Swarm intelligence and evolutionary algorithms on an introduced novel type of time-cost trade-off problem

Hamed Naseri
Amirkabir University of Technology Department of Civil and Environmental Engineering
Amir Golroo (agolroo@aut.ac.ir)
Amirkabir University of Technology Department of Civil and Environmental Engineering
https://orcid.org/0000-0002-4222-9861
Amirhossein Fani
Amirkabir University of Technology Department of Civil and Environmental Engineering
Amir H. Gandomi
University of Technology Sydney

Research Article

Keywords: Time cost trade-off problem (TCTP), evolutionary algorithms, swarm intelligence, project delays

Posted Date: January 31st, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2406469/v1

Swarm intelligence and evolutionary algorithms on an introduced novel type of time-cost trade-off problem

Hamed Naseri[a], Amir Golroo[c*], Amirhossein Fani[b] Amir H. Gandomi[d]

[a] Ph.D. Student, Department of Civil and Environmental Engineering, Amirkabir University of Technology, Tehran, Iran. Tel: (1)514-2315853, Email: h.naseri@aut.ac.ir

[b] Ph.D. Candidate, Department of Civil and Environmental Engineering, Amirkabir University of Technology, Tehran, Iran. Tel: (98)912-2703662, Email: amir.fani@aut.ac.ir

[c*] Professor of Civil Engineering, Department of Civil and Environmental Engineering, Amirkabir University of Technology, Tehran, Iran. Tel: (98)21-64543010, Fax: (98)21-66414213, Email: agolroo@aut.ac.ir (Corresponding author)

[d] Professor of Data Science, Faculty of Engineering & Information Technology, University of Technology Sydney, NSW 2007, Australia. Tel: (61)2-95145081, Email: Amirhossein.Gandomi@uts.edu.au

* *Corresponding Author*

ABSTRACT

Time and cost are essential criteria for analyzing project feasibility. Project managers analyze the cost and duration of projects and make trade-offs between them before project initiation. During the implementation phase of a given project, a delay exists, making the initial plan impractical. Additionally, the contractor must pay a certain amount of money as delay fine based on the contract or spends extra money in order to reduce the duration of the project. This study proposes a new method to consider a trade-off between these two alternatives as a way to minimize the total time and the total extra money that should be paid. To this end, four strategies–minimizing costs, omitting delay under a minimum budget, minimizing cost and delay of the project simultaneously, and reducing the delay up to a particular level under a minimum budget–are taken into account to help decision-makers make the best decision. A case study is presented in this work, and 13 swarm intelligence and evolutionary algorithms are applied to find optimal solutions. A new index is developed and is used to compare various strategies and different algorithms. Based on the results, the introduced approach can reduce project costs and project delays by 28.8% and 85.7%, respectively. Moreover, the cuckoo search algorithm, invasive weed optimization, coyote optimization algorithm, and differential evolutionary algorithm outperform the other algorithms based on outcomes and the Tukey pairwise comparison results. Furthermore, the firefly algorithm is recognized as being the fastest algorithm for solving a delay time-cost trade-off problem.

KEYWORDS

Time cost trade-off problem (TCTP), evolutionary algorithms, swarm intelligence, project delays

1 INTRODUCTION

Cost is one of the most critical criteria of project, and affects all projects significantly. Cost is considered to be one of the sustainability criteria. That is to say, reducing the required cost of projects enhances sustainable development (Naseri et al. 2020). Likewise, the duration of projects (time) is an essential criterion used for analyzing project feasibility. Therefore, project managers and decision-makers analyze the cost and duration of projects meticulously and make trade-offs between them before a project starts. Subsequently, they may opt for the best alternative as the project schedule. This process can lead to completing a project on time with a minimum budget. Additionally, unpredictable events can postpone certain project activities, which may cause a delay in the timetable. Accordingly, this paper introduces a novel methodology that can be used to compensate for the influence of delays in projects.

Various techniques have been used to solve time-cost trade-off problems (TCTPs). These methods can be classified into two groups: mathematical programming and metaheuristic algorithms. Hindelang and Muth (1979) proposed a dynamic programming method that is used to find the optimal solution for a TCTP. Hindelang and Muth (1979) used the critical path method (CPM) to evaluate the total duration of the project, and the minimization of total cost is taken into account by the problem's objective function. Burns et al. (1996) stated that linear-based algorithms, such as the simplex method, can provide optimal solutions for TCTPs. Hafezalkotob et al. (2018) used a cooperative game theory for trade-off problems associated with project resources and total duration.

The TCTP is an integer programming problem, so increasing the dimension of this problem makes it NP-hard. Mathematical methods such as linear programming algorithms and dynamic programming cannot solve the aforementioned problem. To this end, swarm intelligence and evolutionary algorithms have become appropriate approaches to tackle the high-level complexity of large-scale TCTPs.

Agdas et al. (2018) used a genetic algorithm to solve a large-scale TCTP. The results of this investigation indicated that the genetic algorithm is highly effective at solving large-scale construction TCTPs. Total project duration, resource moment deviation, and cost were simultaneously considered in a study conducted by Ghoddousi et al. (2013). The non-dominated sorting genetic algorithm (NSGA-II) was employed to search for optimal solutions, which provided a suitable level for all of the objectives simultaneously. According to the outcomes of the Ghoddousi et al. (2013) study, the NSGA-II is highly qualified for finding optimal solutions for TCTP, which leads to a reduction in project cost and duration and saves resources. Additionally, the genetic algorithm has been applied to analyze

TCTP under different conditions. The results of these investigations revealed that the genetic algorithm can solve various forms of TCTP and provides valuable optimal solutions (Leu et al. 2001; Lo and Kuo 2011; Monghasemi et al. 2015). Aminbakhsh and Sonmez (2016) applied a discrete particle swarm optimization technique to address the high-level complexity of large-scale TCTPs. The discrete particle swarm optimization technique was able to solve a large-scale TCTP within seconds, and it arrived at the desired solutions. The ant colony optimization algorithm has been utilized to solve TCTP problems (Afshar et al. 2007). Similarly, this algorithm can provide appropriate solutions and it can be deduced that various forms of metaheuristic algorithms can tackle the high-level complexity of large-scale TCTPs (Afshar et al. 2007). Although metaheuristic algorithms have been qualified to solve various forms of TCTP, the application of novel evolutionary and swarm intelligence algorithms has not received enough attention in the context of TCTP.

Traditional time-cost trade-off analysis assumes that time, cost, and resource consumption of an option within an activity are deterministic. However, in reality, these parameters are uncertain. Thus, uncertainties should be considered when analyzing the TCTP, and time–cost optimization decisions should be analyzed in a more flexible and realistic manner (Chung-Wei Feng, Liang Liu 2000; Zheng and Ng 2005; Eshtehardian et al. 2009; Kalhor et al. 2011; Ke 2014). Although uncertainty has been taken into account in TCTP, an approach that prevents the detrimental influences of project delays is lacking.

As described in the above references, a time-cost trade-off analysis has been previously used to schedule projects before starting them. The predicted cost and duration of projects are not deterministic. Hence, uncertainty is part of all projects, and the majority of project delays are unavoidable. Delay analysis was neglected in previous studies, although it is an immense concern. To this end, this study introduces a novel approach that helps to make an optimal decision regarding the circumstances that the project does not complete before the deadline. With the aid of proposed method, project management can make optimal decisions dynamically in different implementation phases of projects.

Consequently, four strategies are introduced and these strategies are compared by proposing a novel index. Furthermore, the application of novel and robust metaheuristic algorithms has not been considered for TCTP problems, and comparing the ability of various metaheuristic algorithms has been overlooked. Hence, 13 metaheuristic algorithms, including water cycle algorithm (WCA), invasive weed optimization (IWO), coyote optimization algorithm (COA), soccer league competition algorithm (SLC), ant colony optimization (ACO), particle swarm optimization (PSO), salp swarm algorithm (SSA), marine predators algorithm (MPA), firefly algorithm (FA), cuckoo search algorithm (CS), differential evolutionary (DE), genetic algorithm (GA), and covariance matrix adaptation evolution strategy (CMA-ES) are utilized to prevent the effects of delay on the project by consideration of various strategies. The performance of these algorithms is compared in order to determine their effectiveness and to introduce the best algorithm among these alternatives. This comparison can help decision-makers use the most precise methods to solve the TCTP and to obtain better solutions.

2 OBJECTIVES AND SCOPE

During project implementation, the implemented time of activities and their planning time on timetable should be compared in order to analyze the physical improvement and status of projects. Hence, monitoring and inspecting times are required for most projects. TCTP can be used after each monitoring activity because, after checking out the projects, the critical (most time consuming) path may be changed due to delays. Moreover, the total time of a project may be increased, and TCTP can be utilized to reduce the duration of activities. Employing extra workers, increasing the number of work shifts, and using more efficient equipment are common methods for performing each activity in less time. Hence, this study introduces a new technique to reduce delays and their corresponding negative effects on projects. That is, this paper considers a case study that measures the effectiveness of TCTP at compensating for the impact of a delay on the project. The goal is to balance delay (time) with paying the delay fine (cost). Additionally, the performance of various metaheuristic algorithms is evaluated in order to identify the most valuable algorithms to solve TCTP.

3 METHODOLOGY

Initially, the project's improvement and its timetable are compared in order to identify the existence and the status of a delay. This step categorizes paths into delayed and normal paths. Normal paths are the paths that are finished before the project's deadline, and the completion time of the project cannot be changed by condensing their time. Accordingly, normal paths are not considered in delay analysis. Conversely, delayed paths are paths that are completed after the deadline, and they are a major cause of delay in the project. Therefore, delayed paths and the activities associated with them (delayed activities) are detected. As previously mentioned, the duration of activities can be decreased by increasing resources.

Accordingly, various feasible modes of implementation for each delayed activity are recognized by previous data and resource analysis.

Different strategies are subsequently considered in order to meticulously analyze the model. Strategies consist of different goals, and each of them can be selected according to the situation and the company's purposes. Minimizing total cost, omitting delay, reducing time and cost simultaneously, and reducing delay up to a certain level are the strategies investigated in this study.

