International Journal of Grid and Distributed Computing
Vol. 9, No. 9 (2016), pp.103-114
http://dx.doi.org/10.14257/ijgdc.2016.9.9.10

A DAG based Task Scheduling Algorithms for Multiprocessor System - A Survey

Gurjit Kaur

Research Scholar, Dept. of CSE,
DAV University, Jalandhar, India
jeet.deol@ymail.com

Abstract

The multiprocessor computing is composed of more than one central processing units (CPU) that simultaneously execute the task of a parallel application for obtain quick results, to process a massive amount of data, and to solve a problem in expected time. If Scheduling is done properly in task allocation then they are increase the performance of the system. Task scheduling in a parallel environment is one of the NP-problems, which deals with the optimal assignment of a task. In this paper, various algorithms are surveyed that apportion a parallel program impersonate by an edge-weighted Directed Acyclic Graph (DAG). These include Bounded no. of Processors (BNP), Unbounded no. of Clusters (UNC), Task Duplication Based scheduling (TDB) and Arbitrary Processor Network scheduling algorithm (APN). The objective of this paper is to study and explore several DAG based task scheduling algorithm and the performance of all of the algorithms is evaluated and compared against each other on a unified basis by using various scheduling parameters.

Keywords*: Multiprocessor system, Task Scheduling, Directed Acyclic Graph (DAG), List scheduling, Cluster Scheduling and Duplicate scheduling*

1. Introduction

Task scheduling [1, 2, 3 and 4] can be defined as assigning the task to a processor for executing at a particular time. An execution of task scheduling depends upon the following components [5, 6].

- No. of processors
- Performance of processors
- Mapping the task to the processors
- Sequence of task for execution on a particular processor.

All the above components highly depend on each other's and compute the optimized results. They all are work together so no one considered individual. Parallel processing is an energetic form of information processing, in which problem or task is cleft into parts, which are executed simultaneously each on its individual processor. The processors may be arranged in homogeneous and heterogeneous environment. Task scheduling algorithms can be represented by Directed Acyclic Graphs (DAG). The objectives of this study include the 21 different algorithms under the categories of UNC, BNP, TDB and APN. Parallel task scheduling algorithms are classified into four different groups:

1.1. Bounded Number of Processors (BNP) Scheduling Algorithms

These algorithms [7] schedule the DAG to a bounded number of processors precisely. The processors are imagine to be fully-connected.

ISSN: 2005-4262 IJGDC

1.2. Unbounded Number of Clusters (UNC) Scheduling Algorithms

These algorithms [7] 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.

1.3. Task Duplication Based (TDB) Scheduling Algorithms

These algorithms [7] also schedule the DAG to an unbounded number of clusters but employ task duplication technique to further reduce the completion time.

1.4. Arbitrary Processor Network (APN) Scheduling Algorithms

These algorithms [7] perform scheduling and mapping on the target architectures in which the processors are connected via a network of arbitrary topology.

