ISSN 1512-1232

A STUDY OF TASK SCHEDULING IN MULTIPROCESSOR ENVIROMENT

Ranjit Rajak[1], C.P.Katti[2], Nidhi Rajak[3]

[1]Department of Computer Science & Applications, Dr.H.S.Gour Central University, Sagar, India, ranjit.jnu@gmail.com

[2]School of Computer and System Sciences, Jawaharlal Nehru University, New Delhi India, cpkatti@mail.jnu.ac.in

[3]School of Computing Science & Engineering, Galgotias University, Gr.Noida, India, nidhi.bathre@gmail.com

Abstract

A multiprocessor system is a computer system with more than one processor. Task scheduling on a multiprocessor environment is an important area of research as it has a number of applications in scientific and commercial problems. The objective of task scheduling is to minimize the total completion time of a given application program that is represented by Directed Acyclic Graphs (DAGs). Task scheduling may be classified into static task scheduling and dynamic task scheduling. In this paper, we have studied various types of task scheduling and their properties. Also, we have studied performance metrics for task scheduling.

Keywords: *Task scheduling, DAG, Speedup, Parallel processing.*

I. INTRODUCTION

Parallel computing is an important research area in Computer Science. However, it consist number of problems that are not solved in sequential machine such as designing of parallel algorithms for an application program, dividing of an application program into subtasks, synchronization and coordinating communication, and scheduling of the tasks onto the processors.

Parallel computing is the next generation of computers and has made impact on various areas from scientific and engineering applications to commercial applications.

Scheduling [1] a set of dependent or independent tasks for parallel execution on a set of multiple processors is an important computational problem where a large problem or an application is divided into a set of subtasks and these subtasks are distributed among the processors. The allocation of subtasks on processors and defining their order of execution is called as *Task scheduling*.

Task scheduling is an NP-complete problem of its simple case [2] and some restricted cases [3,4,5,6,7]. It has been used in number of application from scientific to engineering problems.

Figure.1 [8] shows the layout of transformation from an application program to task scheduling. Here, an application program is divided into a number of subtasks that are represented by Directed Acyclic Graph. They are allocated to the multiple processors and should be maintained precedence constraints [9] among the subtasks.