The model is subsequently solved by 13 algorithms, which allowed for the most valuable algorithm to be identified. Additionally, the algorithms are compared based on their convergence speed, ability to find the optimal solution, and their efficiency. In this paper, water cycle algorithm (WCA), invasive weed optimization (IWO), coyote optimization algorithm (COA), soccer league competition algorithm (SLC), ant colony optimization (ACO), particle swarm optimization (PSO), salp swarm algorithm (SSA), marine predators algorithm (MPA), firefly algorithm (FA), cuckoo search algorithm (CS), differential evolutionary (DE), genetic algorithm (GA), and covariance matrix adaptation evolution strategy (CMA-ES) as robust evolutionary and swarm intelligence algorithms are utilized to solve delay time-cost trade-off problem (delay TCTP). Delay TCTP is a new type of time-cost trade-off problem introduced in this study that attempts to compensate for the negative impacts of delay on projects.

Ultimately, a novel equation (improvement) was developed that was used to analyze the value of each solution. The optimal results produced by various algorithms and different strategies are compared with this equation, and the best solution was identified. The steps of the methodology introduced in this investigation are shown in Figure 1.

Insert Figure 1

3.1 Proposed model

On the day of monitoring, the contractor can spend additional money on extra resources to reduce the delay fine, if it is economical. This has two useful aspects: the total money is reduced, and the contractor's credit and prestige are not tarnished. Each activity i has m_i modes. The time and cost pair of activity i for its v^{th} mode is $(t_{i,v}, c_{i,v})$, where $t_{i,v}$ and $c_{i,v}$ are associated time and associated cost, respectively. For each two modes $(i, v1)$ and $(i, v2)$, it is assumed that $t_{i,v1} > t_{i,v2}$ implies $c_{i,v1} < c_{i,v2}$; i.e., shorter durations require extra resources and, accordingly, higher costs. Furthermore, $v1 < v2$ implies $t_{i,v1} > t_{i,v2}$ for all i; that is, the activity modes are indexed according to decreasing order of duration (Hafızo˘glu and Azizo˘glu 2010). The decision variable of this model is as follows:

$$y_{i,v} = \begin{cases} 1 & if\ activity\ i\ is\ assigned\ to\ mode\ v \\ 0 & otherwise \end{cases} \quad (1)$$

Each activity is allowed to be assigned to exactly one mode. Thus:

$$\sum_{v=1}^{m_i} y_{i,v} = 1 \qquad i = 1, 2, \ldots, Z \quad (2)$$

Moreover, in each path, time is the sum of all activities located in that path. Accordingly:

$$T_{Path} = \sum_{v=1}^{m_i} \sum_{s=1}^{Z} y_{s,v} \times t_{s,v} \quad (3)$$

where T_{Path} is the total time of each path and s are the activities that are located in the path. The duration of the project is equal to the most time-consuming path. The extra cost that should be paid for a project delay can be calculated based on the following equation (Naseri 2018):

$$Extra\ Cost = \sum_{v=1}^{m_i} \sum_{i=1}^{Z} y_{i,v} \times c_{i,v} + U_{Pun} \times \left[Max\left(T_j\right) - T_{deadline}\right] \qquad i = 1, 2, \ldots, Z \quad (4)$$

where U_{Pun} is the unit delay fine, which should be paid for each unit time (day) of delay. $T_{deadline}$ is the deadline of the project. $c_{i,v}$ is the cost of each activity i for its v^{th} mode. Z is the number of remaining activities in the delayed paths. T_j is the duration of delayed paths.

3.2 Algorithms process

As previously mentioned, one of the aims of the current study is to compare the effectiveness of evolutionary and swarm intelligence algorithms for solving a delay TCTP. To this end, 13 algorithms were applied. The process of these algorithms is briefly described in this section.
The WCA is a meta-heuristic algorithm and was introduced by Hadi Eskandar et al (2012). This algorithm was inspired by the water cycle in the environment and how water flows from streams and rivers to seas (Sadollah et al. 2015b). In this algorithm, initial data are considered to be raindrops and the best data is associated

with the sea. A number of good raindrops are selected as rivers and the remaining data are considered to streams that flow into rivers and seas. That is to say, each solution vector is associated with a raindrop. Next, the solution vectors are ranked based on their fitness value, and the weaker solution vectors move toward the most valuable solutions. If the distances of the sea and streams are reduced to a particular level, the evaporation operator is performed. The evaporation operator prevents the solution vectors from accumulating in local-minimum areas. Consequently, a rain operator is performed in order to generate new data in random points of the feasible region (Sadollah et al. 2015a).

IWO is inspired by weed colonization. The initial population is considered to be seeds. This population (seeds) is located in different parts of the search region. Then, each seed grows into a flowering plant that may generate new seeds based on its fitness value. The new seeds grow into new plants that are located at various points in the search region randomly. In other words, better solution vectors can survive in each iteration and the best solution vectors can produce a new population. Accordingly, the fitness value of seeds can be gradually enhanced. Consequently, a seed with the best fitness value is considered to be the solution to the problem in the last iteration (Mehrabian and Lucas 2006).

COA is a new metaheuristic algorithm introduced in 2018. This algorithm was inspired by the social behaviors and interactive experience of *Canis latrans*. Coyotes are associated with solution vectors and their fitness value is assessed by their social behavior. First, coyotes are randomly classified into different groups. The social behavior of coyotes is calculated and the most valuable coyote in each group is called alpha. Afterward, each coyote is impacted by its group mates and the alpha of its group. In this regard, solution vectors are moved toward the solution vectors of their group and the best solution of their corresponding group. Furthermore, coyotes are replaced with the coyotes of other groups to transfer different cultures. This replacement leads to checking more area in the feasible region and reduces the possibility of accumulating solution vectors in local-minimum areas. Ultimately, the weakest coyotes die and they are replaced with new generations (Pierezan and Coelho 2018).

SLC is a powerful algorithm, which investigates to obtain optimal solutions in discrete or continuous space. SLC is inspired by the competitieness of soccer players and teams for winning titles and becoming the best player, respectively. The players are the solution vectors and these players are divided into various teams. Each team's players are classified as being either fixed or substitute players. The most powerful players of each team are classified as fixed players and the remaining players are classified as substitutes. The best player of each team and the best player of the league are called star players and superstar players, respectively. First, the fitness value (power) of all players is calculated according to the objective function of the problem. Each team's power is equal to the average power of its fixed players. Next, the league starts and all possible pairs of teams play games. The possibility of each team winning is directly dependent on the corresponding team's power. During these competitions, the power of players is increased by the provocation and imitation processes. Accordingly, the winning team's fixed players imitate (move toward) both their team's star play and superstar player. At the same time, the winning team's substitute players are transferred to the center of their team's fixed players in order to make progress in becoming fixed players (Moosavian and Kasaee Roodsari 2014).

The ACO algorithm is one of the swarm intelligence algorithms inspired by the foraging behavior of certain ant species (Dorigo et al. 1996). In this algorithm, solution vectors and optimal solutions are simulated by ants and sources of food. When ants need to find food, they randomly explore adjacent areas. Whenever an ant discovers a food source, it assesses the quality and quantity of this food source. Furthermore, a portion of the discovered food is carried back to the nest (Dorigo and Socha 2007). During the carrying process, the ant marks the path by dropping a chemical pheromone on the ground. The exact pheromone may depend on the quality and quantity of food at the discovered source. Accordingly, the pheromone can help other ants find valuable food sources in a less amount of time. Hence, the content of pheromone is significantly increased in the paths that go to the best food sources. Moreover, the distance from the nest to the food source is decreased during this process. In the last iteration, the most valuable food source is considered to be the solution to the problem (Dorigo and Blum 2005).

The PSO algorithm is a classic metaheuristic algorithm inspired by flocks of flying birds. In this algorithm, each solution vector is called a particle. Particles cooperate in the swarm and compete with each other for becoming valuable swarms. Particles adjust their movement according to their moving experiences and the movement experienced by their competitors (Shi and Eberhart 1999). That is to say, each solution vector moves toward its best prior (local) location and the location of the best solution vector (global) that had been discovered up to the current iteration. Afterwards, the experiences of particles are enhanced and the best local and global solution vectors are updated (Eberhart and Kennedy 1995). Thus, the fitness value of solution vectors can be steadily improved. Ultimately, the global solution vector of the last iteration is declared to be the solution to the problem (Eberhart and Shi 2001).

The salp swarm algorithm is a novel swarm intelligence algorithm. Development of this algorithm was inspired by the swarm behavior of salps when foraging and navigating in oceans. Each solution vector is considered a salp in this algorithm

(Mirjalili et al. 2017). Salps usually live in chains and stick to each other. In a salp chain, there is a leading salp that other salps follow. First, the salps are generated with random positions. Next, the fitness value of salps are evaluated and the best salp is considered to be the source of food. Consequently, the leading salp chases the food source and the other salps follow the leading salp. The position of the food source is updated in each iteration in the event that a better solution (salp) is found (Sayed et al. 2018).

The MPA is a recently developed algorithm that uses the chasing pattern of ocean predators to solve optimization problems. In other words, this algorithm is inspired by the movement strategies of ocean predators, such as Brownian and Lévy movements, and the reaction of their prey. In this algorithm, both predators and prey are considered to be search agents. However, their movement and, accordingly, their duties change in different iterations (Faramarzi et al. 2020). That is to say, three different moving strategies are considered for predators and prey. In the first strategy (initial iterations), it is assumed that the velocity of prey is higher than that of predators. Hence, the best strategy for predators is to stop moving. In these iterations, exploration matters. In the second strategy (intermediate iterations), it is presumed that the speed of the predators and prey is equal, and they move at the same pace. Ergo, both exploitation and exploration matters. Both the predators and prey are responsible for exploration and exploitation, respectively. In this phase, the algorithm considers prey moves in Lévy while predator moves in Brownian. In the third strategy (final iterations), exploitation is important, and predators move faster than prey. The best movement strategy is Lévy for predators. Ultimately, the most valuable search agent is regarded to be the solution to the optimization problems (Faramarzi et al. 2020).

