Integration of multi-mode resource-constrained project scheduling under bonus-penalty policies with material ordering under quantity discount scheme for minimizing project cost

M. Akhbari*

Department of Industrial Engineering, Electronic Branch, Islamic Azad University, Tehran, Iran.

Received 21 August 2019; received in revised form 24 February 2020; accepted 1 June 2020

KEYWORDS
Project scheduling; Multi-mode resource-constrained; Quantity discount problem in material ordering; Grey wolf optimizer; Coyote optimization algorithm; Genetic algorithm; Particle swarm optimization.

Abstract. In this paper, a mixed binary integer mathematical programming model is developed for integration of Multimode Resource-Constrained Project Scheduling Problem (MRCSP) under bonus-penalty policies and Quantity Discount Problem in Material Ordering (QDMO) with the objective of minimizing the total project cost. By proving a theorem, an important property of the optimum solution of the problem is found, which reduces the search space significantly compared to previous studies. Since the Resource-Constrained Project Scheduling Problem (RCPSP) belongs to the class of problems that are NP-hard, four hybrid meta-heuristic algorithms called COA-GA, GWO-GA, PSO-GA, and GA-GA are developed and tuned to solve the problem. Each of the proposed algorithms consists of outside and inside search components, which determine the best schedule and materials procurement plan, respectively. Finally, a set of standard PROGEN test problems is solved using the proposed hybrid algorithms under fixed CPU time. The results show that the COA-GA algorithm outperforms others.

© 2022 Sharif University of Technology. All rights reserved.

1. Introduction

Over the past decades, project scheduling has attracted extensive attention from both scientific and practical perspectives. The Resource-Constrained Project Scheduling Problem (RCPSP) introduced by Kelley in 1963 is one of the most widely studied cases of project scheduling problems. In its standard form, RCPSP attempts to minimize the makespan of a project by assigning a start time to each non-preemptive activity with respect to the precedence relations and the scarce renewable resource availabilities. While the makespan minimization is one of the most popular objectives, there are various other objectives such as project cost minimization, earliness and tardiness minimization, minimization of resource idle time, or maximization of project Net Present Value (NPV).

Standard RCPSP assumes that each activity can have only one execution way which is determined by a fixed duration and fixed required resources. Elmaghraby [1] made changes to this assumption by allowing several alternatives or modes in which an activity could be executed. This extension led to the introduction of well-known Multi-mode Resource-Constrained Project Scheduling Problem (MRCSP).
Slowinski [2] proposed a linear programming optimization method to solve the MRCPSP, and Talbot [3] and Patterson et al. [4] designed an approach based on an enumeration scheme. Drexel and Gruenewald [5] suggested the application of an efficient stochastic scheduling method that solves MRCPSPs. Also, Speranza and Vaccarini [6] suggested using a depth-first branch-and-bound algorithm; Hartmann and Briskorn [7] examined the algorithm correctness and accuracy and found that the algorithm might not yield an optimal solution when involving more than one renewable resource. Sprecher et al. [8] presented an exact algorithm based on the generalization of the branch-and-bound algorithm, which was proposed by Demenlemeester and Herroelen [9], to solve RCPSP. They enhanced the basic enumeration scheme by defining dominance rules to increase the performance of the algorithm.

Doctor [10] attempted to identify the most efficient heuristics in a comparative study and suggested a combination of five heuristics to solve MRCPSPs. In 1996, he developed another efficient heuristic for solving MRCPSPs which outperformed his previously employed five heuristics [11]. A year later, Kolisch and Drexel [12] proposed a local search method consisting of three phases: (1) A local search phase which tries to reach an initial feasible solution; (2) A construction phase which performs a single-neighborhood search on the sets of feasible mode assignments; and (3) An intensification phase which tries to find a schedule with an improved objective function based on the best mode assignment. Besides, they demonstrated that the problem of discovering a feasible schedule was NP-complete in the presence of at least two non-renewable resources. Moreover, Sprecher and Drexel [13] proved that exact optimization methods could not solve the MRCPSP with more than 20 activities and three modes for each activity in the logical elapsed run time.

Due to the hardness of the problem, various metaheuristic methods have been proposed in the literature. Bouleimen and Lecocq [14] employed a simulated annealing algorithm to solve this problem. In 2001, Hartmann [15] proposed a solution method based on Genetic Algorithm (GA) to solve MRCPSP. Józefowska et al. [16] developed another simulated annealing approach and compared the obtained results with those of two previous algorithms. Allahkar et al. [17] developed a new GA, extending the representation, operators, and fitness function in comparison with Hartmann’s algorithm. In their algorithm, infeasible individuals can participate in the genetic process and transmit their good characteristics to their offspring. Mika et al. [18] considered MRCSP with positive discounted cash flows and different payment models and proposed solution methods based on simulated annealing and tabu search algorithms. Zhang et al. [19] introduced a methodology for solving the MRCPSP based on Particle Swarm Optimization (PSO). Jarboui et al. [20] suggested a combinatorial PSO for solving this problem. Van Peteghem and Vanhoucke [21] presented an Artificial Immune System (AIS) for the MRCPSP. They proposed solution methods based on GA to solve preemptive and non-preemptive MRCPSPs [22]. Barresi et al. [23] developed a two-phase GA for the MRCPSP with maximum time lags. Khalilzadeh et al. [24] presented a metaheuristic algorithm based on the Tchonté and Gourgand’s modified PSO to solve MRCPSP with the objective function of minimizing the total costs of both renewable and nonrenewable resource usage. They developed a prioritization rule for activities and several improvements and local search methods. Wang and Fang [25] designed an Estimation of Distribution Algorithm (EDA) for solving the MRCPSP. Li and Zhang [26] presented another solution method for the MRCPSP based on the Ant Colony Optimization (ACO). Messelis and De Causmaecker [27] constructed an automatic algorithm selection tool for the MRCPSP. This super-algorithm chooses an algorithm from a portfolio of state-of-the-art algorithms based on the characteristics of the given instance. As demonstrated by the results, it outperformed all of the algorithms individually.

In all of the mentioned studies, if non-renewable resources (material) are presented, the availability of them throughout the project lifetime is one of the assumptions of the model. In fact, if the material planning is done before the scheduling of the project activities, it may lead to increased costs such as holding costs because of the time difference between the supply and consumption dates of the materials or the cost of delay in the projects with the bonus-penalty policies due to the lack of the materials to start the activities earlier. This study considers simultaneous multi-mode resource-constrained project scheduling and Material Ordering Problem (MOP).

Aquilano and Smith [28] assumed simultaneous material ordering and Project Scheduling Problem (PSP) first and suggested a hybrid model consisting of Critical Path Method (CPM) and Material Requirements Planning (MRP). Later, their work was improved further by Smith-Daniels and Aquilano [29] who presented a heuristic procedure for scheduling large projects subject to the availability of renewable and non-renewable resources. Smith-Daniels and Smith-Daniels [30] presented a mixed integer binary programming formulation of a Project Scheduling-Materials Ordering Problem (PSMOP) to determine an optimal schedule of both project activities and materials orders. Their research was enhanced further upon maximizing NPV as the objective and suggesting the late start date schedule [31]. Erbasi and Sepill [32] considered a heuristic method to find the trade-off between material ordering expenses and project delay.
In all the studies mentioned above, the price of the material was considered constant, while purchasing options might change according to order quantity in the real world.

Dodin and Elimmam [33] extended the previous works considering variable activity duration, the bonus-penalty policies, and material ordering quantity discounts and formulated the problem as a mixed-integer programming model. They used the conventional branch and bound algorithm to solve the instance problems with up to thirty activities.

Sajadieh et al. [34] extended the research of Dodin and Elimmam [33] by developing a solution approach based on GA so that the model could be solved for large-scale PSMPs. Tabrizi and Ghaderi [35] proposed a mixed-integer programming model for PSMP with the objective of maximizing NPV. They considered the presence of multiple suppliers offering distinctive discount strategies. They developed a GA to solve large-scale problems. Moreover, they tested the effect of inflation on the objective function value (NPV) through sensitivity analysis. In the subsequent study, Tabrizi and Ghaderi [36] developed a robust multi-objective mixed-integer programming mathematical model for PSMP. The purpose of the research was to minimize execution costs and maximize the schedule robustness. They applied the Non-dominated Sorting Genetic Algorithm II (NSGA-II) and a modified version of multi-objective differential evolution algorithm as the solution methodologies.