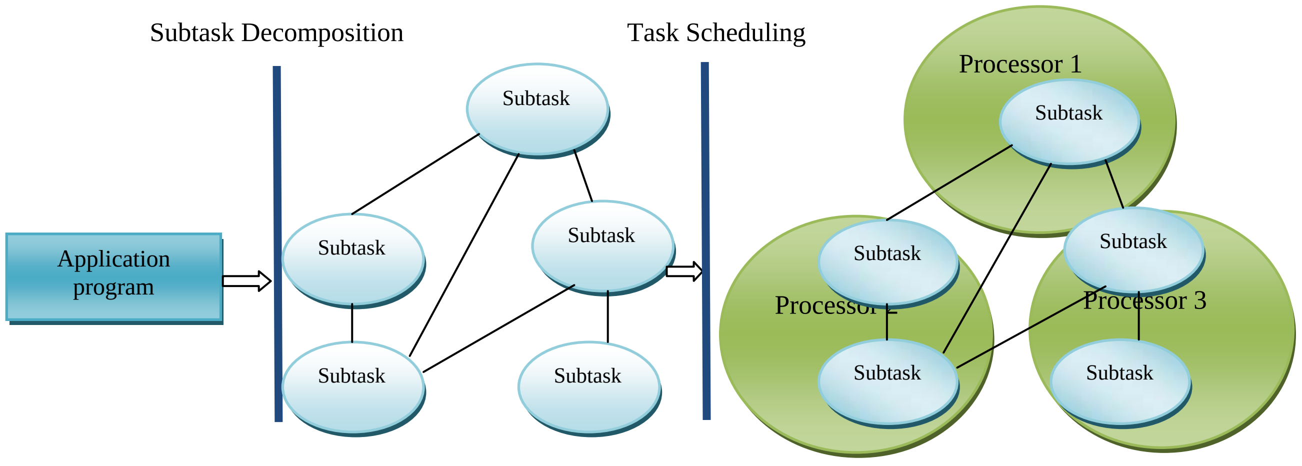

Fig.1. Transforming from an application program to task scheduling

II. TASK SCHEDULING

The major objective of the task scheduling in multiprocessor environment is to minimize the execution time of tasks. There are number of issues in task scheduling such as communication time has to be considered or not and multiprocessors environment is either heterogeneous or homogeneous. Here, heterogeneous means all the processors are different and homogeneous means all the processors are identical.

The generalized model of a task scheduler [10] consists of:

- Task Queue: It contains all the incoming tasks.
- Scheduler: It works simultaneously with the processors and also schedules arrived tasks towards dispatcher of the respective processor.
- Dispatch Queues: It is a mediator between scheduler and processor for the tasks. Each of the processor is associated with it.

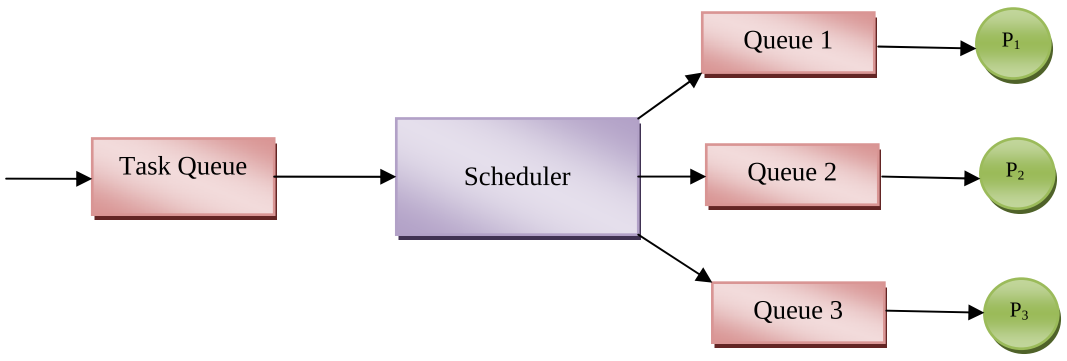

Fig. 2. Task Scheduler Model

There are mainly three components [11] for execution of task scheduling:

- Performance of processors.
- Tasks mapping onto processors.
- Execution order of the tasks on processors.

- Therefore, it deals [12] with the allocation of each task to the suitable processors and assignment of proper order of task.
- Task scheduling is classified on the basis of following characteristics [13]:
- Number of tasks and their dependencies.
- Execution time and communication time of the tasks.
- Number of processors and their uniformity.
- Topology of task graph.

It is classified into two categories: *static task scheduling* [14] and *dynamic task scheduling* [3].

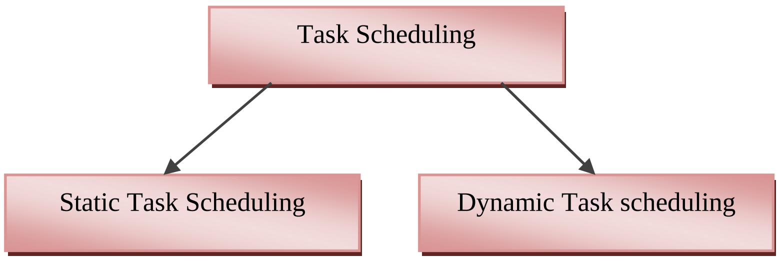

Fig.3. Classification of Task Scheduling.

A. Static Task Scheduling

Static task scheduling is the assignment of various tasks of task graph on the multiple processors during compile time. It is also called as deterministic and compiles time scheduling because the information of computational time of tasks, communication between the tasks and their precedence constraints are known in advance. There are two objective of static task scheduling[15]: minimize the completion time of the task graph and minimize the inter-task communication delay.

