Classification of List Task Scheduling Algorithms: A Short Review Paper

Nidhi Rajak and Anurag Dixit
School of Computing Science & Engineering, Galgotias University, Greater Noida, India
Email: {nidhi.mtech2014, anuradixit}@gmail.com

Ranjit Rajak
Department of Computer Science & Applications, Dr. Hari Singh Gour Central University(MP), Sagar, India
Email: ranjit.jnu@gmail.com

Abstract—Task scheduling algorithms are also known as multiprocessor scheduling algorithms. These are mainly used in scientific and engineering applications. It is also considered as a NP-complete problem. The primary objective of list task scheduling algorithms is to minimize the overall execution time. Task scheduling in multiprocessor environment is represented by directed acyclic graph (DAG). It is classified into static and dynamic task scheduling. List task scheduling algorithms is a type of static task scheduling algorithm. In this paper, we have studied different list of task scheduling algorithms: HLFET, ISH, MCP, ETF, DLS and CNPT algorithms. A comparative study amongst the list task scheduling can be based on the following matrices: scheduling length, speedup, efficiency, load balancing, and normalized scheduling length (NSL).

Index Terms—parallel processing, Directed Acyclic Graph (DAG), scheduling length, speedup, efficiency

I. INTRODUCTION

A large problem cannot be solved on sequential machine in reasonable time, so we need a parallel machine that consists of more than two processors to solve a large problem in reasonable time. Here, a large problem is divided into number of sub problems and assigned these sub problems on distinct processors.

Parallel computing is a very efficient technique that meets the computation constraint of a number of scientific and engineering applications. There are two reasons for the popularity and attractiveness of parallel computing [3]: reducing cost of computer hardware and to perform the application that can’t be performed by conventional computers.

The objective of task scheduling algorithms is to allocate tasks to an available processor such that precedence constraints between tasks are satisfied and total execution time should be minimized. Here, the application program is represented by directed acyclic graph (DAG).

The task scheduling is proven to be NP-complete except in few restricted cases [1]. There are two variants that are related to task scheduling algorithms [2]. The first variant is whether to consider communication time or not and second variant is whether the multiprocessor system is homogeneous or heterogeneous. Task scheduling algorithms are consists of three components [4]: performance of homogeneous processors, mapping of tasks onto processors and sequence of the execution of the tasks on each processor.

In this paper, we will present classification of list task scheduling of algorithms and their properties. Also, we will study the comparison matrices for task scheduling algorithms. This paper is organized in five Sections. Section 2 will discuss on a brief introduction of DAG model and Section 3 will present the classification of task scheduling algorithms. The comparison matrices will be briefly discussed in Section 4 and finally, we will come to the conclusion in Section 5.

II. DIRECTED ACYCLIC GRAPH MODEL

A Directed Acyclic Graph (DAG) is also called a Task Graph (TG). It is a generic model for parallel program and consists of set of vertices and set of edges. A directed acyclic graph consists of four tuples [5] that is G=(V,E, W, C) where each vertex V=(T1, T2,..,Tn) is a task of the graph. E is a set of directed edge eij that show the dependency between two tasks Ti and Tj. Each edge of a graph is associated with weight C which is called communication cost or communication time.
The communication time is represented by $C(T_i, T_j)$. A set of task is mapped on a set $P = (P_1, P_2, \ldots, P_p)$ of $p$ processors and each task is executed on a processor known as computation time $W$. It is denoted by $W(T_i)$. The communication time between two tasks would be zero if they execute on the same processor. Here, precedence constraint [6] always holds that is task $T_j$ cannot be executed until $T_i$ completes if $i < j$. If there is a direct link between $T_i$ and $T_j$ then $T_j$ is successor of $T_i$ and $T_i$ is the predecessor of $T_j$. The layout of a DAG model with six nodes is shown in Fig. 1. [5].

A task is said to be an entry task if it has no predecessors and a task is said to be an exit task if it has no successors. Here entry task is $T_1$ and exit task is $T_6$.