Zoraghi et al. [37] considered the Multi-mode, Resource-Constrained, project-Scheduling and the Material Ordering Problem (MRCSMOP) simultaneously to minimize the total costs of the project consisting of material holding cost, material ordering cost, bonus paid by the client, and penalty of possible delays. To solve this problem, they developed three hybrid meta-heuristic algorithms called PSO-GA, GA-GA, and SA-GA, each of which included two parts: (1) An outside search, for finding the best schedule and mode assignment, and (2) An inside search to determine time and quantity of orders. Shahsavar et al. [38] considered a combination of PSP and MOP with the quantity discount policies and constructed three hybrid methods, namely GA-DP, GA-SA, or GA-PSO. Zoraghi et al. [39] extended the MRCSMOP to a problem with three objectives: minimizing the makespan, maximizing the schedule robustness, and minimizing the total costs including renewable and nonrenewable resources costs. They also investigated the total quantity discount policy. They applied four multi-objective evolutionary algorithms, namely: (1) NSGA-II, (2) Strength Pareto Evolutionary Algorithm II (SPEA-II), (3) Multi-Objective Particle Swarm Optimization (MOPSO), and 4) Multi-Objective Evolutionary Algorithm based on Decomposition (MOEAD), in order to find an optimal Pareto frontier for the developed triple objective model.

Recently, Tabrizi [40] addressed simultaneous planning of the project schedule and material procurement problems and developed a bi-objective mathematical model with the goal to minimize total project costs and the environmental impacts of its execution. They applied NSGA-II and Multi-Objective Migrating Bird Optimization (MOMBO) algorithms to find solutions considering the start time of activities as well as the time and quantity of material orders to each supplier.

In this study, a combination of MRCPS and Quantity Discount Problem in Material Ordering (QDPO) is addressed. It is worth noting that this study is an extension to the problems investigated by Zoraghi et al. [37] and Shahsavar et al. [38] considering multi-mode activities, quantity discount policies, and bonus-penalty policy simultaneously in order to ensure that applications are closer to the real world. Also, according to the assumptions, by proving a theorem, a property of the optimum solution is presented for simplicity and higher efficiency of problem-solving algorithms.

The paper is organized in 8 sections. In Section 2, the problem is defined and the assumptions are introduced. The notations and the integrated model entitled to MRCPS-QDPS are fully described and mathematically formulated in Section 3. Section 4 demonstrates a mathematical proof to narrow the solutions’ search space. Section 5 provides the descriptions of the hybrid algorithms developed for the problem. Section 6 first elaborates the process of tuning the algorithm parameters using statistical techniques and, then, includes the computational results of algorithms. A sensitivity analysis comes next in Section 7 and, finally, the concluding remarks are reported in Section 8.

2. Problem definition

Consider a project consisting of non-preemptable activities each of which can be executed in multiple modes with different durations and resource requirements. These activities need to be scheduled under precedence rules and resource constraints. Each activity could start if all of its predecessors have finished, all of its renewable resources are available, and also its required non-renewable resources or materials are provided. The main assumptions include the following; the initial inventory of all items with the number of non-renewable resources is set to zero; no more than one order can be placed for a specific item in a period of time, and depending on the order quantity, there is an all-unit discount scheme to purchase some items. Also, we defined the total cost of projects as costs of renewable
and non-renewable resources plus the penalty or bonus resulting from the project tardiness or earliness.

The solving procedure for MRCPSP-QDPMO problem attempts to determine a schedule that satisfies relations and resource feasibilities as well as a material ordering plan (times and quantities of orders) with the objective of minimizing the total cost of the project.

3. Mathematical formulation

A summary of all the notations used in the model is given in Table 1. By using the above notations, the proposed Mixed Integer Programming (MIP) model can be formulated as follows:

MRCPSP-QDPMO model:

\[
\begin{align*}
\min Z & = \sum_{i=1}^{n} \sum_{m=1}^{M_i} \sum_{t=1}^{T} \sum_{p=1}^{P} r_{pim} x_{pim} x_{int} \\
& + \sum_{q=1}^{Q} \sum_{t=1}^{T} H_q I_{qt} + \sum_{q=1}^{Q} \sum_{t=1}^{T} A_q \lambda_q t \\
& + \sum_{q=1}^{Q} \sum_{t=1}^{T} \sum_{k=1}^{K} \mu_{qk} P_{qk} Q_{qt} + \varepsilon (FT - D) y \\
& - \theta (D - FT) (1 - y),
\end{align*}
\]

subject to:

\[
\begin{align*}
\sum_{m=1}^{M_i} \sum_{t=1}^{T} x_{int} & = 1 \quad \forall i \in \{0,1,\ldots,n\}, \\
\text{pred}_{ij} \sum_{m=1}^{M_i} \sum_{t=1}^{T} t x_{int} & \leq \sum_{m=1}^{M_i} \sum_{t=1}^{T} (t - d_{jm}) x_{jmt} \\
& \forall i, j \in \{0,1,\ldots,n\}, \\
\sum_{m=1}^{M_i} \sum_{t=1}^{T} t x_{int} & \leq FT \quad \forall i \in \{0,1,\ldots,n\}, \\
\sum_{i=1}^{n} \sum_{m=1}^{M_i} \sum_{b=t}^{t+d_{im}-1} x_{int} & \leq r e_{sp} \\
& \forall i \in \{0,1,\ldots,T\}, \forall p \in \{1,2,\ldots,P\}, \\
I_{qt} & = I_{qt-1} + Q_{qt} - \sum_{i=1}^{n} \sum_{b=1}^{t+d_{im}-1} \sum_{m=1}^{M_i} a_{qim} x_{imt} \\
& \forall i \in \{0,1,\ldots,T\}, \forall q \in \{1,2,\ldots,Q\}, \\
I_{q0} & = 0 \quad \forall q \in \{1,2,\ldots,Q\},
\end{align*}
\]

\[
\begin{align*}
Q_{qt} & \leq \lambda_q t B i g M \\
& \forall t \in \{0,1,\ldots,T\}, \forall q \in \{1,2,\ldots,Q\}, \\
1 - (1 - \lambda_q t) B i g M & \leq Q_{qt} \forall t \in \{0,1,\ldots,T\}, \\
& \forall q \in \{1,2,\ldots,Q\}, \\
\mu_{qk} O_{k-q} & \leq Q_{qt} \forall t \in \{0,1,\ldots,T\}, \\
& \forall q \in \{0,1,\ldots,Q\}, \forall k \in \{1,2,\ldots,K_q\}, \\
\mu_{qk} Q_{qt} & \leq O_{qk} \forall t \in \{0,1,\ldots,T\}, \\
& \forall q \in \{0,1,\ldots,Q\}, \forall k \in \{1,2,\ldots,K_q\}, \\
\sum_{k=1}^{K} \mu_{qk} & = 1 \quad \forall t \in \{0,1,\ldots,T\}, \\
& \forall q \in \{0,1,\ldots,Q\}, \\
FT & (1 - y) \leq D, \\
D y & \leq FT, \\
I_{qt} & \geq 0 \quad \forall t \in \{0,1,\ldots,T\}, \forall q \in \{1,2,\ldots,Q\}, \\
x_{int} & \in \{0,1\} \quad \forall i \in \{0,1,\ldots,n\}, \\
y & \in \{0,1\}, \\
\lambda_q t & \in \{0,1\} \quad \forall t \in \{0,1,\ldots,T\}, \forall q \in \{1,2,\ldots,Q\}, \\
\mu_{qk} & \in \{0,1\} \quad \forall t \in \{0,1,\ldots,T\}, \\
& \forall q \in \{0,1,\ldots,Q\}, \forall k \in \{1,2,\ldots,K_q\}.
\end{align*}
\]

The objective function (1) minimizes the total cost of the project considering the following six factors:

1. The renewable resources cost;
2. The inventory holding cost;
3. The material ordering cost;
4. Procurement cost;
5. The project tardiness penalty; and
6. The project earliness bonus.