The FA is a nature-inspired algorithm that mimics the characteristics and flash patterns of fireflies. Fireflies search for prey, communicate, and mate. The fireflies and their brightness simulate the solution vectors and their fitness value based on the objective function (Senthilnath et al. 2011). In this algorithm, it is assumed that all of the fireflies are unisex and that all of them are attracted to others regardless of sex. Additionally, attractiveness directly correlates with brightness. Accordingly, less bright fireflies moves toward brighter ones. That is to say, attractiveness is proportional to brightness, which decreases with increasing distance between fireflies. If there is no firefly brighter than one specific firefly, it moves randomly in the feasible region. The feasible region is meticulously investigated according to the following rules in order to find optimal or near-optimal solutions for optimization problems (Gandomi et al. 2013b).

The CS is a swarm intelligence optimization algorithm that is inspired by the breeding behavior of particular cuckoo species (Ouaarab et al. 2014). Some cuckoo species lay their eggs in the nests of other host birds (almost other species) and they may eliminate existing eggs so as to increase the hatching likelihood of their eggs. The CS mimics the cuckoo's brood parasitism. There are three types of brood parasitism: intraspecific brood parasitism, cooperative breeding, and nest takeover (Yang et al. 2009). That is, cuckoos simulate solution vectors and nests are particular areas in the feasible region. It is assumed that each cuckoo can lay only one egg at a time, and the generated egg (new solution vector) is dumped into a random nest. The most valuable nest with high-quality solutions will be transferred to subsequent generations. Some host nests may detect an alien egg. If an alien egg is discovered by host nests, the host can throw away this egg or abandon the nest and go find a new nest. The feasible region is investigated using the following process and the optimal or near-optimal solution to the optimization problems is presented (Gandomi et al. 2013a).

The differential evolutionary algorithm is an evolutionary algorithm that was introduced in the 1990s. Although the differential evolutionary algorithm is an old algorithm, it can find valuable solutions to engineering problems (Shirzadi Javid et al. 2020). In this algorithm, solution vectors are responsible for searching in the feasible region. Because of the mutation and crossover operations, the quality of solution vectors is improved and they are transferred to better areas in the feasible space (Varadarajan and Swarup 2008). The goal of crossover is to combine various solution vectors in order to find valuable combinations, whereas mutation changes certain features of solution vectors randomly in order to enhance the possibility of finding the optimal solution to the problems. In this algorithm, the most valuable solution vector is considered to be the solution to the optimization problem (Storn 1997).

The genetic algorithm represents the first generation of metaheuristic algorithms. The genetic algorithm is a classic evolutionary algorithm that has been used to solve various optimization problems. In this algorithm, each chromosome is assigned to a solution vector, which contains a certain number of genes (Holland 2019). Each gene represents the mode of a dimension of the problem. With the help of two operators (mutation and crossover), new generations (chromosomes) are created. The crossover operator combines two chromosomes (parents) that generates new chromosomes (children). Mutation plays a crucial role in the search for new areas in the feasible region. In other words, the mutation operator avoids the algorithm to get stuck in local optimum (Naseri et al. 2020).

The covariance matrix adaptation evolution strategy is an evolutionary algorithm that has served as a standard method for continuous black-box evolutionary optimization. The primary superiority of the covariance matrix adaptation evolution strategy as compared to the classical evolutionary algorithm is related to correlated mutations instead of axis-parallel ones (Loshchilov 2013). Initially, the covariance matrix adaptation evolution strategy generates new populations with a probability distribution. Subsequently, the covariance matrix is adjusted. This algorithm is derived from the concept of self-adaptation in evolution strategies. The covariance matrix adaptation evolution strategy learns correlations between parameters and utilizes the acquired correlations to increase convergence speed. Although the performance of the covariance matrix adaptation evolution strategy has been demonstrated, the performance of this algorithm on continuous problems is more efficient than that of the integer problems (Iruthayarajan and Baskar 2010). This algorithm generates new populations by offspring. Additionally, the covariance matrix and the global step size are updated during the iterations. Updating the aforementioned parameters increases the algorithm's power during the run process (Hansen 2009).

4 MODEL APPLICATION

A case study is presented here to verify and determine the effectiveness of the proposed model. In this project, prior to the implementation phase, the duration of the project was estimated to be 120 days. The sixtieth day after project initiation was considered to be the monitoring and inspecting day. That is to say, $T_{deadline}$ is equal to 60 days. On this day, it was understood that a delay had occurred, resulting in a considerable increase to the total time of the project. A delay fine of $400 was levied for each day in the contract. Therefore, the contractor had to pay the delay fine or had to spend money to decrease the duration of the project by employing additional and expert workers, utilizing more useful gadgets and equipment, and increasing the number of work shifts (Tran et al. 2016). Table 1 presents the delayed activities and their predecessors. Table 1 shows that there were 23 delayed activities that occurred in this project. The predicted time of activities is the estimated durations assigned to them before starting the project. Figure 1 shows the network of the project on the day of monitoring (60^{th} day). This network is based on the predecessors identified in Table 1. As previously mentioned, activities that are not located in delayed paths are not considered in the delay analysis and are therefore overlooked. Based on Figure 2 and Table 1, nine delayed paths existed, as shown in Table 2. As can be seen in Table 2, the range of paths delays is between 4 and 14. Moreover, A-B-C-D-E-F is the most time-consuming path of the project. The duration of this path must be reduced in order to decrease the total duration and delays of the project.

Insert Figure 2

Insert Table 1

Insert Table 2

Table 2 shows that the duration of nine paths exceeds 60 days. Ergo, they are the chief cause of delay in the project and should therefore be analyzed. Finishing time is the summation of the duration time of each path and monitoring day, which is 60 in this project. The most time-consuming delayed path is A-B-C-D-E-F, which takes 74 days. Thus, the project was completed 134 days after it was started. The deadline was 120 days. Hence, a 14-day delay occurred in this project if the remaining activities were implemented based on the timetable. In this case, the contractor had to pay $5600 ($400 per day) as the delay fine.

Following an investigation of different kinds of resources, certain implementation modes are assigned to each delayed activity. The first mode is the primary planning mode and paying extra money for this implementation mode is not needed. In other modes, the time of each activity can be reduced by paying extra money. If more money is spent on each activity, the duration time of that activity is further reduced. Table 3 and Table 4 represent different types of implementing modes for all activities, which are located in delayed paths. These values are extracted from the previous data related to similar projects. The duration of some activities can be reduced by two days, while the duration of others can be decreased even more. The variety of mode numbers is due to the substance of activities and the maximum amounts of resources that can be provided.

Insert Table 3
Insert Table 4

5 STRATEGIES

This study evaluated four different strategies for analyzing the financial benefits and prestige of the company comprehensively. The best strategy can be identified with the assistance of this analysis. Financial profit is one of the essential criteria for every company. Accordingly, project expenditures should be reduced, which benefits the corresponding companies (Shirzadi Javid et al. 2020). Similarly, the credit and prestige of contractors can help the company achieve a prosperous future. The credit and prestige of contractors are consistent with the project's delay. Thus, the delay of the project should be reduced and companies try to complete the project before the deadline.

The goal of the first strategy is to minimize the total cost of the project. This strategy can be used in situations where financial profit is the unique goal of decision-makers. Nevertheless, the contractor's prestige may be tarnished if this strategy is implemented. The objective function of the first strategy is shown in equation (5).

$$Minimize \quad Cost = (400 \times Delay) + \sum_{v=1}^{m_i} \sum_{i=1}^{Z} y_{i,v} \times c_{i,v} \tag{5}$$

The purpose of the second strategy is to omit the delay under the minimum budget. That is to say, the second strategy will find the most economical type of implementation to complete the project by the deadline. This strategy will enhance the prestige of the company. Nonetheless, this strategy may significantly increase the total cost of the project. The corresponding objective function is represented by equation (6).

$$Minimize \quad Cost = \sum_{v=1}^{m_i} \sum_{i=1}^{Z} y_{i,v} \times c_{i,v} \tag{6}$$

s.t: Delay=0

The third strategy uses a multi-objective model to simultaneously reduce the duration and total cost of the project. The time and cost of the project have various ranges. The cost range is much higher than the time range. Hence, to normalize these ranges, equation (7) is used to scale them between 0 and 1 (Naseri et al. 2019).

$$V_s = \frac{V_r - V_{min}}{V_{max} - V_{min}} \tag{7}$$

V_s is the scaled data, V_r is the rough data, and V_{max} and V_{min} are the maximum and minimum values of the rough data, respectively. The maximum and minimum values of delay are 14 (current delay) and 0, respectively. The maximum and minimum costs are extracted from the second and first strategies, respectively. In other words, initially, strategy I and strategy II are solved. Consequently, the maximum logical amounts of cost (cost for omitting delay) that are vital and necessary for modeling the third strategy are extracted from the best solution of the second strategy, because the delay is 0 in this mode and it is not logical to spend more money than this level. Moreover, the minimum value of cost is considered to be the best solution of the first strategy so as to scale the cost objective function. Because the purpose of the first strategy is to minimize the project's total cost, it is not possible to reduce the cost by more than the value introduced in the first strategy. Equation (8) is used to simultaneously optimize cost and time.

$$Minimize \quad Z = w_1 \times Scaled\ Cost + w_2 \times Scaled\ Delay \tag{8}$$

Z is the objective function of the third strategy and w_1 and w_2 are the weights of delay and cost, respectively. These weights are assumed to be equal (0.5).

The fourth strategy uses a comprehensive search method to find optimal solutions to the model. The comprehensive search method is a useful method for calculating the fitness value of all the data that exist in a feasible region. No data is overlooked by this method. However, using this strategy to solve the large-scale problem is not feasible, because it takes a long time. The aim of this this strategy is to minimize the total cost for different amounts of delay. That is to say, possible amounts of delays are considered to be a constraint and minimal costs are identified by various algorithms. This strategy can be useful when the decision-maker intends to reduce the delay up to a certain level.
The objective function of this strategy is shown in equation (9).

$$\text{Minimize} \quad Cost = (400 \times Delay) + \sum_{i=1}^{n} Cost_i \tag{9}$$
$$\text{s.t: Delay} = j \qquad j = 0,1,2,\dots,n$$

j is the feasible amount of delay and n is the maximum amount of delay according to the feasible region.

