THE DISCRETE TIME, COST AND QUALITY TRADE-OFF PROBLEM IN PROJECT SCHEDULING: AN EFFICIENT SOLUTION METHOD BASED ON CELLDE ALGORITHM

Gh. Assadipour[*1] and **H. Iranmanesh**

Department of Industrial Engineering
University of Tehran, Iran
[1]hiranmanesh@ut.ac.ir , [2]ghazal.assadipour@gmail.com

ABSTRACT

The trade-off between time, cost, and quality is one of the important problems of project management. This problem assumes that all project activities can be executed in different modes of cost, time, and quality. Thus a manager should select each activity's mode such that the project can meet the deadline with the minimum possible cost and the maximum achievable quality. As the problem is *NP-hard* and the objectives are in conflict with each other, a multi-objective meta-heuristic called CellDE, which is a hybrid cellular genetic algorithm, is implemented as the optimisation method. The proposed algorithm provides project managers with a set of non-dominated or Pareto-optimal solutions, and enables them to choose the best one according to their preferences. A set of problems of different sizes is generated and solved using the proposed algorithm. Three metrics are employed for evaluating the performance of the algorithm, appraising the diversity and convergence of the achieved Pareto fronts. Finally a comparison is made between CellDE and another meta-heuristic available in the literature. The results show the superiority of CellDE.

OPSOMMING

'n Balans tussen tyd, koste en gehalte is een van die belangrike probleme van projekbestuur. Die vraagstuk maak gewoonlik die aanname dat alle projekaktiwiteite uitgevoer kan word op uiteenlopende wyses wat verband hou met koste, tyd en gehalte. 'n Projekbestuurder selekteer gewoonlik die uitvoeringsmetodes sodanig per aktiwiteit dat gehoor gegegee word aan minimum koste en maksimum gehalte teen die voorwaarde van voltooiingsdatum wat bereik moet word.

Aangesien die beskrewe problem *NP-hard* is, word dit behandel ten opsigte van konflikterende doelwitte met 'n multidoelwit metaheuristiese metode (CellDE). Die metode is 'n hibride-sellulêre genetiese algoritme. Die algoritme lewer aan die besluitvormer 'n versameling van ongedomineerde of Pareto-optimale oplossings vir voorkeurgedrewe besluitvorming. Uiteenlopende probleme word opgelos deur die algoritme. Drie verskillende waardebepalings word toegepas op die gedrag van die algoritme. Die resultate bevestig die voortreflikheid van CellDE.

[*]Corresponding author

1. INTRODUCTION

Since the evolution of the critical path method, the time/cost trade-off problem has received more attention. There is a tradeoff between the cost and time of a project: solutions of shorter duration usually cost more, while solutions with low costs usually take longer [1, 2]. By *crashing* an activity in a project, its duration is reduced by allocating additional resources – money, for example – so a decision problem considered in the project management literature is to determine the activities for crashing and also the extent of crashing. In the discrete time/cost trade-off problem (DTCTP), activities can be executed in different sets of time and cost. In 1961, Kelley and Walker [3] proposed a technique based on a mathematical model. The model considered the assumption of linearity of the activity's utility function, and could be solved using network flow methods. For nonlinear cases, activities with a nonlinear utility function are replaced with a series of *r* activities having linear utility functions. An alternative to what Kelley suggested is the SAM algorithm, which was proposed by Siemens [4]. This algorithm was designed with the purpose of reducing project duration at the minimum cost when it exceeds the predefined due date. Although SAM is simple and less time consuming, there is neither a guarantee of achieving an optimal solution, nor a way to determine the optimality of the achieved solution. Goyal [5] reconsidered SAM, and permitted a shortening of the activities that had been previously shortened. The problem was proved to be *NP-hard* in [6], and subsequently heuristic approaches were offered to optimise the problem. Babu and Suresh [7] suggested that crashing a project might also affect its quality. They developed three linear programming models with the assumption that both cost and quality are linear descending and ascending functions of time respectively. Each model contains a single objective for optimisation, and constrains the values of the other two objectives within desired levels. They evaluated the project quality using arithmetic mean, geometric mean, and the minimum qualities of a project's activities. Evaluating the applicability of the technique proposed by Babu, the method was applied to an actual cement factory construction project in [8]. Analyzing the results, it was shown that there are budget thresholds at different quality levels of the time-cost curve. [2, 10 and 11] proposed meta-heuristic solutions for the discrete time, cost, and quality trade-off problem (DTCQTP). Combining electromagnetism theory with scatter search, Tareghian and Taheri [10] presented an approach to the inter-related integer programming models suggested in [9]. Rahimi and Iranmanesh [2] proposed a multi-objective PSO based algorithm, and compared it with a genetic algorithm (GA) for both small- and large-size problems.