Eq. (2) states that every activity is assigned exactly one mode and exactly one finishing time. Constraint (3) ensures the precedence relations between activities. Inequality (4) ensures that the makespan is the maximum of its activities’ finish times. Constraint (5) enforces the renewable resource constraints.
Table 1. Summary of the notations.

| Indices:                          | Description                                    |
|-----------------------------------|------------------------------------------------|
| $i, j \in \{0, 1, \ldots, n\}$   | Index of project activities                    |
| $t, b \in \{1, \ldots, T\}$      | Index of time periods                          |
| $p \in \{1, 2, \ldots, p\}$      | Index of renewable resources                   |
| $q \in \{1, 2, \ldots, Q\}$      | Index of materials                             |
| $m \in \{1, \ldots, M_i\}$       | Index of modes of activity for the $i$th activity|
| $k \in \{1, \ldots, K_q\}$       | Index of price breakpoint for material $q$      |

| Parameters:                      | Description                                    |
|----------------------------------|------------------------------------------------|
| $d_{im}$                         | Duration of activity $i$ executed in mode $m$  |
| $pred_{ij}$                      | If activity $i$ is the predecessor of activity $j$ |
|                                  | Otherwise                                      |
| $req_{pt}$                       | Number of units of renewable resource $p$ available in period $t$ |
| $rpm_i$                          | Number of units of renewable resource $p$ required by activity $i$ executed in mode $m$ |
| $nqim$                           | Number of units of non-renewable resource $q$ required by activity $i$ executed in mode $m$ |
| $cr_p$                           | Cost of renewable resource $p$ per period       |
| $H_q$                            | Inventory holding cost per unit of material $q$ per period |
| $A_q$                            | Ordering cost per replenishment of material $q$ |
| $BigM$                           | Big number                                     |
| $P_q$ =                           | Purchasing cost of material $q$ under AUD scheme |
| $P_{q1}$                         | $1 \leq Q_{qt} \leq O_{q1}$                    |
| $P_{q2}$                         | $O_{q1} + 1 \leq Q_{qt} \leq O_{q2}$          |
| $\vdots$                         | $\vdots$                                       |
| $P_{qK}$                         | $O_{qK-1} + 1 \leq Q_{qt}$                     |
| $\omega$                        | Penalty per period of delay                    |
| $\vartheta$                      | Bonus per period of early completion            |
| $D$                              | Due date of the project                        |
| $T$                              | Planning horizon                                |
| $\pi$                            | Project income at the end of the project        |

| Binary variables:                | Description                                    |
|----------------------------------|------------------------------------------------|
| $x_{imt}$                        | If activity $i$ is performed in mode $m$ and finished in time period $t$ |
|                                  | Otherwise                                      |
| $y$                              | If the project is finished after due date      |
|                                  | Otherwise                                      |
| $\lambda_{qt}$                   | If material $q$ is ordered in period $t$       |
|                                  | Otherwise                                      |
| $\mu_{qkt}$                      | If material $q$ is purchased at price breakpoint $k$ in period $t$ |
|                                  | Otherwise                                      |

| Continuous variables:            | Description                                    |
|----------------------------------|------------------------------------------------|
| $FT$                             | Malespan                                       |
| $Q_{qt}$                         | Ordered quantity of material $q$ in period $t$ |
| $l_{qt}$                         | Inventory level of material $q$ at the end of period $t$ |
at time interval $t$. Eq. (6) determines the inventory level of materials at the end of each period of the project. Based on Eq. (7), no inventory level is available in the advent of the project. Inequalities (8) and (9) ensure that the order quantity of each material for each period can be provided if it is ordered in that period. Constraints (10) and (11) determine the discount range for each material in each period. Eq. (12) ensures that only one price is allocated to each material in each period. Inequalities (13) and (14) determine the earliness or tardiness of the project. Finally, Constraints (15)–(19) denote the domain of the variables.

4. Optimum solution property

This section demonstrates an important property of the optimum MRCPSP-QDPMO solution, which can reduce the search space significantly. The following theorem establishes that in the optimum solution, only the start time of activities can be used to order tasks.

**Theorem 1.** In the optimum MRCPSP-QDPMO solution, the required quantity of each material in each period is ordered at the beginning of that time period or is merged with the last order of the material.

In other words, if the required quantity of a material is ordered at any time following the last replenishment time(s) and before $t$, the total cost of the project increases.

**Proof.** Let $Q$ be the required quantity of materials in the period $t$ for the optimum MRCPSP-QDPMO solution. Therefore, we have:

$$Q = \sum_{i=1}^{n} \sum_{m=1}^{M_i} a_{im} x_{im(t+d_m-1)}.$$  \hspace{1cm} (20)

Let $PTC^b_i$ be the total cost of $Q$ if $i$ is ordered in period $b$ and consumed in period $t$ ($s \leq b \leq t$).

Thus, $TC^b_i$ indicates the total cost for $Q$ which is ordered and consumed in period $t$:

$$TC^b_i = A_i + P_i Q,$$ \hspace{1cm} (21)

and $TC^e_i$ is the Total Cost of $Q$ if it is ordered in period $e$, $s < e < t$:

$$TC^e_i = A_i + P_i Q + \sum_{j=1}^{i-e} H_j Q, \hspace{1cm} s < e < t,$$ \hspace{1cm} (22)

and:

$$TC^b_i < TC^e_i.$$ \hspace{1cm} (23)

Thus, early ordering in period $e$, when $s < e < t$, is not suggested. However, the ordering of $Q$ in period and buying the material in large quantities may bring about better prices and reduce the total cost ($TC^e_i$) in comparison with $TC^b_i$. Thus, the best ordering of the required material quantity $q$ in a specific period is either $t$ or $s$.

**Remark 1.** Based on Theorem 1, in the optimum solution, only the start times of activities can be used as the ordering points.

The application of this important property of the optimum solution significantly reduces the search space of the algorithm in comparison with the methods used by Tabrizi and Ghaderi [36], Zoraghi et al. [37], Shahsavar et al. [38], and Zoraghi et al. [39], in which all time periods are considered as the possible ordering points.

5. Solving methodologies

This discusses four hybrid meta-heuristic algorithms that have been proposed to solve the problem. Each hybrid solving procedure consists of two optimization levels: an outside search level and an inside search level in which the MRCPSP solution and QDPMO solution parts of the problem are generated, respectively. The MRCPSP solution part represents the best priority of the project activities and their assignment mode, and the QDPMO solution part represents its best ordering plan.

5.1. Solution representation

MRCPSP solution part is represented by a 2-dimensional matrix, $Y = [Y_{ji}]_{2 \times (n)}$, where $n$ is the number of project activities. Each $Y_{ji}$ is generated randomly at an interval of $[0, 1]$. The first row of $Y$ relates to the sequence in which the activities are scheduled, and the second row contributes to the mode of activities and each $Y_{2i}$ is converted to a mode based on $m_i = [Y_{2i} \times M_i] + 1$, where $m_i$ is the execution mode of activity $i$.

The activity sequence list and the activity mode list act as the inputs of the Serial Scheduling Scheme (SSS). In each step of the SSS, the activity with no unscheduled predecessor and the least sequence value is selected to be scheduled based on its mode [41].

Figure 1 represents an example of the MRCPSP solution part of a project with five activities that can be executed in one of three possible modes.

QDPMO solution part is represented by a 2-dimensional matrix, $X = [X_{ij}]_{Q \times U}$, where $Q$ is the number of project material types and $U$ is the number of unique activities’ start time based on the schedule generated by SSS. Each $j \in \{1, 2, ..., U\}$ is a period of time in a project in which at least one activity starts. $X_{ij}$ is a 0-1 integer. If it equals 1, it means the required quantity of material $q$ in the $j$th period will be ordered in the same period. Otherwise, if $X_{ij}$ equals 0, it means the required quantity of material $q$ in the $j$th period will be ordered with the last ordering of
material $q$. $X_{qj}, q \in \{1, 2, \ldots, Q\}$ and $j \in \{2, 3, \ldots, U\}$ are generated randomly, and since there is not any ordering point before the first start time, each $X_{q1}, q \in \{1, 2, \ldots, Q\}$ equals 1.

5.2. Solution procedure

The pseudo-code of the solution procedure is shown in Figure 2.

The characteristics of these algorithms are explained in the following subsections.