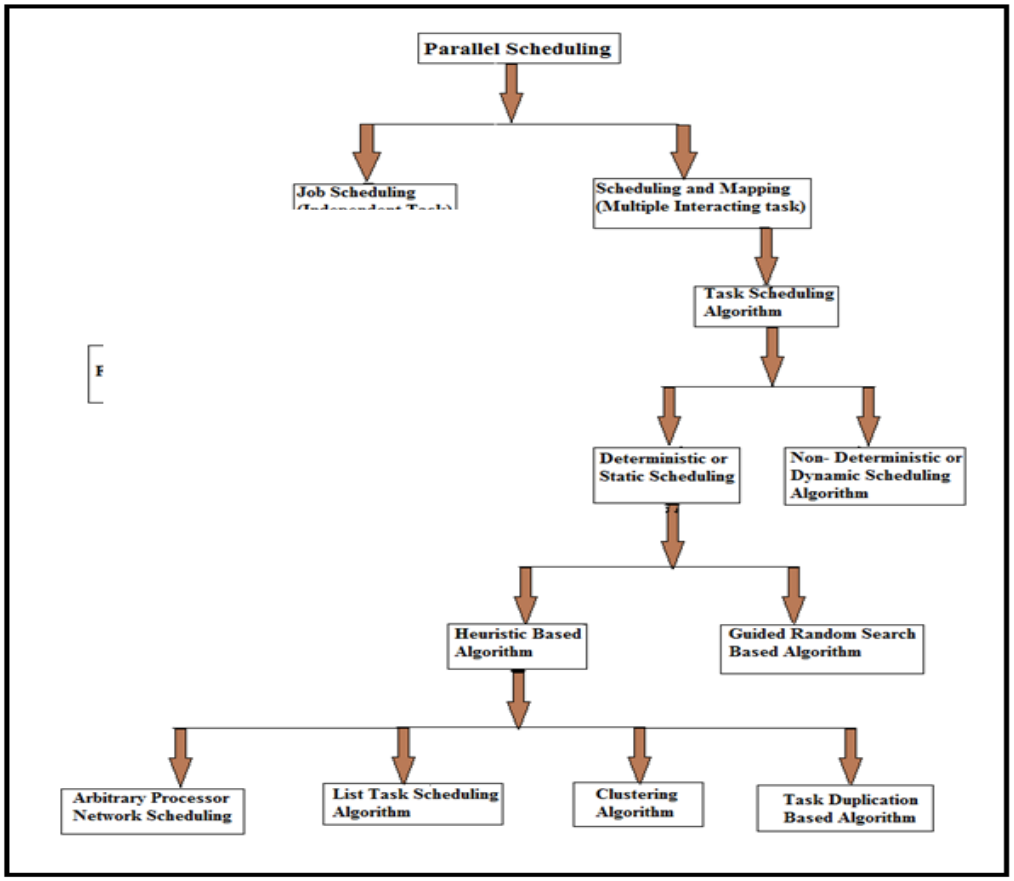

Figure 1. Classification of Various Scheduling Algorithm for parallel Environment.

The resting paper is systematized as follows: First, the introduction of task scheduling and their algorithms are discussed. In Section 2, we can represent a direct acyclic graph (DAG) and also explain DAG based scheduling algorithm. In Section 3, performance parameters are discussed. Lastly, conclusion and future scope is demonstrated in Section 4.

2. The Direct Acyclic Graph (DAG) Model

Direct Acyclic Graph is the legitimate model of a parallel system [10]. Most of the parallel functions can be portray by a DAG [2, 11]. It is a collection of vertex/node and edges [8, 9] as shown in Figure 2.

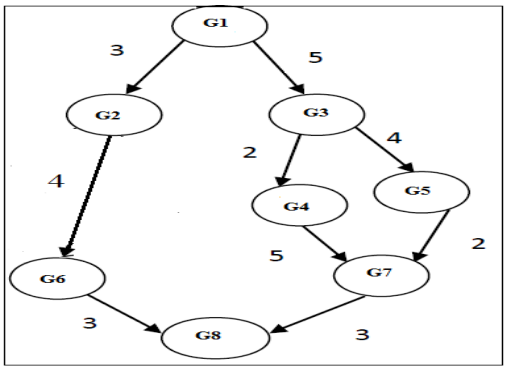

Figure 2. A Sample Directed Acyclic Graph.

In Figure 2, shows the vertex with a weight that represents task processing time and edges represent the communication and dependency time betwixt the tasks [12, 3]. The DAG Graph (G) is defined as G = (V, E), locus V is set of vertex/node and E is set of edges. The starting node (also called source node) of the graph is called the parent node or and end node (also called sink node) of the graph is called the child node. In Figure 5, shows that G1, G2, G3, G4, G5, G6, G7 and G8 all are the node of the graph. G1 is a starting node because it has no parent node and G8 is an end node, with no child node [8, 9].

2.1. Classification of DAG Scheduling Algorithm

An edge weighted DAG is also called a task graph or macro- data flow graph [13, 4]. The DAG based scheduling can be classified into four groups [14, 9], as exhibit in Figure 3. In this section, we can discuss BNP, UNC, TDB, and APN DAG scheduling algorithms [15, 4 and 11]

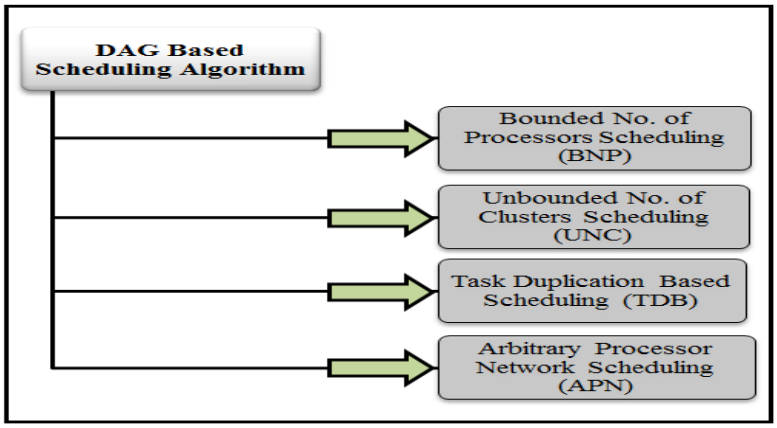

Figure 3. DAG Scheduling Algorithm.