In this paper we find an approximation to the Pareto front for DTCQTP using a multi-objective algorithm, and evaluate the performance using three different metrics. In contrast to other approaches, in which a single solution is offered to the Decision Maker (DM), here we suggest different alternatives, and give the DM the opportunity to choose the one that fits best according to his/her preferences. In the next section the problem is defined. The algorithm that is developed to produce the Pareto front is proposed in section 3. In section 4, three performance metrics are presented, and a comparison between the Pareto sets achieved by the implemented CellDE [13] and a version of FastPGA [11] is made in section 5. Section 6 concludes our work.

2. PROBLEM DEFINITION

A project usually consists of different activities, where activity i can be executed in n different modes. If the time, cost, and quality of mode j of activity are denoted by t_{ij}, c_{ij} and q_{ij} respectively, then in comparison with another mode k ($k > j$) of this activity we should have: $t_{ij} < t_{ik}$, $c_{ij} > c_{ik}$ and $q_{ij} < q_{ik}$. Table 1 shows a sample project with two activities that both can be executed in two modes. The project manager should select modes such that the project meets the deadline with the minimum possible total cost and maximum achievable quality. The mathematical model used here is similar to the one proposed in [11].

3. SOLUTION PROCEDURE

Multi-objective algorithms provide a set of non-dominated solutions. A solution is non-dominated if no other feasible solution is better with respect to all objectives, and moving from one non-dominated solution to another improves at least one objective but degrades one or more others. As evolutionary algorithms (EAs) deal with sets of solutions, they have been reckoned among the best alternatives for multi-objective problems. Their ability to handle complex problems, including those having features such as discontinuities, multimodality, disjoint feasible spaces, and noisy function evaluations, reinforces the potential effectiveness of EAs in multi-objective optimisation [13]. To solve the problem, a hybrid multi-objective cellular genetic algorithm called CellDE is implemented here. CellDE, developed by Durillo et al. [14], is a combination of MOCell, a multi-objective cellular genetic algorithm, and Generalized Differential Evolution 3, GDE3. It inherits good diversity from MOCell and convergence from GDE3 [14]. In the following, a brief description of cellular genetic algorithms as the base of CellDE, and the Differential Evolution method as the reproductive operator of CellDE, is presented.

Activity	Alternative 1			Alternative 2		
	Time	Cost	Quality	Time	Cost	Quality
1	10	2000	0.96	14	1800	0.98
2	11	2100	0.94	13	1900	0.99

Table 1: Sample problem data

3.1 Cellular GA

Cellular optimisation models are structured EAs in which the population is decentralised. Using the concept of neighbourhood, individuals only interact with other nearby individuals. In the primitive cellular genetic algorithm, individuals are structured in a grid of d dimensions (d=1, 2, 3), and a neighbourhood is defined on it. Genetic operators perform the exploitation, while exploration is done by means of overlapped neighbourhoods [15].