5.3. The outside search algorithms

5.3.1. Coyote Optimization Algorithm (COA)

COA proposed by Pirezan and Coelho [42] is a recent population-based algorithm that is inspired by the social structure and experience exchange among coyotes in nature.

In the COA, the population of coyotes is divided into $N_p$ packs with $N_c$ coyotes in each pack. Therefore, the total population is $N_p \times N_c$. The social conditions of the coyotes (soc) are the solutions to the optimization problem and its adaptation to the environment is the cost of the objective function. Thus, each soc is the set of decision variables as follows:

$$soc^t_c = [Y_{ji}^t]_{2 \times n},$$

where $t$ is the current iteration, $c$ and $p$ indicate the related coyote and pack, respectively, and $n$ is the number of project activities and each $Y_{ji}^t, i \in \{1, 2, \ldots, n\}$ and constitute the social conditions in

**Figure 1.** An example of the Multimode Resource-Constraint Project Scheduling Problem (MRCPSP) solution part (Zoraghi et al. (2016a) [37]).

**Figure 2.** The pseudo-code of the solution procedure.
which the random variable is calculated as follows:

\[ Y_{ji} = lb_{ji} + r_{ji} (ub_{ji} - lb_{ji}). \]  

(25)

where \( lb_{ji} \) and \( ub_{ji} \) are lower and upper bounds of \( Y_{ji} \) (in this problem, \( lb_{ji} = 0 \) and \( ub_{ji} = 1 \)), and \( r_{ji} \) is a random number between \([0, 1]\); therefore, \( Y_{ji} = r_{ji} \).

The adaptation of coyotes in the respective current social conditions \( \left(fit^{c,t}_{i}\right) \) is evaluated based on the objective function \( f(x) \):

\[ fit^{c,t}_{i} = f (soc^{p,t}_{i}). \]  

(26)

Initially, coyotes are randomly assigned to the packs; however, the coyotes sometimes leave their pack and join another pack by chance. The coyote eviction probability depends on the number of coyotes in the pack and is calculated as follows:

\[ P_e = 0.005N^2_c. \]  

(27)

Based on Eq. (27), the number of coyotes per pack is at most 14. This process helps COA to diversify the interaction between all the coyotes and simulates a cultural exchange in the global population.

The birth of new coyotes is written as a combination of the social conditions of parents (randomly chosen) plus an environmental influence such that:

\[ pup^{p,t}_{k} = \begin{cases} 
\text{soc}^{p,t}_{r_1,k}, & r_{ndk} < P_{or} \text{or } k = k_1 \\
\text{soc}^{p,t}_{r_2,k}, & r_{ndk} \geq P_o + P_{or} \text{or } k = k_2 \\
R_k, & \text{otherwise}
\end{cases} \]  

(28)

where \( r_1 \) and \( r_2 \) are random coyotes from the \( p \)th pack; \( k_1 \) and \( k_2 \) are two random dimensions of the problem. \( P_{or} \) is the scatter probability, while \( P_o \) is the association probability. Moreover, \( R_k \) is a random number inside the \( k \)th decision variable bound and \( r_{ndk} \) is a random number inside \([0, 1]\). \( P_o \) and \( P_{or} \) are defined as follows:

\[ P_o = \frac{1}{D}, \]  

(29)

\[ P_{or} = \frac{1 - P_o}{2}, \]  

(30)

where \( D \) is the size of the problem which is \( 2n \) in this study. It is worth noting that \( n \) is the number of project activities.

Let \( age^{p,t}_{c} \) be the age of the \( c \)th coyote in the \( p \)th pack at the \( t \)th instant of time. The higher the coyote’s age, the higher the mortality probability.

In order to keep the population size constant, COA performs this procedure: if the new pup has the worst objective value in the pack, it will not have the chance to live. Otherwise, in the group of coyotes with the adaption value worse than the pup, the oldest one dies; and if there is more than one coyote with the same highest age in this group, the coyote with the least adaption value dies.

In this algorithm, \( \text{alpha}^{p,t} \) is the fittest solution in the \( p \)th pack at the \( t \)th instant of time.

Due to the evident signs of swarm intelligence in this species, the COA assumes that the coyotes are sufficiently organized to share the social conditions. Thus, the COA aggregates all information concerning the coyotes and the cultural tendency of each pack is computed as the median social conditions of all coyotes as follows:

\[ cut^{p,t}_{k} = \begin{cases} 
\frac{O^{p,t}_{N_c+1/2}}{O^{p,t}_{N_c} + O^{p,t}_{N_c+1/2}} & N_c \text{ is odd} \\
\frac{O^{p,t}_{N_c} + O^{p,t}_{N_c+1}}{2} & \text{otherwise}
\end{cases} \]  

(31)

where \( O^{p,t}_{j} \) represents the \( j \)th rank of the ordered \( j \)th social condition of all coyotes in the \( p \)th pack at the \( t \)th instant of time.

In each iteration of COA, every coyote is updated using alpha and the cultural tendency of its pack as follows:

\[ new - soc^{p,t}_{c} = soc^{p,t}_{c} + r_1 (\text{alpha}^{p,t} - soc^{p,t}_{r_1}) \]  

\[ + r_2 (cut^{p,t}_{k} - soc^{p,t}_{r_2}), \]  

(32)

where \( r_1 \) and \( r_2 \) are two random numbers inside \([0, 1]\), and \( r_1 \) and \( r_2 \) are two random coyotes in the pack.

The next population would be updated as follows:

\[ soc^{p,t+1}_{c} = \begin{cases} 
new - soc^{p,t}_{c} & f(new - soc^{p,t}_{c}) < f(soc^{p,t}_{c}) \\
soc^{p,t}_{c} & \text{otherwise}
\end{cases} \]  

(33)

The pseudo code of the COA algorithm is shown in Figure 3.

5.3.2. Grey Wolf Optimizer (GWO)

GWO described by Mirjalili et al. [43] is a population-based algorithm which is inspired by the social hierarchy and hunting behavior of grey wolves in nature. Social hierarchy is simulated by defining four types of grey wolves such as alpha, beta, delta, and omega. Moreover, hunting behavior is simulated by implementing the three main steps of hunting, searching for prey, and encircling and attacking prey [44].

Hierarchy of grey wolves

The leaders of wolves include a male and a female, called alphas. They typically responsible for making decisions about hunting, sleeping time and place, and so on. The wolves at the second level are called beta, which are the successor of alphas and help them to make a decision. The lowest on the hierarchy is called omega that plays the role of scapegoat. Omega is able to satisfy all the other dominat wolves. The third level on the hierarchy is called delta, which is subordinate to alpha and beta, but they dominate the omega. Delta should scout to protect and ensure the safety of the whole group of grey wolves.

In the mathematical model of GWO, alpha (\( \alpha \)
is considered as the fittest solution. Consequently, beta (β) and delta (δ) are the second and third best solutions, respectively. The remaining solutions are omega (ω). In the GWO algorithm, the hunting (optimization) is guided by α, β, and δ. The ω wolves follow them.

**Mathematical model of hunting behavior**

Another amusing gregarious characteristic of gray wolves is their flock hunting process. In their hunting process, there are three main phases [45]: first, tracking, chasing, and approaching the prey; then, pursuing, encircling, and harassing the prey until it stops moving and finally, attacking the prey.

In order to mathematically model the encircling behavior, the following formulation is proposed:

\[ Y_k (t + 1) = Y_p (t) - A | CY_p (t) - Y_k (t) |, \]  

where \( t \) is the current iteration, \( Y_k \) is the \( k \)th grey wolf position vector, and \( Y_p \) is the position vector of the prey. \( A \) and \( C \) are coefficient vectors that are calculated as follows:

\[ A = 2a. r_1 - a, \]  
\[ C = 2. r_2, \]

where \( r_1 \) and \( r_2 \) are random vectors at \([0,1]\); components of \( a \) linearly decrease from 2 to 1 over the course of iterations.

