Enhancing Genetic Based Scheduling in Computational Grid with Bacterial Foraging Optimization Technique

Riya Mary Thomas, Joshua Samuel Raj, Dr. V. Vasudevan
1Dept. of Computer Science and Technology, Karunya University, India
2Dept. of Information Technology, KalaSalingam University, Srivilliputhur, India

Abstract
Mapping of jobs to the resources in grid environment is computed under high degree of heterogeneity of resources due to which heuristic and meta-heuristic approaches are the feasible methods to scheduling in grids. In this paper we enhance the genetic based scheduling in computational grids by a bacterial foraging optimization algorithm in grid environment. This paper analyzes the run-length unit parameter of BFO controls and the exploration of the whole search space and then exploits in the promising areas. The HGs scheduling method is improved in order to minimize the makespan and flow time with Bacterial Foraging Optimization algorithm to reduce the cost and complete the process submitted by client. By using this algorithm the scheduler will schedule the job to one of the available resources in which the job can be completed with minimal cost and can further reduce the delay.

Keywords
Bacterial Foraging Optimization (BFO), Execution Time to Compute, Hierarchical Genetic Algorithm (HGS), Makespan, Flowtime

I. Introduction
Grid computing combines computers from multiple administrative domains to reach a common goal to solve very high computational jobs. Grid computing provides a structure for exploiting these underutilized resources and thus has the possibility of substantially increasing the efficiency of resource usage. Grids enable the wide variety of resources including supercomputers, storage systems, data sources, and specialized devices that are geographically distributed and owned by different organizations for solving large-scale computational and data intensive problems in science, engineering, and commerce [1]. Grid computing provides a structure for exploiting these underutilized resources and thus has the possibility of substantially increasing the efficiency of resource usage. The grid is used to create the delusion of a simple yet large and influential self-managing virtual computer out of a large collection of connected heterogeneous systems sharing various combinations of resources. Scheduling is a challenging problem in a Grid environment due to its dynamic nature and the large number of resources to be managed and jobs to be scheduled [2]. Grid scheduling is defined as the process of making scheduling decisions involving resources over multiple administrative domains. The Genetic Algorithm (GA) is a heuristic method based on survival of the fittest. The GA was revealed as a useful tool for search and optimization problem. The GA handles a population of possible solutions [3]. Each solution is called a chromosome. Evaluating the fitness function makes the selection of chromosomes. The genetic algorithm first accepts a set of solutions and then constructs a set of neighbor solutions. A Genetic Algorithm (GA) is a search technique used in computing to find true or approximate solutions to optimization and search problems. GAs combines the utilization of past results with the examination of new areas of the search space. This algorithm provides effective, efficient techniques for optimization and machines learning applications. The efficiency of the GA depends upon an appropriate combination of exploration and exploitation. It is widely used in business, scientific and engineering circles. Three operators to complete this are: selection, crossover, mutation. Selection according to fitness is a source of utilization, and crossover and random mutations promote exploration [4].

Bacterial Foraging Optimization (BFO) is a population-based numerical optimization algorithm. BFOA has been widely established as a global optimization algorithm of recent interest in distributed optimization and control. Natural selection tends to eliminate animals with poor foraging strategies and favor the propagation of genes of those animals that have successful foraging strategies, since they are more likely to enjoy reproductive success. After many generations, poor foraging strategies are either eliminated or shaped into good ones [5].

II. Related Works
Several heuristic and meta-heuristic optimization methods have been proposed for Job Scheduling in computational grids such as Monte Carlo methods, Simulated Annealing (SA), Tabu Search (TS), Genetic Algorithms (GA), and Bacterial Foraging Optimization (BFO) and so on. Amongst these BFO, attempts to avoid the premature convergence to the local minima. GAs is well known for their robustness and has been applied successfully to solve scheduling problems in a variety of fields. The GAs approach for grid scheduling is addressed in several works. Various variants of genetic algorithms are taken. Panmictic Genetic Algorithms exploits the selection mechanisms that use large-scale information about the whole population to perform a global selection [6]. Distributed Genetic Algorithms (DGA) is executed in grid. The issues and problems are Scalability, Dynamic Changes and Heterogeneity [7]. The results of the DGA are not tempted very much by the dynamic reduction of the number of resources. This algorithm is only suitable for parallel execution because of the comparatively coarse-grained communication. The new evolutionary method (HGS) is used, which gives good computation complexity in solving global optimization problems by using different length genotypes. HGS enables a parallel search in the optimization field by many small populations. The creation of these population is governed by the dependent genetic process with low complexity.

