ISSN 1512-1232

COMPARISON OF BOUNDED NUMBER OF PROCESSORS (BNP) CLASS OF SCHEDULING ALGORITHMS BASED ON MATRICES

Ranjit Rajak

Research Scholar, School of Computer and System Sciences, Jawaharlal Nehru University

New Delhi-110067, India, ranjit.jnu@gmail.com.

Tel: +91-7428231979

Abstract

The Task Scheduling in multiprocessor system is also known as multiprocessor scheduling. It is used in large number of applications from scientific and engineering to commercial problems. The major objective of task scheduling is to minimize the program's execution time. The task scheduling is represented by a directed acyclic graph (DAG). There are basically two types of task scheduling algorithms: Deterministic or Static and Non deterministic or dynamic scheduling algorithm. In this paper, we have focused only static scheduling algorithms and it is further classified into heuristic based and guided random search based. In this paper, we have taken, Bounded Number of Processors (BNP) class scheduling algorithms. It is a classification of heuristic based. The BNP scheduling algorithm consists of four scheduling algorithms: Highest Level First with Estimate Time (HLFET) algorithm, Modified Critical Path (MCP) algorithm, Earliest Time First (ETF) algorithm and Dynamic Level Scheduling (DLS) algorithm. We have studied of Bounded Number of Processors (BNP) class scheduling algorithms and find the scheduling length of each algorithm. The scheduling length is used in four matrices: Speedup, Efficiency, Load Balance and Normalized Scheduling Length (NSL). Finally, we have compared these four scheduling algorithm based on four matrices.

Keywords: *Task scheduling, DAG, NP-complete, parallel processing.*

1. Introduction

The task scheduling in multiprocessor system is also known as multiprocessor scheduling, it is used in a large number of computer applications. In parallel processing, it needs to minimized execution time with respect to a minimum number of processors. The major objective of task scheduling in multiprocessor system is to minimized the scheduling length. The scheduling length is also called makespan.

The task scheduling in multiprocessor system has been proven to be NP-Complete except for few restricted cases [1].The scheduling problem is NP –complete into simple cases [2]: scheduling tasks with uniform weights to an arbitrary number of processor and scheduling tasks with weights equal to one or two units to two processors.

In this paper, first we will study the classification of multiprocessors scheduling algorithms and their brief introduction and then BNP class of scheduling algorithms. The BNP class of scheduling algorithms consists of four algorithms: HLEFT, MCP, ETF and DLS scheduling algorithms. We will study each algorithm one by one and find their scheduling length by drawing their Gantt. Chart using exiting algorithms. These scheduling length is used in four matrices: Speedup, Efficiency, Load balancing and Normalized Scheduling Length (NSL). After calculating these matrices, we will compare all four algorithms based on matrices. Finally, we will come to conclusion.

2. Multiprocessors scheduling classification

The multiprocessor schedule can be divided into two main categories: deterministic or static and non deterministic or dynamic scheduling [3]. In static scheduling problem, all the information

ISSN 1512-1232
about the tasks and their precedence constraint, computation time, communication time are known before program execution. In case of dynamic scheduling, all these information is not known in advance or not known till program execution time. In this paper, we are considering only deterministic scheduling or static scheduling problem [4]. The static scheduling has some constraints: the number of task, execution time of task, inter task communication and number of processors [5]. The multiprocessor system consists of a set of m identical processors

$P=\{p_i\}$ where i=1,2,3…m.

which are fully connected with each other through identical links.

In static scheduling problem, the parallel program can be represented by Directed Acyclic Graph (DAG) G. The DAG model consists of four tuples G(V,E,W,C). Where
V: finite set of tasks that is $V=\{T_i\}$, i=1,2,3... Up to n tasks,
E: Set of directed edges from one tasks to another that is $E=\{e_{ij}\}$ edge between task T_i and T_j. It also represents the precedence constraint among the tasks
W: Computational time of each task T_i. It is denoted by $W(T_i)$.
C: Communication time betweens the tasks $C(T_i,T_j)$ which represents weight of edges between T_i and T_j.

The DAG models is also called task graph. The precedence constraint should be always holding in DAG model. That is a task can't be start until get the all information from its parent tasks. When two tasks T_i and T_j are executed in same processor then their communication must be zero.
Considering an example of a DAG model with six tasks.