6 RESULTS AND DISCUSSIONS

The aim of this paper was to compensate for the effect of delays on projects. The contractor has two alternatives in this situation: 1) pay the delay fine; a unit delay fine ($400) should be paid for each day of delay during the project; 2) spend money to increase resources and equipment that will reduce the project's delay. This study investigated the trade-off between these two alternatives. Four strategies were analyzed, including minimizing cost, omitting delay of the project under minimum cost, multi-objective programming to simultaneously reduce the time and cost of the project, and comprehensive search method. The WCA, IWO, COA, SLC, ACO, PSO, SSA, MPA, FA, CS, DE, GA, and CMA-ES were used to find optimal solutions. This section presents the optimal solutions and compares the performance of the aforementioned algorithms. Tukey pairwise comparison is employed to compare the results of all possible algorithm pairs, and to determine the performance of which algorithms are considerably different from others.
The aforementioned algorithms are coded in MATLAB 2019a edition. They were run under the same conditions so that the results could be directly compared. Each

algorithm was run ten times per strategy, and the average value of the objective function, the best achievable solution, and average run time were determined in order to compare their performance. To compare algorithm run time, the population number and the number of solution vectors (population) were considered to be 1,000 and 50 (equally), respectively. Next, the parameters of the other algorithms were tuned and the fastest algorithm was considered to be the algorithm with the lowest average run time.

Table 5 shows the total extra cost, delay, average running time, the average value, the median, the standard deviation, and the minimum value of the objective function corresponding to the algorithms of the first strategy. As can be seen in Table 5, the minimum average value of the objective function is obtained by COA, followed by DE, IWO, MPA, ACO, CS, WCA, SLC, FA, PSO, CMA-ES, SSA, and GA. Likewise, COA generated the lowest median value of the objective function and can be regarded as being the most valuable algorithm in this strategy. The lowest value of the objective function was 3,865. This value was generated by IWO, COA, MPA, and DE. Among these four algorithms (IWO, COA, MPA, and DE), MPA identified the global-optimal solution in the least amount of time. The best objective function value for WCA, FA, CS, SLC, ACO, PSO, CMA-ES, SSA, and GA was $15, which was $15, $20, $20, $20, $35, $75, $95, and $125 more than that of IWO, COA, MPA, and DE.

Accordingly, it can be postulated that COA performed better than the other algorithms in the first strategy because it provided the lowest average objective function value (3,870), the lowest median value of objective function, and it achieved the best solution. The optimal solution to the first strategy reduced the extra cost of the project by 31% (from 5,600 to 3,865). Additionally, MPA was the fastest algorithm, which generated the optimal solution to the first strategy.

Table 6 presents the Tukey pairwise comparison results for the first strategy. Regarding the results of Table 6, the performance of COA and DE is better than other algorithms for the first strategy. Meanwhile, the performance of IWO, MPA, ACO, and CS could be acceptable. On the other hand, GA and SSA are the worst algorithms in the first strategy based on the Tukey pairwise comparison results. The Tukey pairwise comparison outcomes are in line with the results presented in the previous part.

Insert Table 5

Insert Table 6

Table 7 shows the average value of the objective function, the minimum value of the objective function, the median, the standard deviation of solutions, and the lowest amount of cost presented by different algorithms in the second strategy. Each algorithm reduced delay. The presented cost is the extra cost that should be paid in order to finish the project before the deadline. As shown in Table 7, the DE outperformed the other algorithms based on the average value of the objective function among different runs. That is to say, the lowest average value of the objective function is achieved by DE with a value of 4300, followed by COA, IWO, ACO, MPA, CS, SLC, WCA, FA, PSO, SSA, CMA-ES, and GA. Similarly, DE dominates other algorithms, based on the median and standard deviation of the objective function in which the median and standard deviation of the objective function in DE is equal to 3600 and 0, respectively, which indicates that DE reached 3600 in all runs.

In the second strategy, eight algorithms, including WCA, IWO, COA, ACO, SSA, MPA, CS, and DE, provide a solution with the minimum amount of cost. The best value of the objective function identified by these algorithms is equal to $4300. The next most valuable solutions are related to SLC, FA, PSO, CMA-ES, and GA, with the best objective function values of 4330, 4400, 4460, 4510, and 4625, respectively. In this strategy, the best solution (cost = 4300) ultimately removed the project's delay, which could help the company in the future. Furthermore, the extra cost of the project was reduced by 23.2%. In other words, the most valuable solution to the second strategy decreased the project's extra cost from $5600 to $4300 and removed the project's delay simultaneously.

Table 8 shows the Tukey pairwise comparison outcomes for the second strategy. A more detailed look at the results of Table 8 reveals that DE, COA, IWO, ACO, MPA, and CS outperforms other algorithms in the second strategy, and their results are better than the other algorithm results. The weakest algorithm to solve the second strategy is GA, and the performance of GA is by far worse than the other algorithms applied in this study. The Tukey pairwise comparison results are consistent with the results presented in Table 7.

Insert Table 7

Insert Table 8

Table 9 shows the results of algorithms as part of the third strategy. As shown in Table 9, DE is the most valuable algorithm based on the average value of the objective function. MPA is recognized as the second most powerful algorithm, with an average objective function of 0.403. The next most valuable algorithms are IWO, SLC, COA, CS, ACO, FA, WCA, CMA-ES, PSO, SSA, and GA. Furthermore, DE and MPA had the lowest median and standard deviation value of the objective function, respectively. Based on the results, the COA, DE, MPA, IWO, CS, and ACO are capable of finding the third strategy's best solution. These six algorithms generated an objective function of 0.393. This value represents a solution with a two-day delay and a cost of $3,990. Nevertheless, running ACO takes approximately 6.6 times longer than MPA. The aforementioned solution reduced cost by 30.4% and reduced delay by 85.7%. The next valuable solution is obtained by SLC, with an objective function value of 0.4. The SLC optimal solution generated the lowest value of delay. This solution reduced delay to 1 day, and required $4105 to be implemented. The third solution is related to FA. The FA solution can decrease project cost by 28%; however, this expenditure is $20 more expensive than that of the best solution (0.393). In this strategy, the weakest solution is associated with GA because it presents a solution with the lowest fitness value; additionally, it is the only algorithm that provide solutions with an average objective function of greater than 0.8. Moreover, PSO is the only algorithm that reduces the delay from 14 to 3 days. The maximum value of delay belongs to PSO optimal solution in the third strategy.

The Tukey pairwise comparison outcomes for the algorithms in the third strategy are indicated in Table 10. According to the result of Table 10, DE, MPA, IWO, SLC, ACO, and COA outweigh other algorithms in order to find valuable solutions. Similar to the first strategy and the second strategy results, GA is not qualified to solve the delay TCTP for the third strategy.

Insert Table 9

Insert Table 10

The average value of the objective function of algorithms in the fourth strategy is shown in Figure 3. As shown in Figure 3, by reducing the delay up to a certain level, delay and cost are simultaneously reduced. Subsequently, delay reduction leads to an increased required cost. A more detailed examination of the lowest point of different delays reveals that the optimal solutions to ten points (delay from 5 to 14) are dominated by the lowest point in 4 days delay. In other words, the optimal solution of 4 days delay dominates its left points because its delay and its cost are lower than those of the points containing more delays. However, the solution related to delay = 0 (S-4-0), delay = 1 (S-4-1), delay = 2 (S-4-2), delay = 3 (S-4-3), and delay = 4 (S-4-4) cannot dominate each other and they are non-dominated solutions. This result is consistent with the result of the first strategy in which the minimum value of cost is assigned to delay = 4. Accordingly, the optimal solutions of the previously mentioned five delays are located in a Pareto front and a trade-off between these points should be considered in this strategy. Additionally, Figure 3 indicates that the performance of DE, MPA, CS, COA, IWO, and WCA are significantly better than the other algorithms for finding the best solutions. Additionally, these algorithms are capable of finding the optimal or near-optimal solutions of delay TCTPs. Moreover, by increasing the complexity of the problem (increasing delay), the outcomes of algorithms are changed and, in high-level complexity problems, algorithm performance can be compared more easily. Table 11 shows the most valuable solution for a non-dominated sub-strategy (S-4-0, S-4-1, S-4-2, S-4-3, and S-4-4) and the algorithms that acquired the previously mentioned solutions during their ten runs. Based on the results of this table, DE, MPA, CS, IWO, COA, WCA, and SSA are capable of finding the most valuable solution of S-4-0. The optimal solution to S-4-0 can reduce the delay and cost by 14 days and $1300. Additionally, this sub-strategy (S-4-0) is the same as the second strategy, which omits the delay completely. The optimal solution for S-4-1 decreases delay by 13 days and $1495 of additional cost that should be paid as a delay fine. DE, MPA, CS, IWO, COA, WCA, SLC, and ACO were better able to find the optimal solution to S-4-1 as compared to the other algorithms. DE, MPA, CS, IWO, COA, SLC, and ACO were able to achieve the optimal solution of S-4-2, which decreased extra cost and project delay by 28.8% and 85.7%, respectively. The most valuable solution to S-4-3 was found by DE, MPA, CS, IWO, COA, SLC, and WCA. These algorithms reduced the objective function to 3935, which implies that $1665 of extra cost and 11 days of delay can be reduced simultaneously by S-4-3. Furthermore, DE, CS, IWO, COA, and SLC arrived at the optimal solution to S-4-4. The minimum amount of cost is related to the optimal solution to S-4-4, which is the same as the first strategy in which the extra cost reached its lowest level. In this sub-strategy (S-4-4), the required cost was reduced from $5600 to $3865. Based on the following results, the performance of DE, MPA, CS, IWO, and COA for finding the optimal solution to a delay TCTP is better than that of the other algorithms considered in this study. DE, MPA, CS, IWO, and COA can find the most valuable solution for all sub-strategies in the fourth strategy and the average value of the objective function presented by these algorithms is significantly lower than that of the other algorithms.

Insert Figure 3

Insert Table 11

Average run time is one of the essential criteria for comparing the capability and power of evolutionary algorithms. If the population number and the number of iterations are 50 and 1000, respectively, and the algorithms are tunned, the lowest average run time is achieved by FA, followed by SSA, WCA, MPA, DE, CMA-

ES, COA, IWO, CS, PSO, GA, ACO, and SLC. Ergo, the solutions associated with a lower fitness value generated by FA and SSA may be due to the fewer number of evaluations in an iteration. Additionally, the performance of WCA can be accepted because it presents an appropriate solution in less time than most other algorithms. Additionally, it can be postulated that DE outperformed other algorithms because it found the most valuable solution for all strategies and its average running time was lower than the other algorithms that introduce optimal solutions for all strategies.