Various Studies were performed in static benchmark for Independent Job Scheduling in distributed heterogeneous computing environment using Expected Time to Compute (ETC) model in order to observe numerous variations of GAs operators in order to identify a configuration of operators and parameters that works best for the problem. A Hierarchical Genetic Algorithm combined with bacterial foraging optimization has been successfully applied in solving continuous global optimization problems with multi-modal and weakly convex objective functions [8].
III. Problems Background

The purpose of the grid schedulers is to efficiently and optimally allocate tasks originated by applications to a set of available resources. Scheduling in grids remains a challenging NP-complete global optimization problem because of the large-scale heterogeneous structure of the system. The problem of scheduling a set of dependent or independent tasks in a distributed computing system is a well-advanced area. Some advantages of the dynamic techniques over static techniques are, the static techniques should always have a prior knowledge of all the tasks to be executed but dynamic techniques do not require that [9]. Traditional methods used in the optimization are deterministic, fast, and give exact answers but often tend to get stuck on local optima. The dynamic task scheduling also considers the load balancing issue as an NP-hard problem.

The objective is to minimize the total execution and communication cost encountered by the task assignment, subject to the resource constraints. A chromosome is evaluated by calculating its fitness function. Fitness function indicates the goodness of the schedule. The objective function calculates the total finishing time of the set of tasks allocated to each processor. The fitness function calculates the maximum of the total execution time of the set of tasks allocated to the processors. To efficiently and optimally allocate the tasks originated by applications to a set of available resources.

The makespan is defined as the routing that adds on to completion time of the last job in last machine, but it does not determine the waiting time for each job in the process [10].

\[
\text{Makespan} = \min_{S \in \text{Schedules}} \max_{j \in \text{Tasks}} F_j
\]

(1)

Where, \( F_j \) denotes the time when task \( j \) is finalized, Tasks denotes the set of all tasks submitted to the grid system and Schedules is the set of all possible schedules. The flowtime is expressed as the sum of finalization times of all the tasks. Calculated by,

\[
\text{Flowtime} = \min_{S \in \text{Schedules}} \sum_{j \in \text{Tasks}} F_j
\]

(2)

Flowtime has the higher degree order over makespan and its value increases as more jobs and machines are considered.

IV. Methodology

A, Hierarchical Genetic Algorithms (HGA)

The Genetic Algorithm (GA) was proposed by Holland as a heuristic approach based on the “survival of the fittest”. The GA was discovered as a useful tool for the search and optimization problem. HGA is a one of the variant of the genetic algorithm. It is a new evolutionary method which gives good computation complexity in solving global optimization problems by using different length genotypes [11]. HGS enables a parallel search in the optimization field by many small populations. Fig.1. shows the outline of the hierarchic genetic algorithm.

The creation of these populations is governed by the dependent genetic process with low complexity. The HGS is applied for independent job scheduling on computational grids for which both Flowtime and makespan parameters are simultaneously optimized. HGS performs well for instances representing features of reliable and semi consistent computing environments.

B, Bacterial Foraging Optimization (BFO)

Bacterial Foraging Optimization (BFO) is a recently developed nature-inspired optimization algorithm, which is based on the foraging behavior of E. coli bacteria [12]. Bacteria Foraging Optimization Algorithm (BFOA), proposed by Passino, is a new comer to the family of nature-inspired optimization algorithms [13].

BFO has an efficiency to solve the real-world optimization problems arising in several application domains. The poor foraging methods are either eliminated or formed into good ones. The activity of the foraging has led the researchers to use its optimization process [14]. The control system of the bacteria dictates how the foraging should proceed and can be subdivided into four sections; they are chemo taxis, swarming, reproduction, and elimination and dispersal.

In the dynamic case, the numbers of tasks and machines may vary over time. The capacity of the resources and the workload of tasks are randomly generated by a normal distribution. It is also assumed that every tasks submitted to the system must be scheduled and all machines in the system can be used. The Bacterial Foraging Optimization Algorithm is used for minimizing a cost function. It provides the pseudo code listing for the chemotaxis and swim behavior of the BFOA algorithm. A bacteria cost is derated by its communication with other cells. The fig.2 shows the general flow of the BFO scheduler.
The best solution is given to the mutation operator (if has not climbed down too long), the following steps are performed using reproduce from the mutation value obtained.

