A Critical-Path and Top-Level attributes based Task Scheduling Algorithm for DAG (CPTL)
Nidhi Rajak¹, Ranjit Rajak² and Anurag Dixit³
¹,³School of Computing & Engineering
Galgotias University: (U.P. India)
¹nidhi.bathre@gmail.com, ³anuradixit@gmail.com
²Department of Computer Science & Applications
Dr. Harisingh Gour Central University
Sagar: (M.P: India)
²ranjit.jnu@gmail.com

ABSTRACT
Parallel computing is an important area of research. It is used from scientific to commercial applications. Task scheduling is one of the important research areas of parallel computing. The major objective of any task scheduling algorithm is to minimize the execution time i.e. to reduce the scheduling length. Here, an application program is represented by a directed acyclic graph (DAG). In this paper, we have proposed an efficient CPTL scheduling algorithm which is based on two well-known attributes: Critical Path (CP), and Top-level (T-level). This algorithm is calculated the difference between CP and t-level, which is called CPT attribute. This attribute used as a priority of the tasks, and allocated these tasks on multiprocessor system. The proposed algorithm outperforms over the existing heuristic algorithms such as HLEFT, MCP, ETF, and DLS. CPTL algorithm gives minimum scheduling length. The comparison has been done based on following metrics: scheduling length, speedup, efficiency, load balancing, and normalized scheduling length.

KEYWORDS
Scheduling, DAG, Parallel Processing, Speedup, and Efficiency.

1 INTRODUCTION
Task scheduling problem is an NP-hard [1,2]. There are three components [3] of tasks scheduling algorithm:(i) are performance of the processors, (ii) mapping of the tasks on to the processors, and (iii) execution order of the tasks on to the processors. Task scheduling algorithm can be either deterministic or non deterministic. In deterministic algorithm, number of tasks, number of processors, execution time of each task on the processors, and communication time between the tasks that are known at compiled time. But in case of nondeterministic algorithm, all the above information is known at run time. The proposed algorithm is based on deterministic model.
In this paper, we present a new approach for task scheduling algorithm in multiprocessor environment which is known as CPTL algorithms. It is based on CPT attribute which is difference of Critical Path (CP) [4], and Top level (t-level) [4]. We find the priority of the tasks of a given DAG using CPT attribute. Also, we present the comparison among proposed algorithm and three task scheduling heuristic algorithms: HLFET, MCP and DLS task scheduling algorithms.
We consider four performance metrics for comparison and performance analysis of the task scheduling algorithms.
The rest of the paper is organized as follows: Problem formulation have done in section 2. Section 3 consists of proposed algorithm and performance metrics are defined in section 4. The performance analysis and results of the task scheduling algorithms has been done in
section 5. Finally, we come to conclusion part in section 6.

2 PROBLEM FORMULATIONS

2.1. Application Model
An application model is basically used for representing application programs. Here, task scheduling of an application program is represented by a directed acyclic graph (DAG). It consists of two tuples $G_i = \langle T, L \rangle$ where $T$ is a finite set of $n^{th}$ tasks i.e $T = \{T_1, T_2, \ldots, T_n\}$ and $L$ is the communication link between the tasks $T_i$ and $T_j$. Each task of a given DAG is associated with execution time and communication time between the tasks $T_i$ and $T_j$. Layout of a DAG model with six tasks is given below in Figure 1.

When two tasks are allocated on a same processor, their communication time would be negligible. All the tasks should maintain the precedence constraints [6]. Here, an entry task is defined as a task which has not any predecessor task and an exit task is defined as a task which does not any successor task.

2.2 System Computing Model

Multiprocessor environment can be either homogenous or heterogeneous. Homogenous processors work at same execution speed whereas heterogeneous processors work at different execution speed. In this paper, we consider homogeneous processors, and all processors are fully connected via identical link. A layout of fully connected processors [7] is shown in figure 2.

A major objective of task scheduling algorithm is to minimize the overall execution time i.e. to reduce the scheduling length.