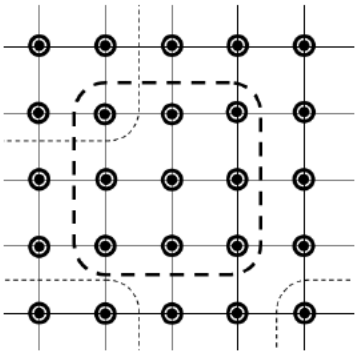

Figure 1: Cellular genetic algorithm

3.2 Differential Evolution

Differential Evolution is a parallel direct search method in which three parents collaborate to produce one offspring. All three parents are selected, using the selection operator, amongst the neighbours of the current individual. By adding the weighted difference vector of two of the parents to the third one, a new individual is generated [16]. The construction of the new individual in Differential Evolution is discussed further in 3.3.2.

3.3 CellDE

Similar to other cellular genetic algorithms, after the creation of an empty front, individuals are settled in a 2-dimensional toroidal grid. Then, from among the nearby neighbours of the current individual, two are chosen and the trio of parents is formed. The pseudo-code of the CellDE algorithm is illustrated in Figure 2. In CellDE, one of the parents is the original individual, while in DE all three members are chosen from among the neighbours of the current individual. The other two parents are chosen from eight neighbours of the current individual using the selection operator. The new offspring's vector is generated by adding a weighted difference vector between two of the parents to the third one (differentialEvolution function). If the original solution is dominated by the new individual, it will be replaced by the new one; but if both are non-dominated, then the neighbour with the worst crowding-distance is replaced (insert function). To compute the crowding-distance, all solutions are assigned a distance value. The boundary solutions (those that have the smallest or largest objective values) for each objective are set to infinite value, while the others are set to the normalised difference in the objective values of two successive solutions.

In the next step, the offspring is added to the external archive (addToArchive function). Through a feedback mechanism implemented after each generation, some of the individuals in the population are randomly selected and replaced by a set of solutions in the archive (replaceIndividuals function).

```
proc stepsUp (CellDE)                          //Algorithm parameters in
'CellDE'
population← randomPopulation ()                //Creates a random initial
population
archive← createFront ()                  //Creates an empty Pareto front
while !terminationCondition() do
    for individuals← 1 to CellDE.populationSize do
        neighborhood← getNeighbors(population, position(individual));
        parent1← selection(neighborhood);
        parent2← selection(neighborhood);
        // parent1 and parent2 might be identical
        while parent1=parent2 do
            parent2← selection(neighborhood);
        end while
        offspring← differentialEvolution(position(individual),
        position(individual),position(parent1), position(parent2));
        evaluateFitness(offspring);
        insert(position(individual), offspring, population);
        addToArchive(individual);
    end for
    population← replaceIndividuals(population, archive);
end while
end_proc stepsUp;
```

Figure 2: The pseudo-code of the CellDE algorithm

3.3.1 Chromosome representation

We used the direct coding to represent the chromosomes. As a consequence, the produced solutions are always feasible. Each chromosome's length is equal to the number of activities, and each allele's value is an integer limited to the number of modes of the related activity.

3.3.2 Reproductive operator

To produce new individuals, the differential evolution operator is used. This operator uses two parameters of *CR* and *F,* where *CR* is the crossover constant and *F* is the mutation's scaling factor. The pseudo-code of producing new individuals is illustrated in Figure 3, where *N*, *G*, *D* are the population size, generation, and dimension of solution respectively.

An example is presented next. Consider the following three parents:

Parent 1 [5, 6, 7, 4, 9, 5]
Parent 2 [1, 4, 1, 2, 5, 3]
Parent 3 [3, 5, 1, 2, 1, 3]

For *CR*= 0.5 and *F*= 0.5, and assuming all generated random values are less than *CR*, the offspring would be:

Offspring [5, 6, 4, 3, 3, 4]

3.3.3 Termination criterion

Different measures have been taken in the literature to set a stopping criterion, among which one may address the maximum number of generations, maximum CPU time, and/or convergence progress. The combination of two criteria is also employed in some cases; for example, the program stops if either the maximum generation or a designated value of the fitness is reached. In this paper, the maximum number of generations is considered as the termination criterion.

```
// r1, r2, r3∈{1, 2,…, N}, randomly selected, except
mutually different from i
proc differentialEvolution (i, r1, r2, r3)
jrand= floor(randi [0,1).D+1
for (j = 1; j ≤ D ; j = j+1) do
    if (randi [0,1) < CR ∨ j = jrand) then
        u_{i[j],G} = x_{r3[i],G} +F.( x_{r1[i],G} - x_{r2[i],G} )
    else
        u_{i[j],G} = x_{i[j],G}
    end if
end for
return u_{i,G}
end proc differentialEvolution
```

