International Journal of Science and Engineering Applications
Volume 4 Issue 3, 2015, ISSN-2319-7560 (Online)

Implementation methodology of Biogeography Based Optimization algorithm for dependent task scheduling

S.Selvi,
Associate Professor
Department of Electronics and
Communication Engineering,
Dr.Sivanthi Aditanar College of
Engineering, Tiruchendur -
628215,Tamilnadu, India.

Abstract: Biogeography Based Optimization (BBO) is a new evolutionary algorithm for global optimization that was introduced in 2008. BBO is an application of biogeography to evolutionary algorithms. Biogeography is the study of the distribution of biodiversity over space and time. It aims to analyze where organisms live, and in what abundance. BBO has certain features in common with other population-based optimization methods. Like GA and PSO, BBO can share information between solutions. This makes BBO applicable to many of the same types of problems that GA and PSO are used for, including unimodal, multimodal and deceptive functions. This paper explains the methodology of application of BBO algorithm for the constrained task scheduling problems.

Keywords: Biogeography Based Optimization, Constrained Task Scheduling, DAG, Makespan, Ranking,

1. INTRODUCTION

A task scheduling is the mapping of tasks to a selected group of resources which may be distributed in multiple administrative domains. A scheduling problem is specified by a set of machines, a set of jobs/operations, optimality criteria, environmental specifications, and by other constraints[1]. Given an application modeled by the Directed Acyclic Graph (DAG), the scheduling problem deals with mapping each task of the application onto the available heterogeneous systems in order to minimize makespan [2]. DAG includes the characteristics of an application program such as the execution time of tasks, the data size to communicate between tasks and task dependencies. The task scheduling problem has been solved several years ago and is known to be NP-complete [3,4]. In general, task scheduling algorithm for heterogeneous systems is classified into two classes: static and dynamic [5]. In static scheduling algorithms, all information needed for scheduling must be known in advance [6]. Static task scheduling takes place during compile time before running the parallel application. In contrast, scheduling decisions in dynamic scheduling algorithms are made at run time.

2. PROBLEM DEFINITION

The task scheduling problem is the process of assigning a set of *v* tasks in a DAG to a set of *q* computing nodes, which have diverse characteristics, without violating the precedence constraints. Before scheduling, the priority of execution of tasks is calculated based on the upward ranking methodology [4].The tasks are sorted in the decreasing order of the upward rank value. The highest priority task (with high rank value), has the highest scheduling priority. If more than one task has equal upward rank value, the scheduling priority of the task is decided randomly.

In this paper, the schedule length of the given DAG application, namely makespan, is the largest finish time among all tasks, which is the actual finish time of the exit task, n_{exit} .The objective of the task scheduling problem is to minimise the makespan (fitness), without violating the precedence constraints of the tasks. The objective function is defined in Equation (1) [4].

$$fitness = Makespan = f(x) = \begin{cases} EFT(n_{exit}), & for\ single\ \ n_{exit} \\ max\{EFT(n_{exit})\}, & for\ multiple\ n_{exit} \end{cases} \tag{1}$$

where EFT is the Earliest Finish Time of the task n_i on the computing node p_j, defined in the Equation (2) [4].

$$EFT(n_i, p_j) = w_{i,j} + EST(n_i, p_j) \tag{2}$$

where $EST(n_i, p_j)$ is the Earliest Start Time of the task n_i on the computing node p_j, defined in the Equation (3) ([4].

$$EST(n_i, p_j) = \begin{cases} max\{avail\_time(p_j), ready\_time(n_i)\}, & if\ n_i \neq n_{entry} \\ 0, & if\ n_i = n_{entry} \end{cases} \tag{3}$$

where $avail\_time(p_j)$ is the earliest time at which the computing node p_j is ready for the task execution and $ready\_time(n_i)$ is the time when all data needed by n_i has arrived at the computing node p_j, defined in the Equation (4) [4].$ready\_time(n_i) = \overset{max}{n_m \epsilon pred(n_i)} (EFT(n_m) + c_{m,i})$ (4)

where $pred(n_i)$ is the set of predecessor tasks of the task n_i.