In order to mathematically simulate the hunting behavior of grey wolves to determine the location of the optimum (prey), the first three best solutions obtained \((Y_a, Y_b, \text{ and } Y_c)\) are saved and others including omegas should update their positions according to the positions of the best search agents based on the following formulas:

\[ Y_1 = Y_a - A_1 | C_1 Y_a - Y |, \]
\[ Y_2 = Y_b - A_2 | C_2 Y_b - Y |, \]
\[ Y_3 = Y_c - A_1 | C_3 Y_c - Y |. \]

\[ Y (t + 1) = \frac{Y_1 + Y_2 + Y_3}{3}. \]  

When the prey stops moving, the grey wolves attack it and finish the hunting. As mentioned earlier, the values of the components of \( a \) are reduced linearly over the course of iterations, and based on Eq. (35), the fluctuating range of \( A \) also decreases; therefore, approaching the prey is mathematically modeled. When random values of \( A \) are in the range of \([-1,1] \) or \(|A| \leq 1\), the next position of a search agent can be in any position between its current position and the position of the prey and it forces the wolves to attack the prey.

In order to mathematically model divergence, the random values of \( A \) with \(|A| > 1\) oblige the search agent to diverge from the prey to find a better prey hopefully. This highlights exploration and allows the GWO algorithm to search globally.

Another component of GWO that helps exploration is. This component emphasizes (when \(|C| > 1\)) or deemphasizes (when \(|C| < 1\)) the effect of prey position in defining the distance in Eq. (34).

The pseudocode code of the GWO algorithm is given in Figure 4.
5.3.3. The Particle Swarm Optimization (PSO)

PSO described by Eberhart and Kennedy [46] is a swarm-based algorithm that is inspired by the social behavior of bird flocking and fish schooling. Each particle in PSO has its own velocity, position vector, and a fitness value determined by the objective function. Moreover, PSO has memory and the previous particle information is reflected in it and is used in general backpropagation processes. Hence, a continuous progression towards the global optimum point is provided. The following equations demonstrate how the velocity and new position of each particle are calculated in every iteration:

\[
V(t + 1) = \omega V(t) + c_1 r_1 (Y_{\text{best}}(t) - Y(t)) + c_2 r_2 (Y_{\text{global}}(t) - Y(t)),
\]

\[
Y(t + 1) = Y(t) + V(t + 1),
\]

where \( \omega \) is inertia weight, \( V(t) \) is former velocity, and \( c_1 \) and \( c_2 \) are acceleration constants that control how the particles approach the local best and global best positions, respectively. \( r_1 \) and \( r_2 \) are random values within \( [0, 1] \). \( Y(t) \) is the former position. \( Y_{\text{best}}(t) \) is the best position of the particle and \( Y_{\text{global}}(t) \) is the global best position of the whole swarm.

5.3.4. The genetic algorithm (GA)

GA is the evolutionary methodology introduced by Holland in 1975 and is based on natural selection and genetics. A GA works on an initial population; it selects parents, applies crossover and mutation operators, and evaluates the children. The goal is to successively produce better solutions by selecting the better ones among existing solutions with a higher chance for recombination. For the description about GA used in our study, we refer readers to the work of Zoraghi et al. [37].

5.3.5. The inside search: GA

This section describes another GA that is applied at the second level of hybrid algorithms to discover the best ordering policy of the schedule.

First QDPMO solution part \( X = [X_{ij}]_{Q \times U} \), which was explained in Section 5.1, is encoded to a chromosome as shown in Figure 5 (as mentioned before, \( X_{ij} \) is equal to 1; therefore, it is omitted from the chromosome).

This algorithm begins with a random population of initial chromosomes and continues to produce new generations using crossover and mutation operators.

In each generation, a set of chromosomes is generated through a recombination process. To this end, the roulette wheel selection pattern is used to select parents and two kinds of crossover operators are applied with the same probability to produce offspring (Figure 3).

In the one-point crossover, a number \( r \) is randomly generated in the range \( \{1, 2, \ldots, Q(U-1)\} \). The data for each offspring is a mixed combination of one parent (before \( r \)) and another parent (after \( r \)). Figure 6(a) shows a simple example of this operator.
Also, the two-point crossover in which two points \( r_1, r_2 \in \{1, 2, ..., Q(U - 1)\} \) of the parents are randomly selected such that \( r_1 < r_2 \) is employed. The columns between these two points of parents are exchanged and two offspring are created. Figure 6(b) shows a simple example of this operator.

In addition, a single-point mutation is applied with a certain probability that changes 1 to 0, and vice versa.

6. Computational experiments

In this step, the model is implemented using hybrid algorithms to demonstrate their performance and applicability in practice. To do so, first, a parameter tuning procedure is utilized based on Taguchi method to provide robust solutions. Next, the developed mathematical model is solved using the BARON solver in GAMS and the obtained results are compared with those obtained from hybrid algorithms. The comparisons were carried out for problems with different sizes to point to the performance of meta-heuristics. All computations were performed on a Core (TM) i7-2600K PC with 3.4 GHz CPU speed and 8 GB of RAM. MATLAB software was employed to code the proposed meta-heuristics.

6.1. Parameter tuning

The performance of meta-heuristic algorithms is highly related to the values of their structural parameters (factors). In this regard, there are different ways for tuning the parameters in order to improve the robustness of the algorithms. In this study, we applied the Taguchi method, which can consider a large number of decision variables with a small number of experiments [47].

In Taguchi method, the factors are separated into two main categories: controllable and noise factors. However, noise factors do not have direct effect on results and their removal is impractical and often impossible. The Taguchi method attempts to minimize the effect of noise and determine the optimal level of significant controllable factors to achieve robustness.

Taguchi method uses a special set of standard arrays called orthogonal arrays. The orthogonal arrays determine how to conduct the minimal number of experiments that could give the full information of all the factors that affect the response variables. An appropriate orthogonal array is determined based on the number of factors and their levels. In the orthogonal array, the columns correspond to the parameters, the entries in the columns correspond to the levels of the factors, and the rows correspond to the experiments.

This method uses the Signal-to-Noise (S/N) ratio to reflect the extant variation in the response variable, where a higher S/N ratio implies smaller variation. Eq. (43) shows the calculation for S/N ratio in the minimization (the smaller-the-better type) problem:

\[
SN_j = -10 \log \left( \frac{\sum_{i=1}^{n} y_{ij}^2}{n} \right),
\]

where \( y_{ij} \) is the response value of the \( i \)th observation at trial \( j \) of the orthogonal array and \( n \) is the number of observations in each trial.

In this study, the relative deviation percentage for a single problem \( i \) is assumed at trial \( j \) \( (RDP_{ij}) \) of the orthogonal array as the response value \( (y_{ij}) \), defined as follows:

\[
RDP_{ij} = \frac{Z_{ij}^* - \min \left( Z_{ik}^* \right)}{\min \left( Z_{ik}^* \right)},
\]

where \( Z_{ij}^* \) is the objective value for the problem \( i \) at the trial \( j \) of the orthogonal array.

We tuned the hybrid algorithms’ parameters in two steps:

**Step 1: Tuning of the inside search algorithm (GA)** Inside search uses the schedule of activities and their modes generated by outside search to solve the MOP. Therefore, we assumed some problems with different sizes and their schedules and modes of activities and employed Taguchi method to tune the parameters of inside search algorithm (GA).
Table 2. Inside GA factors and their levels.

| Factors | Description       | Levels |
|---------|-------------------|--------|
| $P_c$   | Crossover probability of GA | 0.7 0.8 0.9 |
| $P_m$   | Mutation probability of GA | 0.1 0.2 0.3 |
| $MG$    | Population size of GA | 10 30 60 |
| NGA     | Maximum iteration of GA | 100 150 200 |

Table 3. The orthogonal array for parameters of the inside Genetic Algorithm (GA) and results.

| Trial | $P_c$ | $P_m$ | $MG$ | NGA | RDP | S/N |
|-------|-------|-------|------|-----|-----|-----|
| 1     | 1     | 1     | 1    | 1   | 0.038 | 28,312 |
| 2     | 1     | 2     | 2    | 2   | 0.023 | 32,670 |
| 3     | 1     | 3     | 3    | 3   | 0.014 | 36,002 |
| 4     | 2     | 1     | 2    | 3   | 0.014 | 35,330 |
| 5     | 2     | 2     | 3    | 1   | 0.019 | 34,507 |
| 6     | 2     | 3     | 1    | 2   | 0.017 | 30,682 |
| 7     | 3     | 1     | 3    | 2   | 0.006 | 39,702 |
| 8     | 3     | 2     | 1    | 3   | 0.012 | 37,007 |
| 9     | 3     | 3     | 2    | 1   | 0.011 | 37,728 |