2.1.1. BNP Scheduling Algorithm: BNP stands for the Bounded number of processors [13]. It is based on the list scheduling techniques [16, 17, 18 and 15]. The BNP algorithms schedule the DAG to a bounded number of processors directly. The processors are counterfeited to be full connected. List scheduling is a class of

scheduling heuristics in which the nodes are assigned in priorities and placed in a list arranged in a descending order of priority. The node with a higher priority will be examined for scheduling first rather than a node with a lower priority. All the BNP scheduling algorithms are explained in Table 2.

Two considerable attributes for accrediting priority are the t-level (top level) and b-level (bottom level). The t-level of a node 'n' is the length of the longest path from an entry node to 'n' node in the DAG excluding 'n' node. The t-level of 'n' is also known as earliest start time, denoted by T (n), which is determined by 'n' is scheduled to a processor. The b-level of a node 'n' is the length of the longest path from node 'n' to an exit node. In b-level, only the weights of the nodes are considered not weights of the edges while measured. The b-level of a node is bounded by length of the critical path.

A critical path of a DAG is a path from an entry node to an exit node, whose length is a maximum. There are various notations that can be used in these algorithms are as illustrated in Table 1. Most scheduling algorithms attempt to minimize the start-time of a node for assigning a node to a processor. This is a greedy strategy. But in Non greedy, the algorithms do not minimize the start-time of a node but consider other factors as well.

Table 1. Notation used in DAG Algorithms.

Symbol	Description
-b-level	- Bottom level of a node
-t- level	- Top level of a node
-V	- No. of vertex in the graph
-e	- No. of edges in the graph
-DL	- Dynamic level
-ALAP	- As late as possible start time of the node
-CP	- Critical path
-P	- No. of processors
-f(p)	- Time complexity of the message routing algorithm.
-CPN	- Critical Path Nodes
-IBN	- In-Branch Nodes
-OBN	- Out-Branch Nodes

2.1.2. UNC Scheduling Algorithm: UNC stands for Unbounded Number of Clusters [13, 15]. The UNC algorithms schedule the DAG to a bounded number of bunches. The processors are pretended to be entirely connected. The technique hired by the UNC algorithms is also called clustering.

- At the starting point of the programming process, particular node is considered as a group.
- In the next stage, two groups are merged if the merger minimize the completion time.
- This procedure continues until no cluster can be merged.

The main goal of these algorithms is to reduce the number of clusters. In Table 2, we discuss some UNC scheduling algorithms.

2.1.3. TDB Scheduling Algorithm: TDB stands for Task Duplication Based scheduling algorithm [13, 19, 20 and 15]. TDB is scheduled the DAG to an unbounded number of clusters but hires duplication method to reduce the completion time. The main purpose of this algorithm is to reduce the communication overhead by allocating a redundant task to different processors. This different strategy can be used to select parent nodes for duplication. We survey some of the TDB scheduling algorithms. We

describe six TDB scheduling algorithms in detail: the PY, LWB, DSH, BTDH, LCTD, and CPFD algorithm as shown in Figure 4. And detail demonstrates in Table 2.

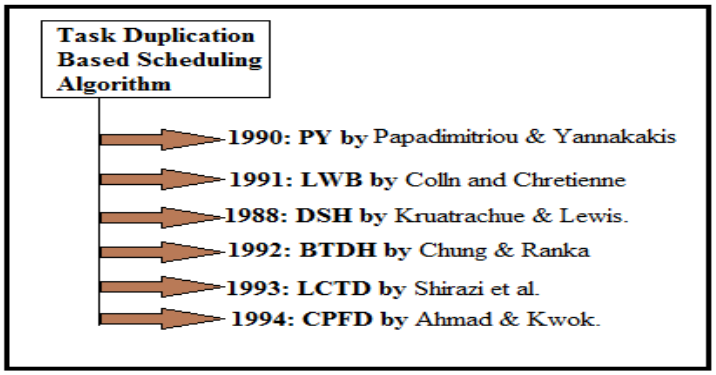

Figure 4. TDB Scheduling Algorithm.

2.1.4. APN Scheduling Algorithm: APN stands for Arbitrary Processor Network scheduling algorithm. These algorithms perform scheduling and mapping task on processors and passing message via a network of arbitrary topology. The algorithms in this category include specific architectural features such as the number of processors as well as their interconnection topology. Scheduling of messages may be reliant on the routing scenario used by the underlying network. The mapping, including the temporal dependencies, is therefore implicit - without going through a separate clustering phase. In Table 2, we discuss some APN algorithms and also shown in Figure 5.