B. Dynamic Task Scheduling

Dynamic task scheduling is reassignment of various tasks of task graph on the multiple processors during the time of execution. It is also called as non-deterministic and dynamic load balancing scheduling because the information of computational time of tasks, communication between the tasks and their precedence constraints are not known in advance. The major objective of dynamic task scheduling is to maximize the utilization of the processor in the system at all the times.

Following policies [15] are used for dynamic task scheduling:

- Load estimation policy: It determines how to estimate the load of each processor.
- Task transfer policy: It determines whether or not a task is to be transferred to another processor.
- State information exchange policy: It determines how to exchange the system load information among the processors.
- Location policy: It determines the processor to which a task should be transferred.

III. FORMULATION OF TASK SCHEDULING PROBLEM

A task scheduling problem consists of the application model, system computing model and performance evaluation metrics. This section will discuss an application model, a system computing model followed by performance evaluation metrics.

A. Application model

Task scheduling of a given application is represented by directed acyclic graph (DAG) $G_1 = (T, E)$, where T is the finite set of m tasks $\{T_1, T_2, T_3 \ldots T_m\}$ and E is the set of edges $\{e_{ij}\}$between the tasks T_i and T_j. Here, each edge represents the precedence constraints between the tasks T_i and T_j such that T_j can not start until T_i completes its execution. Each task T_i is associated with an execution time $ET(T_i)$ and each edge e_{ij} is associated with a communication time $CT(T_i, T_j)$ for data transfer from T_i to T_j. If there is a direct path from T_i to T_j then T_i is the predecessor of T_j and T_j is the successor of T_i. A entry task does not any predecessor, similarly, an exit task does not any successors. Layout of a DAG with six tasks is shown in figure 4 [16],

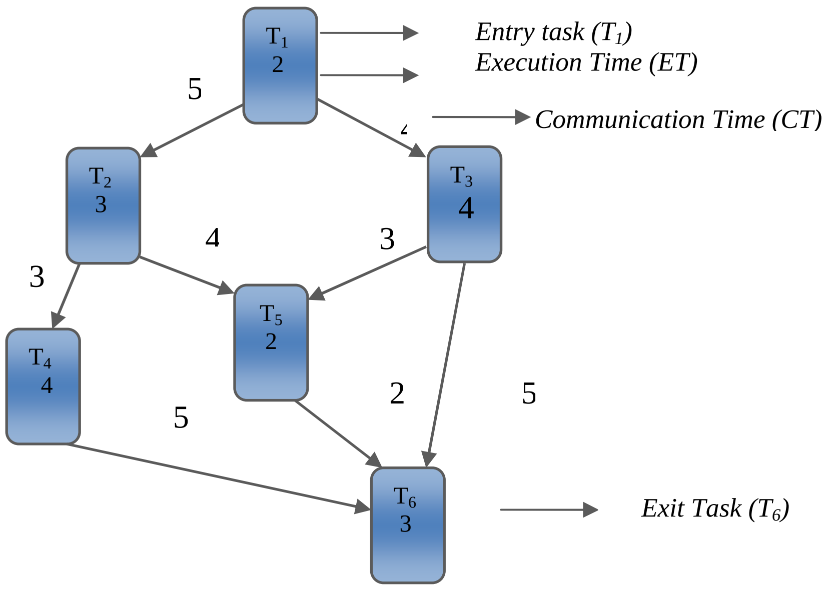

Fig.4. DAG Model with six tasks

where T_1, T_2, T_3, T_4, T_5 and T_6 are different tasks of the given DAG. The execution time ET (T_i) and communication time CT (T_i, T_j) [16] of tasks are:

ET (T_1) = 2	ET (T_2) = 3	ET (T_3) = 4	
ET (T_4) = 4	ET (T_5) = 2	ET (T_6) = 3	
CT (T_1, T_2) = 5	CT (T_1, T_3) = 4	CT (T_2, T_4) = 3	CT (T_2, T_5) = 4
CT (T_3, T_5) = 3	CT (T_3, T_6) = 5	CT (T_4, T_6) = 5	CT (T5, T6) = 2