Table 4. Optimum factor levels and values for inside Genetic Algorithm (GA) algorithm.

| Factors | Inside search |
|---------|---------------|
| $P_c$   | 0.9           |
| $P_m$   | 0.3           |
| $MG$    | 10            |
| NGA     | 150           |

Table 5. Optimum factor levels and values for Coyote Optimization Algorithm (COA) algorithm.

| Factors | Iter | $N_p$ | $N_c$ |
|---------|------|-------|-------|
|          | Optimal levels | 200 | 15 | 14 |

Table 6. Optimum factor levels and values for GWO algorithm.

| Factors | Iter | Search Agents_no |
|---------|------|------------------|
|          | Optimal levels | 200 | 30 |

Table 7. Optimum factor levels and values for Particle Swarm Optimization (PSO) algorithm.

| Factors | Iter | Swarm size | $c_1$ | $c_2$ | $\omega$ |
|---------|------|------------|-------|-------|---------|
|          | Optimal levels | 400 | 30 | 2.9 | 1.9 | 0.65 |

Table 8. Optimum factor levels and values for Genetic Algorithm (GA) algorithm.

| Factors | Iter | $S/N$ |
|---------|------|-------|
| $P_c$   | 0.9  |       |
| $P_m$   | 0.3  |       |
| $MG$    | 20   |       |
| NGA     | 200  |       |

Step 2: Tuning of the outside search algorithms

This section describes the Taguchi method for tuning the parameters influencing the performance of the inside algorithm, each at three levels. Table 2 presents four parameters affecting the performance of the GA and their different levels. Based on Taguchi method, at first, an orthogonal array, L9(3^4), is selected as the fittest design. Then, the experiments are performed for a set of 5 problems from the PSPLIB library with 10, 16, 20, 30, and 60 activities. However, other essential assumptions such as holding costs, ordering costs, material prices, and a schedule for activities and their modes for each problem are added to complete the matter.

The data in Table 3 show the orthogonal array of parameters for the Inside GA and observations’ results including RDP’s mean and $S/N$ ratio in each trial.

In Figure 7, there is a $S/N$ ratio plot resulting from the inside GA and its best parameter levels are shown in Table 4.

The same approach was applied to tune parameters for outside search algorithms and the optimum levels are given in Tables 5–8.

Performance evaluation of the hybrid algorithms uses these optimal parameter values.

6.2. Performance analysis of the hybrid algorithms

Initially, the performance of the aforementioned meta-heuristics to solve the MRCPSP-QDPMO problem is considered for a typical instance, and the results are compared with that of BARON solver.

Consider a project including 16 activities with an
Table 9. The list of activities with their durations and required renewable and non-renewable resources with respect to each mode.

| Activity | Duration | Mode 1 Required resources | Mode 2 Required resources | Mode 3 Required resources |
|----------|----------|---------------------------|---------------------------|---------------------------|
|          |          | R1* R2 NR1* NR2           | R1  R2 NR1  NR2           | R1  R2 NR1  NR2           |
| 1        | 1        | 9 3 10 5                 | 5 2 12 3                 | 3 3 1 12 3               |
| 2        | 2        | 6 0 3                    | 4 4 0 3                 | 5 3 0 0 2               |
| 3        | 2        | 5 10 6 6                | 3 3 5 5 6               | 3 6 4 4 0               |
| 4        | 2        | 10 8 0 4                | 4 8 0 4                 | 6 6 8 0 3               |
| 5        | 8        | 0 2 8 2                 | 10 8 1 7 5              | 10 5 2 8 4             |
| 6        | 1        | 0 5 8 0                 | 3 0 5 4 0              | 5 0 4 3 0             |
| 7        | 7        | 7 5 0 5                 | 9 5 3 0 4              | 10 1 0 2 0             |
| 8        | 1        | 7 9 0 7                 | 2 5 5 3 0             | 4 2 2 3 0             |
| 9        | 5        | 5 6 4 0                 | 7 5 3 2 1              | 10 3 2 3 1             |
| 10       | 4        | 8 0 0 8                 | 6 6 0 0 6              | 8 5 0 0 5             |
| 11       | 2        | 10 0 0 8                | 2 0 8 0 3              | 5 12 0 0 6             |
| 12       | 5        | 2 10 7 0                | 9 6 5 4 1              | 9 5 4 6 3             |
| 13       | 6        | 8 8 4 1                 | 8 12 5 5 2              | 10 10 4 15 0           |
| 14       | 2        | 7 8 6 0                | 3 5 4 10 0             | 5 3 3 3 0             |
| 15       | 5        | 6 7 0 3                 | 6 3 5 5 3             | 9 6 2 15 3             |
| 16       | 2        | 5 2 5 1                 | 4 3 1 5 0             | 7 5 0 2 1             |

*R: Renewable resource; NR: Non-Renewable resource.

Figure 8. AON precedence network of the typical instance.

AON precedence network as shown in Figure 8. Each activity has three possible execution modes.

Table 9 presents a list of activities with their durations and required renewable and non-renewable resources separately for each mode. Project’s due date is at the end of time unit 25. Delay penalty is 10,000 per time unit and earliness bonus is $15,000 per time unit.

Table 10 presents the costs of materials and Table 11 shows the prices for each of them at different break points. Table 12 includes the available number of renewable resources and their cost per time unit. Tables 13–15 show the results after implementing the meta-heuristics and BARON.

According to the results in Table 13, COA-GA and GWO-GA outperformed BARON significantly.

For the purpose of evaluating the performance of four solving algorithms, the project generator software (PROGEN) developed by Kolisch and Sprecher in 1996 was used, which resulted in 270 numerical MRCPSP problems having 10, 12, 14, 16, 18, 20, 30, 60, and 90 non-dummy activities; each one can be executed in one of three modes. According to Table 16, the problems differ with respect to the number of renewable and nonrenewable resources. For each of 27 classes of problems, 10 instances were generated, resulting in 270 test problems. To solve each problem, every hybrid

Table 10. The costs of materials.

| Non-renewable resources | 1 | 2 |
|-------------------------|---|---|
| Holding cost ($) per time unit | 100 | 1,200 |
| Ordering cost ($) | 1,000 | 1,500 |

Table 11. The prices of materials at different break points.

| Non-renewable resources | 1 | 2 |
|-------------------------|---|---|
| Range | Price ($) | Range | Price ($) |
| $1 \leq O \leq 10$ | 700 | $1 \leq O \leq 5$ | 3,200 |
| $11 \leq O \leq 20$ | 650 | $6 \leq O \leq 10$ | 2,800 |
| $21 \leq O \leq 30$ | 630 | $11 \leq O$ | 2,500 |
| $31 \leq O \leq 40$ | 600 | $41 \leq O$ | 500 |

*O: Order quantity

Table 12. The cost and maximum level of each renewable resource.

| Renewable resources | 1 | 2 |
|---------------------|---|---|
| $8$/time unit | 100 | 200 |
| Max level | 19 | 15 |
Table 13. The costs and the finish time of the problem found by the metaheuristics and BARON.

| Method                  | BARON | COA-GA | GWO-GA | PSO-GA | GA-GA |
|-------------------------|-------|--------|--------|--------|-------|
| Total cost (\$)         | 227,650 | 223,000 | 223,000 | 233,400 | 235,500 |
| FT                      | 23     | 23     | 23     | 24     | 24    |
| Renewable resource cost | 93,500 | 90,600 | 90,600 | 92,500 | 93,400 |
| Holding cost            | 7,300  | 700    | 700    | 1,100  | 5,100 |
| Ordering cost           | 16,000 | 14,000 | 14,000 | 14,000 | 15,000 |
| Procurement cost        | 130,850 | 137,700 | 137,700 | 135,800 | 132,400 |
| Penalty (Bonus)         | (20,000) | (20,000) | (20,000) | (10,000) | (10,000) |

Table 14. The schedules of the problem found by the metaheuristics and BARON.