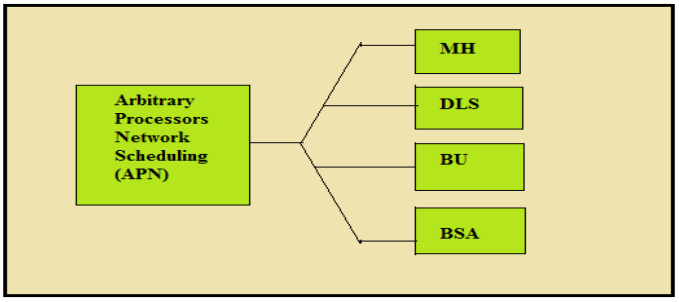

Figure 5. APN Scheduling Algorithm.

Table 2. DAG based Task Scheduling Algorithm.

Sr. No.	Authors	Algorithm	Year	Description	Priority attribute	Time Complexity	Limitations	Greedy	Duplication	Type of algorithm
1	Thomas L. Adam et al.	Highest level first with estimated times (HLFET) [22]	1974	It schedules the first node from ready queue to that processor that allows the earliest execution.	Static b-level	$O(V^{2)}$	The communication time is not considered.	Yes	No	BNP
2	B. Kruatrachue & T. G. Lewis	Insertion scheduling heuristic algorithm (ISH)[21]	1987	It utilizes the idle time that is created by partial schedule on processors.	Static b-level	$O(V^2)$	Cannot give a guarantee optimal schedule.	Yes	No	BNP

3	Kruatrachue and Lewis	Duplication Scheduling Heuristic (DSH) Algorithm [24]	1988	Determines the beginning time of the node on the processor without duplication of any ancestor and then it tries to duplicate the ancestors of the node into the duplication time slot.	b- level	$O(V^4)$	It does not improve the start time of the node.	Yes	Yes	TDB
4	S.J.Kim, J.C. Browne	Linear Clustering algorithm (LC) [23]	1988	It work using iteration method and create a cluster by zeroing all the edges based on the CP	CP	O(v(e+v))	It does not schedule nodes on different paths to the same processor.	No	No	UNC
5	Jing jang hwang, et al.	Earliest time first algorithm (ETF) [26]	1989	Determines the earliest start time of all the tasks and then selects the task with the smallest start time.	Static t-level	$O(PV^3)$	This algorithm is to not being able to minimize the scheduling length at each step.	Yes	No	BNP
6	V. Sarkar	Edge-zeroing algorithm (EZ) [25]	1989	It selects clusters for merging based on edge weights. Zero the highest edge weight if the completion time does not increase.	Static b-level	O(e(e+v))	Does not provide guarantee optimal schedules.	No	No	UNC
7	Baxter and Patel	Localized allocation of static tasks (LAST) algorithm [29]	1989	The main goal of this is to minimize the overall communication.	Dynamic edge weight	O(v(e+v))	It schedule nodes only based on edge weight only.	Yes	No	BNP
8	Min-you Wu. Et al.	Modified Critical Path Algorithm (MCP) [28]	1990	This algorithm gives the highest priority to task which takes minimum start time.	ALAP (based on b-level)	$O(V^2 (\log V)+P)$	HLFET algorithm performs better as compared to MCP.	Yes	No	BNP
9	Papadimitriou and Yannakakis	Papadimitriou and Yannakakis (PY) [27]	1990	It use e-value attribute. This attribute is computed recursively beginning from the starting nodes to the end nodes.	e- Value	$O(V^2 (e+V\log V))$	When it inserts each node into a cluster and due to duplication of ancestors data arrival times larger than the e value of the node.	No	Yes	TDB
10	M.Y.Wu and D.D. Gadjski.	Mobility directed algorithm (MD) [28, 14]	1990	This algorithm scans from the earliest idle time slot on each cluster and schedules the node into the first idle time slot that is large enough for the node.	Relative mobility	$O(V^3)$	NA	No	No	UNC
11	El-Rewini and Lewis	Mapping Heuristic algorithm (MH)[34]	1990	It initializes a ready node list that contains all entry nodes ordered in decreasing priorities. Each node is scheduled to a processor that gives the smallest start time.	static b-levels	$O(v(p^3v + e))$.	The MH may take super polynomial time to solve task scheduling. This is because they are not scalable in nature.	-	No	APN
12	Colin and Chretienne	Lower Bound algorithm (LWB)[31]	1991	It first determines the lower bound start time for each node and then identifies a set of critical edges in the DAG.	Lower bound start time	$O(V^2)$	In which node weights are strictly larger than any edge weight	No	Yes	TDB