Table 12 shows the optimal mode of activities for the most valuable solution achieved in various strategies. As can be seen in Table 12, the cost and delay introduced by the first strategy and S-4-4 are the same. Similarly, the solution and corresponding mode of activities presented by the second strategy and S-4-0 are identical. Likewise, the optimal solution presented by the third strategy and S-4-2 are the same. However, their implementation mode of activities is not unique. That is to say, the third strategy and S-4-2 reduced the cost and delay to $4300 and four days, respectively, by introducing different modes of activities. The modes assigned to activities J and L are dissimilar in the optimal solution associated with them. The mode assigned to activities D, G, O, and Q do not change in different strategies and it is not recommended to modify these activities. This process may be due to the higher price that needs to crash these activities rather than other activities located in their paths.

Insert Table 12

In the results, eight solutions are presented based on various strategies and different algorithms. Some of these solutions are the same and the number of solutions is five. In other words, by analyzing the fourth strategy, all optimal solutions are analyzed, and the fourth strategy encompasses the solutions introduced by the first, second, and third strategies. The resulting solutions are better than paying the delay fine. They reduce both the cost and duration of the project on the day of inspection and monitoring. Thus, time-cost trade-off analysis is a powerful technique that compensates for the effects of delay on projects. Table 13 shows the percentage of time (delay) and cost reduction achieved by different strategies.

Insert Table 13

As can be seen in Table 13, all of the solutions decreased the total cost and duration of the project. The delay and cost were also reduced in all of the solutions presented in Table 13 by more than 71.4% and 23.2%, respectively. The most economical solution is related to the first strategy and S-4-4, which reduced the extra cost of the project by 31%. The lowest value of cost reduction is related to the optimal solution and to the second strategy and S-4-0, which reduced the total cost of the project by 23.2%. The second strategy provided solutions that completely omitted project delays. In contrast, the lowest delay reduction was related to the optimal solution and to the first strategy, which decreased the project's delay by 71.4%.

It is a difficult and challenging decision to opt for one of the previously mentioned solutions. Accordingly, a novel index was developed in order to compare the optimal solutions and assess their improvement. The previously mentioned index (equation (10)) was developed by sensitivity analysis, expert justice, and engineering analysis.

$$Improvement = \left(\frac{Initial\ cost - optimal\ cost}{Initial\ cost}\right) \times \left(1 + \frac{Initial\ delay - optimal\ delay}{10}\right) \times 100 \quad (10)$$

$Improvement$ is the value of each solution. $Initial\ cost$ and $Initial\ delay$ represent the delay fine (5600) and delay time (14) on the day of inspection and monitoring, respectively. The $optimal\ cost$ and $optimal\ delay$ are the cost and delay introduced by the algorithms. The values of $optimal\ cost$ and $optimal\ delay$ are shown in Table 12.

Using equation (10), the improvement achieved by optimal solutions under different strategies was calculated and is displayed in Figure 4. Based on Figure 4, the highest degree of improvement is relevant to the optimal solution to the third strategy (S-4-2), with an improvement value of 63.25%. The optimal solution to S-4-3 represents the second valuable solution based on improvement. The improvement of the S-4-3 optimal solution is equal to 62.44%. The improvement of the first strategy optimal solution is 61.96%. Additionally, solving the sub-strategy of S-4-1 provides a solution with an improvement value of 61.40%. The lowest improvement value is associated with the second strategy (S-4-0), with an improvement of 55.71%, which is lower than the improvement values generated by the other strategies. The second strategy optimal solution is the only model that had an improvement value lower than 60%. It can be postulated that omitting delay may not be an appropriate strategy in delay TCTPs. The third and fourth strategies are the best strategies, since they provided the highest level of improvement.

Although the fourth strategy found the most valuable solution and in S-4-2 the highest level of improvement was detected, solving a delay TCTP with the fourth strategy (comprehensive search method) takes a long time, and it may be impractical to use this strategy for solving large-scale networks in logical time. In this study, solving the fourth strategy required approximately 14 times longer than that of the third strategy, while both of them arrive at a similar improvement. Hence, it can be postulated that the third strategy (multi-objective optimization) is the best method for compensating for the negative impacts of delay on projects.
To compare the outcomes of algorithms and analyze their introduced solutions, the improvement value of the algorithm's optimal solution for the points located in the Pareto front (S-4-0, S-4-1, S-4-2, S-4-3, and S-4-4) was calculated, and the results are shown in Table 14. As can be seen in Table 14, DE, CS, COA, and IWO are highly qualified for finding the optimal solution of delay TCTPs because these algorithms provide solutions with a higher level of improvement. That is to say, the average value of improvement related to DE, CS, COA, and IWO is 60.95, which is more than the other algorithms. The result of this index is consistent with the results presented in previous sections of this report.

Insert Figure 4

Insert Table 14

7 Data and Availability Statement

The data that support the findings of this study are available on request from the corresponding author.

8 CONCLUSIONS

This study used a variety of algorithms to minimize the influences of delay on cost and duration of TCTP, including WCA, IWO, COA, SLC, ACO, PSO, SSA, MPA, FA, CS, DE, GA, and CMA-ES. Four strategies were analyzed so that the best decision could be made regarding compensating for the effects of delay on projects. The Tukey pairwise comparison is employed to analyze the algorithm' performance. A novel index was used to scrutinize the results generated by various strategies and different algorithms. The following conclusions can be drawn from the results of this study:

- DE, CS, COA, and IWO are the best algorithms for solving the delay TCTP because they provide the most valuable solutions, with an average improvement value of 60.95%. The average value of the objective function presented by these algorithms is lower than that of other algorithms. The next powerful algorithms for solving a delay TCTP are MPA, SLC, WCA, ACO, PSO, FA, SSA, CMA-ES, and GA, with average improvement values of 60.81%, 60.70%, 60.44%, 59.81%, 58.56%, 56.21%, 55.89%, 54.68%, 54.65%, and 52.03%, respectively.

- The Tukey pairwise comparison indicates that COA and DE are the most valuable algorithms to solve the first strategy. In the second strategy, DE, COA, IWO, ACO, MPA, and CS are better than other algorithms so as to find precious solutions. The Tukey pairwise comparison determines DE, MPA, IWO, SLC, ACO, and COA as the most valuable algorithms in the third strategy. However, GA is recognized as the worst algorithm by the Tukey pairwise comparison.
- Based on a comparison of the various strategies, multi-objective optimization and comprehensive search method generated the best solutions, followed by minimizing cost and omitting delay. Furthermore, the multi-objective (third) strategy outperforms the comprehensive search method because solving a large-scale network under a comprehensive search method is not feasible. While the third strategy can generate the most valuable solution to delay TCTPs in logical time. Thus, it can be theorized that the solution introduced by multi-objective optimization is ideal because it was associated with the highest level of improvement (63.25%). The first strategy (cost minimization) ranked third among the proposed strategies due to providing an improvement value of 61.96%. Besides, the second strategy (omitting delay) may not be a suitable method for compensating for the negative influences of project delays because the improvement it introduced was equal to 55.71%.
- Eight solutions were presented for the problem. The best solution is achieved by the optimal solution of the multi-objective strategy with an improvement value of 63.25%. The best solution reduced project delays by 85.7% (from 14 to 2 days) and decreased project costs by 28.75% (from $5600 to $3990).
- With the same number of iterations and solution vectors and after the tuning and calibration process, FA is the fastest algorithm for the delay TCTP, followed by SSA, WCA, DE, CMA-ES, COA, IWO, MPA, CS, PSO, GA, ACO, and SLC.

Funding: Not applicable.

Conflict of Interest: Not applicable.

Ethical approval: Not applicable.

Ethical approval: Not applicable.

Ethical approval: This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent: Not applicable.

REFERENCES

Afshar A, Kaveh A, Shoghli OR (2007) MULTI-OBJECTIVE OPTIMIZATION OF TIME-COST-QUALITY USING MULTI-COLONY ANT ALGORITHM. Fuzzy Sets Syst. https://doi.org/Doi 10.1016/S0165-0114(99)00119-0

Agdas D, Warne DJ, Osio-Norgaard J, Masters FJ (2018) Utility of Genetic Algorithms for Solving Large-Scale Construction Time-Cost Trade-Off Problems. J Comput Civ Eng 32:04017072. https://doi.org/10.1061/(asce)cp.1943-5487.0000718

Aminbakhsh S, Sonmez R (2016) Discrete particle swarm optimization method for the large-scale discrete time-cost trade-off problem. Expert Syst Appl 51:177–185. https://doi.org/10.1016/j.eswa.2015.12.041

Burns SA, Liu L, Feng CW (1996) The LP/IP hybrid method for construction time-cost trade-off analysis. Constr Manag Econ 14:265–276. https://doi.org/10.1080/014461996373511

Chung-Wei Feng, Liang Liu SAB (2000) STOCHASTIC CONSTRUCTION TIME-COST TRADE-OFF ANALYSIS. J Comput Civ Eng 117–126

Dorigo M, Blum C (2005) Ant colony optimization theory: A survey. Theor Comput Sci. https://doi.org/10.1016/j.tcs.2005.05.020

Dorigo M, Maniezzo V, Colorni A (1996) Ant system: Optimization by a colony of cooperating agents. IEEE Trans Syst Man, Cybern Part B Cybern. https://doi.org/10.1109/3477.484436

Dorigo M, Socha K (2007) Ant colony optimization. In: Handbook of Approximation Algorithms and Metaheuristics

Eberhart R, Kennedy J (1995) New optimizer using particle swarm theory. In: Proceedings of the International Symposium on Micro Machine and Human Science

Eberhart RC, Shi Y (2001) Particle swarm optimization: Developments, applications and resources. In: Proceedings of the IEEE Conference on