The computational time $W(T_i)$ and communication time $C(T_i, T_j)$ of a given DAG is as follows [5]:

- $W(T_1) = 2$
- $W(T_2) = 3$
- $W(T_3) = 4$
- $W(T_4) = 4$
- $W(T_5) = 3$
- $W(T_6) = 3$

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

In this paper, considering that the mapping tasks with communication time on to the homogeneous processors. Fig. 2 is three fully connected homogeneous processors via common link.

![Figure 2. A fully connected homogeneous processors [15]](image)

### III. CLASSIFICATION OF LIST TASK SCHEDULING ALGORITHMS

Task scheduling algorithms are classified into two categories: deterministic task scheduling and nondeterministic task scheduling algorithms [7]. Deterministic task scheduling is also known as static scheduling and compile time scheduling algorithm. In deterministic scheduling, the following information of tasks are known in advance: communication time between tasks, computation time of tasks, and the precedence constraints of tasks. Nondeterministic task scheduling is also known as dynamic scheduling and run time scheduling algorithms. In nondeterministic scheduling algorithm, all the information of tasks are known at run time and not in advance.

Fig. 3 shows the classification of task scheduling algorithms. The list task scheduling algorithms fall in heuristic based algorithms.

Deterministic scheduling algorithms are further classified in two parts: Heuristic based algorithms [8] and Guided random search based algorithms [8]. Heuristic based algorithm that give near – optimal solution. It has acceptable performance and has polynomial time complexity rather than exponential time complexity. This algorithm is classified in three categories: List task scheduling algorithms, Clustering algorithms and Task duplication algorithms. List task scheduling algorithms [8] is a scheduling algorithm in which the given DAG is assigned priority of tasks and it is allocated to processors. It is simple and gives very low complexity with comparisons to other algorithms. Clustering algorithms [8] reduces that communication time between the tasks of a given DAG. It is a very efficient algorithm. It works in two phases [3]: Task clustering phase and post clustering phase. Here task clustering is either linear or non linear.

Task duplication based algorithms [9] gives better efficiency and minimum scheduling length because it reduces the communication time between the tasks. Also, it helps to reduce the start time of the waiting tasks of a given DAG. The major objective of these algorithms is to utilize the processors ideal time.

#### A. Highest Level First with Estimate Times (HLFET) Algorithm

The HLFET algorithm [10] is the first algorithm and the simplest algorithm of list task scheduling algorithms. The task priority of this algorithm is decided by static level attribute [7]. In this algorithm, the communication time is not considered.

#### B. Insertion Scheduling Heuristic (ISH) Algorithm

The ISH algorithm [7] is a very effective algorithm because it utilizes the idle time that is created by partial schedule on processors. The tasks priority of this algorithm is decided by static b-level attribute [7]. It gives better result in comparison to HLFET algorithm.

#### C. Modified Critical Path (MCP) Algorithm

The MCP algorithm [11] determines the task priority using As Late As Possible (ALAP) attribute [5]. This algorithm gives the highest priority to task which takes minimum start time. The major drawback of the MCP algorithm is that it does not include the communication time for computing the priority of the tasks.

#### D. Earliest Time First (ETF) Algorithm

The ETF algorithm [12] determines the earliest start time of all the tasks and then selects the task with the smallest start time. This algorithm is based on the static level attribute. The major drawback of this algorithm is to...
not being able to minimize the scheduling length at every step.

E. Dynamic Level Scheduling (DLS) Algorithm

The DLS algorithm [13] determines the priority of tasks on the basis of dynamic level (DL) [5]. It is similar to ETF algorithm but one difference is that DLS uses DL attribute whereas ETF uses static level attribute. The major drawback of DLS algorithm which does not maintain a scheduling list on scheduling process.

F. Critical Node Parent Tree (CNPT) Algorithm

The major objective of the CNPT algorithm [14] is to achieve greater performance and less complexity. Its task priority is decided by the Critical Node (CN) attribute. This algorithm consists of two phases: listing phase and processors assigning phase. The CNPT gives better performance in comparison to MCP, DLS and ETF algorithms.

In Table I and Table II, time complexity and priority attribute of the list task scheduling algorithms is given in summarized form. Time complexity in term of p, v and e where p is processors, e is edges and v is tasks.

