Multiprocessor Task Scheduling Using Hybrid Genetic Algorithm

Yogesh R. Shahare
Department of Computer Engineering
yogeshshahare@gmail.com

Abstract – In multiprocessor system are widely used in parallel computing. In many cases we can divide a huge problem into small portion and assign these small problems to processors. Task scheduling is crucial and complex problem in multiprocessor system and is defined as a NP-hard problem. Nowadays, with increasing size of program and information to this algorithm is use task graph is various restriction of these problem is Independent task, dependent task and with and without communication time. This is important algorithm since there should be a balance between the quality of solutions and execution time of algorithm.

In this paper we present a new genetic algorithm that employs neighborhood search and tabu list for performing task scheduling it means that both the combination of genetic algorithm and tabu search algorithm. This newly proposed algorithm has solved the mentioned problem, as well as having a reasonable execution time versus the makespan and also we use to comparing the some other optimization technique for our proposed algorithm such as Tabu search, Genetic algorithm, Simulated Annealing which is the better solution to these problems. In this paper is proposed algorithm will be given a feasible solution to compare the all these algorithms and find the minimum makespan to solve the multiprocessor task scheduling problem.

I. INTRODUCTION

Scheduling for a set of dependent and independent task that execute parallely on a set of processors is important and computationally complex. Multiprocessor system have widely been used in parallel computation. This means that makespan must be minimum level. Another objective of the scheduling is to provide precedence relationship among the tasks when allocating them to processors.

In this context there are a lot of issues that should be addressed in terms of dependent or independent tasks, task graph produced randomly, with and without communication delay or homogeneity or heterogeneity of the multiprocessor system. In this paper we proposed a scheduling algorithm for allocation of tasks to set of processors which computes the communication cost between dependent tasks when these tasks were not executed in the same processor. In this paper we proposed a various comparing algorithm such as genetic algorithm and tabu search algorithm and Simulated Annealing and to find the better solution of these problems.

The result of this paper is shows that the genetic algorithm and tabu search due to their nondeterministic structure are appropriate candidate for NP-complete problems. Genetic algorithm have been proposed for solving such problems and each suggestion has some advantages and disadvantages. The main difference between these algorithm is on chromosome encoding have important effect on genetic operators. Doing a comparison between these algorithm is difficult because of some reasons: First majority of algorithm are based on Authors assumption and do not consider all aspects of the problem. Second there exist no standard benchmark for evaluation of algorithm.

II. TASK SCHEDULING PROBLEM

In this paper we formulate the scheduling problem. Let \( P = \{ p_i : i = 1, \ldots, m \} \) be a set of \( m \) homogeneous fully connected processors and let the application program be modeled by directed acyclic graph \( T = \{ T_j : j = 1, \ldots, n \} \) of \( n \) tasks. For any two task \( i, j \) \( T \) \( i \rightarrow j \) means that task \( j \) cannot be a schedule until \( i \) has been complete, \( i \) is a predecessor of \( j \) and \( j \) is a successor of \( i \). Weight associated with the nodes represent the computation cost and weight associated with edge represent the communication cost. An example of Directed acyclic graph (DAG) consisting of 11 tasks shown in Fig 1. The multiprocessor scheduling is to assign the set of tasks \( T \) onto the set of processor \( P \) in such a way that precedence constrain are maintained and to determine the start time and finish time of each task with the objective to minimize the completion time[1][2][4].
We assume that the communication system is contention free and it permits the overlap communication with computation. Task execution is started only after all the data have been received from its predecessor. The communication links is full duplex, duplication of same task is not allowed. Communication is zero when two task are assigned to the same processor, otherwise they incur the communication cost is equal to the edge weight.

![Fig 1. A Task Graph](image)

III. NEW GENETIC ALGORITHM BASED ON GENETIC AND TABU SEARCH ALGORITHM

In this study we proposed a new genetic algorithm and tabu search algorithm. In this Algorithm in order to accelerate evolutionary process and to reach an efficient Solution with a reasonable execution time, several techniques are used. Firstly the use of an initial population in which chromosome is produced in the basis of the earliest start time of each node (task)[5]. Secondly, using neighborhood search [7]. for those parents whose fitness value is more than 75% of mean population fitness. This is because the possibility of finding the best solution from doing genetic operators on these parents is higher than that of rest of population. Therefore this algorithm performs all the crossovers with considering an amax moves constant. Thirdly, a tabu list[1][7]. is used for preventing the repetition of the parents in the next generation whose children have been searched in the previous stage.

This prevention is because performing operation on such chromosomes will not produce children whose fitness is better than current generation. Furthermore, in order to perform a more precise evaluation of fitness value of chromosomes, another parameter other than makespan i.e. waiting time has been used. The coincident use of these parameters evaluates fitness of chromosomes more precisely. Another important factor in scheduling algorithms is the execution time of algorithm which should be scalable to solution quality. We have addressed this objective by restricting searches to those solutions which do not result in the considered value. Structure of proposed genetic algorithm is shown as below[1][9].

Step 1:
- Create initial population as randomly.

Step 2:
- While termination criteria not satisfied do
  - For each chromosome in current population do
    - Calculate its fitness value.
    - Create intermediate generation.
    - Add the fittest chromosome to the intermediate population.
- Repeat
  - For all chromosomes that fitnesses > 75% average fitness.
  - Apply all crossover for this chromosome considering amax moves.
  - and calculate fitness for these chromosome.
  - Replace the best chromosome in the population and put it parents to tabu list.
  - Apply tournament selection to selected two chromosomes.
  - Apply crossover operator.
  - Calculate fitness value.
  - Apply mutation and calculate fitnesses of this chromosome.
- Until the intermediate population size is complete.
- Copy the intermediate population over current population.