Evolutionary Computation, ICEC
Eshtehardian E, Afshar A, Abbasnia R (2009) Fuzzy-based MOGA approach to stochastic time-cost trade-off problem. Autom Constr 18:692–701. https://doi.org/10.1016/j.autcon.2009.02.001
Eskandar H, Sadollah A, Bahreininejad A, Hamdi M (2012) Water cycle algorithm--A novel metaheuristic optimization method for solving constrained engineering optimization problems. Comput Struct 110:151–166
Faramarzi A, Heidarinejad M, Mirjalili S, Gandomi AH (2020) Marine Predators Algorithm: A Nature-inspired Metaheuristic. Expert Syst Appl 113377. https://doi.org/10.1016/j.eswa.2020.113377
Gandomi AH, Yang XS, Alavi AH (2013a) Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. Eng Comput 29:17–35. https://doi.org/10.1007/s00366-011-0241-y
Gandomi AH, Yang XS, Talatahari S, Alavi AH (2013b) Firefly algorithm with chaos. Commun Nonlinear Sci Numer Simul 18:89–98. https://doi.org/10.1016/j.cnsns.2012.06.009
Ghoddousi P, Eshtehardian E, Jooybanpour S, Javanmardi A (2013) Multi-mode resource-constrained discrete time-cost-resource optimization in project scheduling using non-dominated sorting genetic algorithm. Autom Constr 30:216–227. https://doi.org/10.1016/j.autcon.2012.11.014
Hafezalkotob A, Hosseinpour E, Moradi M, Khalili-Damghani K (2018) Multi-resource trade-off problem of the project contractors in a cooperative environment: highway construction case study. Int J Manag Sci Eng Manag 13:129–138. https://doi.org/10.1080/17509653.2017.1323240
Hafızo˘glu AB, Azizo˘glu M (2010) Linear programming based approaches for the discrete time / cost trade-off problem in project networks. 676–685. https://doi.org/10.1057/jors.2009.20
Hansen N (2009) Benchmarking a BI-Population CMA-ES on the BBOB-2009 Function Testbed. 2389–2395
Hindelang TJ, Muth JF (1979) Dynamic Programming Algorithm for Decision Cpm Networks. Oper Res 27:225–241. https://doi.org/10.1287/opre.27.2.225
Holland JH (2019) Adaptation in Natural and Artificial Systems
Iruthayarajan MW, Baskar S (2010) Expert Systems with Applications Covariance matrix adaptation evolution strategy based design of centralized PID controller. Expert Syst Appl 37:5775–5781. https://doi.org/10.1016/j.eswa.2010.02.031
Kalhor E, Khanzadi M, Eshtehardian E, Afshar A (2011) Stochastic time-cost optimization using non-dominated archiving ant colony approach. Autom Constr 20:1193–1203. https://doi.org/10.1016/j.autcon.2011.05.003
Ke H (2014) Uncertain random time-cost trade-off problem. 1–10
Leu S Sen, Chen AT, Yang CH (2001) A GA-based fuzzy optimal model for construction time-cost trade-off. Int J Proj Manag 19:47–58. https://doi.org/10.1016/S0263-7863(99)00035-6

Lo W, Kuo ME (2011) Assessing the time-cost trade-off of construction projects. Adv Mater Res 243–249:6394–6399. https://doi.org/10.4028/www.scientific.net/AMR.243-249.6394

Loshchilov I (2013) CMA-ES with Restarts for Solving CEC 2013 Benchmark Problems. 369–376

Mehrabian AR, Lucas C (2006) A novel numerical optimization algorithm inspired from weed colonization. Ecol Inform 1:355–366. https://doi.org/10.1016/j.ecoinf.2006.07.003

Mirjalili S, Gandomi AH, Mirjalili SZ, et al (2017) Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002

Monghasemi S, Nikoo MR, Khaksar Fasaee MA, Adamowski J (2015) A novel multi criteria decision making model for optimizing time-cost-quality trade-off problems in construction projects. Expert Syst Appl 42:3089–3104. https://doi.org/10.1016/j.eswa.2014.11.032

Moosavian N, Kasaee Roodsari B (2014) Soccer league competition algorithm: A novel meta-heuristic algorithm for optimal design of water distribution networks. Swarm Evol Comput 17:14–24. https://doi.org/10.1016/j.swevo.2014.02.002

Naseri H (2018) Time-Cost Trade off to Compensate Delay of Project Using Genetic Algorithm and Linear Programming. Int J Innov Manag Technol 9:285–290. https://doi.org/10.18178/ijimt.2018.9.6.826

Naseri H, Jahanbakhsh H, Hosseini P, Moghadas Nejad F (2020) Designing sustainable concrete mixture by developing a new machine learning technique. J Clean Prod 258:120578. https://doi.org/10.1016/j.jclepro.2020.120578

Naseri H, Jahanbakhsh H, Moghadas Nejad F (2019) Developing a novel machine learning method to predict the compressive strength of fly ash concrete in different ages. AUT J Civ Eng 0: https://doi.org/10.22060/AJCE.2019.16124.5569

Ouaarab A, Ahiod B, Yang XS (2014) Discrete cuckoo search algorithm for the travelling salesman problem. Neural Comput Appl 24:1659–1669. https://doi.org/10.1007/s00521-013-1402-2

Pierezan J, Coelho L dos S (2018) Coyote Optimization Algorithm : A new metaheuristic for global optimization problems. 2018 IEEE Congr Evol Comput 1–8

Sadollah A, Eskandar H, Bahreininejad A, Kim JH (2015a) Water cycle algorithm for solving multi-objective optimization problems. Soft Comput 19:2587–2603. https://doi.org/10.1007/s00500-014-1424-4

Sadollah A, Eskandar H, Kim JH (2015b) Water cycle algorithm for solving constrained multi-objective optimization problems. Appl Soft Comput J 27:279–298. https://doi.org/10.1016/j.asoc.2014.10.042

Sayed GI, Khoriba G, Haggag MH (2018) A novel chaotic salp swarm algorithm

for global optimization and feature selection. Appl Intell 48:3462–3481. https://doi.org/10.1007/s10489-018-1158-6
Senthilnath J, Omkar SN, Mani V (2011) Clustering using firefly algorithm: Performance study. Swarm Evol Comput 1:164–171. https://doi.org/10.1016/j.swevo.2011.06.003
Shi Y, Eberhart RC (1999) Empirical study of particle swarm optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation, CEC 1999
Shirzadi Javid AA, Naseri H, Etebari Ghasbeh MA (2020) Estimating the Optimal Mixture Design of Concrete Pavements Using a Numerical Method and Meta-heuristic Algorithms. Iran J Sci Technol - Trans Civ Eng. https://doi.org/10.1007/s40996-020-00352-6
Storn R (1997) Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. 341–359
Tran DH, Cheng MY, Prayogo D (2016) A novel Multiple Objective Symbiotic Organisms Search (MOSOS) for time-cost-labor utilization tradeoff problem. Knowledge-Based Syst 94:132–145. https://doi.org/10.1016/j.knosys.2015.11.016
Varadarajan M, Swarup KS (2008) Electrical Power and Energy Systems Differential evolutionary algorithm for optimal reactive power dispatch. 30:435–441. https://doi.org/10.1016/j.ijepes.2008.03.003
Yang X, Deb S, Behaviour ACB (2009) Cuckoo Search via L ´ evy Flights. Ieee 210–214
Zheng DXM, Ng ST (2005) Stochastic time-cost optimization model incorporating fuzzy sets theory and nonreplaceable front. J Constr Eng Manag 131:176–186. https://doi.org/10.1061/(ASCE)0733-9364(2005)131:2(176)

Figure captions:

Figure 1: Methodology flowchart

Figure 2: The network of project on the inspection day (60^{th} day), NDA are activities that are not located in delayed paths

Figure 3: Average value of the objective function concerning various amounts of delay

Figure 4: Improvement values of optimal solutions generated by various strategies

