Scheduling with Heuristic Technique using Parallel Environment

Rachhpal Singh
Asstt. Professor (Sr.), P.G. Department of Computer Science & Applications, Khalsa College (KCA), City: Amritsar (State-Punjab-INDIA).

Abstract
Job of Parallel Processing is to speed-up the execution of a large task by dividing the task into small and multiple sub-tasks that execute concurrently[1][2]. Here each sub task has its own processor for a faster output. By doing so the effectiveness becomes more beneficial for large number of calculations having some constraints like time constraints etc. Scheduling is the crucial step of parallel programming that deals with the assignment of a set of tasks in the parallel system environment and set the execution in such a way that the total execution time is minimized. The proposed technique has the efficient execution of the schedule on parallel system that takes the structure of the application and the performance characteristics. Number of approximations, heuristics and comparisons have been proposed to fulfill the task scheduling problem. It is well known NP-complete problem (NP-Hard problem) in task scheduling for optimization of the result[3][4]. With a fast heuristic technique known as GA (Genetic Algorithm), it will be proposed to schedule all the parallel tasks in homogenous parallel environment having comparison with the First Come First Serve (FCFS), Shortest Job First (SJF) and Round Robin (RR) scheduling methods. By comparing all the algorithms, optimized result can be evaluated and decision can be done for the best optimized selection.

Key words
Parallel homogeneous System, DAG (Directed acyclic graph), heuristics serach, Parallel processing, Parallel Genetic Algorithm, Task scheduling.

I. INTRODUCTION

Task scheduling can be defined as assigning the different tasks in parallel environment onto a set of homogeneous processor. By doing so it will determine the sequence of execution of the task at each processor. While the makespan of the tasks determined by the sequence of the task in which these are processing in a particular order and further the performance of the processors also effects the total finish time. So an execution of task scheduling depends upon four major components:

i). Sequence of the task in particular order.
ii). Number of processors.
iv). Scheduling and Mapping of the tasks onto the processors [31].

All the above four major components plays vital role in finding the optimized result in solving such type of optimization problem [31]. These all components are interdependent on each other and to compute the optimized results these are not be considered individual. Task scheduling in parallel environment having a set of homogeneous processor is being proposed by using the Genetic Algorithm (GA) approach. A GA approach [7][8][32] is a heuristic technique which deals with a generation of individual. In this technique a set of genes are generate which further creates a string and a string is encoded as individuals. Such type of strings are known as chromosome. With the help of these generated chromosome (generation by generation) gives us an optimized result. A fitness function is used for generating the set of individuals from which another generation of chromosome be generated. Three major operations selection, crossover and mutation operations [15] part of the GA based on some major attributes such as population of chromosome, fitness function, selection criteria, probability of occurrence of crossover and probability of occurrence of mutation. All these attributes are used for the optimization of task scheduling.

This paper is organized in five sections and rest four are as follows: Section II gives an overview of the problem along with brief description of the solution methodology. In the section III detailed proposed heuristic technique explained. Section IV provides the experimental results and performance analysis of the study. At end in the section V conclusion was done.

II. PROBLEM DEFINITION
A brief review of classification of parallel algorithms, task scheduling and methods based on the characteristics of the tasks to be scheduled, the parallel environment and the availability of the information be presented in this
The main idea behind this theory is to subdivide the huge task into a set of sub-tasks in such a way so that execution of the tasks can be done efficiently in the homogeneous parallel computing system. Further this will create a criterion for ordering the task in a fine grain size manner and that criteria be represented in a graphical model. That an abstract model of the partitioned tasks can be represented by a term DAG (Directed Acyclic Graph) [25][28][29]. The main idea behind this is to generate a deterministic scheduling criterion in which there exist a scheduling of a set of precedence relations among the sub-tasks in parallel system. A deterministic scheduling problem [16] is one in which all information about the tasks and the relation to each such as execution time and precedence relation are known to the scheduling algorithm in advance and the processor environment is homogenous[23][24][26][27][33]. The homogeneous setup of processors means that the processors have same processing speeds or processing capabilities for finding the accurate results in time. So here, a study has been done having the deterministic task scheduling concept in the homogeneous parallel multiprocessor environment. This will give accuracy of results in an efficient manner. The major goal by using such type of system is to minimize the makespan, in otherwords it will minimize the total finish time of every task in the parallel multiprocessor system. Here the makespan or the total finish time is the combination of execution time and waiting time or idle time in parallel environment. By taking a set of x homogeneous multiprocessors in the parallel multiprocessor computing environment, an equation can be derived as written below:

\[ P = \{ p \ i = 1, 2, 3...x \} \]

A fully connected processors with each other via identical links or communication lines of three parallel system is as shown in the Figure 1 with the help of a Directed Acyclic Graph. A DAG has the parallel application connected with links and can be represented by a set of nodes G and links E. Let by taking G = (T, E, W, C), where the vertices set T consist of x tasks and are represented as:

\[ T = \{ t \ i = 1, 2, 3...x \} \]

A directed edge set E consist of k edges and all are denoted as:

\[ E = \{ e \ k = 1, 2, 3...t \} \]

Both T and E are deeply interconnected in the parallel multiprocessor system [31].

Above figure gives the complete information about the precedence relationships among tasks in multiprocessor network system. Two task can be represented having link with the directed edges. These edge creates a system in system that no cycle will occurs. By taking the linkage between two different task occurring in a order as task \( t_i \) and next task \( t_j \) and these two task are from the subset task group T. These set of two task will be connected by a directed edge \( e_i \). So here in the occurrence of these two task represent that task \( t_j \) cannot be scheduled until \( t_i \) has been completed. In other words task \( t_i \) is a predecessor of task \( t_j \) i.e. task \( t_j \) is a successor of task \( t_i \). Similarly the task \( t_2, t_3......t_p \) connectivity with edges \( e_2, e_3,......e_r \) can be represented which generate a well designed DAG having tasks and edges. From communication point of view every successor task \( t_m \) sends a message having a set of contents according to the requirement of next task i.e. to the next task \( t_n \). This will generates execution set up between tasks in this graph. Every edge has a weight or the data label between two task can be represented by the term W. It represents the execution duration the corresponding task and are varies from processor to processor in the multiprocessor parallel system. In other words every elements set has the weight between nodes of vertices of the graph in the DAG and is represented as:

\[ W = \{ w_{ij} \ i = 1, 2, 3...m, j: 1, 2, 3...n \} \]

There is a data communication between the tasks and Data communication between the two tasks can be represented by using this technique i.e. if they are scheduled to different processors, but in case of if both tasks are scheduled to the same processor, then the weight associated to the edge becomes null. There is an example of a complete DAG as shown in the Figure 2. The weight between a set of tasks be denoted by the term communication cost CC and can be represented as:

\[ CC = \{ c_{ij} \ i = 1, 2, 3...n \} \]

As shown in Figure 1, DAG naming D consist of a set of tasks \( T = \{ t_j \ j = 1, 2...x \} \) and set of processors \( P = \{ p_i \ i = 1, 2, 3,4,5 \} \). The relationship or association between processor and tasks is represented by processor-task execution time matrix and is as shown in the Table 1. The execution time of each set of tasks is different due to different weight or communication.
cost in the system whether a homogeneous environment. It is assumed due to any reason, processor display different processing speed in the homogeneous system. Let us suppose that processor $p_1$ is much faster than $p_2$, $p_3$ and so on. Processor $p_2$ is faster than $p_3$, $p_4$ and so on. (i.e., the order of speed and processing capabilities can be expressed as $p_1 > p_2 > p_3 > p_4$). As given in Table 1 task $t_1$ takes 4 time units to complete their execution on processor $p_1$ and takes 9 time units and 10 time units to complete their execution on processor $p_9$ and $p_{10}$ respectively. On the basis of the size of the tasks processed on same processors, the execution time has been calculated.

