Reference Point Based Evolutionary Approach for Workflow Grid Scheduling

R. Garg and A. K. Singh

Abstract—**Grid computing facilitates the users to consume the services over the network. In order to optimize the workflow execution, multi-objective scheduling algorithm is required. In this paper, we considered two conflicting objectives of execution time (makespan) and total cost. We propose a scheduling algorithm, using Reference Point Based multi-objective evolutionary algorithm (R-NSGA-II), which provides the optimal scheduling solutions near the regions of user preference within the given quality of service constraints. The simulation results show the multiple solutions are obtained near each user specified regions of interest.**

Index Terms—**Multi-objective scheduling; DAG; grid computing; MOEA**

I. Introduction

With the rapid development of networking technology, grid computing [1] has emerged as a promising distributed computing paradigm that enables large-scale resource sharing and collaboration. Grid resources are normally highly dynamic and heterogeneous. One of the key challenges of heterogeneous systems is the scheduling problem. Scheduling of computational tasks on the Grid is a complex optimization problem, which may require consideration of different scheduling criteria. Usually, the most important are the task execution time, cost of running a task on a machine, reliability, resource utilization etc.

The optimization of scheduling problem is NP-complete, so numerous heuristic algorithms have been proposed in literature [2]. Many heuristics have also been proposed for workflow scheduling in order to optimize a single objective. Defining the multiple objectives for the task scheduling problem for generating efficient schedules at reduced computational times are of research interest in the recent days. In a multi-dimensional parameter space, it is in general not possible to find a solution that is best with respect to all the objectives, which makes the problem of requirements specification a real challenge. User may prefer the solution with slightly higher value for one objective but with large savings in other. With this motivation, this paper considers the two objectives for task scheduling keeping in view the tradeoff between two conflicting objectives of minimizing the makespan and total cost under the specified deadline and budget constraint.

We consider the preference set of solutions near the user specified regions of interest for that we used Reference Point Based Non-dominated Sort Genetic Algorithm (R-NSGA-II).

Rest of the paper is organized as follows. Section II specifies some of the related work. In section III, we introduced the grid scheduling problem formulation. Section IV, describes the technique of multi objective optimization and different multi objective evolutionary algorithms used. In section V we described the implementation details of MOEA for the problem of workflow scheduling. Section VI discusses the simulation analysis of proposed scheduling approach. Finally section VII gives the conclusion.

Manuscript received March 8, 2012; revised May 11, 2012.

The Authors are with the Department of Computer Engineering, National Institute of Technology, Kurukshetra, and Haryana, India-136119 (e-mail: ritu.59@gmail.com, aksinreck@rediffmail.com).

II. Related Work

The problem of Grid scheduling for directed acyclic graph (DAG) based task graph has already been addressed in the literature. Most of the related work considers execution time (makespan) and economic cost as two independent scheduling criteria. To schedule scientific workflow applications in Grid Computing environments, Heterogeneous Earliest Finish Time (HEFT) and Genetic Algorithms have been applied with extension by the ASKALON project [3]. The solution proposed by E. Tsiakkouri et al. [4] addresses a similar problem of bi-criteria budget-constrained workflow scheduling, by applying a two-phase optimization. Depending on which criterion is optimized in the first phase (either execution time or economic cost), one of two proposed versions of the scheduling algorithm is used (called LOSS and GAIN respectively). The work presented in [5] addresses tradeoff between execution time and reliability. The authors propose two workflow scheduling algorithms; one of which is called BDLS and the second one called BGA is a genetic algorithm. [6] Proposes a new bi-criterion workflow scheduling algorithm that performs optimization based on a flexible sliding constraint, and they apply a dynamic programming method to the entire workflow to enable an extensive exploration of the search space within the optimization process. The work presented in [7] proposes evolutionary algorithms as powerful heuristics which can address the general multi-criteria scheduling problems for workflows. The presented algorithms aim at providing a wide spread of alternative solutions (a Pareto set) rather than a single solution. It investigated and compared three major well known MOEA approaches to solve the workflow scheduling problem in grid. A. Talukder et al. [8] proposed a workflow execution planning approach using Multi-objective Differential Evolution (MODE) to generate a set of tradeoff schedules within the user specified constraints (deadline and budget), which will offer more choices to user when estimating QOS requirements.