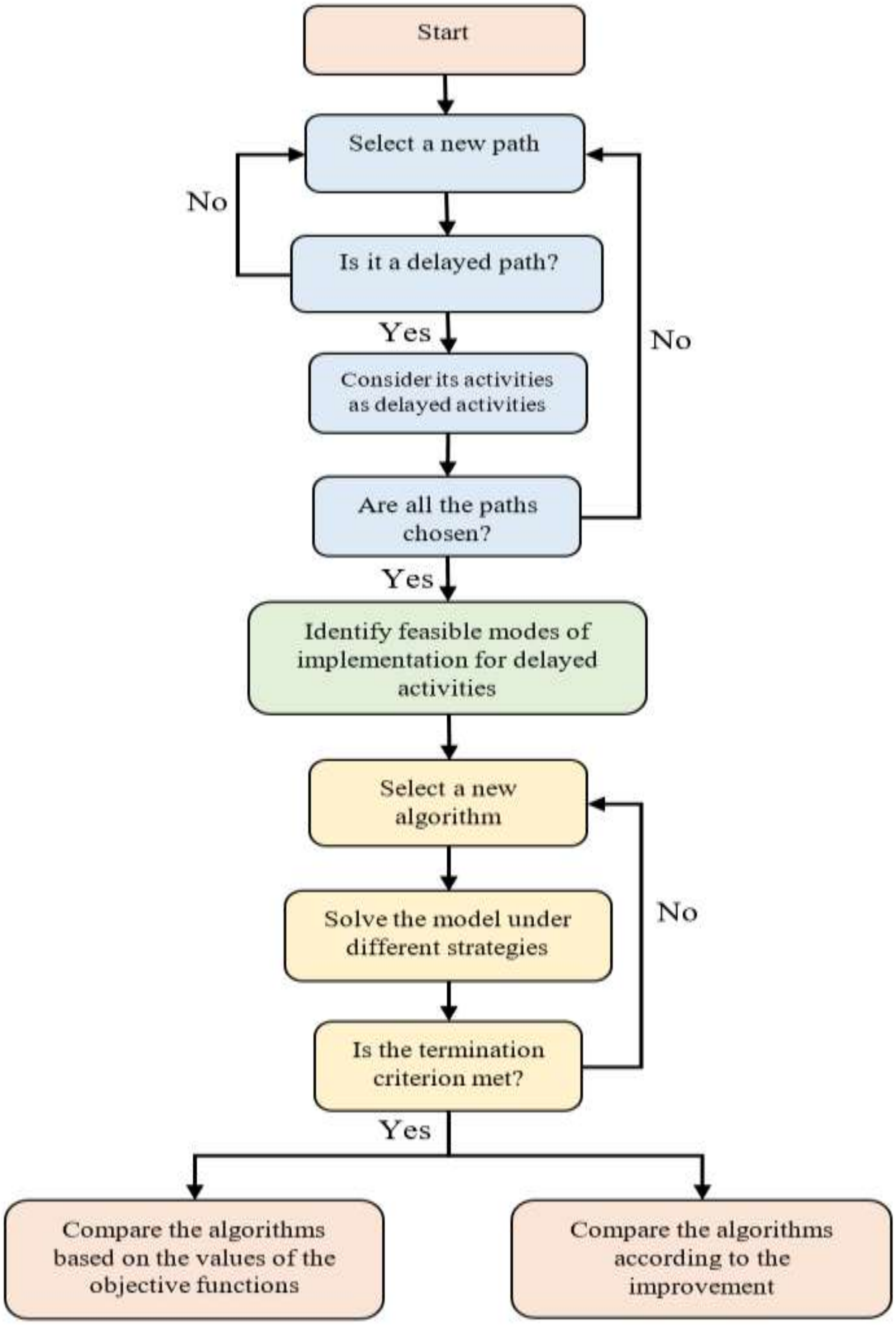

Figure 1: Methodology flowchart

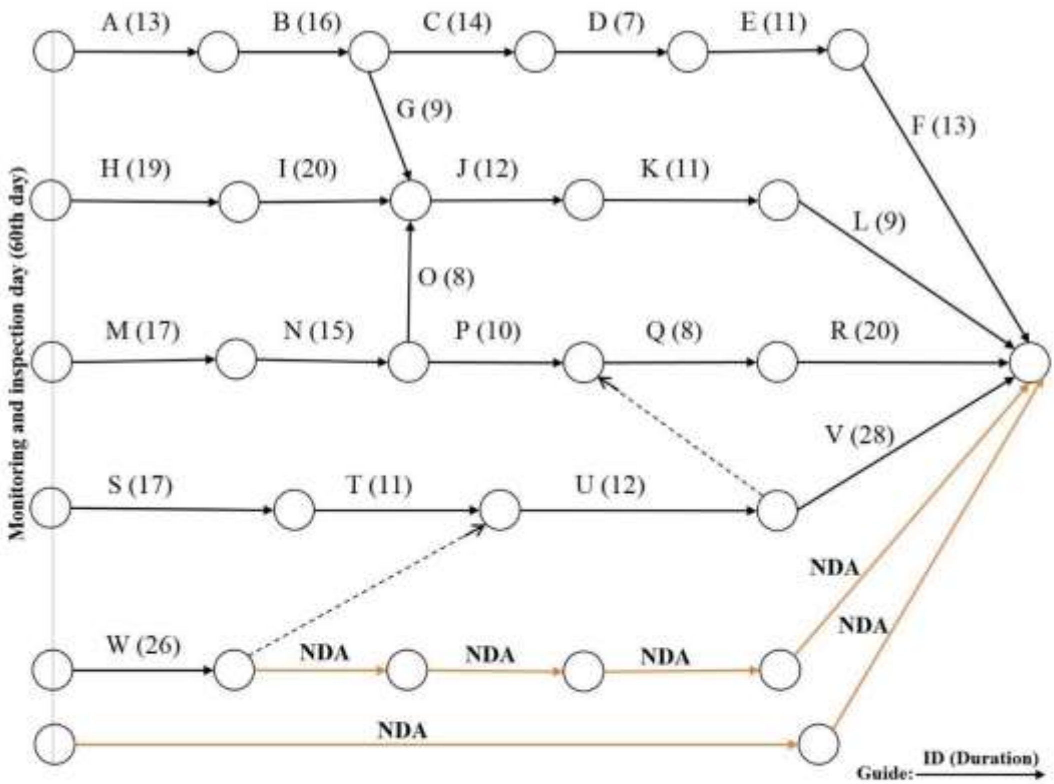

Figure 2: The network of project on the inspection day (60^{th} day), NDA are activities that are not located in delayed paths

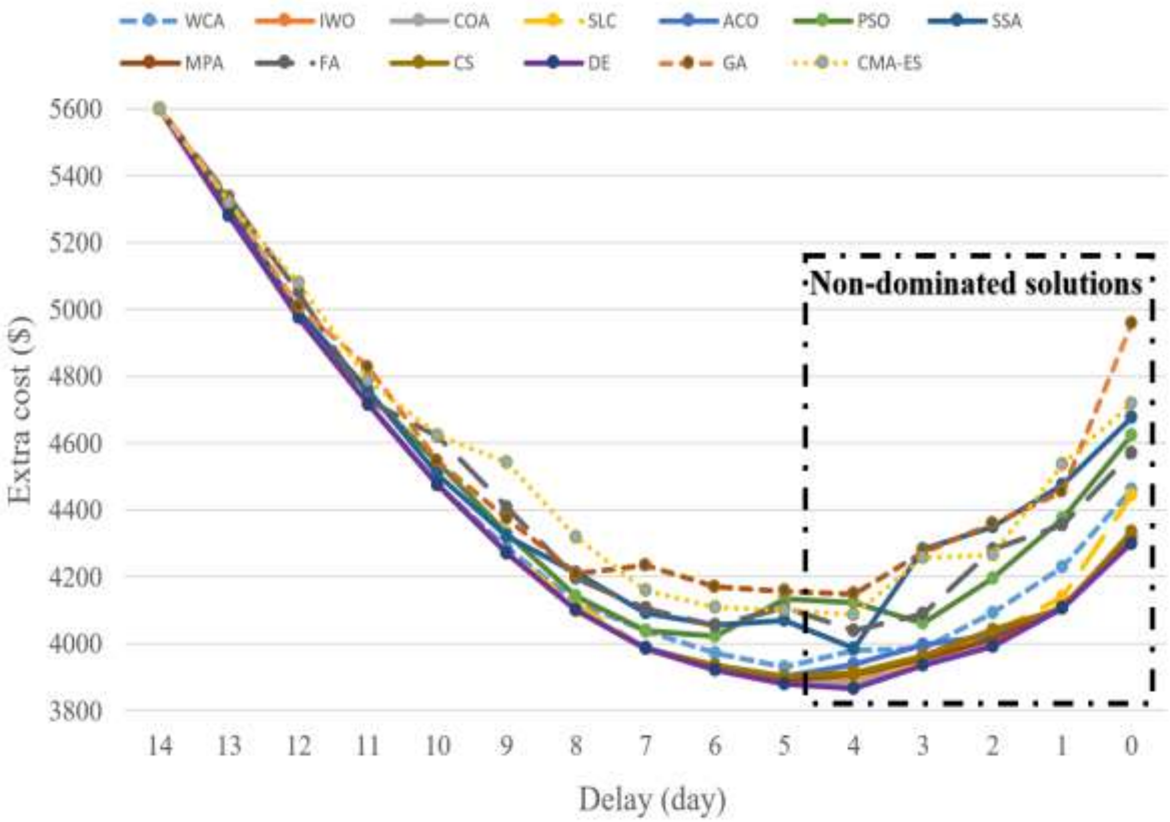

Figure 3: Average value of the objective function concerning various amounts of delay

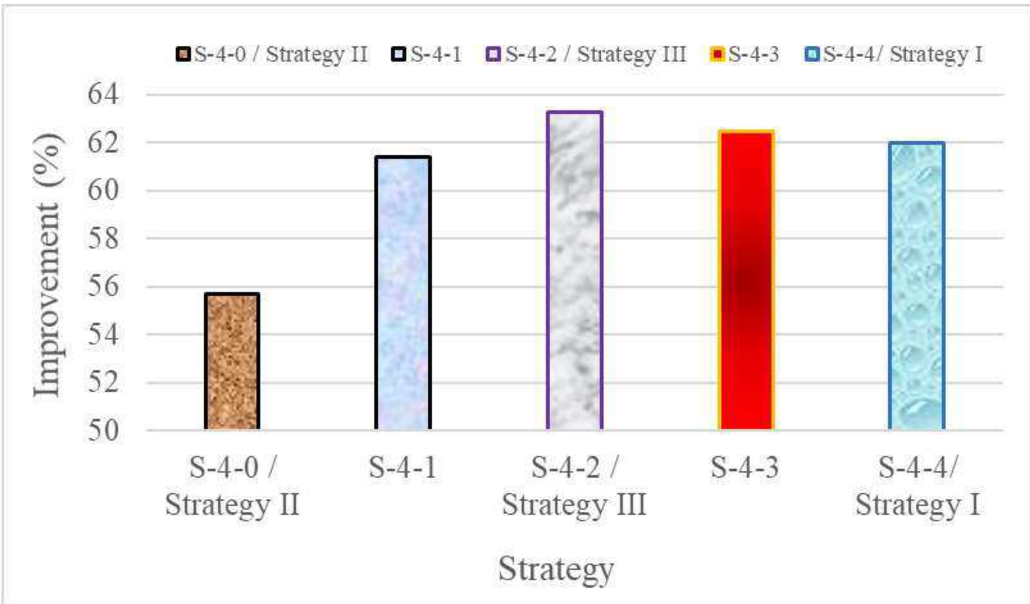

Figure 4: Improvement values of optimal solutions generated by various strategies

Table captions:

Table 1: Delayed activities ID, predecessors, and predicting time
Table 2: Delayed paths
Table 3: Different modes of implementation for activities in delayed paths (A-L)
Table 4: Different modes of implementation for activities in delayed paths (M-W)
Table 5: Algorithm results for the first strategy
Table 6: Tukey pairwise comparison outcomes for the first strategy
Table 7: Optimal solutions to the second strategy
Table 8: Tukey pairwise comparison outcomes for the second strategy
Table 9: Results of the algorithm that was used to solve the problem under the third strategy
Table 10: Tukey pairwise comparison outcomes for the third strategy
Table 11: The most valuable solutions to the non-dominated sub-strategies of the fourth strategy
Table 12: Optimal mode of activities for the first, second, and third strategies
Table 13: Percentage of delay and cost reduction
Table 14: Improvement values of optimal solutions to non-dominated sub-strategies

Table 1: Delayed activities ID, predecessors, and predicting time

Delayed Activities ID	Predecessors	Predicting time
A	-	13
H	-	19
M	-	17
S	-	17
W	-	26
B	A	16
I	H	20
N	M	15
T	S	11
G	B	9
O	N	8
C	B	14
J	G,I,O	12
P	N	10
D	C	7
K	J	11
U	T,W	12
Q	U,P	8
E	D	11
L	K	9
R	Q	20
V	U	28
F	E	13