B. System Computing Model

A multiprocessor system is either heterogeneous or homogeneous. Considered homogeneous processors for the purpose of methodology that can be represented by a directed graph $G2 = (Proc, Link)$, where *Proc* is a set of p processors $\{P_1, P_2, P_3 \ldots Pp\}$ connected to each other through a network. *Link* is a set of links between the processors P_i and P_j . If both of the tasks T_i and T_j are scheduling on the same processor, then communication time of the direct path between T_i and T_j is negligible. Figure 5 shows [17] three fully connected homogenous processors via common link.

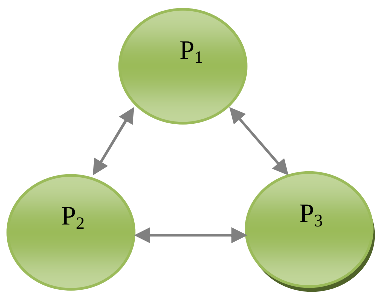

Fig.5. Fully connected homogeneous processors

The objective of multiprocessors task scheduling is to minimize the overall execution time of the tasks as well as to preserve the precedence constraints. If a task T_i is scheduled on a processor P, then starting time of task and finishing time of task is denoted by STT (T_i, P) and FTT (T_i, P), respectively. The scheduling length is defined by $Slen$ = Max { FTT (T_i, P)}for all the processors.

IV. TASK SCHEDULING EXAMPLES

This section is considered two examples for task scheduling: a task graph with communication time and task graph without communication time with the assumption that number of processors is finite and homogeneous. The schedule of tasks on the processor is shown with help of a Gantt chart. A Gantt chart is a common graphical representation of task schedule. It consists of start time and finish time of each task on the available processor. It also represents the idle time between the tasks.

A. Task Scheduling with Communication Time

Task scheduling with communication time means consideration of communication time during assignment of tasks on the processors. The communication time is required due to inter-dependency of tasks. A dependent task will not start execution until it gets all the information from previous tasks. If two tasks are scheduled on same processor then their communication time would be negligible. Consider an example of DAG that consists of four tasks and also consider two processors for scheduling as shown in Figure 6 [18].

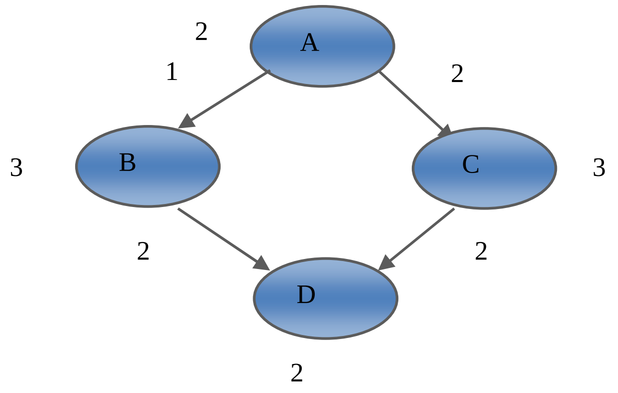

Fig.6 DAG with four tasks

0	1	2	3	4	5	6	7	8	9	10
P1	A	A	B	B	B	▨	▨	▨	▨	▨
P2	▨	▨	▨	▨	C	C	C	D	D	▨

Fig.7. Scheduling length is nine time units.

B. Task Scheduling without Communication Time

Task scheduling without communication time means do not consider communication time during the scheduling of tasks on the processors. It reduces the scheduling length. Consider the same example as shown in Figure 8 without communication time with the same number of processors.