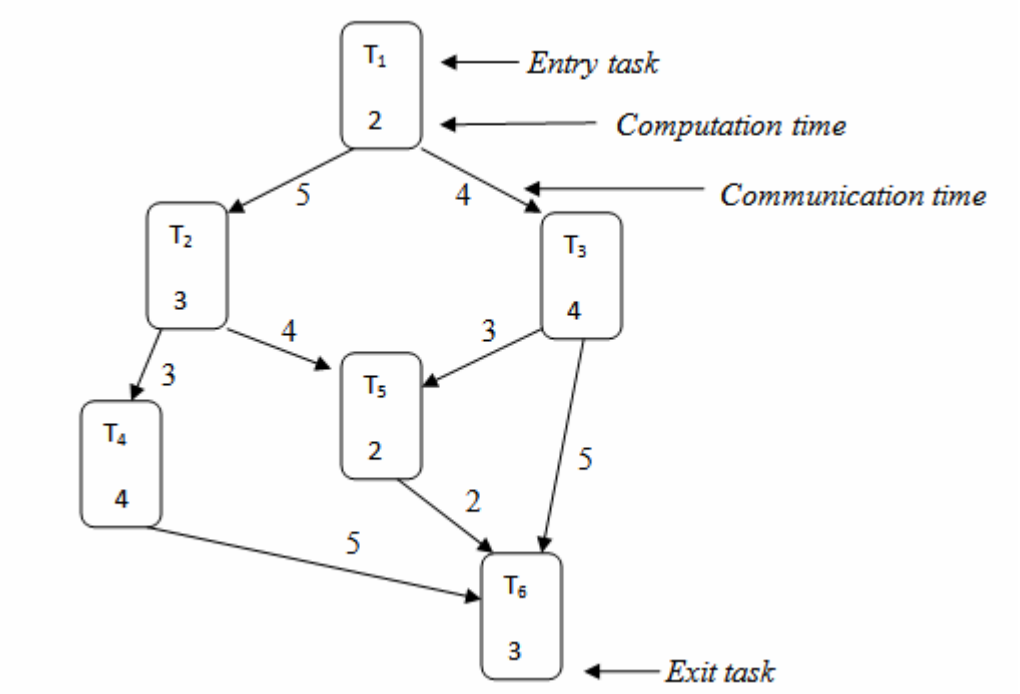

Figure 1. DAG Model

Here T_1, T_2,T_3,T_4,T_5 and T_6 are set of tasks of a given DAG. The computational time $W(T_i)$ and communication time $C(T_i,T_j)$ of tasks are following :

$W(T_1)=2$ $W(T_2)=3$ $W(T_3)=4$
$W(T_4)=3$ $W(T_5)=2$ $W(T_6)=3$
$C(T_1,T_2)=5$ $C(T_1,T_3)=4$ $C(T_2,T_4)=3$ $C(T_2,T_5)=4$
$C(T_3,T_5)=3$ $C(T_3,T_6)=5$ $C(T_4,T_6)=5$ $C(T_5,T_6)=2$

Tasks T_1 is an entry task and T_6 is an exit task of DAG.

Since, the multiprocessor scheduling can be classified into two main categories: Deterministic or Static scheduling and Non deterministic or dynamic scheduling. The Static scheduling is further classified into two categories: Heuristic based [6] and guided random search based algorithm [6]. In

ISSN 1512-1232
heuristic based algorithm produced solution depend on the assumption has been used like tasks of application and target computing system. Also, it can be classified into three categories: *List Scheduling* [6] is scheduling algorithm in which assigned the priority to every tasks of DAG and sort the list in decreasing order of priorities of the tasks. *Clustering [6]* is efficient methods that reduce the communication time between the tasks in the DAG. *Task duplication based algorithms* [7] is used where needs to reduce the communication time and also reduces the start time of waiting tasks. In case of guided random search based algorithm, it is search problem which consists of an exponential number of possible schedules. It is mainly used to solve very complex problem. Genetic algorithms [8,9] are the most popular random search techniques. The list scheduling algorithm is classified into Bounded Number of Processors (BNP) class of scheduling algorithms and Arbitrary Processor Network(APN) class of scheduling algorithms

In the figure 2, shows the classification of multiprocessor scheduling algorithms. The BNP class of scheduling algorithm is fall into list scheduling algorithm. The BNP class of scheduling is applicable for homogeneous processors and fully connected. The APN class of scheduling [3] is also fall into list scheduling algorithms but not considering fully connected and message routing must be considered.