### III. Heuristic Algorithm

Every GAs passes through a cycle of phases and these phases are

1. from the string population creation of strings,
2. string evaluation,
3. best string or strings selection and
4. reproduction to create a new population.

Here the concept of chromosome i.e. the individuals are encoded in the population string known as chromosomes. It is possible to evaluate the performance or fitness of individuals in a population after a chromosome has been coded. A good coding scheme [17][18] will benefit operators and make the objective function easy to calculate. Each individual is assigned a fitness value given by the objective function during the selection operation and choose the fittest individual of the current population to serve as parent of the next generation. Crossover and mutation are the two prime operators of the reproduction.

Individual’s fitness in a population generates a good coding scheme [17][18] which will benefit operators. It makes the objective function easy to calculate. During selection, each individual is assigned a fitness value given by the objective function and choose the fittest individual of the current population to serve as parent of the next generation. Reproduction involves two types of operators namely crossover and mutation. The crossover operator chooses randomly a pair of individuals among those selected previously and exchange some part of the information. The mutation operator takes an individual randomly and works with task duplication heuristics, so that, the total execution time of the schedule should be minimum [10].

**A.** Creation of the initial population is the first step in the GA. Some of the parameters like number of processors, number of tasks and population size are needed to generate initial population. The initial population is initialized with randomly generated individuals. The length of all individuals in an initial population is equal to the number of tasks in the DAG. Each task is randomly assigned to a processor [19].

**B.** Proposed Genetic Algorithm uses the fitness function which is based on the total completion time for the schedule. It includes execution time and communication delay time. The fitness function separates the evaluation into two parts: Task fitness and processor fitness. The task fitness focuses on ensuring that all tasks are performed and scheduled in valid order. A valid order means that a precedence relations are satisfied i.e. successor task cannot scheduled until predecessor has been completed. The processor fitness component attempts to minimize the processing time. Consider the following schedule S1 and S2 for single processor and multiprocessor parallel system tasks schedules with task size equal to 50 tasks respectively (here, consider the case when fitness function assigned all tasks to a single processor and randomly generated tasks to heterogeneous parallel system.) The processor chosen for scheduler S1 is $p_1$ and the execution time for all task are given in Table 1.

![Figure 2. DAG of task size=10 with task precedence.](image-url)
The execution time of different tasks on different processors as shown in Table 1.

<table>
<thead>
<tr>
<th>t1</th>
<th>t2</th>
<th>t3</th>
<th>t4</th>
<th>t5</th>
<th>t6</th>
<th>t7</th>
<th>t8</th>
<th>t9</th>
<th>t10</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3</td>
<td>8</td>
<td>5</td>
<td>2</td>
<td>8</td>
<td>7</td>
<td>4</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>7</td>
<td>6</td>
<td>3</td>
<td>9</td>
<td>8</td>
<td>5</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>7</td>
<td>7</td>
<td>3</td>
<td>9</td>
<td>9</td>
<td>5</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>8</td>
<td>7</td>
<td>4</td>
<td>10</td>
<td>9</td>
<td>6</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>9</td>
<td>8</td>
<td>4</td>
<td>11</td>
<td>10</td>
<td>6</td>
<td>11</td>
<td>13</td>
</tr>
</tbody>
</table>

Table 1: Shows a tasks execution matrix on different processors with task size = 10.

The total finish time of scheduler S1 and S2 is:

**S1:** t1 → t2 → t3 → t4 → t5 → t6 → t7 → t8 → t9 → t10

Total Finish Time = Execution time + Comm. time = 4 + 3 + 6 + 5 + 2 + 8 + 7 + 4 + 9 + 10 = 58 time units

As here all the tasks are executed on same processor, so the communication time is zero. The processors chosen for schedule S2 are same as given in table 1. Here proposed GA use task duplication heuristics to reduce the data response time of its descendant tasks, permitting them to start earlier.

