International Journal of Business, Human and Social Sciences
ISSN: 2517-9411
Vol:2, No:2, 2008

Finding Pareto Optimal Front for the Multi-Mode Time, Cost Quality Trade-off in Project Scheduling

H. Iranmanesh, M. R. Skandari, and M. Allahverdiloo

Abstract*—**Project managers are the ultimate responsible for the overall characteristics of a project, i.e. they should deliver the project on time with minimum cost and with maximum quality. It is vital for any manager to decide a trade-off between these conflicting objectives and they will be benefited of any scientific decision support tool. Our work will try to determine optimal solutions (rather than a single optimal solution) from which the project manager will select his desirable choice to run the project. In this paper, the problem in project scheduling notated as (1,*T*|cpm,disc,mu|curve:quality,time,cost) will be studied. The problem is multi-objective and the purpose is finding the Pareto optimal front of time, cost and quality of a project (curve:quality,time,cost***), whose activities belong to a start to finish activity relationship network (***cpm)*** and they can be done in different possible modes (***mu***) which are non-continuous or discrete (***disc***), and each mode has a different cost, time and quality . The project is constrained to a non-renewable resource i.e. money (**1,*T***). Because the problem is *NP-Hard*, to solve the problem, a meta-heuristic is developed based on a version of genetic algorithm specially adapted to solve multi-objective problems namely FastPGA. A sample project with 30 activities is generated and then solved by the proposed method.

***Keywords*—**FastPGA, Multi-Execution Activity Mode, Pareto Optimality, Project Scheduling, Time-Cost-Quality Trade-Off.

I. Introduction

THE American National Standard Institution defines project management as “the application of knowledge, skills, tools and techniques to project activities to meet project requirements” [1]. Project scheduling, an integral part of project management, is intended to balance the competing objectives of a project, namely quality, time and cost while maintaining the project scope. The trio of project scheduling (see Fig. 1) and consequently the trade-off between them has been the subject of several studies so far.

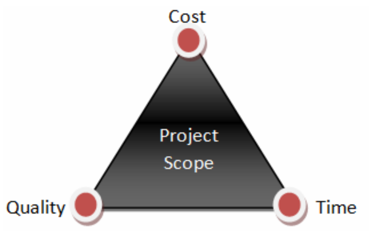


Fig. 1 The trio of project objectives

The critical path method (CPM) is a fundamental quantitative technique developed for project management. Assuming deterministic activity completion times, CPM determines the minimum time needed to complete the project.

In the management of a project, it is often possible to compress the duration of some of the activities at an additional expense in order to reduce the total project’s duration, and generally there is a due date (or in some cases called soft deadline) for project completion, so a decision problem considered in the project management literature is to determine the activities for crashing and the extent of crashing. By assuming that the direct cost of an activity varies with time (limited by normal and crash times), mathematical programming models were developed to minimize the project direct cost [2]. The problem is known as time/cost tradeoff problem in the literature. This problem was first studies by [3]. He assumed a linear relation between time and cost of an activity and offered a mathematical modeling and a heuristic algorithm (but one that leads to optimal solutions) for solving the problem. In 1961, Fulkerson [4] presented a solution method to find the curve of time-cost trade-off, i.e. for each time realization of the project, the best time and cost of each activity to minimize the total cost of the project subject to the given due date were found. In most practical cases, resources are available in discrete units, such as a number of machines, a number of workers and so on [5] .This kind of problem is called multi-execution mode for activities in the literature and the best execution mode (time, cost) of the activities should be determined to optimize an objective subjected to some constraints. An objective that was studied by [6] was to construct the complete and efficient time/cost profile over the set of feasible project durations. De [7] proved that the problem is strongly *NP-hard*. So, heuristic approaches were also presented.

H. Iranmanesh, Assistant Professor, "University of Tehran" & "Institute for Trade Studies & Research", Tehran, Iran (corresponding author, phone: +9821-88021067, fax: +9821- 88013102, e-mail: hiranmanesh@ut.ac.ir).