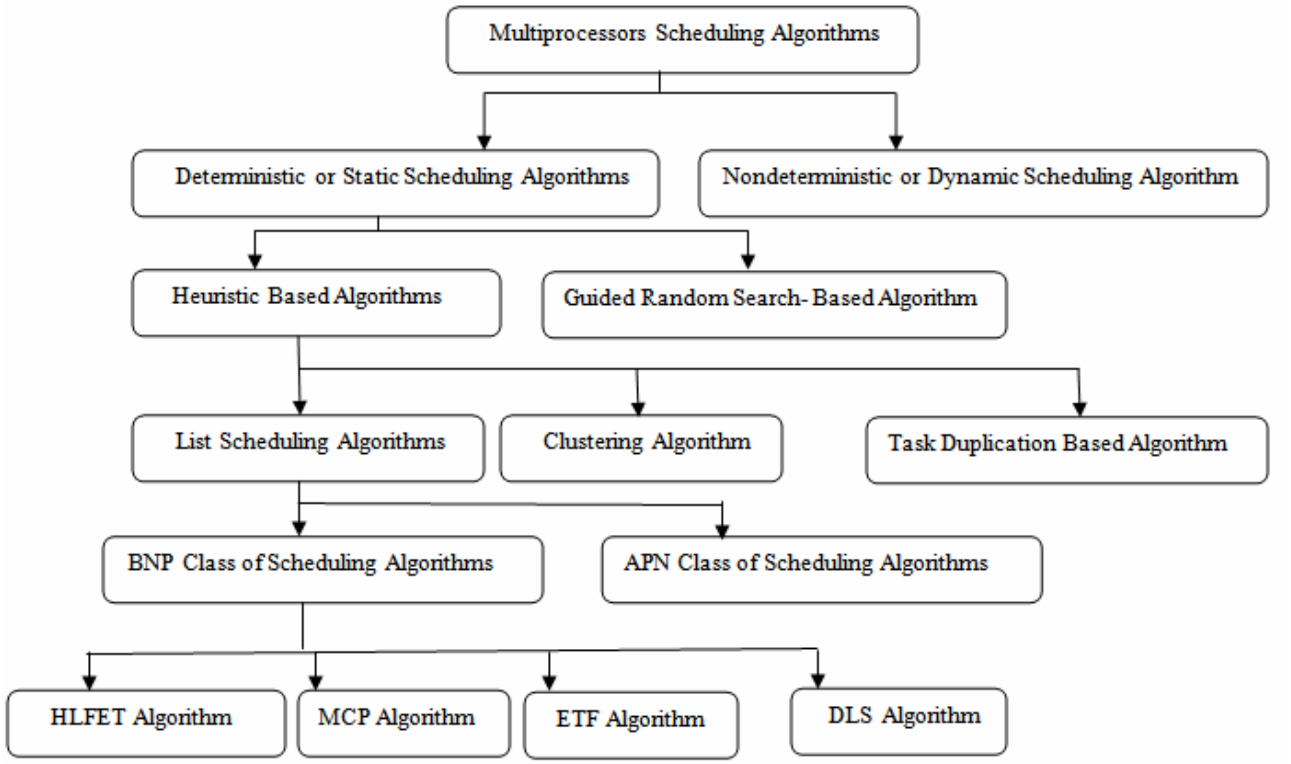

Figure 2. Multiprocessor Scheduling Algorithms

The next sub sequent section, we will describe each class of BNP scheduling algorithms. Each BNP class of scheduling algorithms is based on four priorities attribute. These are used to decide the priority of a task of DAG and it is depends on the algorithm that to decide which priority will use. The t-level (top level)and b-level (bottom level) [10] are two frequently used in assigning the priority to a task. The *t-level* of a task T_i is defined to be the length of longest path in DAG from entry task to T_i but there should be excluded the computation time of T_i .Similarly, we can define *b-level* is the length of longest path from T_i to exit task but it would be included the computation time of T_i .If computing b-level of task and not include the edge weights, then it is called *static b-level* or simply *static level(sl*). It is the third attribute. The last attribute is *ALAP* (As –Late-As-Possible) start time [11,12]. The *ALAP= CP - b-level (T_i),* where CP is Critical Path of graph the and b-level(T_i) is bottom level of each task. Now we will take an example of a DAG to illustrate the all four attributes of every task T_i.

ISSN 1512-1232

Consider DAG model with eleven tasks T_1, T_2, T_3…T_{11} and their computation and communication time are given in the figure 3. Now, we can calculate all four attributes of every task. After calculating the priority, it will use in BNP class of scheduling algorithms either increasing or decreasing order of priority of task or as per algorithms.