Unlike the aforementioned work, we have proposed workflow scheduling using Referenced Point Based NSGA-II (R-NSGA-II) [9]. This approach delivers scheduling solutions in the multiple region of interest of user simultaneously and it also generates multiple trade-off schedules in each region that minimizes the execution time and cost in the given quality of service constraints.

III. Problem Definition

We define workflow Grid scheduling as the problem of assigning different precedence constraint tasks in the workflow to different available grid resources. We model the application as a task graph: Let G = (V, E) be a directed acyclic graph (DAG), with V as the set of n tasks $t_i \in V$, $1 \le i \le n$ and E is the set of edges representing precedence constraint among the tasks $e = (t_i, t_j) \in E$, $1 \le i \le n$, $1 \le j \le n$, $i \ne j$. Associated to each edge is the amount of data required to send from task t_i to t_j if they are not executed on the same resource. Let set R represent the m number of resources which are available in the grid and resource $r_j \in R$ is associated with two values: Time and cost of executing the task on resource r_j. Every task t_i has to be processed on resource r_j until completion. In our work, each solution is represented as two strings, the task assignment string and the scheduling order string. Task assignment string is the allocation of each task to the available time slot of the resource capable of executing the task and the scheduling order string encodes the order to schedule tasks. The order in the scheduling order string must satisfy the dependencies. We denote that time (t_i) is the completion time of task t_i and cost (t_i) is the total cost which include input data transmission cost and service cost for processing t_i. The execution optimization problem is to generate task assignment string S, which maps every t_i onto a suitable r_j to achieve the multi-objective below:

$$Minimize\ Time(S) = maxtime\ (t_i) \tag{1}$$

where $t_i \in V$ *and* $1 \le i \le n$

$$Minimize\ Cost(S) = \sum cost\ (t_i) \tag{2}$$

where $t_i \in V$ and $1 \le i \le n$

Subject to Cost(S) < B and Time(S) < D, where B is the cost constraint (Budget) and D is the time constraint (Deadline) required by users for workflow execution.

IV. Evolutionary Multi-Objective Workflow Grid Scheduling

A. Multi-Objective Optimization

In this paper we consider a bi-objective minimization problem with the task of minimization of makespan and cost. In a Multi-objective Optimization Problem (MOP), one solution that is the best with respect to all objectives may not be achieved. Usually the aim is to determine the trade-off surface, which is a set of non-dominated solution points, known as Pareto optimal solutions. Every individual in this set is an acceptable solution.

Mathematically a MOP can be formulated as follows:

$$Minimize\ f_i(x) \quad i = 1, 2, N_{obj} \tag{3}$$

where f_i is the i^{th} objective function, x is a decision vector that represents a solution; N_{obj} is the number of objectives.

MOP uses the concept of Pareto dominance which is defined as:

Definition (Pareto Dominance)

Consider the minimization problem. Let x_1 and x_2 are two decision vectors (solutions) from definition domain. Solution x_1 dominates x_2 if the following two conditions hold:

$$\forall i \in \{1, 2, \ldots . N_{obj}\} \rightarrow f_i(x_1) \le f_i(x_2) \tag{4}$$

$$\exists j \in \{1, 2, \ldots . N_{obj}\} \rightarrow f_j(x_1) \le f_j(x_2) \tag{5}$$

If x_1 dominates x_2 then x_1 is known as non dominant solution. The solutions that are non-dominated within the entire search space are denoted as Pareto optimal set or Pareto optimal front. The concept of Pareto optimal front is shown in Fig. 1.