3. BIOGEOGRAPHY BASED OPTIMIZATION ALGORITHM

Biogeography describes how species migrate from one island to another, how new species arise and how species become extinct. An island is any habitat that is geographically isolated from other habitats. Geographical areas that are well suited as residences for biological species are said to have a high Habitat Suitability Index (HSI). The variables that characterize habitability are called Suitability Index Variables (SIVs). SIVs can be considered as the independent variables of the habitat, and HSI can be calculated using these variables. Habitats with a high HSI tend to have large number of species, while those with a low HSI have a small number of species. Habitats with a high HSI have many species that migrate to nearby habitats, simply by virtue of the large number of species that they host. Migration of some species from one habitat to other habitat is known as emigration process. When some species enter into one habitat from any other outside habitat, it is known as immigration process[7]. The pseudo code for BBO algorithm is illustrated in Algorithm 1.

Algorithm 1. Biogeography Based Optimization Algorithm

```
Initialize the BBO parameters
Create a random set of habitats (population) $H_1, H_2, \ldots, H_n$.
Compute HSI values;
**While** the halting criterion is not satisfied **do**
    Compute immigration rate $\lambda$ and emigration rate $\mu$ for each habitat based on HSI;
    **For** each habitat (solution)
        **For** each SIV (solution feature)
                Select habitat $H_i$ with probability $\propto \lambda_i$
                **If** $H_i$ is selected **then**
                    Select $H_j$ with probability $\propto \mu_j$
                    **If** $H_j$ is selected then
                        $H_i(SIV) \leftarrow H_j(SIV)$
                    **End if**
                **End if**
                Select $H_i(SIV)$ based on mutation probability $m_i$;
                **If** $H_i(SIV)$ is selected **then**
                    Replace $H_i(SIV)$ with a randomly generated SIV;
                **End if**
            **Next for**
            Recompute HSI values;
    **Next for**
**End while**
```

4. IMPLEMENTATION OF BBO ALGORITHM FOR SCHEDULING DEPENDENT TASKS

The following subsections deal with the representation of solution, and the generation of initial solution.

4.1 Solution representation

The solution is represented as an array of length equal to the number of jobs [8]. The value corresponding to each position *i* in the array represent the node to which task *i* was allocated. The representation of the solution for the problem of scheduling 13 tasks to 3 computing nodes is illustrated in Figure 1. The first element of the array denotes the first task (n_1) in a batch which is allocated to the computing node 2; the second element of the array denotes the second job (n_2) which is assigned to the computing node 1, and so on.

J_1	J_2	J_3	J_4	J_5	…	J_i	…
G_2	G_5	G_9	G_1	G_7	…	G_j	…

(a)

2	1	2	3	1	2	3	1	2	3	2	1	1

(b)

Grid Node 1	J_2	J_5	J_8	J_{12}	J_{13}
Grid Node 2	J_1	J_3	J_6	J_9	J_{11}
Grid Node 3	J_4	J_7	J_{10}		

(c)

Fig. 1. (a) Solution Representation (b) Solution for the problem of 13 tasks and 3 computing nodes (c) Mapping of tasks with computing nodes for the solution given in (b)

4.2 Initial solution generation

Numerous methods have been proposed to generate the initial solution when applying meta heuristics to the scheduling problem in the heterogeneous environment[9,10]. Random solution may also be generated to initiate the process.

4.3 Computational experiments

To illustrate, a small scale DAG scheduling problem involving 3 nodes and 10 tasks is considered (Fig 2) with the computation cost matrix given in Table 1.The upward rank and the order of the tasks for execution are given in Table 2 and 3 respectively. BBO algorithm was executed with the following parameters. Habitat size-15, Habitat Modification Probability-1,Immigration probability bounds per gene-[0-1],Step size for numerical Integration-0.2,Maximum immigration and emigration rate for each island-1,Mutation probability-0.005, Number of iterations- 50. The makespan value obtained for the example problem is found to be 73.