M. R. Skandari is with Socio-Economic Systems Engineering, University of Tehran, Tehran, Iran (e-mail: r.skandari@ie.sharif.edu).

M. Allahverdiloo is with Socio-Economic Systems Engineering, University of Tehran, Tehran, Iran (e-mail: morteza.allahverdilooi@gmail.com).

Babu and Suresh [2] were the first who suggested that the quality of a completed project may be affected by project crashing. For simplicity, they adapted the continuous scale from Zero to One to specify quality attained at each activity. The overall project quality is a function of quality levels attained at the individual activities. They developed optimization models involving the project time-cost-quality trade-off which would assist in expediting a project, weighing time-cost-quality trio. Each of the three models developed optimizes one of these three entities by assigning desired levels (bounds) on the other two.

Later, [8] assumed the duration and quality of project activities to be discrete, non-increasing functions of a single non-renewable resource (for example money). Three inter-related integer programming models were developed such that each model optimizes one of the given entities by assigning desired bounds on the other two. A solution method was presented in [9] for the above mentioned problem. A meta-heuristic solution procedure based on electromagnetic scatter search was developed to solve the problem.

In solving a problem with multiple objectives, different methods can be deployed such as considering a new objective defined by weighted sum of the multiple objectives or bounding all but one of the objectives and trying to optimize the selected one. One other approach is to find the Pareto optimal solution set. A solution is Pareto optimal, if there is no feasible solution, for which an improvement in one objective does not lead to a simultaneous degradation in one or more of the remaining objectives. The so called curve of time/cost trade-off is a good example of a Pareto optimal set (or better to say a Pareto optima front). In facing the trio features of a project and trade-off between them, a multi-objective problem, the second approach (bounding all but one of the objectives and trying to optimize the selected one) has been studied so far. In this paper we are trying to find the Pareto optimal set for the Time-Cost-Quality trade-off problem with the following assumptions:

- Activities can be done in different execution modes
- Each execution mode is a triplet of cost and quality and time by which they demonstrate the features of doing the activity in that mode

The problem notation based on the [10] will be (1,*T*|cpm,disc,mu|curve:quality,time,cost). As proved in [7], the easier problem, trade-off between time and cost is *NP-Hard*, so the trade-off between time, cost and quality is *NP-Hard* obviously. To solve the problem, a newly developed genetic algorithm, Fast Pareto Genetic Algorithm Approach, developed by Eskandari and Geiger [11], is deployed to solve the problem heuristically. The remaining parts of the paper are organized as the following: first the problem will be defined with details, then a mathematical model of the problem will be presented and then the solution method and a numerical example will be given.

II. Proposed Solution

A. Problem Definition

A project is defined as a directed acyclic graph *G= (V, E)* in which *V* is the set of nodes and *E* is the set of the arcs. In this case, the project is modeled by an activity-on-node network (AON) where its nodes represent project activities and its arcs are used to define precedents for activities. Each project activity, say $i \in V$, has *r(i)* different modes of execution of which mode *k* has t_{ik} time, c_{ik} cost and q_{ik} quality respectively.

It is assumed that if *k* and r are two alternatives for activity *i* such that $k \prec r$, then $t_{ik} > t_{ir}$ and $c_{ik} < c_{ir}$, but $q_{ik} > q_{ir}$. The objective is to construct the complete and efficient time, cost and quality profile to offer decision support in crashing a project. For example the following table shows an activity with four different modes:

TABLE I
EXECUTION MODE DATA OF A SAMPLE ACTIVITY

Mode	Time	Cost	Quality
1	*14*	*2772*	*0.95*
2	*24*	*2762*	*0.96*
3	*30*	*2750*	*0.9*
4	*69*	*2672*	*0.87*

B. Solution Representation

A solution to the problem is a set of integer values which shows the selected mode for each activity that is between 1 and the number of defined modes for that activity (1).