3 PROPOSED ALGORITHM

The proposed algorithm is based on two attributes Critical Path (CP) [4], and Top-Level (T-Level) [4]. T-level of a task $T_i$ in a DAG is the longest path from entry task to $T_i$, and it does not include execution time of $T_i$, and CP is the longest path from entry task to exit task. We have abbreviated this algorithm as Critical Path and T-Level Attributes based Task Scheduling algorithm (CPTL). The CPTL scheduling algorithm uses CPT attribute which is the difference between CP and T-Level attributes. i.e.

$$CPT(T_i) = CP - T_{Level(T_i)}(i)$$

Where $T_i$ is number of tasks of a given DAG.

CPT attribute is used to computing the priority of the tasks. These tasks are sorted and allocated to the available processors. The tasks are allocated to the processors whose the earliest start time (EST) [8] is smaller.
CPTL Scheduling Algorithm:

**Step 1:** Read DAG \((t_n,\text{ number of tasks})\)
- Entry Task: first task or starting task of given DAG
- Exit Task: last task or end task of given DAG

**Step 2:** Compute CP of given DAG.

**Step 3:** Compute \(t\)-level of given DAG.

**Step 4:** Compute CPT

**Step 5:** Sorting the tasks of CPT in decreasing order.

**Step 6:** Add the tasks to a Queue.

**Step 7:** Remove tasks one by one from the Queue.

**Step 8:** Check Precedence Constraint (PC)
- If PC is satisfied then
  - Allocate to the processor
- Else
  - Removed task insert into end of the Queue.

**Step 9:** When last a task of the Queue is allocated to processor
- Find the completion time of the last task.

**Step 10:** Stop.

3.1 Numerical Examples with Nine and Eleven Tasks.

We have taken two DAG1 and DAG2 with nine and eleven tasks respectively, and considered four homogeneous processor for allocation of the tasks. The communication and execution times are shown in the DAGs.

### Table 1. Computing CPT(Ti) Attribute for DAG1 with nine tasks

<table>
<thead>
<tr>
<th>Tasks(Ti)</th>
<th>T-Level(Ti)</th>
<th>CPT(Ti)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>0</td>
<td>23</td>
</tr>
<tr>
<td>T2</td>
<td>6</td>
<td>17</td>
</tr>
<tr>
<td>T3</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>T4</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>T5</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>T6</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>T7</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>T8</td>
<td>8</td>
<td>15</td>
</tr>
<tr>
<td>T9</td>
<td>22</td>
<td>1</td>
</tr>
</tbody>
</table>

**Figure 3.** DAG1 with nine tasks [4]

**Figure 4.** Scheduling Length of Proposed Algorithm is 17 units
Table 2. Computing CPT(Ti) Attribute for DAG2 with eleven tasks

<table>
<thead>
<tr>
<th>Tasks(Ti)</th>
<th>T-Level(Ti)</th>
<th>CPT(Ti)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>0</td>
<td>34</td>
</tr>
<tr>
<td>T2</td>
<td>8</td>
<td>26</td>
</tr>
<tr>
<td>T3</td>
<td>4</td>
<td>30</td>
</tr>
<tr>
<td>T4</td>
<td>5</td>
<td>29</td>
</tr>
<tr>
<td>T5</td>
<td>6</td>
<td>28</td>
</tr>
<tr>
<td>T6</td>
<td>16</td>
<td>18</td>
</tr>
<tr>
<td>T7</td>
<td>10</td>
<td>24</td>
</tr>
<tr>
<td>T8</td>
<td>14</td>
<td>20</td>
</tr>
<tr>
<td>T9</td>
<td>13</td>
<td>21</td>
</tr>
<tr>
<td>T10</td>
<td>23</td>
<td>11</td>
</tr>
<tr>
<td>T11</td>
<td>32</td>
<td>2</td>
</tr>
</tbody>
</table>

The scheduling length of proposed algorithm is given in the figure 6.

4 PERFORMANCE METRICS

This section discusses the performance metrics for evaluating the performance of proposed algorithm and heuristic algorithms. We discuss five performance metrics: Scheduling length, Speedup, Efficiency, Load Balancing, and Normalized Scheduling Length (NSL).