13	Chung and Ranka	Bottom-up Top-down Duplication Heuristic algorithm (BTDH)[32,24]	1992	To reduce the start time, the duplication time slot is filled up and the start time of the task under consideration temporarily increases.	b- level	$O(V^4)$	BTDH algorithm does not stop the duplication process even though the start-time increases.	Yes	Yes	TDB
14	Shirazi et al	Linear Clustering with Task Duplication algorithm (LCTD)[33]	1993	Tasks on the longest path are clustered and removed from the task graph. Then, duplication of the parents corresponding to the edges is done to reduce the start time for nodes in the cluster.	Linear Cluster	$O(V^3 \log V)$	NA	Yes	Yes	TDB
15	Sih and Lee	Dynamic level Scheduling algorithm (DLS) [30]	1993	It required the message routing method. These routing methods are supplied by the user.	-	$O(v^3 pf(p))$	Insufficient routing strategy will increases the schedule length.	-	No	APN
16	Mehdirata & Ghose	Bottom Up algorithm (BU) [35]	1994	Firstly, finds the CP of the DAG and then assigns all the nodes of CP to the same processor. Afterward the algorithm assigns the remaining nodes to the processors in a reversed topological order.	CP method	$O(v^2 \log v)$	The schedule length is considerably longer than that of the MH and DLS algorithms	-	No	APN
17	Ahmad and Kwok	Critical path fast duplication algorithm (CPFD) [37]	1994	An OBN is a node which is neither a CPN nor an IBN. CPN nodes are lie on the critical path and their finish times effectively determine the final makespan.	CPN, IBN and OBN	$O(V^4)$	NA	NA	Yes	TDB
18	Gilbert C. sih, et al.	Dynamic level scheduling algorithm (DLS) [30]	1993	DLS uses DL attribute for priority. DL is calculated by subtracting the Earliest Start Time from Static b-Level.	Dynamic level	$O(PV^3)$	It uses the dynamic level, but they can't give a guarantee optimal schedule.	Yes	No	BNP
19	T.Yang, A. Gerasoulis.	Dominant sequence clustering algorithm (DSC) [39]	1994	The Dominant Sequence is the CP of the partially scheduled DAG. A distinctive feature of the algorithm is that in order to lower the time complexity.	t- level and b-level	$O((e+v)\log v)$	If no zeroing is accepted, the node remains in a single node cluster.	Yes	No	UNC
20	Kwok and Ahmad	Bubble Scheduling and Allocation (BSA) [36]	1995	It constructs a schedule incrementally. The task has highest connectivity in the processor network is called the pivot processor and by migrating it to one of the adjacent processors.	-	$O(p^2 ev)$	More cost and large schedule length.	Yes	-	APN
21	Yu-Kwong Kwok and I. Ahmad.	Dynamic Critical Path Algorithm (DCP)[38,14]	1996	It's design based on the value of mobility. This algorithms use a lookahead strategy to find a better cluster for a given node.	b- level & t-level	$O(V^3)$	More cost and effort are obligation for this algorithm.	No	No	UNC

3. Performance Metrics

The Performance is constantly a central factor in determining the smash of any system. We must early choose some criteria. Those are called metrics for performance

evaluation [1, 40, 41, 42 and 43]. There are different metrics used for measure the performance of the various parallel scheduling algorithms.

- **Processor utilization:** In this parameter focuses on the Resource. Using a resource in a way that increases throughput and any resources should not remain idle for a long time.
- **Speed up:** Speedup is the proportion of sequential execution time and parallel execution time.

Speed Up = Time taken by serial algorithm / Time taken by parallel Algorithm

- **Makespan:** The total scheduling length is called makespan. Makespan is calculated by measuring the finishing time of the exit task by the algorithm.
- **Scheduled length Ratio (SLR):** It is defined as the ratio of Makespan of the algorithm to Critical Path values of the DAG.
- **Normalized schedule length (NSL):** The NSL of an algorithm is obtained by dividing the schedule length produced by the algorithm and the sum of weights of the nodes on the original critical-path.
- **Average Running Time of the algorithm:** The running time of an algorithm is its execution time for obtaining the output schedule of a given task graph.

For the comparative analysis of BNP algorithm the following criteria is use [44, 41, 18, 15, 45]. The results include the makespan time of the algorithm, scheduled length ratio (SLR), speed up and processor utilization by the algorithm as shown in Table 3 [43].

Table 3. Comparison of BNP Scheduling Algorithm.

S. No.	Algorithm	Makespan			SLR			Speed up			Processor utilization		
		L	M	H	L	M	H	L	M	H	L	M	H
1.	HLEFT	✓					✓	✓			✓		
2.	MCP	✓				✓		✓				✓	
3.	ETF		✓		✓					✓			✓
4.	DLS			✓	✓					✓			✓
Note: L- Low, M- Medium, H- High													

The UNC algorithm compare with their own classes, the comparison are made using the following criteria [46, 15]: - the normalized schedule length (NSL) [43], Processors utilizations, average running time and overall performance of the algorithm as shown in Table 4.

Table No. 4. Comparison of UNC Scheduling Algorithm.