$$[m_1, m_2, ..., m_n] : m_i \in [1, M_i] \tag{1}$$

For example the following is a valid solution to a project with 12 activities and with a minimum of 5 modes for each activity:

[1, 2, 3, 1, 4, 5, 2, 3, 2, 4, 1, 3]

C. Mathematical Modeling

The mathematical model of problem is presented here:

Parameters :

n : number of activities

r(i) : number of modes for activity i

c_{ik} : cost of doing activity i in mode k

q_{ik} : quuality of activity i done in mode k

t_{ik} : time of doing activity i in mode k

Variables :

s_i : start time of activity i

$$x_{ik} : \begin{cases} 1 : \text{if activity } i \text{ is done in mode } k \\ 0 : \text{otherwise} \end{cases}$$

Equation 2 is the objective functions, (3) forces activities to select one and only one mode, (4) is used for the precedence relationships, and equations (5-6) are sign constraints.

$$\min\ s_n + \sum_{k=1}^{r(n)} x_{nk} t_{nk},\ \min \sum_{i=1}^{n} \sum_{k=1}^{r(i)} x_{ik} c_{ik},\ \max \frac{\sum_{i=1}^{n} \sum_{k=1}^{r(i)} x_{ik} q_{ik}}{n} \quad (2)$$

$s.t.$

$$\begin{cases} \sum_{k=1}^{r(i)} x_{ik} = 1 & \forall i \in V & (3) \\ s_i + \sum_{k=1}^{r(i)} t_{ik} x_{ik} \le s_j & \forall (i,j) \in E & (4) \\ s_i \ge 0 & \forall i \in V & (5) \\ x_{ik} \in \{0,1\} & \forall i \in V, k \in [1,2,\dots,r(i)] & (6) \end{cases}$$

D. Solution Method

Evolutionary algorithms such as genetic algorithm are proved to be able to balance exploration and exploitation of the solution search space. Other advantages of using EAs to solve Multi objective optimization problems include:

1. They explore the search space more accurately than point-to-point local search procedures such as simulated annealing and tabu search [12]
2. They can find a set of good solutions rather than a single solution (an intrinsic attribute of evolutionary algorithms) [13]
3. They are less dependent (and in some cases independent) on the selection of the starting solutions (and it is common to use a random scheme to create initial population), and they do not require definition of a neighborhood for searching the solution space [12]

Because of proved superiority of Fast Pareto Genetic Algorithm (FastPGA) over other alternatives to use multi-objective problems [11], we have deployed it as the solution method and will explain it in details. The superiority of FastPGA is most likely due to (according to [11]):

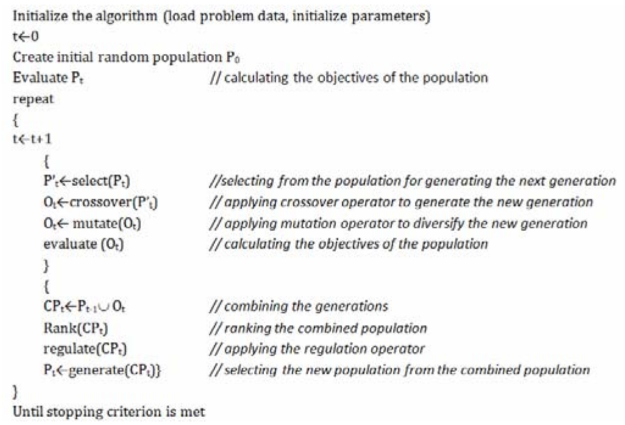
1. Employment of high intensity of elitism
2. Utilization of a new ranking strategy
3. Adaptive population sizing
4. Conservative solution evaluation scheme

Pseudo-code of the proposed fast Pareto genetic algorithm is illustrated in Fig. 2.

To define how our devised algorithm works, the operators of the FastPGA should be defined.