Structure of proposed genetic algorithm

Chromosome Encoding

In this algorithm, a new string encoding for chromosome is used which employs the advantage of clear global precedence of each task. In the chromosome encoding only the precedence of the task that are executed on the same processor are clear while the global precedence of task was not achievable and additional string should have been used for achieving the global precedence. The new string of the chromosome encoding is demonstrated in fig.2 as below.

![Fig 2. Chromosome Encoding for Task Graph](image)
Each element of array indicates the allocation of a task to the corresponding processor. One advantage of such an encoding is the constant length of chromosome during the genetic operation. Chromosome fitness value by incorporating two fields, S and W which indicate makespan and waiting time, respectively.

Fig represent the fitness value which is achieved through S and W fields. For instance fig.2. demonstrates an exemplatory chromosome for task graph demonstrated in fig.1.

**Production of Initial Population**
In this algorithm the node on the critical path as well its length are calculated. For each node a reference counter which initially equal the number of its parent is considered. Starting from the initial point of the queue, a task whose parent has finished execution is chosen randomly and allocated to a processor. Consequently reference belonging to all children of the chosen task is reduced by one mark. Thus reference count of the ready task equal to zero.

If the task does not belong to critical path it is allocated to a processor which the presented task on it has the maximum communication cost. In case there were several communication cost, one of them was chosen randomly. This algorithm tries to minimize the communication and produce a better solution. If all the members of initial population are produced in this algorithm, there will be possibility of premature convergence and repeated identical solutions. Therefore it will be better to produced some initial population randomly[1][6][7].

**Genetic Operators:** The operators of selection, crossover and mutation are described as follows.

**Selection Operator:** The selection operator used in this algorithm is the tournament algorithm.

**Crossover Operator:** In proposed algorithm for performing combination of two chromosome, the two point crossover method is used. This operator is performed on the basis of crossover rate (denoted by P_c). On this operator two point of chromosome is selected randomly. The interval between these point to are exchanged while the sequence of task remains intact. The selected point on each chromosome should be in identical position compared with other. This necessary so that the crossover operator can produce valid chromosome.

**Mutation Operator:** The mutation rate is denoted by P_m provides a variation and possibility of avoiding local optimum. If the task is selected to be mutated then its processor number will be randomly changed. Fig.4, shows an example of mutation.

Fig. 4. Performing Mutation

**Fitness Function:** In this algorithm to achieve a more precise fitness function, waiting time as well as makespan is considered. Waiting time is focuses on the time in which the task could be executed but is delayed due to scheduling policy. This waiting is either because of precedence of task graph, the way of allocating the task of processor or communication delay between the task. In other word no task can be executed. If two communicatory task do not executed on the same processor, then the communication has to be spent.

If the related chromosome is involve fitness of F_2, F_2,...,F_m and each one share a portion of 100% as C_2, C_2,...,C_m then general fitness can be considered as shown in equation 1.

\[
\text{Fitness} = \sum_{i=1}^{m} C_i F_i \quad (1)
\]

**IV. EXPERIMENTAL RESULTS**

In this study, the proposed algorithm is compared with genetic algorithm and Tabu search algorithm and Simulated Annealing Algorithm and find the minimum makespan which algorithm is called a better solution. In this paper is considered communication cost and computation cost. In this paper is implemented to solve the multiprocessor task scheduling problems is given a number of processor is assumed to be 4 and the various size of the task to the task graph is assumed the task is 15, 21, 28, 36. Therefore the proposed algorithm is implemented in java Genetic Library and Tabu Search Algorithm and Simulated Annealing Algorithm with java. Comparing to all these algorithm with the our proposed Algorithm and find the feasible solution of these problems and which Algorithm is the better solution of a Makespan and Execution time are as shown in the following graphs[1][8].
In above results shows a comparison of our proposed algorithm and the other Algorithm. In this paper is propose a combination of Genetic Algorithm and Tabu Search Algorithm (HGTS) and find the the makespan and execution time to the assign the processors to the task graph and compare the basic GA and basic Tabu Search Algorithm and Simulated Annealing algorithm. In this paper is given a better feasible solution of our HGTS algorithm as compared to other algorithm. Therefore to solve the multiprocessor task scheduling problems to find the minimum makespan and execution time.

V. CONCLUSION

Task graph scheduling is an NP-hard problem, therefore nondeterministic approaches e.g., genetic algorithm, tabu list and neighborhood search are applicable to this context. Therefore in this paper we study a new hybrid algorithm i.e. combination of genetic and tabu search algorithm. In this algorithm is capable of achieving an appropriate scheduling while spending less execution time since there should be balance between solution space and execution time of algorithm. It is based on reduction of communication cost between processors and neighborhood search for parents whose fitness value more than 75% of means current population fitness. Experimental result from implementation of this algorithm is indicated that the better solution with less execution time are feasible by a combination of tabu search and genetic algorithm. Comparison of HGTS and the other algorithm such as tabu search and simulated annealing and find the minimum makespan and execution time to solve the multiprocessor task scheduling problem.

Future work is the Consider the many applications attributed to scheduling problem, proposed algorithm can be adapted in structures such as mesh connected processors platform. This structure is famous and more usable and many algorithms has been implemented in such structures. Future work is comparing the another optimization algorithm and find the finish time i.e. finishing time and to solve a multiprocessor scheduling problems. It will use mesh structure and assign the different processor to each task and find the finish time.

VI. REFERENCES