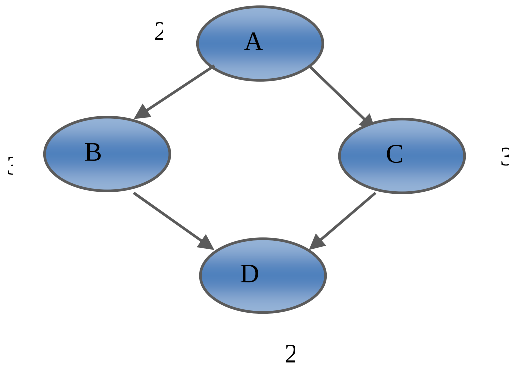

Fig.8. DAG with four tasks.

0	1	2	3	4	5	6	7	8	9	10
P1	A	A	B	B	B	▨	▨	▨	▨	▨
P2	▨	▨	C	C	C	D	D	▨	▨	▨

Scheduling length of this system is decreased by two units as compared to previous example.

V. PERFORMANCE METRICS OF TASK SCHEDULING

Following metrics are used for performance evaluation of task scheduling algorithms:

A. Scheduling Length

Scheduling length is the maximum time required to execute the last task on a processor as shown in (1).

$$SLen = Max\{FTT(Ti, P). \qquad (1)$$

B. Total Parallel Overhead

Total parallel overhead [19] is the total time spent by all PEs over and above that required by the fastest known sequential algorithms for solving the same problem on a single PE.

It is denoted ToT and can be expressed as shown in (2).

$$To = p.Tp - Ts \ , \qquad (2)$$

Where T_s is sequential time, T_p is the parallel execution time and p is total processors for T_p.

C. Cost

Cost [20] is defined as the product of parallel execution time (T_p) and the number of processing elements (P) used as shown in (3).

$$Cost = Tp.P \ . \qquad (3)$$

D. Speedup

Speedup [21] is as the ratio of sequential execution time (Ts) and parallel execution time (Tp). Sequential execution time is the total execution time of each task in uni-processor environment and a parallel execution time is the last execution time of a task on bounded number of processors in multiprocessors environment. It can be expressed as shown in (4).

$$Speedup = \frac{Ts}{Tp} \ . \qquad (4)$$

E. Efficiency

Efficiency [21] of a parallel program is the ratio of speedup and the number of processors used as shown in (5).

$$Efficiency = \frac{Speedup}{Number of Processors} \ . \qquad (5)$$

F. Normalized Scheduling length (NSL)

Normalized Scheduling Length (NSL) [22] of a scheduling algorithm is written as shown in (6).

$$NSL = \frac{Scheduling\ length\ of\ a\ particular\ algorithm}{Max\{sum\ of\ execution\ time\ along\ a\ path\}} \ . \qquad (6)$$

G. Load Balancing

Load balancing [23] is the ratio of scheduling length to the average execution time over all the processors as shown in (7).

$$Load\ Balancing = \frac{SchedulingLength}{Average} \ , \qquad (7)$$

where Average is the ratio of sum of processing time of each processor to the number of processors are used.

CONCLUSION

In this paper, we have studied task scheduling, components and types of task scheduling. Here, an application program is represented by DAG. A DAG model consists of tasks and edges represent the communication time between the tasks.

The edges also represent the data dependency between two tasks. Each task and edge is associated with execution time of each task and communication time between the two tasks. The multiprocessor model may be either homogeneous or heterogeneous. We have evaluated task scheduling with considered communication time and without considered communication time. The Gantt. Charts for both cases showed that the task scheduling without communication time gives minimum scheduling length as compared to task scheduling with communication time. We can also evaluated different task scheduling algorithms on the basis of performance metrics.