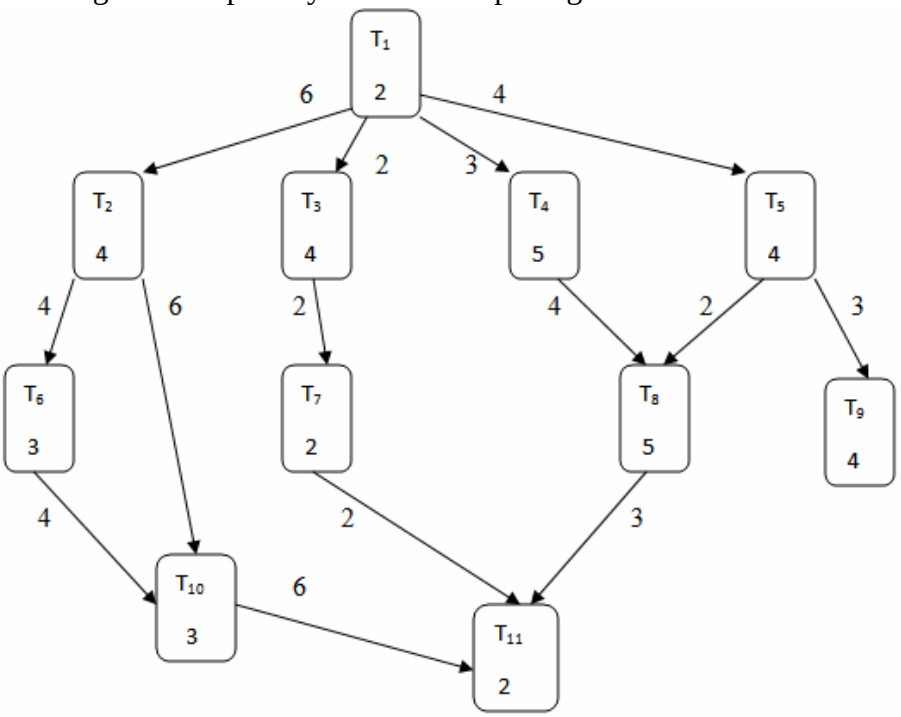

Figure 3. A Task Graph with 11 tasks

Table 1. Priority Attributes of Every Task (T_i)

Task(T_i)	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}
t-level	0	8	4	5	6	16	10	14	13	23	32
b-level	34	26	12	19	16	18	6	10	4	11	2
ALAP	0	8	22	15	18	16	28	24	30	23	32
Static Length	14	12	8	12	11	8	4	7	4	5	2

3. BNP class of Scheduling Algorithms

The BNP classes of scheduling algorithms are developed for homogeneous and limited number of processors. It also considered that all processors are fully connected. Now we have to analysis all four scheduling algorithms as per their priority attribute and scheduling length and comparison matrices. The time complexity of the algorithms are given in terms of p, e and v where p is processor, e is the edges and v is number of tasks.

3.1.The Highest Level First with Estimate Times (HLFET)algorithm:

This is the simplest algorithm among the BNP class of scheduling algorithms. It was developed for full connected identical processors. In this algorithm, priority of the task is decided by using *static level* (sl) attribute. The major problem with this algorithm, it does not use communication cost during the entire process. i.e. ignore the communication cost. The HLFET algorithm [13] is given below.

ISSN 1512-1232

Step1.Calculate the static level (sl) of each task.
Step2.Make a ready list in a descending order of static level(sl).Initially, the ready list contains only the entry task. Ties are broken randomly.
Repeat
Step3.Schedule the first task in the ready list to a processors that allows the earliest execution, using the non-insertion approach.
Step4.Update the ready list by inserting the tasks that are now ready .
Until all tasks are scheduled

The time complexity of HLFET is O (v^2).

3.2. The Modified Critical Path (MCP) Algorithm

The MCP uses ALAP attribute to calculate the priority of each task. It can be computed by using formula :*ALAP= CP - b-level (T_i),* where CP is longest path from entry to exits task of the DAG and b-level(Ti) is bottom level of each task {T_i: i=1,2…n}. It includes the communication time in the scheduling process. The major problem with the MCP algorithm that it includes the communication time for computing the priority of tasks but does not assign task priorities accurately. The MCP algorithm [12] is given below:

Step1.Compute the ALAP time of each task.
Step2.For each task, create a list which consists of the ALAP times of the tasks itself and all its children in a descending order.
Step3.Sort these lists in an ascending lexicographical order. Create a task list according to this order.
Repeat
Step4.Schedule the first task in the tasks list to a processor that allows the earliest execution, using the insertion approach.
Step5. Remove the task from the task list .
Until the task list is empty.