4.1 Scheduling length (SL)

Scheduling length is defined as the execution time taken by the last task executed on a processor.

4.2 Speedup (Sp)

Speedup [9] is the ratio of time taken by sequential and parallel execution. It is denoted by following:
\[ Sp = \frac{T_s}{T_p} \]

Where Ts is the sequential execution time and Tp is the parallel execution time.

### 4.3 Efficiency (Eff.)

Efficiency \([9]\) is the ratio of Speedup and total number of processors used. It is also called as processor utilization.

\[ Eff = \frac{Sp}{P_n} \]

Where Sp is Speedup and Pn is the number of processors used.

### 4.4 Load Balancing (LB)

Load Balancing \([10]\) is the ratio of Scheduling length and mean of execution time of all processors.

\[ LB = \frac{SL}{Mean} \]

\[ Mean = \frac{\sum_{i=1}^{n} P_i}{P_n} \]

Where Pi is a sum of processing time of each processors and Pn is number of processors are used.

### 4.5 Normalized Scheduling Length (NSL)

Normalized Scheduling length (NSL) \([11]\) of task scheduling algorithm is defined by following:

\[ NSL = \frac{SL}{\max\{\text{Sum of computation costs along a path}\}} \]

### 5 PERFORMANCE ANALYSIS

This section presents comparisons among proposed algorithm: CPTL Scheduling algorithm and heuristic algorithms: HLFET, MCP, ETF, and DLS Scheduling algorithms. CPTL scheduling algorithm gives minimum scheduling length as compared to heuristic scheduling algorithms. All comparisons have been done based on performance metrics in Section 4.

We have tested proposed algorithm on two DAG models with nine and eleven tasks. Also, we have taken four processors for allocation of the priority tasks of given DAGs. Table 3 shows comparison of the task scheduling algorithms. We have drawn graphs based on the results of Table 3.

### Table 3. Result of Task Scheduling Algorithms for DAG1

<table>
<thead>
<tr>
<th>Scheduling Algorithms</th>
<th>Performance Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SL</td>
</tr>
<tr>
<td>CPTL</td>
<td>17</td>
</tr>
<tr>
<td>HLFET[6]</td>
<td>19</td>
</tr>
<tr>
<td>MCP[6]</td>
<td>20</td>
</tr>
<tr>
<td>ETF[6]</td>
<td>19</td>
</tr>
<tr>
<td>DLS[6]</td>
<td>19</td>
</tr>
</tbody>
</table>

**Figure 7. Scheduling Length**

**Figure 8. Speedup**
Similarly, we can compute the performance metrics for DAG2 which is shown in Table 4.

**Table 4.** Results of Task Scheduling Algorithms for DAG2

<table>
<thead>
<tr>
<th>Scheduling Algorithms</th>
<th>Performance Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SL</td>
</tr>
<tr>
<td>CPTL</td>
<td>22</td>
</tr>
<tr>
<td>HLFET[5]</td>
<td>22</td>
</tr>
<tr>
<td>MCP[5]</td>
<td>24</td>
</tr>
<tr>
<td>ETF[5]</td>
<td>25</td>
</tr>
<tr>
<td>DLS[5]</td>
<td>24</td>
</tr>
</tbody>
</table>
6 CONCLUSIONS

We present new algorithm based on two attributes: Critical Path (CP) and Top level (T-level), called CPTL scheduling algorithm. It uses new attribute: CPT for computing priority of the tasks. It is a difference between a CP and a T-level attributes. Based on the priority attribute CPT, we allocated tasks on the given processors.

The CPTL scheduling algorithm gives lesser scheduling length as compared to heuristic algorithms: HLFET, MCP, ETF, and DLS. We have compared all task scheduling algorithms based on performance metrics illustrated in Section 4 considering two DAG models.

We achieved higher speedup, and efficiency for CPTL algorithms compared to heuristic algorithms.

Additionally, the proposed algorithm also gives less load balancing and NSL as compared to heuristic algorithms.

7 REFERENCES