1) Chromosome Definition

To explain crossover and mutation operators, we first should identify the chromosome definition or in other words the coding system. We have used direct coding of the solutions rather than binary or other coding systems, because based on this type of coding and special definition of the operators, all the created solutions will be feasible and we do not need to do any repairs for the offspring solutions.



```
Initialize the algorithm (load problem data, initialize parameters)
t←0
Create initial random population P_0
Evaluate P_t                    // calculating the objectives of the population
repeat
{
t←t+1
    {
    P'_t←select(P_t)            //selecting from the population for generating the next generation
    O_t←crossover(P'_t)         // applying crossover operator to generate the new generation
    O_t← mutate(O_t)            // applying mutation operator to diversify the new generation
    evaluate (O_t)              // calculating the objectives of the population
    }
    {
    CP_t←P_{t-1}∪O_t            // combining the generations
    Rank(CP_t)                  // ranking the combined population
    regulate(CP_t)              // applying the regulation operator
    P_t←generate(CP_t)}         // selecting the new population from the combined population
}
Until stopping criterion is met
```

Fig. 2 Pseudo-code of the proposed fast Pareto genetic algorithm

2) Ranking Operator

Because we are facing multi-objective optimization problems with the approach of Pareto optimality, we need a new ranking strategy based on the classification of candidate solutions. This ranking system classifies the composite population CP_t into two different categories according to solution dominance i.e. non-dominated rank (first rank) and dominated rank (second rank). These ranks are used to evaluate solution fitness for the purpose of solution reproduction. The aim of this ranking strategy is to keep the diversity of the population as much as possible.

The fitness of the non-dominated solutions in the first rank is calculated by comparing each non-dominated solution with one another and assigning a fitness value computed using the crowding distance approach which helps maintain diversity among the non-dominated solutions in the Pareto optimal front. The greater the distance a solution is from its neighboring non-dominated solutions along the Pareto front, the larger the solution's fitness value. Briefly explained, the crowding distance of a set of solutions estimates *the density of the solutions surrounding any one particular solution in the population* (for more information about crowding distance operator we confer the reader to [11]).

Each dominated solution in the second rank is compared to all other solutions in the composite population and assigned a fitness value depending on the number of solutions it dominates and number of the solutions in the population that dominates the solution. More precisely, the fitness assignment takes into account both dominating and dominated solutions for any dominated solution. Here, each solution in the composite population CP_t is assigned a strength value $S(x_i)$, that indicates the number of solutions it dominates (equation 7).

$$S(x_i) = \left| \left\{ X_j \middle| \forall X_j \in CP_t \wedge X_i \succ X_j \wedge j \ne i \right\} \right| \quad (7)$$

So the fitness value of each dominated solution is equal to the summation of the strength values of all solutions it dominates minus the summation of the strength values of all solutions by which it is dominated. In other words:

$$F(x_i) = \sum_{x_i \succ x_j} S(x_i) - \sum_{x_k \succ x_i} S(x_k), \quad (8)$$
$$\forall x_j, x_k \in CP_t \wedge j \ne i \ne k$$

3) Regulation Operator

To make the search algorithm more efficient we need to adapt the population size during the search time. It is because the number of non-dominated solutions usually increases as the number of generations increases, which consequently results in low elitism intensity (ratio of non-dominated solutions to the population size) in early generations if the population size is quite large and kept fixed so it demands an adaptive population sizing strategy to place an appropriate emphasis of elitism intensity on the non-dominated solutions. If elitism intensity is too high, premature convergence (stopping of the algorithm while it has not found the optimal solutions) may occur, and on the other hand if elitism intensity is too low, convergence may slow down. Therefore, FastPGA employs a regulation operator to dynamically adjust the population size until it reaches a user-specified maximum population size as calculated by (9)

$$|P_t| = \min\{a_t + \lceil b_t \times |\{x_i \,|\, x_i \in CP_t \wedge x_i \text{ is nondominated}\}| \rceil \text{, maxpopsize}\} \quad (9)$$

In this study, we set a_t =20, b_t =1 and maxpopsize =100.