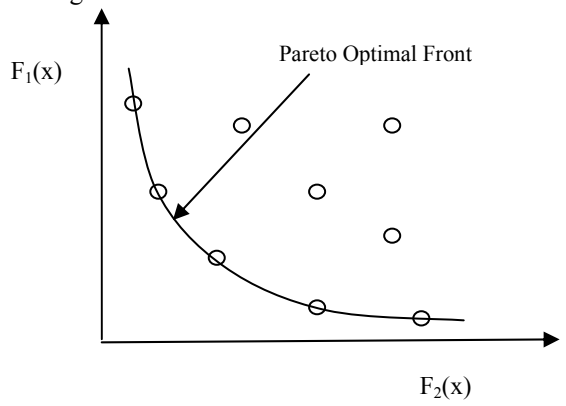


Fig. 1. Pareto optimal front

B. Multi-objective Evolutionary Algorithms

In this paper we are using the two well known multi-objective algorithms. These are Non Dominated Sort Genetic Algorithm (NSGA-II) and Reference Point Based Non Dominated Sort Genetic Algorithm (R-NSGA-II). It is an extension of NSGA-II with some user specified preferences or reference points.

The idea of elitist non-dominated sorting genetic algorithm (NSGA-II) was introduced in [10]. NSGA-II evaluates solutions based on the values of each objective by ensuring elitism and diversity among the solutions. In NSGA-II, initially a random parent population of size N is generated and sorted using fast non-dominated sort procedure. Each solution or schedule is assigned a fitness value using fitness function. The offspring population is then created by applying genetic operations i.e. selection, crossover and mutation on parent population. The procedure is different from the first generation onward; the parent populations and offspring population are combined to form population of size 2N in the t^{th} generation. Then, the non-domination sorting is applied. Since all previous best and current population members are included, so elitism is ensured. The new parent population is created by adding solutions from the first front (Best front). If size of thus created new parent population is

less than N, Then remaining members of population are selected from next non-dominated front and so on. This procedure is continued until no more fronts can be accommodated. Thereafter, last non-accepted fronts' solutions are sorted in descending order based on crowding distance measurement discussed in[10]and best solutions of the sorted front are included until size of population exceeds N. This algorithm is repeated until maximum number of generations M. NSGA-II has advantage that a set of solutions is generated while preserving uniform diversity among solutions but failed to generate solutions according to user choices.

In order to solve problems such as workflow scheduling where user has some preferences for solutions, R-NSGA-II [9] is more suitable. In R-NSGA-II, a user or decision maker simply provides some clues in terms of reference directions or reference points which represent the region of interest of the user. So the algorithm is able to generate the solutions in the region of user interest rather than wasting time to find other solutions which are not of user interest. Moreover, multiple reference points can be specified by the user. The generic overview of R-NSGA-II procedure is shown in Fig. 2.

1. generate initial parent population
2. **repeat**
3. generate offspring population from parent population by applying Selection, Crossover and Mutation operators
4. combine parent and offspring population
5. place each individual in its respective front by applying fast non-dominated sort on combined population
6. calculate preference distance of each front's individual using nitching strategy specified in Fig. 3
7. make new parent population by selecting individuals which are in better front and having least preference distance
8. **until** (maximum number of generations)

Fig. 2. Overview of R-NSGA-II procedure

In order to generate solutions near the user specified reference point, the preference operator in NSGA-II is used to select the subset of solutions from the last front which cannot be accommodated entirely to maintain the population size in the next population. Here preference distance is measured instead of crowding distance. The preference distance in the preference operator represents the closeness of solution from the user specified region. The modification performed is shown in Fig. 3 by R-NSGA-II nitching (diversity) strategy.

1. Assign rank to each solution with respect to each reference point according to calculated Euclidean distance (Eq. 6).
2. Determine preference distance of each solution by selecting minimum rank with respect to all reference points.
3. Make groups of solutions by applying ε-clearing idea and retain one random solution from each group.