REFERENCES

1. Shiyuan Jin, Guy Schiavona and DamlaTurgut, “A performance study of multiprocessor task scheduling algorithms”, Journal of Supercomputing ,Vol.43,pp.77-97,2008.
2. M.R.Garey and D.S.Johson,Computers and Intractability: A Guide to the Theory of NP - completeness,1979.
3. G.N Srinivas and Bruce R. Musicus. “Generalized Multiprocessor Scheduling for Directed Acyclic Graphs” In Third Annual ACM Symposium on Parallel Algorithms and Architectures, pp.237-246, 1994.
4. G.L.Park, BehroozShirazi, Jeff Marquis and H.Choo. “Decisive Path Scheduling: A new List Scheduling Method” Communication of the ACM, vol.17, no.12, pp. 472-480,Dec1974.
5. Min You Wu “On Parallelization of static scheduling Algorithms”, IEEE Transactions on Parallel and Distributed Systems, Vol.10, No.4, , pp 517-528,April 1999
6. C.H.Papadimitrious and M.Yannakakis, “Scheduling Interval-Ordered Tasks, “ SIAM Journal of Computing,Vol.8,pp.405-409,1979.
7. R.Sethi, “Scheduling Graphs on Two Processors,” SIAM Journal of Computing,Vol.5, No.1,pp.73-82, Mar.1976.
8. Oliver Sinnen,”Task Scheduling for Parallel Systems” Wiley-Interscience Pulication, 2007.
9. Edwin S.H and Nirwan Ansari, “A Genetic Algorithm for Multiprocessor Scheduling", IEEE Transaction on Parallel and Distributed Systems, Vol.5, No.2, Feb.1994.
10. Hadis Heidari and Abdolah Chaechale , ” Scheduling in Multiprocessor System Using Genetic Algorithm,” International Journal of Advanced Science and Technology,Vol.43, pp.81-93,2012.
11. RavneetKaur and RamneekKaur,”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,pp.311-317,April,2013.
12. XiaoyongTang,Kenli Li and Guiping Liao ,” List Scheduling with duplication for heterogeneous computing systems”, Journal of Parallel and Distribute Computing,Vol.70, pp.323-329,2010.
13. MostafaR.Mohamed and MedhatH.A,”Hybrid Algorithms for Multiprocessor Task Scheduling”, International Journal of Computer Science Issues, Vol.8, Issue.3 No.2 May,2011.
14. Y.C.Chung and S.Ranka, “Application and Performance Analysis of a Compile-Time Optimization Approach for List Scheduling Algorithms on Distributed–Memory Multiprocessors”, Proceedings of Supercomputing’92, , pp.512-521,1992.
15. V.Rajaraman and C.S.R Murthy, “ Parallel Computers : Architecture and Programming, “ PHI Publication , May.2012.

ISSN 1512-1232

16. RanjitRajak, “Comparison of Bounded Number of Processors (BNP) Class of Scheduling Algorithms Based on Metrics”, GESJ: Computer Science and Telecommunication, Vol.34, No.2, 2012.
17. RanjitRajak and C.P.Katti,”Task Scheduling in Multiprocessor System using Fork-Join Method(TSFJ)”, *International Journal of New Computer Architectures and Their Applications*, Vol.3 No.3, pp.47-53, 2013.
18. AhmedZakiSemarShaul and Oliver Sinnen,”Optimal Scheduling of Task Graphs on Parallel System”, 9th International Conference on Parallel and Distributed Computing ,Applications and Technologies,pp.323-328,2008.
19. Vipin Kumar and Anshul Gupta,” Analysis of scalability of parallel algorithms and architectures: a survey” ICS '91 Proceedings of the 5th international conference on Supercomputing, pp. 396-405, 1991.
20. Ananth Gramma, Vipin Kumar and Anshul Gupta,” Introduction to Parallel Computing, Pearson Edition,2009.
21. M.J.Quinn,Parallel Programming in C with MPI and OpenMP,Tata McGraw-Hill, Edition 2003.
22. K.SikS,Myong, J.Cha, M.S.Jang,”Task Scheduling Algorithm using Minimized Duplication in Homogeneous Systems,” Journal of Parallel and Distributed Computing ,Vol.68, pp. 1146-1156 , 2008.
23. F.A.Omara and M.Arafa,”Genetic Algorithm for Task scheduling Problem”, Journal of Parallel and Distributed Computing .Vol.70, pp.13-22, 2010.

Article received: 2014-01-26