4) Crossover Operator

A simple two-point crossover operator is used here, in which two random numbers r_1 and r_2 are created that specify the position and length of the string that is to be swapped to create offspring, respectively. Since the parents' chromosomes involve a string of modes for activities so there are no repair requirements for preserving the feasibility and the offspring will be always feasible. For instance, the following two parents:

Parent 1	[1, 1, 3, 1, 1, 2, 2, 1, 3, 1, 3, 1]
Parent 2	[1, 2, 3, 1, 1, 2, 1, 2, 1, 3, 3, 1]

Result in the following offspring given that r_1 =2 and r_2 =7:

Offspring 1	[1, 1, 3, 1, 1, 2, 1, 1, 3, 1, 3, 1]
Offspring 2	[1, 2, 3, 1, 1, 2, 2, 2, 1, 3, 3, 1]

5) Mutation Operator

To search the solution space more thoroughly and diversely, a mutation operator is used here. The random integer operator is used for mutation. It works as the following: first a random number between 1 and total number of the activities is selected by random let say r_1, and then a set of random integers with r_1 members is created showing the activities that should undertake mutation. Then for each selected activity its mode will be generated by random in the permissible interval. For example, consider the following solution:

Raw Solution: [1, 1, 3, 1, 1, 2, 2, 1, 3, 1, 3, 1]

First r_1 is generated from [1, 12] let say 3, then a set of activities which their mode will be regenerated will be generated let say {1, 4, 11}. At last the mutated solution will be (the regenerated modes are 4, 3, and 2 respectively):

Mutated Solution: [4, 1, 3, 3, 1, 2, 2, 1, 3, 1, 2, 1]

III. Numerical Example

To test the performance of the algorithm, a network with 30 activities was used here. We have used the smart algorithm proposed in [9] to generate execution modes, smart because it does not produce dominated modes. The steps of the algorithm that creates modes are the following:

1) The number of execution modes per activity is randomly chosen from [2, 15]. Then the duration of each mode is randomly sampled from *DU*[10, 100]. After durations for all the execution modes are generated, they are sorted in ascending order, i.e.

$$tos_{ij} \equiv \{t_1, t_2, ..., t_{r(i)}; t_i \le t_{i+1} \; \forall i = 1...r(i) - 1\} \quad (10)$$

2) The corresponding cost for the first mode which is incidentally the maximum cost mode is sampled randomly from *DU*[1000; 5000]. The execution costs for the remaining modes are obtained by (11), in which s_i is from U(1,5), s_1=1 and c_i and t_i are the cost and the duration for mode *i* respectively.

$$c_i = c_{i-1} - s_i.(t_{i-1} - t_i) \quad (11)$$

3) It is assumed that quality attained by each activity under normal condition is 99%. To generate quality for other modes, it is assumed that the activity duration compressions do not adversely affect the activity qualities by the same magnitude. So we classified activities into five different categories. The quality attained by activities in categories 1–5 is allowed to vary between 0:95 … 0:99, 0:90 … 0:99, 0:85 … 0:99, 0:80 … 0:99 and finally 0:75… 0:99 respectively. To assign quality to the execution modes of activity *i,* we first randomly sample *DU*[1,5] to determine the category to which activity *i* belongs. We then take *r(i)* random samples from the quality range of the corresponding category.

After the problem is generated (both the network and the corresponding modes), we solved the problem by the proposed algorithm. The graph of the Pareto optimal points for the generated problem is depicted in Fig. 3. Note that, for coordinating the direction of objectives (we want to maximize the quality but minimize both time and cost of the project), we used the supplement of the quality (1-$q_{project}$). To make the solution's costs and times more tangible we then normalized cost and time in [0, 1] scale by dividing each objective of the solutions into its maximum attained by the Pareto optimal solutions.