Fig. 3. R-NSGA-II Nitching strategy

$$d_{x,R} = \sqrt{\sum_{i=1}^{Nobj} \left(\frac{f_i(x) - f_i(R)}{f_i^{max} - f_i^{min}} \right)^2} \quad (6)$$

where dx, R - normalized Euclidean distance from solution x to reference point R, Nobj - number of objectives, fimax - maximum value of ith objective in the population, fimin - minimum value of i^{th} objective in the population.

The ε-clearing idea specified in the Fig. 3 is used to control the spread of solutions near the preferred Pareto optimal regions and is based upon the concept of ε-dominance. This diversification between solutions is maintained by ε-value gap which implies the tolerance or precision for objective values specified by the user. The ε-value is chosen according the application and it may be different for each objective.

V. Implementation Of Multi-Objective Evolutionary Algorithms To Workflow Scheduling

Multi-objective Evolutionary Algorithms (MOEA) is well known meta-heuristic global search technique. This starts with a population of randomly generated candidate solutions and move towards the better set of solutions (Pareto optimal front) over number of generations. Its implementation for workflow task scheduling is as given below.

Step1 (Initialization): The chromosome consists of two parts: the task assignment string and scheduling order string. The task assignment string S is obtained by randomly assigning each task to available resource. It is a vector of length equal to V that represents the number of tasks. $S(i)=j$ means that i^{th} task is assigned to j^{th} resource. The random scheduling order string SS, which is also of size V, is formed by performing the topological sort of the DAG which signifies that the ordering of tasks obeys the precedence constraints. $SS(i)=k$ represents that i^{th} task is at the k^{th} position in the scheduling order.

Step2 (Selection): For the task assignment string, the combination of parent and child solutions is selected by two objective functions FTIME(S) and FCOST(S). These functions assume their values using (1) and (2) respectively by adding the penalty value if they go beyond the deadline constraint or budget constraint.

Step3 (Crossover): Crossovers are used to generate new solutions by rearranging the part of existing fittest solutions. We have used one point crossover for task assignment string. The probability of crossover has been set to 0.8.

Step4 (Mutation): For mutation single chromosome is selected to generate new chromosome that, possibly, is genetically better. We used the replacing mutation operator with the probability of mutation set to 0.5.

VI. Simulation Results and Discussion

We used GridSim [11] toolkit in our experiment to simulate the scheduling of workflow tasks. GridSim is a java based toolkit for modeling and simulation of resource and application scheduling in large-scale parallel and distributed computing environment such as Grid. It is flexible to support simulation of grid entities like resources, users, application tasks, resource brokers or schedulers and their behavior using

discrete events.

In our test environment, we simulated complex workflow applications consisting of 20 tasks on 8 virtual resources. Links between resources are established through a router so that direct communication can take place between resources. Computational rating (Million instructions per second) and computational cost (in dollars) of each resource is generated with non-uniform distribution. Communication baud rate between resources is specified in terms of Mbps. Number of data units required by one task from another task in the workflow are also generated non-uniformly.

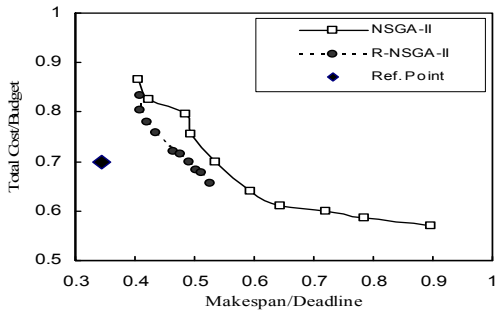
In order to generate valid schedule which can meet both deadline and budget constraints specified by the user, two algorithms HEFT[12] and Greedy Cost were used to make deadline and budget effectively. HEFT is a time optimization scheduling algorithm which gives minimum makespan (Timemin) and maximum total cost (Costmax) of the workflow schedule. Greedy Cost is a cost optimization scheduling algorithm which gives maximum makespan (Timemax) and minimum total cost (Costmin) of the workflow schedule Thus Deadline (D) and Budget (B) are specified as:

$$Deadline = Timemax - P\ (Timemax - Timemin) \quad (7)$$

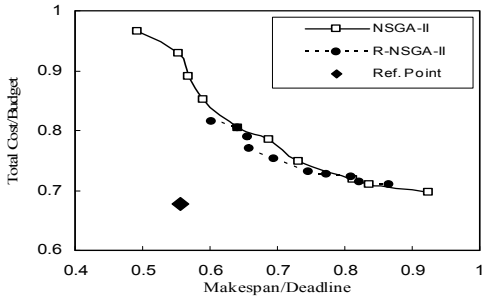
$$Budget = Costmax - P\ (Costmax - Costmin) \quad (8)$$

The value of parameter P can vary from 0.1 to 0.7 for both deadline and budget, we used 0.1, 0.4 and 0.7 to make loose, intermediate and stiff constraints respectively. Loose constraint means that the user requires large values for deadline and budget while stiff means small values respectively. We evaluated R-NSGA-II with single reference point and two reference points. The objective values for these reference points are also established using Timemax, Timemin , Costmax and Costmin in order to generate valid schedules (solutions).

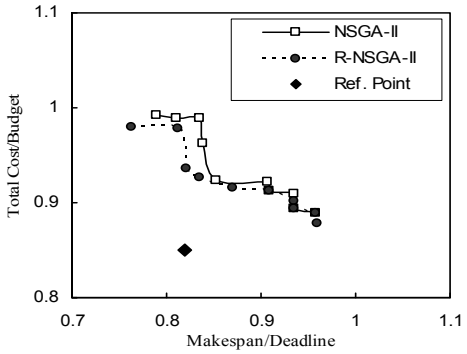
In order to compare the performance of algorithms NSGA-II and R-NSGA-II, we conducted our experiment over 10 runs and then optimal solutions were found by merging the non dominated solutions of each run. Fig. 4 shows the non dominated solutions obtained with NSGA-II and R-NSGA-II at different constraints. Fig. 4(a) and 4(b) shows, non-dominated solutions generated by R-NSGA-II are only in region of user interest rather than generation of wide spread non-dominated solution over the entire Pareto front as produced by NSGA-II at loose and intermediate constraints. But in Fig. 4(c) on stiff constraint, solutions obtained by both algorithms are in same region with slight difference because only a few non-dominated solutions exist with very small value of budget and deadline constraint.



a) Pareto optimal front at loose constraint



b) Pareto optimal front at intermediate constraint



c) Pareto optimal front at stiff CONSTRAINT

Fig. 4. Obtained pareto optimal solutions with two MOEA approaches on different constraint levels

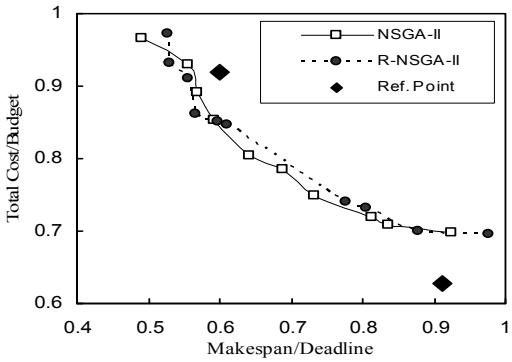


Fig. 5. Obtained pareto optimal solutions with two reference points

Further Fig. 5, shows solutions generated with NSGA-II and R-NSGA-II when user have more than one preference region i.e. two reference points. Here we get multiple solutions in each region of interest simultaneously with same computation time.