Table 2: Delayed paths

Delayed Paths	Duration Time	Finishing Time	Delay
A-B-C-D-E-F	74	134	14
A-B-G-J-K-L	70	130	10
H-I-J-K-L	71	131	11
M-N-O-J-K-L	72	132	12
M-N-P-Q-R	70	130	10
S-T-U-Q-R	68	128	8
W-U-Q-R	64	124	4
S-T-U-V	68	128	8
W-U-V	66	126	6

Table 3: Different modes of implementation for activities in delayed paths (A-L)

Activity	Mode	Time	Extra Cost	Activity	Mode	Time	Extra Cost
	1	13	0		1	9	0
A	2	12	100	G	2	8	60
	3	11	220		3	7	135
	4	10	405		1	19	0
	1	16	0	H	2	18	60
B	2	15	80		3	17	135
	3	14	190		4	16	230
	4	13	330		1	20	0
	1	14	0	I	2	19	55
C	2	13	160		3	18	125
	3	12	350		4	17	250
	4	11	610		1	12	0
	1	7	0	J	2	11	85
D	2	6	205		3	10	200
	3	5	520		4	9	390
	4	4	1200		1	11	0
	1	11	0	K	2	10	140
E	2	10	95		3	9	315
	3	9	225		4	8	530
	4	8	415		1	9	0
	1	13	0	L	2	8	50
F	2	12	115		3	7	130
	3	11	250		4	6	245
	4	10	435				

Table 4: Different modes of implementation for activities in delayed paths (M-W)

Activity	Mode	Time	Extra Cost	Activity	Mode	Time	Extra Cost
	1	17	0		1	17	0
M	2	16	40	S	2	16	35
	3	15	95		3	15	90
	4	14	190		4	14	230
	1	15	0		1	11	0
N	2	14	60	T	2	10	120
	3	13	135		3	9	265
	4	12	210		4	8	415
	1	8	0		1	12	0
O	2	7	170	U	2	11	75
	3	6	455		3	10	175
	1	10	0		4	9	400
P	2	9	115		1	28	0
	3	8	280		2	27	60
	4	7	465	V	3	26	140
	1	8	0		4	25	245
Q	2	7	165		5	24	530
	3	6	390		6	23	995
	1	20	0		1	26	0
R	2	19	75	W	2	25	35
	3	18	165		3	24	85
	4	17	275		4	23	150

Table 5: Algorithm results for the first strategy

Algorithm	of the objective function	the objective function	Median	Standard deviation	Average run time (second)	Cost ($)	Delay (day)
WCA	3880	3929.5	3932.5	28.58758	1.029	3880	5
IWO	**3865**	3882	3885	**6**	2.291	3865	4
COA	**3865**	**3870**	**3865**	7.745967	2.3258	3865	4
SLC	3885	3933.5	3927.5	35.64057	8.729	3885	5
ACO	3885	3892	3885	9.797959	7.374	3885	5
PSO	3900	3980	3965	62.00806	3.383	3900	5
SSA	3960	4059	4020	92.35259	0.491	3960	4
MPA	**3865**	3882.5	3885	10.78193	1.235	3865	4
FA	3880	3946.5	3957.5	32.40756	**0.323**	3880	5
CS	3885	3895.5	3895	11.92686	3.055	3885	5
DE	**3865**	3872.5	3872.5	7.5	1.900	3865	4
GA	3990	4072	4040	72.11796	4.188	3990	5
CMA-ES	3940	3988.5	3982.5	40.99085	1.947	3940	4

Table 6: Tukey pairwise comparison outcomes for the first strategy

The first strategy Factor	Grouping			
GA	A			
SSA	A			
CMA-ES		B		
PSO		B		
FA		B	C	
SLC		B	C	D
WCA		B	C	D
CS			C	D
ACO			C	D
MPA			C	D
IWO			C	D
DE				D
COA				D

Table 7: Optimal solutions to the second strategy

Algorithm	of the objective function	the objective function	Median	Standard deviation	Average run time (second)	Cost ($)
WCA	**4300**	4461	4492.5	110.2905	1.055	4300
IWO	**4300**	4308.5	**4300**	10.5	2.327	4300
COA	**4300**	4302	**4300**	2.44949	2.310	4300
SLC	4330	4444	4407.5	117.3627	8.788	4330
ACO	**4300**	4319	4312.5	23.5372	7.528	4300
PSO	4460	4623.5	4582.5	176.2108	3.146	4460
SSA	**4300**	4677	4582.5	241.8181	0.660	4300
MPA	**4300**	4322.5	4312.5	23.26478	1.220	4300
FA	4400	4569.5	4532.5	130.9284	**0.348**	4400
CS	**4300**	4336	4315	39.35734	3.122	4300
DE	**4300**	**4300**	**4300**	**0**	1.739	4300
GA	4625	4957	4965	217.1428	4.088	4625
CMA-ES	4510	4718	4675	196.7765	1.883	4510

Table 8: Tukey pairwise comparison outcomes for the second strategy

The second strategy Factor	Grouping			
GA	A			
CMA-ES		B		
SSA		B		
PSO		B	C	
FA		B	C	
WCA			C	D
SLC			C	D
CS				D
MPA				D
ACO				D
IWO				D
COA				D
DE				D

Table 9: Results of the algorithm that was used to solve the problem under the third strategy

Algorithm	of the objective function	the objective function	Median	Standard deviation	Average run time (second)	Cost ($)	Delay (day)
WCA	0.422	0.540	0.533	0.078	1.061	4015	2
IWO	**0.393**	0.405	0.400	0.012	2.408	3990	2
COA	**0.393**	0.421	0.414	0.029	2.437	3990	2
SLC	0.400	0.407	0.400	0.015	9.608	4105	1
ACO	**0.393**	0.431	0.400	0.014	8.114	3990	2
PSO	0.501	0.627	0.604	0.100	3.410	3975	3
SSA	0.491	0.658	0.648	0.117	0.623	4075	2
MPA	**0.393**	0.403	0.400	**0.008**	1.235	3990	2
FA	0.416	0.537	0.505	0.077	**0.340**	4010	2
CS	**0.393**	0.423	0.456	0.044	3.231	3990	2
DE	**0.393**	**0.399**	**0.393**	0.009	1.784	3990	2
GA	0.560	0.808	0.795	0.126	4.041	4135	2
CMA-ES	0.422	0.544	0.556	0.098	1.902	4015	2

Table 10: Tukey pairwise comparison outcomes for the third strategy

The third strategy Factor	Grouping				
GA	A				
SSA		B			
PSO		B	C		
CMA-ES			C	D	
WCA			C	D	
FA			C	D	
CS				D	E
COA					E
ACO					E
SLC					E
IWO					E
MPA					E
DE					E

Table 11: The most valuable solutions to the non-dominated sub-strategies of the fourth strategy

Non-dominated solutions	Best value of the objective function	Algorithms that find the most valuable solution	Cost ($)	Delay (day)
S-4-0	4300	DE, MPA, CS, IWO, COA, WCA, SSA	4300	0
S-4-1	4105	DE, MPA, CS, IWO, COA, WCA, SLC, ACO	4105	1
S-4-2	3990	DE, MPA, CS, IWO, COA, SLC, ACO	3990	2
S-4-3	3935	DE, MPA, CS, IWO, COA, SLC, WCA	3935	3
S-4-4	3865	DE, MPA, CS, IWO, COA, SLC	3865	4

Table 12: Optimal mode of activities for the first, second, and third strategies

Activity	Strategy I	Strategy II	Strategy III	Strategy IIII				
				S-4-0	S-4-1	S-4-2	S-4-3	S-4-4
A	3	4	3	4	3	3	3	3
B	4	4	4	4	4	4	4	4
C	2	4	4	4	4	4	2	2
D	1	1	1	1	1	1	1	1
E	3	3	3	3	3	3	3	3
F	3	4	3	4	4	3	4	3
G	1	1	1	1	1	1	1	1
H	3	4	4	4	4	4	3	3
I	3	3	3	3	3	3	3	3
J	2	3	3	3	3	2	2	2
K	1	2	1	2	1	1	1	1
L	3	4	3	4	4	4	4	3
M	3	4	4	4	4	4	3	3
N	4	4	4	4	4	4	4	4
O	1	1	1	1	1	1	1	1
P	1	2	1	2	1	1	1	1
Q	1	1	1	1	1	1	1	1
R	2	4	3	4	4	3	3	2
S	3	3	3	3	3	3	3	3
T	1	2	1	2	1	1	1	1
U	2	3	3	3	3	3	2	2
V	2	4	3	4	4	3	3	2
W	1	2	1	2	1	1	1	1
Cost	3865	4300	3990	4300	4105	3990	3935	3865
Delay	4	0	2	0	1	2	3	4

Table 13: Percentage of delay and cost reduction

Improvement	Cost reduction (%)	Delay reduction (%)
S-4-0 / Strategy II	23.2	100
S-4-1	26.7	92.9
S-4-2 / Strategy III	28.8	85.7
S-4-3	29.7	78.6
S-4-4 / Strategy I	31	71.4

Table 14: Improvement values of optimal solutions to non-dominated sub-strategies

Algorithm	S-4-0	S-4-1	S-4-2	S-4-3	S-4-4	Average	Ranking
WCA	55.71	61.40	62.27	62.44	60.36	60.44	7
IWO	55.71	61.40	63.25	62.44	61.96	60.95	1
COA	55.71	61.40	63.25	62.44	61.96	60.95	1
SLC	54.43	61.40	63.25	62.44	61.96	60.70	6
ACO	55.71	61.40	63.25	62.06	60.89	59.81	8
PSO	48.86	60.38	61.48	62.25	59.82	58.56	9
SSA	55.71	53.80	55.98	56.25	57.68	55.89	11
MPA	61.25	61.40	63.25	62.44	61.96	60.81	5
FA	51.43	56.47	54.61	59.25	59.29	56.21	10
CS	55.71	61.40	63.25	62.44	61.96	60.95	1
DE	55.71	61.40	63.25	62.44	61.96	60.95	1
GA	58.04	54.94	53.82	51.54	41.79	52.03	13
CMA-ES	56.61	56.81	59.13	54.01	46.71	54.65	12