S. No.	Algorithm	NSL			Processors Utilizations		Average Running time of the algorithm		Overall Performance	
		L	M	H	L	H	L	H	L	H
1.	LC		✓			✓	✓		✓	
2.	EZ			✓		✓		✓	✓	
3.	MD			✓	✓			✓		✓
4.	DSC	✓				✓	✓		✓	
5.	DCP	✓			✓			✓		✓
Note: L- Low, H- High, M- Medium										

To analyse the performance of task duplication based parallel algorithms. We can use the NLS, processors utilizations, makespan, overall performance, and speed up criteria for comparing the algorithms in table 5 [19, 47, 15].

Table No. 5. Comparison of TDB Scheduling Algorithm

S. No.	Algorithm	NSL		Processors utilizations		Makespan		Speed up		Overall Performance	
		L	H	L	H	L	H	L	H	L	H
1.	LWB		✓		✓		✓	✓		✓	
2.	LCTD		✓	✓		NA		✓		✓	
3.	DSH	✓		✓		✓			✓		✓
4.	BTDH	✓		✓		NA			✓		✓
5.	PY		✓		✓		✓	✓		✓	
6.	CPFD	✓			✓	✓			✓		✓
Note: L – low, H- High											

The APN scheduling algorithm can be fairly complicated because they take into account more parameters. In table 6 show the comparison of APN based classes of algorithm [46, 48, 15].

Table No. 6. Comparison of APN Scheduling Algorithm.

S. No.	Algorithm	NSL		Processor utilizations		Average Running time of the Algorithm		Overall Performance	
		L	H	L	H	L	H	L	H
1.	MH		✓	✓		✓		✓	
2.	DLS	✓		NA			✓		✓
3.	BSA	✓		NA		✓			✓
4.	BU	✓			✓	✓		✓	
Note: L- Low, H- High									

4. Conclusions

Parallel task scheduling is a key factor for achieving the high performance in a multiprocessor system. The multiprocessor scheduling can be explained with DAG model. A taxonomy of DAG scheduling algorithms is studied in this paper which classifies the algorithms into four categories: the UNC (unbounded number of clusters) scheduling, the BNP (bounded number of processors) scheduling, the TDB (task duplication based) scheduling, and APN (arbitrary processor network) scheduling. The objective of task scheduling is to map tasks in parallel on the multiprocessors and order their execution so that a minimum schedule length. These algorithms have been compared against each other based on the following parameters: Processors utilizations, makespan, speed up, overall performance, Normalized schedule length etc. In this survey, it has been observed that various researchers are proposed various task scheduling algorithms with heuristics. Hence, the future work will focus on developing efficient task scheduling algorithms for higher performance by applying heuristic methods and evolutionary algorithm for better optimization in the task scheduling that will reduce the schedule length.