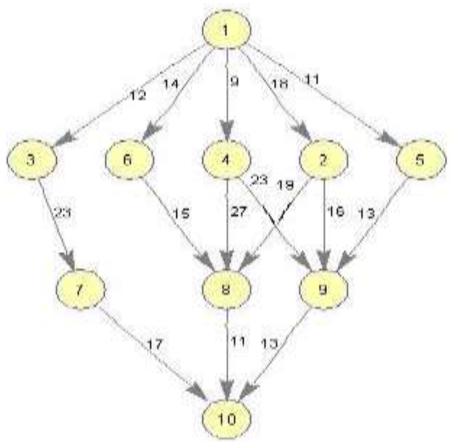

Figure 2. An example DAG

Table 1
Computation cost matrix for random DAG

Task id	P1	P2	P3
1	14	16	9
2	13	19	18
3	11	13	19
4	13	8	17
5	12	13	10
6	13	16	9
7	7	15	11
8	5	11	14
9	18	12	20
10	21	7	16

Table 2
Upward rank of the tasks

Task id	Upward Rank
1	108.00000
2	77.00000
3	80.00000
4	80.00000
5	69.00000
6	63.33333
7	42.66667
8	35.66667
9	44.33333
10	14.66667

Table 3
Execution Order of tasks

Order of Task id
1
4
3
2
5
6
9
7
8
10

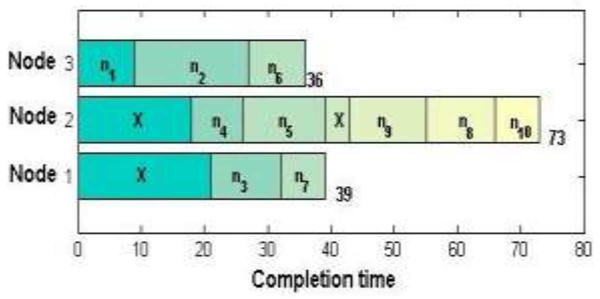

Figure 3. A schedule produced by the BBO algorithm for the example DAG

5. CONCLUSION

The methodology of implementation of BBO algorithm for the constrained dependent task scheduling problem had been discussed. The methodology adopted for this algorithm gives rise to the development of scheduling algorithms using other meta heuristic methods.

6. REFERENCES

[1] Selvi, S,Manimegalai, D.Research Journal of Applied Sciences, Engineering and Technology 8(8): 964-975, 2014

[2] P. Chitra, R. Rajaram, P. Venkatesh, Application and comparison of hybrid evolutionary multiobjective optimization algorithms for solving task scheduling problem on heterogeneous systems, Applied Soft Computing 11 (2011) 2725–2734

[3] E. Ilavarasan, P. Thambidurai, R. Mahilmannan, Performance effective task scheduling algorithm for heterogeneous computing system, in: Proc. 4th International Symposium on Parallel and Distributed Computing, France, 2005, pp. 28–38.

[4] Topcuoglu, H., Hariri, S., Wu, M.Y.: Performance-Effective and Low-Complexity Task Scheduling for Heterogeneous Computing. IEEE Trans. Parallel and Distributed Systems 13(3), 260–274 (2002)

[5] B.Hamidzadeh, L.Y.Kit, D.J.Lija, Dynamic task scheduling using online optimization, IEEE Trans. Parallel Distributed systems 11(11)(2000) 1151-1163.

[6] Mohammad I. Daouda, Nawwaf Kharma ,A hybrid heuristic–genetic algorithm for task scheduling in heterogeneous processor networks, J. Parallel Distrib. Comput. 71 (2011) 1518–1531

[7] Simon, D., (2008). Biogeography-based optimization. IEEE Transactions on Evolutionary Computation.12(6): 702–713.

[8] Selvi, S,Manimegalai, D. Task Scheduling using Two Phase Variable Neighborhood Search Algorithm on heterogeneous computing and grid environments, Arabian journal for science and engineering, March 2015, 40(3):817-844.

[9]Abraham A, Liu H, Zhao M. Particle swarm scheduling for work-flow applications in distributed computing environments. Studies in Computational Intelligence 2008; 128: 327–342.

[10] Xhafa F, Duran B, Parallel memetic algorithms for independent job scheduling in computational grids, in: Recent Advances in Evolutionary Computation for Combinatorial Optimization, vol. 153 of Studies in Computational Intelligence, Springer, 2008, pp. 219–239.