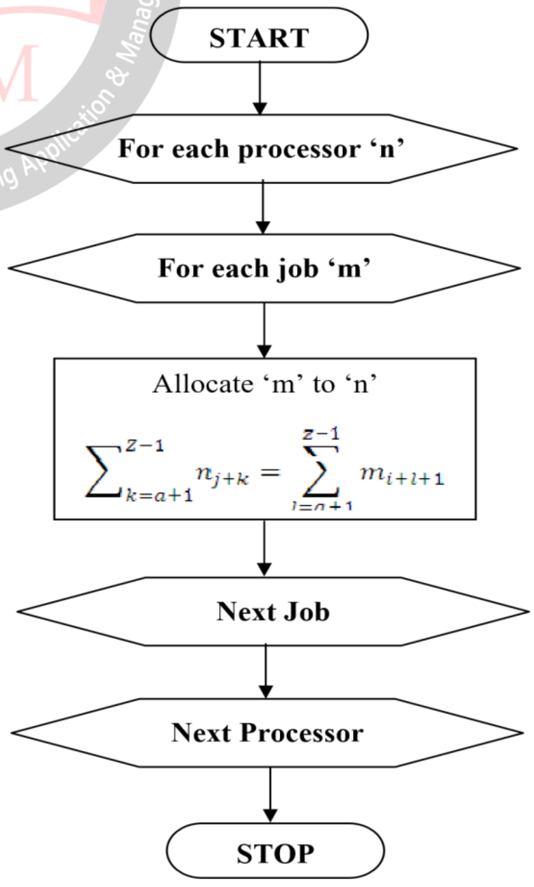
In the proposed work redundancy is introduced to improve the job execution rate, the system design is explained in the following sections.

A. METHODOLOGY

Consider 'n' number of processors are available in which 'm' number of jobs are scheduled in the following fashion .In P1 the jobs are arranged as J1, J2, J3,…., Jn-1, and in Pn the jobs are arranged as Jn, Jn-1, Jn-2,…J1 are distributed, if any of the processor gat crashed as well as any number of the processors malpractice, the job will be executed ,which in turn job failure rate is reduced as per Fig. 1.

B. ALGORITHM

```
STEP 1. Indentifying 'n' number of processors.
STEP 2. Allocate 'm' number of jobs to 'n' number of
        processors.
STEP 3. For each processor.
STEP 4. Allocate '$m_1$' to '$m_{n-1}$'.
STEP 5. Increment step4 up to '$m_n$'.
STEP 6. Terminate the Iteration.
```



IV. EXPERIMENTAL SETUP AND RESULTS

On implementing single server queuing concept in the existing algorithm shows the fluctuation rate in a linearly increasing fusion. In the same manner the fluctuation rate is getting decreased on implementing multiple server queuing theory in the heterogeneous environment. In the proposed work multiple queuing server concept is integrated with the scheduling algorithm. It shows further linearity in the error free execution.

V. PERFORMANCE ANALYSIS

On comparing with the existing algorithms the fluctuation rate is decreased in the proposed environment moreover when the number of processors or number of jobs are increased the same latency will exist in the proposed work observed from Table 1. and Fig. 2.

Table 1: Comparison

PROPOSED SYSTEM$_{ET}$	EXISTING SYSTEM$_{ET}$
0.0000321502057613169000000	2
0.0000021522038567493100000	2
0.0000001284105851413540000	2
0.0000974018068035162000000	6
0.0000085733882030178300000	6
0.0000006755270483141170000	6
0.0001368539143598620000000	12
0.0000134526466836735000000	12
0.0000011865234375000000000	12
0.0001547996976568410000000	20
0.0000161577909270217000000	20
0.0000015154584345011000000	20
0.0001607510288065840000000	30
0.0000174244524807692000000	30
0.0000016988169420115300000	30

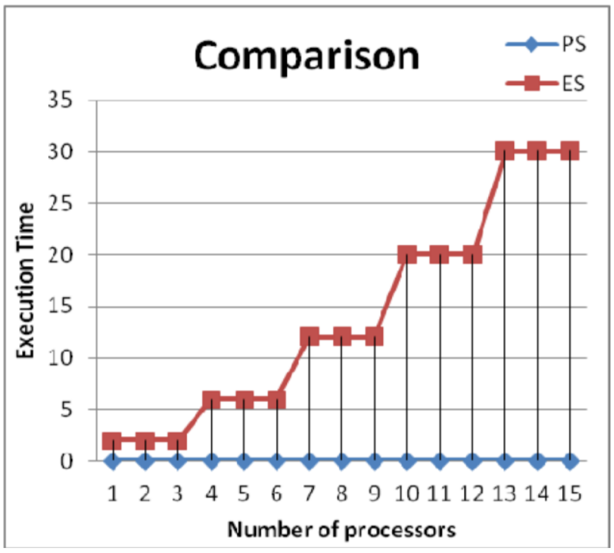


Fig. 2: Execution Summary

VI. CONCLUSION

In this paper, redundancy based scheduling algorithm is implemented with the concept of multiple server queuing model also. The comparison between existing and proposed work is represented. Linearity is the main factor accountable for performance consideration with respect to fluctuation rate.

On extending the proposed work the same models can be implanted for grid computing or cloud computing environment.

REFERENCES

[1] Chakrabarthi et al, "Hybrid Scheduling of Dynamic Task Graphs with Selective Duplication for Multiprocessors Under Memory and Time Constraints" , *IEEE Transactions on Parallel and distributed Systems, Vol. 19, no. 7, pp. 967980, 2008.*

[2] Lewis et al, "Scheduling Parallel Program Tasks onto Arbitrary Target Machines," *J. parallel and Distributing Computing, vol. 9, no.2 pp. 138-153, 1990.*

[3] Gupta et al, "Real Time Scheduling Using Compact Task Graphs," *proc. 16th Int'l Conf. Distributed Computing Systems (ICDCS' 96), p. 53, 1996.*

[4] Kumar et al, "Dealing with Heterogeneity Through Limited Duplication for Scheduling Precedence-Constrained Task Graphs," J. *Parallel and Distributed Computing, vol. 65, no. 6, pp. 479-491, 2005.*

[5] Ahmad et al, "Optimal and Near-Optimal Allocation of Precedence-Constrained Tasks to Parallel Processing, "*ICPP `98, pp. 424-431, 1998.*