Comparative Study of Task Duplication based Scheduling Algorithms for Parallel Systems

Nidhi Arora
Department of Computer Science and Engineering
M.M.U. Mullana (Ambala) – India

ABSTRACT
Scheduling of tasks is done by mapping tasks on multiple processors so that it requires least time for completion of all processes. Multiprocessors are used to run real time applications that uniprocessor systems would not be competent to execute. This paper presents various scheduling algorithms that schedule an edge-weighted Directed Acyclic Graph (DAG) to a number of processors. In this paper, task duplication based scheduling algorithms like PY algorithm and DSH algorithm are analyzed and studied for various performance matrices. Also, the effect of varying number of processors is examined on these algorithms.

Keywords
Task Scheduling, DAG, Parallel Processing, Multiprocessor Scheduling, Performance Evaluation, Scalability.

1. INTRODUCTION
Parallel computing which has been considered as high end of computing is used to execute a large number of tasks on different processors. The processors may be arranged in homogeneous environment as well as in heterogeneous environment. Parallelization of a multiprocessor scheduling algorithm is done to reduce the time for completion of all processes, to improve the execution speed and to increase the efficiency of the entire system. Task scheduling algorithms take care of all the above said factors and can be represented by Directed Acyclic Graphs (DAG). Parallel task scheduling algorithms are classified into four different groups:

Bounded Number of Processors (BNP) scheduling algorithms: These algorithms [3] schedule the DAG to a bounded number of processors directly. The processors are assumed to be fully-connected.

Unbounded Number of Clusters (UNC) scheduling algorithms: These algorithms [3] schedule the DAG to an unbounded number of clusters. The processors are assumed to be fully-connected. The technique employed by these algorithms is also called clustering**.**

Task Duplication Based (TDB) scheduling algorithms: These algorithms [3] also schedule the DAG to an unbounded number of clusters but employ task duplication technique to further reduce the completion time.

Arbitrary Processor Network (APN) scheduling algorithms: These algorithms [3] perform scheduling and mapping on the target architectures in which the processors are connected via a network of arbitrary topology.

In this paper TDB scheduling algorithms are discussed and their performance is analyzed.

The rest of the paper is organized as follows. In the next section task scheduling algorithms are discussed. DAG model is also represented. In section 3, performance parameters are discussed and results are shown graphically. In last section, overall results are concluded based upon different outputs for various cases.

2. TASK SCHEDULING ALGORITHMS

2.1 DAG Model
A scheduling system model consists of an application program which is represented by DAG $(G = V, E)$ where V is a set of nodes and E is a set of directed edges. In the model [7], a set of n nodes $\{n_1, n_2, ...n_n\}$ are connected by a set of e directed edges, which are represented by n_i, n_j, where n_i is called the parent node and n_j is called the child node. A node n_i represents a task and its weight $w(n_i)$ is called s computation cost. The weight of an edge is called the communication cost and is denoted as $c(n_i, n_j)$. There are various ways to determine the priorities of nodes. A DAG model having 5 nodes with their respective weights is shown in Figure 1.

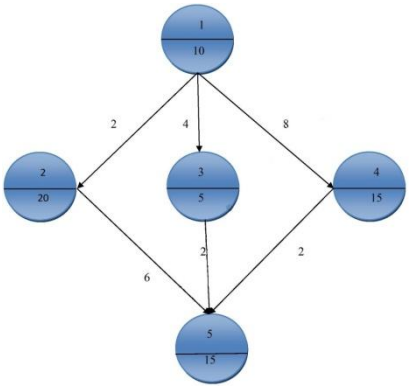

Figure 1: DAG model representing 5 nodes

The main scheduling attributes used in DAG for assigning priority are top level of the node n_i in DAG which is the length of the longest path from entry node to n_i not including n_i. Another attribute used to assign priority to a node is bottom level of a node n_i which is defined as the length of the longest path from n_i to the exit node. If the edges weights are not taken while considering the b-level, then it is called Static level. The length of the longest path from starting node to the exit node in DAG is the critical path.

2.2 Task Duplication Based Scheduling Algorithms

The TDB stands for Task Duplication Based scheduling algorithms. The principle behind these algorithms is to reduce the communication overhead by redundantly allocating some tasks to multiple processors (Ahmad and Wu, 1996). In duplication different strategies can be employed to select ancestor nodes for duplication. Some of the algorithms duplicate only the direct predecessors while some other algorithms duplicate all possible ancestors.

The PY Algorithm: The PY algorithm (named after Papaadimitriou and Yannakakis) is task duplication based scheduling algorithm that uses an attribute [7] called e-value, to approximate the absolute achievable lower bound of the start-time of a node. This attribute is computed recursively beginning from the entry nodes to the exit nodes. After computing e-values, the algorithm inserts each node into a cluster, in which a group of ancestors are to be duplicated such that the ancestors have data arrival times larger than the e value of node.