References

[1] H. Topcuoglu and S. Hariri, "Task scheduling algorithms for heterogeneous processors," Proceedings. Eighth Heterogeneous Computing Workshop (HCW'99), **(1999),** pp. 3–14.
[2] J. Singh and G. Singh, "Task Scheduling using Performance Effective Genetic Algorithm for Parallel Heterogeneous System". International Journal of Computer Science and Telecommunications, Vol. 3, Issue 3**,(2012).**
[3] R. Singh, "Task Scheduling in Parallel Systems using Genetic Algorithm*,"* International Journal of Computer Applications, Vol. 108, No. 16, **(2014)** pp. 34–40.
[4] X. Tang, K. Li, and D. Padua, "Communication contention in APN list scheduling algorithm," Science in China Series F: Information Sciences, Vol. 52, Issue 1, **(2009**) pp. 59–69.
[5] F. Bonomi, "On job assignment for a parallel system of processor sharing queues," IEEE Transactions on Computers, Vol. 39, No. 7, (**1990**) pp. 858–869.
[6] R. Singh, "Scheduling with Heuristic Technique using Parallel Environment," International journal of computers and distributed systems, Vol. 4, Issue 1, (**2013**) pp. 1–6.
[7] I. Ahmad and Y. K. Kwok "Analysis, evaluation, and comparison of algorithms for scheduling task graphs on parallel processors." InParallel Architectures, Algorithms, and Networks, 1996. Proceedings., IEEE Second International Symposium on, **(1996)** pp. 207-213.
[8] D. I. G. Amalarethinam and A. M. Josphin, "Dynamic Task Scheduling Methods in Heterogeneous Systems- A Survey," International journal of computer applications, Vol. 110, No. 6, **(2015)** pp. 12–18.
[9] Minakshi and G. Singh, "A Study On Evolutionary Optimization Based Scheduling Algorithm Techniques For Parallel Processors," International Journal of Advance Foundation And Research In Science & Engineering (IJAFRSE), Vol. 1, **(2015)** pp. 1–11.
[10] M. Wolfe and U. Banerjee, "Data Dependence and Its Application to Parallel Processing," International Journal of Parallel Processing, Vol. 16, No. 2, **(1987)** pp. 137–178.
[11] R. Kaur, P. Kachroo, " Evaluation and Comparison of Make Span Time in BNP Scheduling Algorithms ," International Journal of Futuristic Science Engineering and Technology, Vol. 1, Issue 6, **(2013)** pp. 376–380.
[12] D. I. G. Amalarethinam and G. J. J. Mary, "A new DAG based Dynamic Task Scheduling Algorithm (DYTAS) for Multiprocessor Systems," International Journal of Computer Applications, Vol. 19, No. 8, **(2011**) pp. 24–28.
[13] A. Dogra and K. Dhiman, "Scheduling Using Multi Objective Genetic Algorithm," IOSR Journal of Computer Engineering Ver. II, Vol. 17, No. 3, **(2015)** pp 73-78.
[14] S. Sharma, "Critical Analysis of Various Parallel Scheduling Algorithms," International Journal of Engineering and Computer Science, Vol. 3, No. 6, **(2014)** pp. 6615–6619.
[15] Y. K. Kwok and I. Ahmad, "Benchmarking and Comparison of the Task Graph Scheduling Algorithms*,"* Journal of Parallel and Distributed Computing, Vol. 59, Issue 3, **(1999)** pp. 381–422.
[16] J. Barbosa and A. P. Monteiro, "A List Scheduling Algorithm for Scheduling Multi-user Jobs on Clusters," High Performance Computing for Computational Science – Vecpar, Vol. 5336, (**2008)** pp. 123–136.
[17] C. Science and N. Paltz, "Analysis of the List Scheduling Algorithm for Precedence Constrained Parallel Tasks," Journal of Combinatorial Optimization, Vol. 88, **(1999)** pp. 73–88.
[18] S. Gill, A. Bharadwaj, N. Singh, H. Singh and J. Singh, "Analysis of HLFET and MCP Task Scheduling Algorithms," International Journal of Modern Engineering Research (IJMER), Vol. 2, Issue 3, **(2012)** pp 1176–1180.
[19] N. Arora, "Comparative Study of Task Duplication based Scheduling Algorithms for Parallel Systems," International Journal of Computer Applications (0975 – 8887), Volume 58, Issue 19, **(2012)** pp. 1-3.
[20] S. Gupta, R. Rajak, K.G. Singh and S. Jain Sanjay, "Review of Task Duplication Based (TDB) Scheduling Algorithms.," The Smart Computing Review, Vol. 5, No. 1, **(2015)** pp. 67–75.
[21] Y. K.. Kwok, and I. Ahmad, "Static scheduling algorithms for allocating directed task graphs to multiprocessors", ACM Computing Surveys, Vol. 31, No. 4, **(1999**) pp. 406–471.
[22] T.L. Adam, K. Clhandy and J. Dickson, "A Comparison of List Scheduling for Parallel Processing Systems," Communications of the ACM**,** vol. 17, no. 12, (1974) pp. 685-690.
[23] S.J. Kim and J.C. Browne, "A General Approach to Mapping of Parallel Computation upon Multiprocessor Architectures," Proc. of Int '1 Conference on Parallel Processing, vol. 11, **(1988)** pp. 1-8.
[24] B. Kruatrachue and T.G. Lewis, "Duplication Scheduling Heuristics (DSH): A New Precedence Task Scheduler for Parallel Processor Systems," Technical Report, Oregon State University, Corvallis, OR 97331, **(1987).**
[25] V. Sarkar, "Partitioning and scheduling parallel programs for execution on multiprocessors," MIT Press, Cambridge, MA, **(1989).**