<table>
<thead>
<tr>
<th>S. No</th>
<th>Algorithms</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>HLFET</td>
<td>O(v^2)</td>
</tr>
<tr>
<td>2</td>
<td>ISH</td>
<td>O(v^2)</td>
</tr>
<tr>
<td>3</td>
<td>MCP</td>
<td>O(v^2(log v)+p)</td>
</tr>
<tr>
<td>4</td>
<td>ETF</td>
<td>O(pv^2)</td>
</tr>
<tr>
<td>5</td>
<td>DLS</td>
<td>O(pv^2)</td>
</tr>
<tr>
<td>6</td>
<td>CNPT</td>
<td>O(v^2)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S. No</th>
<th>Algorithms</th>
<th>Priority Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>HLFET</td>
<td>Static Level [10]</td>
</tr>
<tr>
<td>2</td>
<td>ISH</td>
<td>Static Level [10]</td>
</tr>
<tr>
<td>3</td>
<td>MCP</td>
<td>ALAP [5]</td>
</tr>
<tr>
<td>4</td>
<td>ETF</td>
<td>Static Level [12]</td>
</tr>
<tr>
<td>5</td>
<td>DLS</td>
<td>Dynamic Level[5]</td>
</tr>
<tr>
<td>6</td>
<td>CNPT</td>
<td>Critical Node[14]</td>
</tr>
</tbody>
</table>

IV. COMPARISON METRICS

The performance analysis can be done amongst the list task scheduling algorithms based on the comparison matrices: scheduling length, speedup, efficiency, normalized scheduling length (NSL) and load balancing.

A. Scheduling Length

Scheduling length of a list task scheduling algorithm is the maximum finishing time of a last task on a processor. It is also called makespan, and defined by following equation [18]:

\[ SchedulingLength = \max\{FT(T_i, P)\} \]

Here, FT is the finishing time of a last task Ti on a processor P.

B. Speedup

Speedup [16] is the ratio of sequential execution time (SET) and parallel execution time (PET). A SET is the total execution time of a task graph on uniprocessor system and PET is the execution time of a last task on a processor in multiprocessor system. It is defined by following:

\[ Speedup = \frac{SET}{PET} \]

C. Efficiency

Efficiency [16] of a parallel program is the ratio of speedup and number of processors used. It is defined by following:

\[ Efficiency = \frac{Speedup}{Number\ of\ Processors\ used} \]

D. Normalized Scheduling Length (NSL)

NSL [17] of a list task scheduling algorithm is defined by following:

\[ NSL = \frac{SchedulingLength}{Max[\text{Sum of execution time along a path}]} \]

E. Load Balancing

Load balancing [18] is the ratio of scheduling length and average execution time over all the processors. It is defined by following:

\[ LoadBalancing = \frac{SchedulingLength}{Average} \]

where Average is the ratio of sum of processing time of each processor and number of processors.

V. CONCLUSION

In this paper, we have studied classification of list task scheduling algorithms: HLFET, ISH, MCP, ETF, DLS and CNPT algorithms. All these algorithms are for homogeneous environments. Also, we have studied their limitations and time complexity. These algorithms are based on some priority attributes. According to priority attributes, priority is assigned to the tasks. We also studied some comparison matrices: speedup, scheduling length, efficiency, normalized scheduling length, and load balancing. These comparison matrices will help to differentiate between algorithms. The list task scheduling algorithms gives better efficiency and minimum scheduling length as compared to other scheduling algorithms. List task scheduling algorithms for heterogeneous environments may be studied in future.

REFERENCES


**Nidhi Rajak** is a student of M.Tech (CSE) in School of Computing Sciences &Engg. Galgotias University, India. She is completed B.Tech and Diploma in Electronics & Communication from RGPV, Bhopal.

**Anurag Dixit** is a Professor in School of Computing Sciences &Engg. Galgotias University, India. He is completed PhD in Computer Science from JNU and MCA from DU. He has 20 years teaching experience. His research area: Software Engg., Mobile Computing, Parallel Processing & Object Oriented Design.

**Ranjit Rajak** is an Assistant Professor in Department of Computer Science & Application, Dr. Hari Singh Gour Central University, Sagar(MP), India. He is completed MCA, M.Tech(CSE) from JNU, New Delhi, and pursuing PhD in Computer Science from JNU. His area of research: parallel computing, genetic algorithm, scheduling problems.