DSH Algorithm: Another TDB algorithm is Duplication Scheduling Heuristic Algorithm that considers each node in a descending order of their priorities [7]. The DSH algorithm first determines the start time of the node on the processor without duplication of any ancestor. Then, it considers the duplication in the idle time period from the finish time of the last scheduled node on the processor and the start time of the node currently under consideration. After that, it duplicates the node ancestors into duplication time slot until the slot is used.

3. PERFORMANCE RESULTS AND COMPARISONS

This section analyzes the performance of task duplication based parallel algorithms. There are various methods to measure the performance of these algorithms. In each case different number of nodes i.e. 10, 20 and 30 are assigned to 3 processors which are homogeneous in nature. Following parameters are implemented for given scheduling algorithms and results are shown graphically.

Overall execution time: It is the amount of time consumed in execution of an algorithm for a given input on the N-processor based parallel computer. In figure 2, overall execution time of parallel algorithms is derived for both PY and DSH algorithms. Also, it is compared with uniprocessor system in which it increases constantly. For a better algorithm, execution time must be small. Therefore DSH algorithm proves to be better as it has lowest value for execution time.

Speedup: It is defined as the ratio of the execution time on the single processor to that of the multiprocessor [9].

$$Speedup = \frac{sequential\ execution\ \ time}{parallel\ execution\ \ time}$$

The purpose of a parallel computer is to speed up the computer processing capability, so for a better algorithm Speedup value must be high. As shown in figure 3, Speedup for DSH algorithm is higher than PY algorithm and Serial algorithm implemented on uniprocessor system.

Scalability: A computer system is scalable [9], if it can increase its resources to accommodate performance and to decrease its resources to reduce cost. Scalability indicates how well the performance will improve on adding processors. Figure 4 represents the performance of the algorithm in terms of Speedup by varying number of processors. The Speedup increases as the number of processors gets increased. So, overall performance of the system improves on adding number of processors.

Efficiency: It is defined as the fraction of time that is usefully employed by the processors for a given task. It means how the resources of the parallel system are being utilized. Usually, efficiency is between 0 and 1. From the experimental values shown in figure 5, it is depicted that DSH algorithm is more efficient than PY algorithm.

Processor Utilization: It is another important factor to measure the performance of an algorithm. Maximum number of processors must be utilized in case of a high performance system. Figure 6 shows average processor utilization of PY and DSH algorithms with varying nodes. All processors are utilized equally in case of PY algorithm.