VII. Conclusion And Future Work

Here, we focused on the optimization of the workflow scheduling in a grid. Existing bi-criteria workflow scheduling algorithm generates the wide spread alternative solutions. However, we used the evolutionary algorithms approach to find more than one solutions near the user specified region of interest. With reference point based non dominated sorting algorithm (R-NSGA-II) we are able to generate the number of Pareto optimal scheduling solutions in the multiple regions of interest simultaneously and that minimize the total execution time (makespan) and cost along each desired region of interest of the user. In future work we will evaluate the scheduling approach using more than two

objectives simultaneously. We will also enhance the approach by using other optimization approaches.

REFERENCES

[1] R. Buyya and S. Venugopal, "A Gentle Introduction to Grid Computing and Technologies*," CSI Communications*, vol. 29, pp. 9-19, July 2005.

[2] T. D. Braun, H. J. Siegal, and N. Beck, "A comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems," *Journal of Parallel and Distributed Computing*, vol. 61, pp. 810-837, 2001.

[3] M. Wieczorek, R. Prodan, and T. Fahringer, "Scheduling of Scientific Workflows in the ASKALON Grid Environment," *SIGMOD*, vol. 34, no. 4, pp. 56-62, 2005.

[4] R. Prodan and T. Fahringer, "Dynamic Scheduling of Scientific Workflow Applications on the Grid: A case study," *SAC '05: Proceedings of the 2005 ACM Symposium on Applied Computing*, Santa Fe, New Mexico, ACM: New York, U.S.A, pp. 687–694, 2005.

[5] A. Dogan and F. Ozguner, "Biobjective Scheduling Algorithms for Execution Time-Reliability trade-off in Heterogeneous Computing Systems," *Comput. J*, vol. 48, no. 3, pp. 300-314, 2005.

[6] M. Wieczorek, S. Podlipning, R. Prodan, and T. Fahringer, "Bi-criteria Scheduling of Scientific Workflows for the Grid," *IEEE* 978-0-7675-3156-4/08, 2008.

[7] J. Yu, M. Kirley, and R. Buyya, "Multi-objective Planning for Workflow Execution on Grids," in *Proc. of the 8th IEEE/ACM International conference on Grid Computing*, 2007 ISBN: 978-1-4244-1559-5, doi. 10.1109/GRID.2007.4354110.

[8] A. K. Talukder, M. Kirley, and R. Buyya, "Multi-Objective Differential Evolution for Scheduling Workflow Applications on Global Grids," *John Wiley and Sons, Ltd., DOI: 10.1002/cpe.1417*, 2009.

[9] K. Deb, J. Sundar, U. R. Rao, and S. Choudhuri, "Reference Point Based Multi-Objective Optimization Using Evolutionary algorithms," *International Journal of Computational Intelligence Research*, vol. 2, no.3, pp. 273–286, 2006.

[10] K. Deb, A. Pratap, S. Aggarwal, and T. Meyarivan, "A Fast Elitist Multi-Objective Genetic Algorithm: NSGA-II," *Parallel Problems Solving from Nature VI*, pp.849-858, 2000.

[11] R. Buyya, "GridSim: A Toolkit for Modeling and Simulation of Grid Resource Management and Scheduling," [Online]. Available: http://www.buyya.com/gridsim

[12] T. Haluk, S. Hariri, and M. Y. Wu, "Performance-Effective and Low-Complexity Task Scheduling for Heterogeneous Computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, pp. 260-274, 2002.

R. Garg The corresponding author did M.Tech from Kurukshetra University, Kurukshetra. She is pursuing her PhD degree from N.I.T. Kurukshetra in the field of Grid Computing. Since 2002, she is teaching Computer Engineering Subjects. Presently she is working as an Assistant Professor in Computer Engineering Department at N.I.T, Kurukshetra, and Haryana, India.

A. K. Singh Presently the author is Associate Professor with Computer Engineering Department at N.I.T. Kurukshetra. He is having about 15 years of experience in the field of Computer Engineering. He is guiding a number of PhD's and M.Tech students. He has many research publications to his contribution. He has worked as Chairman, Computer Engineering Department at N.I.T. Kurukshetra.