The time complexity of MCP algorithm is O(v^2logv).

3.3. The Earliest Time First (ETF) Algorithm

The ETF algorithm uses static level(sl) attribute to compute the priority of each task of graph. It is not need to be schedule task with highest priority due to each step of scheduling. The ETF algorithm is compute the earliest start time of each task of graph and select the smallest among them for schedule. If two or more task having same earliest start time , then ETE algorithm select the task as per static level (sl) with the highest priority. The major problem ETF algorithm that it can not be reduce partial scheduling length at each step of scheduling process. The ETF algorithm[14] is explained below :

Step1.Compute the static level of each task.
Step2.Initially , the pool of the ready tasks includes only the entry task.
Repeat
Step3.Calculate the earliest start time on each processor for each task in the ready pool. Pick the task-processor pair that gives the earliest time using the non-insertion approach . Ties are broken by selecting the task with a higher static level (sl). Schedule the node to the corresponding processor.
Step4 Add the newly ready task to the ready task pool.
Until all tasks are scheduled.

The time complexity of ETF algorithm is O (pv^2).

ISSN 1512-1232

3.4. The Dynamic Level Scheduling (DLS) Algorithm

The DLS algorithm determines the priority of task by using dynamic attribute is called Dynamic Level (DL). It is computed by static level and earliest time first. Formally, the DL is defined as the difference between static level and earliest start time of every task. After computed DL of each task, then select the task with the highest DL value. The problem with DLS algorithm, it does not maintain a scheduling list during scheduling process. The DLS algorithm [15] is explained below:

Step1. Calculate the static level of each task.
Step2. Initially , the ready node pool includes only the entry tasks.
Repeat
Step3. Calculate the earliest start time for every ready task on each processor. Hence, compute the DL of every node-processor pair by subtracting the earliest start time from static level of task.
Step4. Select the node –processor pair that gives the largest DL Schedule the task to the corresponding processor.
Step5. Add the newly ready tasks to the ready pool
Until all task are scheduled
The time complexity of DLS algorithm is O (pv^3)

4. An Illustrated Example

Consider task graph of figure 3. Now we will compute the scheduling length of each algorithm. Also make Gantt. Chart for each algorithm. Here, also considering that we have a multiprocessor system consisting four processors. They are homogeneous and fully connected. We have traced the Gantt Chart as per algorithms [12,13,14,15].

Time	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P1	T_1	T_1	T_2	T_2	T_2	T_2	T_6	T_6	T_6	T_{10}	T_{10}	T_{10}									T_{11}	T_{11}	
P2				T_4	T_4	T_4	T_4				T_8	T_8	T_8	T_8	T_8								
P3					T_5	T_5	T_5	T_5	T_9	T_9	T_9												
P4			T_3	T_3	T_3	T_3	T_7	T_7															

Figure 4. Gantt. Chart of HLFET Algorithm with scheduling length is 22

Time	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
P1	T_1	T_1	T_2	T_2	T_2	T_2	T_6	T_6	T_6	T_6	T_{10}	T_{10}	T_{10}		T_8	T_8	T_8	T_8	T_8				T_{11}	T_{11}	
P2						T_4	T_4	T_4	T_4	T_4															
P3							T_5	T_5	T_5	T_5	T_9	T_9	T_9	T_9											
P4					T_3	T_3	T_3	T_7	T_7																

Figure 5.Gantt. Chart of MCP Algorithm with scheduling length is 24

Time	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
P1	T_1	T_1	T_3	T_3	T_3	T_3	T_5	T_5	T_5	T_5	T_7	T_7	T_9	T_9	T_9	T_9										
P2						T_4	T_4	T_4	T_4	T_4					T_8	T_8	T_8	T_8	T_8							
P3									T_2	T_2	T_2	T_2	T_6	T_6	T_6						T_{10}	T_{10}	T_{10}	T_{11}	T_{11}	
P4																										

Figure 6. Gantt. Chart of ETF Algorithm with scheduling length is 25

Time	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
P1	T_1	T_1	T_4	T_4	T_4	T_4	T_4	T_2	T_2	T_2	T_2				T_9	T_9	T_9	T_9							
P2							T_5	T_5	T_5	T_5		T_8	T_8	T_8	T_8	T_8	T_6	T_6	T_6	T_{10}	T_{10}	T_{10}	T_{11}	T_{11}	
P3					T_3	T_3	T_3	T_3	T_7	T_7															
P4																									

Figure 7. Gantt. Chart of DLS Algorithm with scheduling length is 24

ISSN 1512-1232

5. Comparison Matrices

In this section, we will use four types of matrices to compare the BNP class of scheduling algorithms which are based on scheduling length. They are speedup, efficiency, normalized scheduling length (NSL) and load balancing.