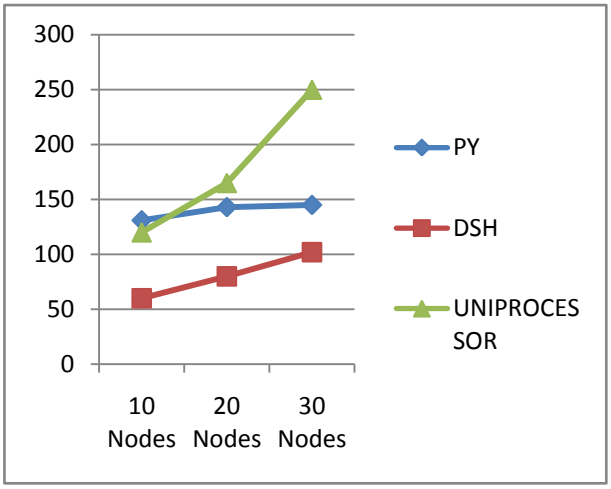

Figure 2: Overall Execution time

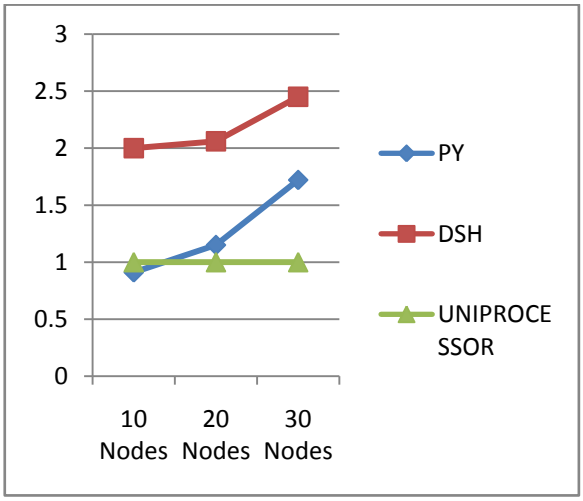

Figure 3: Speedup

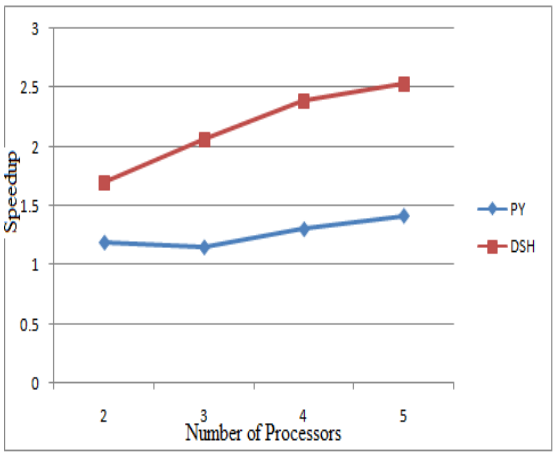

Figure 4: Speedup vs Number of processors

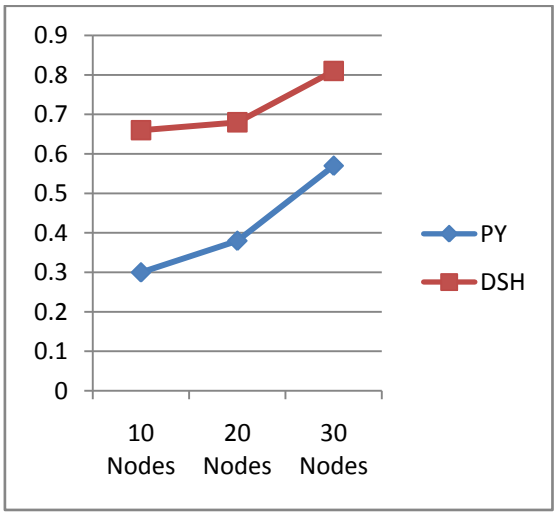

Figure 5: Efficiency

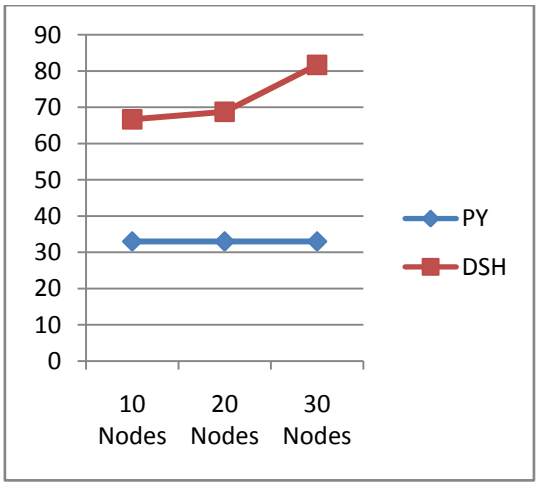

Figure 6: Processor Utilization

4. CONCLUSION

The study shows that parallel scheduling algorithms reduce the execution time of jobs. From the above analysis, it is concluded that on scheduling tasks in an effective way, the parallel system is able to process more number of jobs in lesser time as compared with uniprocessor system. Also DSH algorithm proves to be better as its overall execution time is less and Speedup is better than PY algorithm. The study also shows that the Speedup increases with the increase in number of processors on parallel systems.

5. REFERENCES

[1] Hwang, K. and Briggs, F. A. 1984 Computer architecture and Parallel Processing, McGrawHill.

[2] Ahmad, I. and Kwok, Yu-K. 1995 Performance Comparison of Algorithm for Static Scheduling of DAGs to multiprocessor, Proceedings of the Second Australian Conference on Parallel and Real-Time Systems, Perth, Australia, 185-192.

[3] Ahmad, I. and Wu, M. Y. 1996 Analysis, Evaluation and Comparison of algorithm for Scheduling Task Graph on Parallel Processor, IEEE Conference Publications, 1087-4087.

[4] Wu, M. Y. 1997 On parallelization of Static Scheduling Algorithm, IEEE, vol 23, pp. 517 – 528.

[5] Ahmad, I. and Kwok, Yu-K. 1998 Benchmarking and Comparison of the Task Graph Scheduling Algorithms, IEEE Conference Publications, pp. 1063-7133.

[6] Ahmad, I. and Kwok, Yu-K. 1999 On Parallelizing the Multiprocessor Scheduling Problem, IEEE Transactions on parallel and distributed systems, vol 10(4) 414-431.

[7] Kwok, Y. K and Ahmad, I. 1999 Static Scheduling Algorithms for Allocating Directed Task Graphs to Multiprocessors, ACM Computing Surveys, vol.31(4) 406-471.

[8] Hagras, T. and Janeek, J. 2003 Static Vs. Dynamic List-scheduling Performance Comparison Acta Polytechnica vol. 43(6)

[9] Chen et al. 2006 Study on Parallel Computing, Journal of Computer science and technology, vol 21(5)

[10] Shyiyuan Jin, S, Schiavone, S. and Turgut, D. 2008 A performance study of multiprocessor task scheduling algorithm, Journal of Supercomputing, vol 43(1) 77-97.

[11] Padmavathi, S. and Shalinie, S. M. 2010 Scalable low complexity task scheduling algorithm for cluster of workstations, Journal of Engineering Science and Technology, vol 5(3) 332 – 341.

[12] Arora, N. et al 2012, Performance Comparison of BNP Scheduling Algorithms in homogeneous environment, Global Journal of Computer Science and Technology, vol 12(8) 47-55.

[13] Arora, N. 2012, Analysis and performance comparison of algorithms for scheduling directed task graphs to parallel processors, International journal of emerging trends in Engineering and development, vol 4 (2) 793-802.

[14] Samriti et al. 2012 Analysis of HLFET and MCP Task Scheduling Algorithms, International Journal of Modern Engineering Research, vol 2(3) 1176-1180.