[26] 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, vol. 18, no. 2, **(1989)** pp. 244-257.
[27] C. H. Papadimitriou and M. Yannakakis, "Towards an Architecture-Independent Analysis of Parallel Algorithms," SIAM Journal on Computing, Vol. 19, No. 2, **(1990)** pp. 22–328.
[28] M.Y. Wu and D.D. Gajski, "Hypertool: A Programming Aid for Message-Passing Systems," IEEE Trans. On Parallel and Distributed Systems, vol. 1, no. 3, **(1990)** pp. 330-343.
[29] J. Baxter and J.H. Patel, "The LAST Algorithm: A Heuristic-Based Static Task Allocation Algorithm," Proc. Of Int'l Conference on Parallel Processing, vol. **11, (1989)** pp. 217-222.
[30] G.C. Sih and E.A. Lee, "A Compile-Time Sicheduling Heuristic for Interconnection-Coxtrained Heterogeneous Processor Architectures," IEEE Trans, on Parallel and Distributed Systems, vol. 4, no. 2, **(1993)** pp. 75-87.
[31] J.Y. Colin and P. Chretienne, "C.P.M. Scheduling with Small Computation Delays and Task Duplication," Operations Research, Vol.39, No.4, **(1991)** pp. 680-684.
[32] Y. Chung and S. Ranka, "Applications and performance analysis of a compile-time optimization approach for list scheduling algorithms on distributed memory multiprocessors," Supercomputing'92., Proceedings. **(1992)** pp 512-521.
[33] H. Chen, B. Shirazi and J. Marquis, "Performance Evaluation of A Novel Scheduling Method: Linear Clustering with Task Duplication," Proc. of Int'l Conf on Parallel and Distributed Systems, **(1993)** pp. 270-275.
[34] H. El-rewini and T.G. Lewis, "Scheduling Parallel Program Tasks onto Arbitrary Target Machines," Journal of Parallel and Distributed computing, vol. 9, no. 2, **(1990)** pp. 138- 153.
[35] N. Mehdiratta and K. Ghose, "A Bottom-Up Approach to Task Scheduling on Distributed Memory Multiprocessors," International Conference on Parallel Processing (ICPP'94), Vol. 2, (**1994)** pp. 151–154.
[36] Y. Kwok and I. Ahmad, "Bubble scheduling: A quasi dynamic algorithm for static allocation of tasks to parallel architectures", Proceedings Seventh IEEE Symposium on Parallel and Distributed Processing, **(1995)** pp. 36–43.
[37] I. Ahmad and Y. K. Kwok, "A New Approach to Scheduling Parallel Programs Using Task Duplication,"(ICPP) International Conference on Parallel Processing, Vol. 2, **(1994)** pp. 47–51.
[38] Y. K. Kwok And Ahmad, I. "Dynamic critical-path scheduling: An effective technique for allocating task graphs to multiprocessors," IEEE Trans. Parallel Distrib. Syst. Vol. 7, No. 5, **(1996)** pp. 506-521.
[39] T. Yang and A. Gerasoulis, " DSC: Scheduling parallel tasks on an unbounded number of processors", IEEE Transactions on Parallel and Distributed Systems, Vol. 5, No. 9, **(1994)** pp. 951–967.
[40] Igor Grudenić "Scheduling Algorithms and Support Tools for Parallel Systems," www.fer.unizg.hr/ _download/ repository /Grudenickvalifikacijski.pdf.
[41] T. Hagras and J. Janecek, "Static vs. Dynamic List-Scheduling Performance Comparison," Acta Polytechnica Vol. 43, No. 6, **(2003)** pp. 16–21.
[42] R. Kaur and R. Kaur, "Multiprocessor Scheduling Using Task Duplication Based Scheduling Algorithms : A Review Paper," International Journal of Application or Innovation in Engineering and management(IJAIEM) Vol. 2, Issue 4, **(2013)** pp. 311–317.
[43] A. A. Nasr, N. A. El-Bahnasawy and A. El-Sayed, "Task Scheduling Algorithm for High Performance Heterogeneous Distributed Computing Systems," International Journal of Computer Applications, Vol. 110, No. 16, (**2015)** pp. 23–29.
[44] P. Kaur, D. Singh, G. Singh and N. Singh, "Analysis , Comparison and Performance Evaluation of Bnp Scheduling Algorithms in Parallel Processing," International Journal of Information Technology and Knowledge Management, Volume 4, No. 1, **(2013)** pp. 279-284.
[45] M. Sharma & H. Kaur, " A Study of Bnp Parallel Task Scheduling Algorithms Metric's for Distributed Database System," International Journal of Distributed and Parallel Systems (IJDPS) Vol. *3, Issue* 1, **(2012)** pp. 157–166.
[46] Y. Kwok and I. Ahmad, "Benchmarking the task graph scheduling algorithms," Proceedings of the International Parallel Processing Symposium, **(1998)** pp 531- 537.
[47] I. Ahmad and Y. K. Kwok, " On exploiting task duplication in parallel program scheduling," IEEE Transactions on Parallel and Distributed Systems, Vol. 9, Issue 9, (**1998**) pp. 872–892.
[48] I. Ahmad, Y. K. Kwok and M. Wu, "Performance Comparison of Algorithms for Static Scheduling of DAGs to Multiprocessors," In Second Australasian Conference on Parallel and Real-Time Systems, (**1995)** pp. 185–192.

Author

Gurjit kaur, She is an M. Tech student in DAV University, Jalandhar, Punjab (India). She has completed her B. Tech in Information Technology from Sant Baba Bhag singh Institute of Engineering ns Technology, Punjab (India) under Punjab Technical University in 2014. She is currently pursuing her research career in Parallel Computing.