5.1 Speedup

Speedup is the ratio between sequential execution time and parallel execution time [16] where the sequential time execution time is sum of total computation time of each task and parallel time execution is the scheduling length on limited number of processors. Speedup= Sequential time execution/ Parallel time execution

5.2. Efficiency

The efficiency of a parallel program is a measure of processor utilization [16]. Efficiency=Speedup/ number of processors.

5.3. The Normalized scheduling length (NSL)

The Normalized scheduling length (NSL) [17] of a scheduling algorithm is given by NSL=Scheduling length of particular algorithm / Max { sum of computation costs along a path}.

5.4. Load Balancing

The load balancing [18] is calculated by the ratio of scheduling length to average execution time over all processors. Load Balance= Scheduling length /Average. Where Average = sum of processing time each processor / number of processors.

6. Comparison of BNP class of Scheduling

The BNP class of scheduling algorithms will be compared based on comparison matrices. *HLFTE Scheduling:* The HLFET algorithm has the scheduling length is 22, So that we can calculate the matrices based on scheduling length.

Speedup= Sequential Execution Time/ Parallel Execution Time= 38/22=1.727

Efficiency= (Speedup/Number of processors)*100= (1.727/4)x 100=43.18%

Load Balance=Scheduling length/Average=22/14=1.571

NSL=Scheduling length/ Max {sum of computation cost along path} =22/34=0.647

Similarly we can also calculate the matrices of other three BNP class of scheduling algorithm. *The MCP algorithm:* The MCP algorithm has scheduling length is 24 so that

Speedup=1.583, Efficiency=39.58%, Load Balance=1.655, NSL=0.706

The ETF algorithm: The ETF algorithm has scheduling length is 25 so that

Speedup=1.520, Efficiency=38 %, Load Balance=1.666, NSL=0.735

The DLS Algorithm: The DLS algorithm has scheduling length is 24 so that

Speedup=1.583, Efficiency=39.58 %, Load Balance =1.846, NSL=0.706

In the table 2, we have given the comparison based on the matrices. The HLFET algorithm gives minimum scheduling length, maximum speedup, maximum efficiency, minimum load balance and NSL. The ETF algorithm gives maximum scheduling length, minimum speedup, minimum efficiency and maximum NSL. Both MCP and DLS algorithms have same scheduling length, speedup, efficiency and NSL but differ in load balancing.

Table 2: Comparison based on the comparison matrices

S.No	Algorithms	Speedup	Efficiency	Load Balance	NSL
1	HLFET	1.727	43.18%	1.571	0.647
2	MCP	1.583	39.58%	1.655	0.706
3	ETE	1.520	38%	1.666	0.732
4	DLS	1.583	39.58%	1.846	0.706