Figure 3: The pseudo-code of producing new individuals in DE

3.3.4 Parameterisation

Other considerations for the implemented CellDE are the following:

- Population size: 100 individuals. For more information about the selection of the population size, we refer the reader to [17].
- Neighbourhood: individuals located at North, East, West, South, Northwest, Southwest, Northeast, and Southeast of the current solution.
- Binary tournament selection: selects two individuals randomly. The one that is fitter is selected as a parent.
- Reproductive operator: differential evolution. Tests for real values in range [0, 1] lead to selection of 0.05 for *CR* and 0.95 for *F*. More explanation for the values of *CR* and *F* is presented in [18].

- Archive size: 100 individuals.
- Feedback: 20 individuals.
- Termination criteria: 100000 evaluations.

4. PERFORMANCE METRICS

Three metrics are implemented here for evaluating the performance of the developed algorithm. To calculate diversity and hypervolume, the Ideal and Nadir points should be calculated first. The vector that contains the best value of each objective in the objective space is considered as an Ideal point. The opposite of the Ideal point is the Nadir point, which contains the worst of objective values. In this problem, the Ideal point is a vector that contains the minimum possible time and cost, and the maximum achievable quality. As execution modes of activities are predefined, the Ideal and Nadir points can be calculated easily. A project completes at the minimum time when all the activities forming the critical path are executed in the modes with the minimum time, and it has the minimum cost when all the activities of a project are executed in the modes with the minimum cost. The same can be said about quality: a project has the maximum quality while all activities are executed in the modes with the maximum quality. A brief description of each metric is provided next.

4.1 Hypervolume metric

Zitzler and Thiele [19] introduced a metric called hypervolume, which measures the size of the space dominated by the Pareto front. Considering a cuboid between the Ideal and Nadir points, this indicator calculates the fraction of the cuboid that is dominated by the obtained non-dominated solutions. Since this metric is not free from arbitrary scaling of objectives, we evaluate this metric by using normalised objective function values. Furthermore, the achieved Pareto front is inverted, as we are minimizing cost, time, and the inverse of quality.

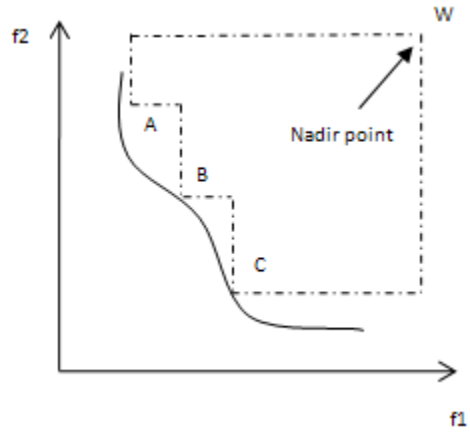

Figure 4: A hypervolume indicator in the two-objective case

4.2 Spread metric

Diversity metric Δ was first introduced by Deb et al. [20] and computes the non-uniformity of spread through the Pareto front. It is defined as Equation (1).

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} \left| d_i - \bar{d} \right|}{d_f + d_l + (N-1)\bar{d}} \tag{1}$$

where d_i is the Euclidean distance of two successive points. To define the successive points in 3-D space, the achieved front is normalised and then sorted through a lexicographical comparator. $\overline{d}$ is the average of distances, and d_f and d_l are the distances between bounding solutions. To define d_f and d_l, the Euclidean distance between the first solution of sorted front and the Ideal point, and the last solution of the sorted front and the Nadir point, should be calculated respectively. In well distributed fronts Δ is zero, because d_i is equal to $\overline{d}$ and $d_f = d_l = 0$, while in other cases Δ would be greater than zero.