Step5: Mutation
Substep5.1: It contains already available solutions then
If \( m < C_s \), then mutate the solution
Else back
Substep5.2: Elitism
If \( B_{sol} < B_{reploid} \)
Then override the solution
Else move to step6
Step6: Reproduction: Move chromosomes to next population using reproduce from the mutation value obtained.

In this algorithm the uniform crossover operation enables the parent chromosomes to contribute the gene level rather than the segment level. After selecting the chromosome then apply the chemotaxis operation. This step performs two operations such as tumble and swimming. Store the chromosomal values in an array \( A \) in ascending order in that the least values are assigned as best solution \( B_{sol} \). The best solution is given to the mutation operator which truncates the values greater than the medium value. If \( m < C_s \) then mutate the solution. Also the solution is better than the previous one then replaces the solution as \( B_{reploid} \).

V. Experimental Results
The tasks are created based on the size selected by the user. The simulator schedules the problem instance and sends the job to the scheduler. Scheduler schedules the job based on the hierarchical genetic algorithm and calculates the population for \( n \) jobs. Then the tasks are processed in Batch mode. The problem is then formalized using the Expected Time to Compute (ETC) [15] matrix model. The instance of problem in the model is defined by the following:

- The estimation of workload of each task.
- The computing capacity of the each machine defined.
- The estimation of the prior load of each available machine in the scheduler.

The ETC matrix model is calculated as follows:

\[
ECC[t][i] = \frac{w_l}{c_e}.
\]

Where, \( w_l \) is the workload of tasks and is specified as the workload vector \( [w_1, . . . , w_n] \), \( c_e \) denoting the computational load of the task \( t \) and \( n \) is the total number of tasks. Also \( c_e \) denotes the computing capacity of machine \( i \) and \( m \) is the total number of machines. The computing capacity of the grid resources can be characterized by the vector \( CC = [c_1, . . . , c_e] \). Here the job will be submitted on machine \( i \) which has minimum ECC. Repeat the above process until all jobs are submitted to the available resources and also do the same using the bacterial foraging optimization technique. Then the cost function is evaluated basing on the below mentioned expression [16].

\[
j_{sw}(\theta, j, k, l) - j(\theta, j, k, l) + j_{ce}(\theta, j, k, l),
\]

\( \theta \) is the position of the bacterium and \( j \) represents the chemotactic step, \( k \) for reproduction, \( l \) for elimination-dispersal event. Here initially, all the values are initialized with \( j = k = l = \theta = 0 \).

The performance evaluation graph is based on the cost for processing in hierarchical genetic algorithm and the Bacterial Foraging optimization technique. Table 1 shows the results of comparing the costs of the HGS and BFO and also shows the performance graph in fig.3.

<table>
<thead>
<tr>
<th>No of jobs</th>
<th>Computing Cost of HGS</th>
<th>Computing Cost of Enhanced GA-BFO</th>
</tr>
</thead>
<tbody>
<tr>
<td>150</td>
<td>2000</td>
<td>1800</td>
</tr>
<tr>
<td>300</td>
<td>4000</td>
<td>3500</td>
</tr>
<tr>
<td>450</td>
<td>6000</td>
<td>5200</td>
</tr>
<tr>
<td>600</td>
<td>8000</td>
<td>7000</td>
</tr>
<tr>
<td>750</td>
<td>10000</td>
<td>8700</td>
</tr>
</tbody>
</table>

**Table 1: Results of Comparing the Costs of the HGS and BFO**
VI. Conclusion

In this paper, a bacterial foraging optimization technique is combined with the hierarchical genetic algorithm to improve the performance of the optimization of cost and then efficiently allocate the tasks. Compared to the hierarchical genetic algorithm the GA enhanced BFO reduced the cost. In the future we plan to include scalability by extending the algorithm and bringing in the ant colony optimization for the same.

References


[6] Travis Desell, Boleslaw Szymanski, Carlos Varela, "The Effects of Heterogeneity on Asynchronous Panmictic Genetic Search", Department of Computer Science, Rensselaer Polytechnic Institute, Troy NY 12180, USA.


Dr. V. Vasudevan, Senior Professor, Department of Information Technology, and Kalasalingam University received his Ph.D degree from Madurai Kamaraj University in the year 1992. He is the Project Director for the Software Technologies Group of TIFAC Core in Network Engineering. His areas of interest include Grid computing, Mobile Adhoc Networking, Multicasting and so forth.