ISSN 1512-1232

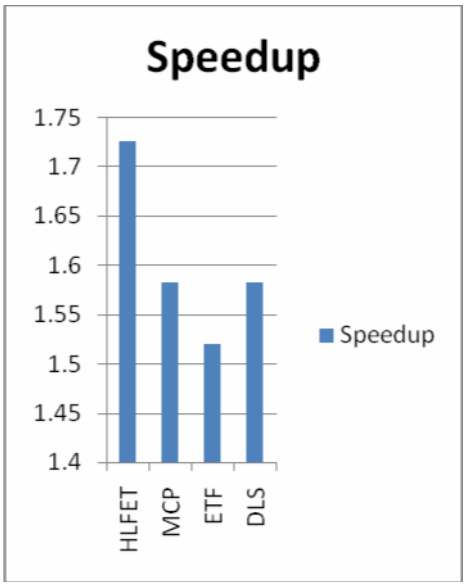

Figure 8 (a).Speedup

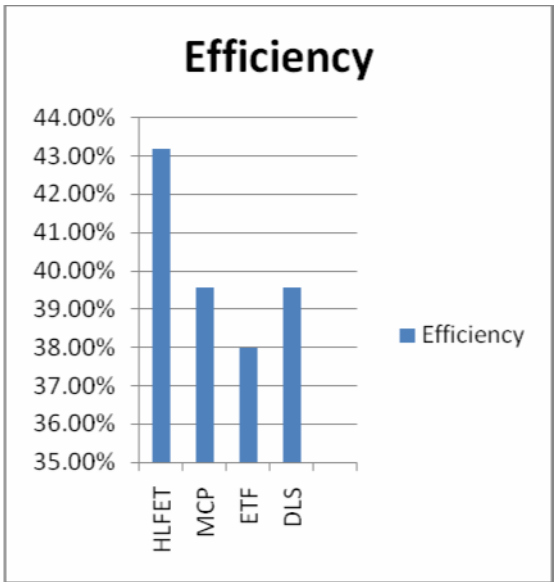

Figure 8(b). Efficiency

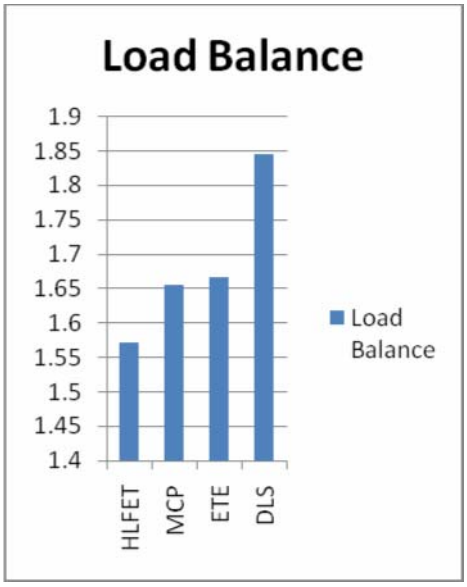

Figure 8(c). Load Balance

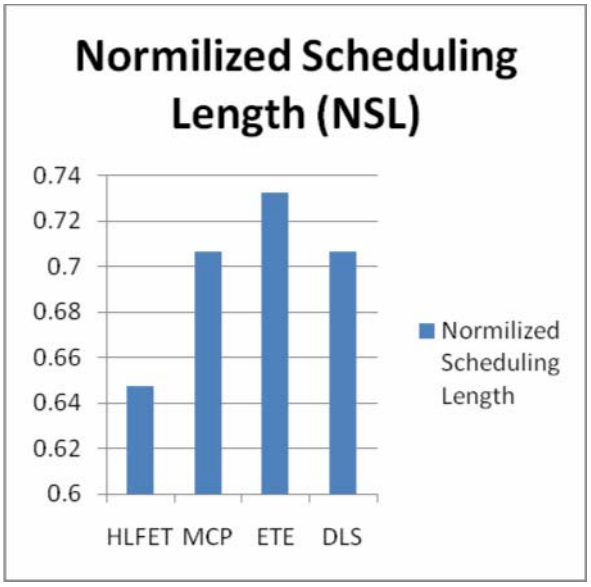

Figure 8 (d). NSL

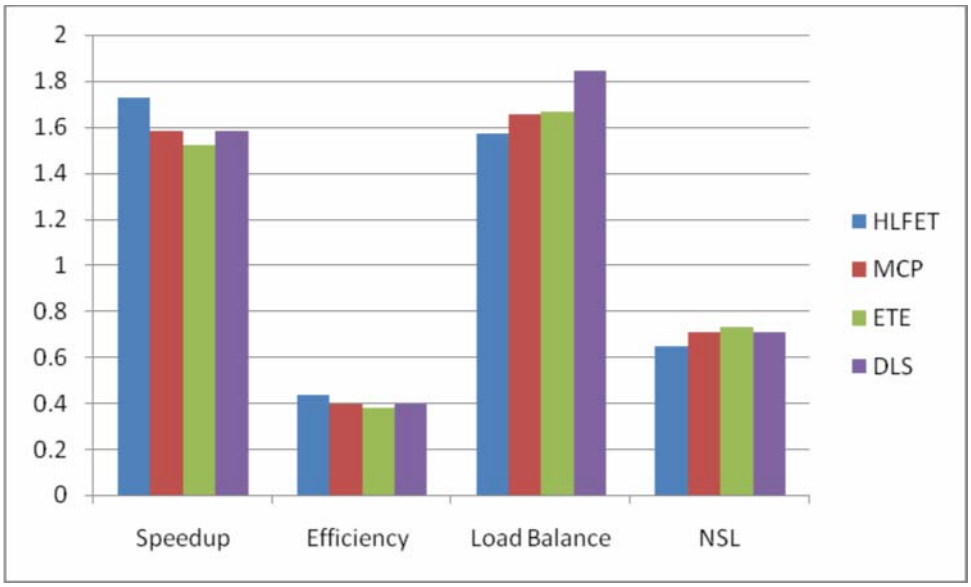

Figure 8. Performance Analysis of BNP Class Scheduling Algorithms

ISSN 1512-1232

Conclusion

In this paper, we have studied BNP (Bounded Number Processors) Classes of scheduling: HLFET, MCP, ETE and DLS scheduling algorithms. We compared their performance based on speedup, efficiency, load balancing and normalized schedule length. The HLFET scheduling are the highest speedup and efficiency than MCP, ETF, and DLS scheduling. The MCP and DLS scheduling are same speedup and efficiency. The DLS scheduling gives better load balance than HLFET, MCP and ETF scheduling. The ETF scheduling gives better NSL than HLFET, MCP, DLS scheduling. All these comparison, we have studied on eleven tasks DAG model. The future wok would be, we will also comparison other class of multiprocessor scheduling algorithms based on matrices with more task graph.

References:

1. M.R.Gary and D.S.Johnson, "Computer and Intractability: A Guide to the theory of NP Completeness ", San Francisco .CA, W.H freeman, 1989.
2. Ullman, J., "NP-Complete Scheduling Problem", Journal of Computer System Science 10,pp384-393,1975.
3. Yu-Kwong Kwok and Ishfaq Ahmad," Static Scheduling Algorithms for Allocating Directed Task Graphs to Mulitprocessors", ACM Conputing Surveys, Vol.31 N0.4, December 1999.
4. H.El-Rewini,T.G.Lewis and H.H.Ali,"Task Scheduling in Parallel and Distributed System" Prentice Hall 1994.
5. S.R.Vijaylakshmi and G.Padmavarth", International Journal of Computer Science and Security",Vol.7,pp125-132,2009
6. [6] H.Topcuoglu, and M.Y.Wu, "Performance-Effective and Low Complexity task scheduling for heterogeneous computing ",IEEE Transaction on Parallel and Distributed Computing ,Vol.13,pp 260-274,3,2002.