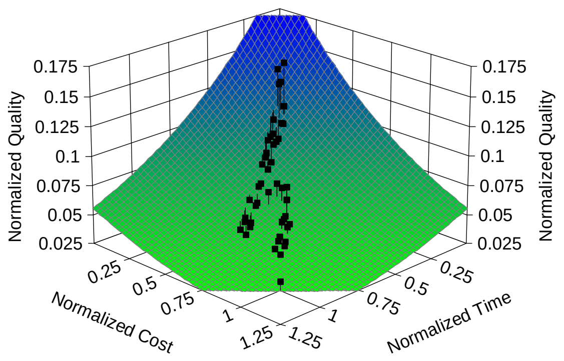


Fig. 3 Pareto Solution for the Sample Problem

IV. Conclusion and Further Work

In this paper, the problem in project scheduling notated as (1,*T*|cpm,disc,mu|curve:quality,time,cost) was studied. The problem is multi-objective and the purpose was finding the Pareto optimal front of time, cost and quality of a project, whose activities can be done in different discrete modes (execution modes), and each mode has a different cost, time and quality. To solve the problem, a meta-heuristic was developed based on a version of genetic algorithm specially adopted to solve multi-objective problems namely FastPGA. A sample project with 30 activities was generated and solved by the method. To challenge the solution method, a set of problems with different characteristics can be generated and the efficiency of the method can be compared with other meta-heuristics. Another suggestion to extend this work is to change the modes' nature from discrete modes to continuous ones which will make continuous fronts instead and will offer an easier way to assess the efficiency of solution method.

References

[1] Project Management Institute, "A guide to the project management body of knowledge", Third Edition, PMI Publisher, 2004, pp. 110-117
[2] Babu, A.J.G., and Nalina Suresh. "Project management with time, cost, and quality considerations." Journal of Operational Research 88, 1996: 320-327.
[3] Kelly, J. "Critical-path planning and scheduling: Mathematical basis." (Operations Research) 9, no. 3 (1961): 296-320.
[4] Fulkerson, D.R. "A network flow computation for project cost curves." (Management Science) 7, no. 2 (1961): 167-181.
[5] Demeulemeester, Erik l., and Willy S. Herroelen. "Project Scheduling: A research handbook". Kluwer Academy Publishers inc., 2002.
[6] Demeulemeester, E., S. E Elmaghraby, and W. Herroelen. "Optimal procedures for the discrete time/cost trade-off problem in project networks." Vol. 88. European Journal of Operational Research, 1996
[7] De, P., E. J. Dunne, J. B. Ghosh, and C. E. Wells. "Complexity of the discrete time/cost trade-off problem for project networks." (Operations Research) 45 (1997): 302–306.
[8] Tareghian, Hamed R., and Seyyed Hassan Taheri. "On the discrete time, cost and quality trade-off problem."Applied Mathematics and Computation 181 (2006): 1305–1312.
[9] Tareghian, Hamed R., and Seyyed Hassan Taheri. "A solution procedure for the discrete time,cost and quality tradeoff problem using electromagnetic scatter." Applied Mathematics and Computation 190 (2007): 1136–1145.
[10] Herroelen, W., E. Demeulemeester, and B. De Reyck. "A classification scheme for project scheduling problems." J. Weglarz (Ed.), Handbook on recent advances in project scheduling, 1999.
[11] Eskandari, H., and C.D. Geiger. "A Fast Pareto Genetic Algorithm Approach for Solving Expensive Multiobjective Optimization Problems." (Journal of Heuristics) in press (2006).
[12] April, J., F. Glover, J. Kelly, and M. Laguna. "Practical introduction to simulation optimization." Practical introduction to simulation optimization. Piscataway: Chick, S.(ed.), 2003. 71-78.
[13] Deb, K. "Multi-Objective Optimization using Evolutionary Algorithms." Chichester, UK: John Wiley & Sons, 2001.