4.3 Coverage metric

Comparing two Pareto sets achieved by the implemented CellDE and the FastPGA developed in [11], the coverage metric, first introduced in [19], is used (see Equation (2)). Zitzler and Thiele defined the function $C(X', X'')$, where X' and X'' are two sets of decision vectors, which calculates the percent of points in X'' that are dominated by at least a point in X'. In our problem a solution is dominated if there is at least one other individual with less cost and time, and greater quality.

$$C(X', X'') = \frac{\left|\{a'' \in X''; \exists a' \in X' : a' \geq a''\}\right|}{\left|X''\right|} \quad (2)$$

5. COMPUTATIONAL EXPERIMENTS

The CellDE algorithm was tested with multiple problems of different sizes, and the results were compared with the solutions obtained by the FastPGA [11], an adapted version of the original FastPGA [12] for the DTCQTP problem. Four types of networks, containing 30, 60, 90, and 120 activities respectively, were downloaded from psplib[1]. The procedure implemented to generate the execution modes is the same as the one used in [9]. Each experiment was repeated 40 times in order to restrict the influence of random effects. The parameter settings used for the developed algorithm are described in 3.3.4, while the parameters' values for the FastPGA are taken from the reference paper [11]. While the running times of two algorithms are similar, the results of the implemented performance metrics are presented in Tables 2, 3, and 4.

#Activity	CellDE	FastPGA
30	0.674±0.076	0.628±0.070
60	0.605±0.032	0.570±0.081
90	0.568±0.019	0.515±0.073
120	0.512±0.054	0.462±0.065

Table 2: Mean and standard deviation of the hypervolume metrics for the sample problems

#Activity	CellDE	FastPGA
30	0.466±0.076	0.662±0.051
60	0.511±0.069	0.716±0.022
90	0.639±0.027	0.742±0.039
120	0.670±0.051	0.744±0.026

Table 3: Mean and standard deviation of the spread metrics for the sample problems

[1] http://129.187.106.231/psplib

#Activity	C(X', X'')	CellDE	FastPGA
30	CellDE	-	24%
	FastPGA	4%	-
60	CellDE	-	24%
	FastPGA	9%	-
90	CellDE	-	21%
	FastPGA	5%	-
120	CellDE	-	22%
	FastPGA	6%	-

Table 4: Mean of the coverage metric for the sample problems

According to the size of the covered space, which is measured by hypervolume, CellDE is more efficient than FastPGA. About 59% of the space on average is dominated by CellDE, while this value is about 54% for FastPGA. The main conclusion that can be drawn from Table 2 is that the difference in hypervolume of CellDeE and FastPGA statistically proves the superiority of the Pareto front achieved by the proposed algorithm.

Regarding the spread metric, the difference between the values achieved by the two algorithms is meaningful, and the results indicate that CellDE outperforms FastPGA on all problems. The lower value of spread shows better distribution of the solutions; therefore solutions achieved by CellDE are spread broadly and more uniformly through the Pareto front.

In addition, on average 22.75% of solutions achieved by FastPGA are dominated by at least one solution obtained by CellDE, while the opposite is 6%, according to the coverage metric presented in Table 4. As mentioned earlier, a solution is dominated if there is at least another solution with greater quality but less cost and time.

6. CONCLUSION

In this paper, a hybrid multi-objective cellular genetic algorithm called CellDE was implemented to solve the time, cost, and quality trade-off problem. A set of randomly-generated problems of different sizes was generated and solved using CellDE and a meta-heuristic available in the literature called FastPGA. The Pareto solution set, which is achieved by minimising time and cost while maximising the quality of a project, gives the decision maker the opportunity to choose the best solution according to his/her preferences. Implementing three different metrics, the performance of the proposed algorithm was evaluated, and the results showed acceptable convergence and diversity of the obtained Pareto front. Then a comparison was made between the obtained Pareto set and the one achieved using FastPGA. The metrics demonstrated the comparative superiority of CellDE over the existing FastPGA for networks of different sizes.