**S2:** Total finish time = Execution time + Comm. Time = 42 time units

The scheduler S1 shows a total finish time of 58 time units, where as scheduler S2 shows a total finish time of just 42 time units. Therefore, proper fitness function reduces the total finish time very well. Therefore, the fitness values (task and processor) have been evaluated for all chromosomes and the probability of higher fitness is to be selected for reproduction from current generation to the next generation.

C. Selection operation is the basic design of fitness function, so how to design the fitness function will directly affect the performance of genetic algorithm. To select the superior and eliminate the inferior, GA uses the selection operator. According to their fitness value individual are selected. Once fitness values have been evaluated for all chromosomes, we can select good chromosomes through rotating roulette wheel strategy. This operator generate next generationby selecting best chromosomes from parents and offsprings.

D. Crossover operator randomly selects two parent chromosomes (chromosomes with higher values have more chance to be selected) and randomly chooses their crossover points, and mates them to produce two child (offspring) chromosomes. We examine one and two point crossover operators. In one point crossover, the segments to the right of the crossover points are exchanged to form two offspring as shown in Figure 3(a) and in two point crossover [19][20], the middle portions of the crossover points are exchanged to form two offspring as shown in Figure 3(b).

E. To reduce the idle time of a processor waiting for the data from other processors a mutation operation is designed. It works by randomly selecting two tasks and swapping them. Firstly, it randomly selects a processor, and then randomly selects a task [21] on that processor. This task is the first task of the pair to be swapped. Secondly, it randomly selects a second processor (it may be the same as the first), and randomly selects a task. If the two selected tasks are the same task the search continues on. If the two tasks are different then they are swapped over (provided that the precedence relations must satisfy).

**Figure 3(a). One Point Crossover**
The phases of the Genetic Algorithm (GA) [30] are as follows:

**Step 1.** Read the DAG.

**Step 2.** Set the parameters.

**Step 3.** Initialize a list of chromosome having its p-size after selecting the chromosome randomly.

**Step 4.** Compute Fitness function on the basis of number of processor and number of tasks.

**Step 5.** Crossover swapping or Crossover Operation on the selected population.

**Step 6.** Mutation process or the mutation operation or swap mutation process on chromosome selected.

**Step 7.** At end apply the last heuristic operation.

**Step 8.** Last but not least testing procedure and mapping be done.

**IV. EXPERIMENTAL RESULTS AND PERFORMANCE ANALYSIS**

The performance comparisons with Shortest job First (SJF), First Come First Serve (FCFS) and Round Robin (RR) scheduling method [5] on parallel systems and execution of the schedules are shown in Figure 4 and 5 for task size equal to 10.

**Performance analysis:** Speed up ($T_{sp}$): Speed up[22] is defined as the completion time on a uniprocessor divided by completion time on a multiprocessor. In case of homogeneous system, it is denoted as: $T_{sp} = p(1)/p(m)$. But in case of heterogeneous system, it is denoted as $T_{sp} = (\min (p(1)) / p(m)$ i.e., the best uniprocessor completion time divided by the completion time on a heterogeneous multiprocessor system. The speedup is measured with the execution of tasks on single processor which shows 58 time units for task size equal 10 tasks divided by execution time units on PGA, SJF, FCFS and Round Robin (RR) scheduler as shown in Figure 5.

**Efficiency ($\epsilon$):** ($T_{sp} / m$), where $m$ is the number of processors.
Here, the Genetic Algorithm (GA) has been proposed for scheduling in homogeneous parallel multiprocessor systems to minimize the makespan including execution time and waiting or idle time and increase the throughput of the system. It is found that a better solution for assigning the tasks to the homogeneous parallel multiprocessor system. After the discussion, experimental results and Genetic Algorithm are compared with FCFS, SJF, and Round Robin (RR) scheduling methods. Finally, the performance study is based on the best randomly generated schedule of the Genetic Algorithm.

### V. Conclusion

![Performance Comparison](image)

**Figure 5.** Performance comparisons of the GA, SJF, FCFS, and RR for task size = 10 tasks

---

### VI. References

[12] David E Culler, “Parallel Computer Architecture”, Published by Morgan Kaufmann & Elsevier India.