| Activity | Start time | Mode | Start time | Mode | Start time | Mode | Start time | Mode | Start time | Mode |
|----------|------------|------|------------|------|------------|------|------------|------|------------|------|
| 1        | 0          | 2    | 0          | 2    | 0          | 2    | 0          | 2    | 0          | 2    |
| 2        | 0          | 1    | 0          | 1    | 0          | 1    | 0          | 1    | 0          | 1    |
| 3        | 0          | 1    | 0          | 1    | 0          | 1    | 0          | 1    | 0          | 1    |
| 4        | 2          | 1    | 2          | 1    | 2          | 1    | 2          | 1    | 2          | 1    |
| 5        | 4          | 1    | 4          | 1    | 4          | 1    | 4          | 1    | 4          | 1    |
| 6        | 4          | 2    | 10         | 1    | 10         | 1    | 4          | 1    | 10         | 1    |
| 7        | 2          | 2    | 2          | 2    | 2          | 2    | 2          | 2    | 2          | 2    |
| 8        | 17         | 1    | 11         | 2    | 11         | 2    | 5          | 3    | 11         | 3    |
| 9        | 11         | 1    | 11         | 1    | 11         | 1    | 12         | 1    | 12         | 1    |
| 10       | 17         | 1    | 16         | 1    | 16         | 1    | 17         | 1    | 14         | 3    |
| 11       | 15         | 2    | 14         | 2    | 14         | 2    | 15         | 2    | 18         | 1    |
| 12       | 16         | 1    | 16         | 1    | 16         | 1    | 17         | 1    | 14         | 1    |
| 13       | 10         | 1    | 4          | 1    | 4          | 1    | 9          | 1    | 4          | 1    |
| 14       | 21         | 1    | 21         | 1    | 21         | 1    | 22         | 1    | 22         | 1    |
| 15       | 17         | 2    | 16         | 2    | 16         | 2    | 17         | 2    | 20         | 1    |
| 16       | 21         | 1    | 21         | 1    | 21         | 1    | 22         | 1    | 20         | 1    |

Table 15. The procurement plan of the problem obtained from the metaheuristics and BARON.

| Date | Material 1 | Material 2 | Material 1 | Material 2 | Material 1 | Material 2 | Material 1 | Material 2 | Material 1 | Material 2 |
|------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|
| 1    | 18         | 0          | 18         | 0          | 18         | 0          | 18         | 0          | 18         | 0          |
| 2    | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 3    | 0          | 10         | 0          | 8          | 0          | 8          | 0          | 8          | 0          | 11         |
| 4    | 3          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 5    | 9          | 0          | 12         | 5          | 12         | 5          | 12         | 5          | 19         | 0          |
| 6    | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 7    | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 8    | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 9    | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 10   | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 11   | 11         | 15         | 0          | 15         | 0          | 15         | 0          | 8          | 0          | 0          |
| 12   | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 13   | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 14   | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 15   | 0          | 0          | 3          | 0          | 3          | 0          | 3          | 0          | 0          | 0          |
| 16   | 7          | 3          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 17   | 0          | 0          | 12         | 11         | 12         | 11         | 19         | 9          | 0          | 0          |
| 18   | 12         | 11         | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 19   | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 20   | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 21   | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 22   | 0          | 1          | 11         | 11         | 11         | 11         | 0          | 0          | 0          | 0          |
| 23   |          |            |            |            |            |            |            |            |            |            |
algorithm runs for a constant CPU time based on its size. Table 16 demonstrates the mean of the relative deviation percentage from the best-known solution (RDP), resulting from every four algorithms for each class. In this table, the algorithms are compared using RDP. RDP of 0.00% means the best result or the minimum total project cost.

According to Figure 9, we can strongly claim that the proposed hybrid COA-GA and GWO-GA algorithms are quite effective in solving MRCPSP-
QDPMO problems, evidently outperforming PSO-GA and GA-GA algorithms.

To compare the COA-GA and GWO-GA algorithms, a paired sample T-test is conducted to test the equality of the \( \overline{RDP} \) means obtained by the two algorithms \( (H_0) \) against the COA-GA outperforming GWO-GA or the \( \overline{RDP} \) mean of COA-GA is lower than the mean of GWO-GA \( (H_1) \). The two hypotheses for this test are as follows:

\[
\begin{align*}
H_0: & \quad \mu_d = 0 \\
H_1: & \quad \mu_d > 0
\end{align*}
\]

where \( d \) is the difference between two paired \( \overline{RDP} \) \( (\overline{RDP}_{GWO-GA} - \overline{RDP}_{COA-GA}) \).

The test statistic \( (2.88) \) denotes that null hypothesis is rejected at a confidence level of 95%, indicating that COA-GA algorithm significantly outperforms GWO-GA.

7. Sensitivity analysis

This section seeks to determine the effect of different features of the problem on the total project cost. To deal with this issue, again, we used the typical instance illustrated in Section 6.2 to assess the sensitivity of the objective function with respect to changing five factors including holding cost, ordering cost, material price, penalty, and bonus. Figures 10–14 (Logarithmic diagrams are used to represent all the cases in a graph according to different cost scales.) show the sensitivity of objective function to each factor based on changing its value by a different percentage compared to its base value, while other factors remain unaltered. Figures 10–14 show the results after solving the problems with the best metaheuristic, COA-GA.

According to the results in Figure 10, increasing the holding costs in the presence of fixed other costs leads to smaller order sizes to reduce the material inventories and projects’ holding costs.

As shown in Figure 11, increase of the ordering costs results in reducing the number of orders and increasing their amount, hence a greater chance to use better discounts. The increase in project’s holding cost actually represents more inventories due to increase in the number of orders.

Figure 9. Comparison of the algorithms based on the mean of the relative deviation percentage.

Figure 10. The costs and the finish time of the problem found by COA-GA with different values of holding costs.

Figure 11. The costs and the finish time of the problem found by COA-GA with different values of ordering costs.
Figure 12. The costs and the finish time of the problem found by COA-GA with different values of material prices.

Figure 13. The costs and the finish time of the problem found by COA-GA with different values of penalty.

Figure 14. The costs and the finish time of the problem found by COA-GA with different values of bonus.

As Figure 12 shows, higher material prices lead to increase in the number of orders in order to make use of more discounts. In such situations, higher holding costs are acceptable.

The impact of the penalty factor can be seen in its elimination. When there is no tardiness penalty in this project, the activities are carried out in modes with lower costs and longer durations; thus, it is obvious that the project finish time increases (Figure 13).

According to Figure 14, with low earliness bonus, selection of activity modes with a lower duration and higher costs is not justifiable, simply because the bonus cannot compensate for the increased project cost.

8. Conclusions

This study investigated a more realistic class of project scheduling problems, a combination of Multi-mode Resource-Constraint Project Scheduling Problem (MRCPSP) under bonus-penalty policies and Quantity Discount Problem in Material Ordering (QDPMO) under an all-unit discount scheme, which is called MRCPSQDPMO problem. The model was mathematically formulated with the objective of minimizing the total cost of the project including costs of renewable resources, non-renewable resources (ordering, holding, and purchasing) and the penalty or bonus resulting from the project’s tardiness or earliness.

By proving a theorem, we found that an important property of the optimum solution is the fact that activities’ start times are the only choice for ordering points. This property reduces the search space significantly compared to previous studies.

Since this problem is NP-hard, four hybrid algo-
rithms (COA-GA, GWO-GA, PSO-GA, and GA-GA) incorporating an outside and an inside search were presented to solve the model.

In fact, the outside search (COA, GWO, PSO, and GA) focuses on finding solutions to the MRCPSP to present the schedules and the inside search (GA), which is the same among all the proposed hybrid algorithms focusing on solving (QDPMO) to discover the best ordering plan for the generated schedules.

After tuning the parameters of the algorithms by the Taguchi method, for solving a typical instance, the algorithms were applied and their results were compared with the solution obtained from BARON solver of GAMS software. The comparison demonstrated that COA-GA and GWO-GA outperformed BARON. Finally, hybrid metaheuristics were compared with each other in order to determine the best solutions in a fixed time. In this regard, the developed algorithms were tested on a set of problems generated by the PROGEN with some assumptions, and it was observed that the hybrid COA-GA outperformed others.