7. REFERENCES

[1] **De, P., Dunne, E.J., Ghosh, J.B.& Wells, C.E.** 1995. The discrete time-cost tradeoff problem revisited, *European Journal of Operational Research*, 81, pp 225-238.

[2] **Rahimi, M. & Iranmanesh, H.** 2008. Multi objective particle swarm optimization for a discrete time, cost and quality trade-off problem, *World Applied Sciences Journal*, 4(2), pp 270-276.

[3] **Kelly, J.E. & Walker, M.R.** 1959. *Critical path planning and scheduling: An introduction*, Mauchly Associates, Ambler, PA.

[4] **Siemens, N.** 1971. A simple CPM time/cost trade-off algorithm, *Management Science*, 17, pp B-354–B-363.
[5] **Goyal, S.K.** 1975. A note on the paper: A simple CPM time/cost trade-off algorithm, *Management Science*, 21, pp 718–722.
[6] **De, P., Dunne, E.J., Ghosh, J.B. & Wells, C.** 1997. Complexity of the discrete time-cost tradeoff problem for project networks, *Operations Research*, 45, pp 302–306.
[7] **Babu, A.J.G. & Suresh, N.** 1996. Project management with time, cost and quality considerations, *European Journal of Operational*, 88, pp 320–327.
[8] **Khang, D.B. & Myint, Y.M.** 1999. Time, cost and quality trade-off in project management: A case study, *International Journal of Project Management*, 17(4), pp 249–256.
[9] **Tareghian, H.R. & Taheri, H.** 2006. On the discrete time, cost and quality trade-off problem, *Applied Mathematics and Computation*, 181, pp 1305–1312.
[10] **Tareghian, H.R. & Taheri, H.** 2007. A solution procedure for the discrete time, cost and quality tradeoff problem using electromagnetic scatter, *Applied Mathematics and Computation*, 190, pp 1136–1145.
[11] **Iranmanesh, H., Skandari, M.R. & Allahverdiloo, M.** 2008. Finding Pareto optimal front for the multi-mode time cost and quality trade-off in project scheduling, *International Journal of Computer, Information, and Systems Science, and Engineering* 2, pp 118-122.
[12] **Eskandari, H. & Geiger, C.D.** 2008. A fast Pareto genetic algorithm approach for solving expensive multiobjective optimization problems, *Journal of Heuristics*, 14(3), pp 203–241.
[13] **Fonseca, C.M. & Fleming, P.J.** 1995. An overview of evolutionary algorithms in multiobjective optimization, *Evolutionary Computation*, 1(1), pp 1–16.
[14] **Durillo, J.J., Nebro, A.J., Luna, F. & Alba, E.** 2008. Solving three-objective optimization problems using a new hybrid cellular genetic algorithm, PPSN 2008, pp 661-670.
[15] **Nebro, A.J., Durillo, J.J., Luna, F., Dorronsoro, B. & Alba, E.** 2007. Design issues in a multiobjective cellular genetic algorithm, *Evolutionary Multi-Criterion Optimization*, LNCS 4403, pp 126–140.
[16] **Storn, R. & Price, K.** 1995. Differential evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces, Technical Report TR-95-012, Berkeley, CA.
[17] **Neri, F. & Tirronen, V.** 2008. On memetic differential evolution frameworks: A study of advantages and limitations in hybridization, *Proceedings of the IEEE World Congress on Computational Intelligence*, pp 2135-2142.
[18] **Zielinski, K., Weitkemper, P., Laur, R. & Kammeyer, K.D.** 2006. Parameter study for differential evolution using a power allocation problem including interference cancellation, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp 1857–1864.
[19] **Zitzler, E. & Thiele, L.** 1999. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach, *IEEE Transactions on Evolutionary Computation*, 3(4), pp 257-271.
[20] **Deb, K., Pratap, A., Agrawal, S. & Meyarivan, T.** 2002. A fast and elitist multi-objective genetic algorithm: NSGA II, *IEEE Transactions on Evolutionary Computation*, 6(2), pp 182-197.