7. B.Kruatrachue and T.G.lewis, "Duplication Scheduling Heuristic , a New Precedence Task Scheduling for Parallel System", Techinical Report 87-60-3, Oregon State University,1987.
8. D.C.Goldberg," Genetic Algorithm in Search , Optimization and Machine Learning", Add Wesley Publication 1987
9. M.Srinivas and L.M.Pathnaik," Genetic Algorithm: A Survey Computer", Vol. 27 pp-17-26,1994
10. Amir Masoud Rahman and Mohammad Ali Vahedi," A Noval Task Scheduling in Multiprocessors System with Genetic Algorithm by Using Elistism Stepping Method", Science and Research Branch,Tehran, Iran,May 26,2008.
11. Y-K.Kwok and I.Ahmad, "Dynamic Critical Path Scheduling : An Effective Techniques for Allocating Tasks Graph onto Multiprocessor", IEEE Transaction on Parallel and Distributed System,Vol.7(5) pages 506-621, May 1996.
12. M-Y.Wu and D.D.Gajski,"Hyperpool: A programming Aid for Message Passing", IEEE Transaction on Parallel and Distributed System.Vol1 (3) pages 330-343,July 1990
13. T.L.Adam,K.M.Candy and J.Dickson, "A Comparison of list scheduling for parallel processing system", Communication ACM 17, No.12 (dec 1974) pp 685-690.
14. J.J.Hwang.Y.C.Chow,F.D.Anger and C.Y.Lee, "Scheduling precedence graph in systems with interprocessor communication times", SIAM Journal of Computing. 18 No 2(April 1989),pp244-257.
15. G,C.Sih and E.A.Lee," Compile time scheduling heuristic for interconnection –constrained heterogeneous processor architecture", IEEE Transaction on Parallel and Distributed System 4. No 2(Feb.1993),pp-75-87
16. M.J.Quinn,"Parallel Programming in C with MPI and OpenMP",Tata McGraw-Hill Edition 2003.

ISSN 1512-1232
17. KwangSik Shin,MyongJin Cha, MunSuck Jang,"Task Scheduling Algorithm using Minimized Duplication in Homogeneous Systems", Journal of Parallel and Distributed Computing, 68(2008) 1146-1156.
18. Fatma A.Omara, Mona M.Arafa, "Genetic Algorithm for Task scheduling Problem", Journal of Parallel and Distributed Computing 70(2010),13-22

Article received: 2011-09-15