References
1. Elmaghraby, S.E., *Activity Networks: Project Planning and Control by Network Models*, John Wiley & Sons (1977).
2. Słowinski, R. “Two approaches to problems of resource allocation among project activities—a comparative study”, *Journal of the Operational Research Society*, 31(8), pp. 711–723 (1980).
3. Talbot, F.B. “Resource-constrained project scheduling with time-resource tradeoffs: The non-preemptive case”, *Management Science*, 28(10), pp. 1197–1210 (1982).
4. Patterson, J.H., Słowinski, R., Talbot, F.B., and Weglarz, J. “An algorithm for a general class of precedence and resource constrained scheduling problems”, *Advances in Project Scheduling*, Elsevier, Amsterdam, pp. 3–28 (1989).
5. Dred, A. and Gruenwald, J. “Nonpreemptive multi-mode resource-constrained project scheduling”, *IIE Transactions*, 25(5), pp. 74–81 (1993).
6. Speranza, M.G. and Vercellis, C. “Hierarchical models for multi-project planning and scheduling”, *European Journal of Operational Research*, 64(2), pp. 312–325 (1993).
7. Hartmann, S. and Briskorn, D. “A survey of variants and extensions of the resource-constrained project scheduling problem”, *European Journal of Operational Research*, 207, pp. 1–14 (2010).
8. Sprecher, A., Hartmann, S., and Dred, A. “A new algorithm for project scheduling with multiple modes”, *Operations-Research-Spektrum*, 19(3), pp. 195–203 (1997).
9. Demeulemeester, E.L. and Herroelen, W.S. “An efficient optimal solution procedure for the preemptive resource-constrained project scheduling problem”, *European Journal of Operational Research*, 90(2), pp. 331–348 (1996).
10. Boctor, F.F. “Heuristics for scheduling projects with resource restrictions and several resource-duration modes”, *The International Journal of Production Research*, 31(11), pp. 2547–2558 (1993).
11. Boctor F.F. “A new and efficient heuristic for scheduling projects with resource restrictions and multiple execution modes”, *European Journal of Operational Research*, 90(2), pp. 349–361 (1996).
12. Kolisch, R. and Dred, A. “Local search for non-preemptive multi-mode resource-constrained project scheduling”, *IIE Transactions*, 29(11), pp. 987–999 (1997).
13. Sprecher, A. and Dred, A. “Solving multi-mode resource-constrained project scheduling by a simple, general and powerful sequencing algorithm”, *European Journal of Operational Research*, 107(2), pp. 431–450 (1998).
14. Bouleimen, K. and Leccoeq, H.A. “New efficient simulated annealing algorithm for resource constrained scheduling problem”, *Technical Report, Service de Robotique et Automatisation*, University de Liège, pp. 1–10 (1998).
15. Hartmann, S. “Project scheduling with multiple modes: a genetic algorithm”, *Annals of Operations Research*, 102(1-4), pp. 111–135 (2001).
16. Józefowska, J., Mila, M., Różycki, R., Waligóra, G., and Weglarz, J. “Simulated annealing for multi-mode resource-constrained project scheduling”, *Annals of Operations Research*, 102(1-4), pp. 137–155 (2001).
17. Alcaraz, J., Maroto, C., and Ruiz, R. “Solving the multi-mode resource-constrained project scheduling problem with genetic algorithms”, *Journal of the Operational Research*, 54(6), pp. 614–626 (2003).
18. Mila, M., Waligóra, G., and Weglarz, J. “Simulated annealing and Tabu-search for multi-mode resource constrained project scheduling with positive discounted cash flows and different payment models”, *European Journal of Operational Research*, 164(3), pp. 630–668 (2005).
19. Zhang, H., Tam, C.M., and Li, H. “Multi-mode project scheduling based on particle swarm optimization”, *Computer-Aided Civil and Infrastructure Engineering*, 21(2), pp. 93–105 (2006).
20. Jarboui, B., Damak, N., Siarry, P., and Rebai, A. “A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems”, *Applied Mathematics and Computation*, 195(1), pp. 299–308 (2008).
21. Van Peteghem, V. and Vanhoucke, M. “An artificial immune system for the multi-mode resource-constrained project scheduling problem”, *In European*
Conference on Evolutionary Computation in Combinatorial Optimization, Springer, Berlin, Heidelberg, pp. 85–96 (2009, April).

22. Van Peteghem, V. and Vanhoucke, M. “Genetic algorithm for the preemptive and non-preemptive multimode resource-constrained project scheduling problem”, European Journal of Operational Research, 201(2), pp. 409–418 (2010).

23. Barrios, A., Balestini, F., and Valls, V. “A double genetic algorithm for the MRCPSP/max”, Computers & Operations Research, 38(1), pp. 33–43 (2011).

24. Khojasteh, M., Kianfar, F., Shirzadeh Chaleshtari, A., Shadrokh, S., and Ranjbar, M. “A modified PSO algorithm for minimizing the total costs of resources in MRCPSP”, Mathematical Problems in Engineering, 2012, pp. 1–18 (2012). DOI: 10.1155/2012/367597

25. Wang, L. and Fang, C. “An effective estimation of distribution algorithm for the multi-mode resource-constrained project scheduling problem”, Computers & Operations Research, 39(2), pp. 449–460 (2012).

26. Li, H. and Zhang, H. “Ant colony optimization-based multi-mode scheduling under renewable and nonrenewable resource constraints”, Automation in Construction, 35, pp. 431–438 (2013).

27. Messelis, T. and De Causmaecker, P. “An automatic algorithm selection approach for the multi-mode resource-constrained project scheduling problem”, European Journal of Operational Research, 233(3), pp. 511–528 (2014).

28. Aquilano, N.J. and Smith, D.E. “A formal set of algorithms for project scheduling with critical path method material requirements planning”, Journal of Operations Management, 1(2), pp. 57–67 (1980).

29. Smith-Daniels, D.E. and Aquilano, N.J. “Constrained resource project subject to material constraints”, Journal of Operations Management, 4(1), pp. 309–387 (1984).

30. Smith-Daniels, D.E. and Smith-Daniels, V.L. “Optimal project scheduling with materials ordering”, IE Transactions, 19(2), pp. 122–129 (1987).

31. Smith-Daniels, D.E. and Smith-Daniels, V.L. “Maximizing the net present value of a project subject to materials and capital constraints”, Journal of Operations Management, 7(1–2), pp. 33–45 (1987).

32. Ertas, A. and Sepil, C. “A modified heuristic procedure for materials management in project networks”, International Journal of Industrial Engineering: Theory, Applications and Practice, 6(2), pp. 132–140 (1999).

33. Dodin, B. and Elimam, A.A. “Integrated project scheduling and material planning with variable activity duration and rewards”, IE Transactions, 35(11), pp. 1005–1018 (2001).

34. Sajadieh, M.S., Shadrokh, S., and Hassanazadeh, F. “Concurrent project scheduling and material planning: A genetic algorithm approach”, Scientia Iranica, Transaction E, Industrial Engineering, 16(2), pp. 91–99 (2009).

35. Tabrizi, B.H. and Ghaderi, S.F. “An integrated mixed-integer programming model to address concurrent project scheduling and material ordering”, International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering, 9, pp. 1960–1963 (2015).

36. Tabrizi, B.H. and Ghaderi, S.F. “A robust bi-objective model for concurrent planning of project scheduling and material procurement”, Computers & Industrial Engineering, 98, pp. 11–29 (2016).

37. Zanghi, N., Shamsavar, A., Abbasi, B., and Van Peteghem, V. “Multi-mode resource-constrained project scheduling problem with material ordering under bonus-penalty policies”, Top, 25(1), pp. 49–79 (2017).

38. Shamsavar, A., Zanghi, N., and Abbasi, B. “Integration of resource investment problem with quantity discount problem in material ordering for minimizing resource costs of projects”, Operational Research, 18(2), pp. 315–342 (2018).

39. Zanghi, N., Shamsavar, A., and Niazi, S.T.A. “A hybrid project scheduling and material ordering problem: Modeling and solution algorithms”, Applied Soft Computing, 58, pp. 700–715 (2017).

40. Tabrizi, B.H. “Integrated planning of project scheduling and material procurement considering the environmental impacts”, Computers & Industrial Engineering, 120, pp. 103–115 (2018).

41. Kolisch, R. “Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation”, European Journal of Operational Research, 90(2), pp. 320–333 (1996).

42. Pierozan, J. and Cools, L. “Coyote Optimization Algorithm: A new metaheuristic for global optimization problems”, IEEE Congress on Evolutionary Computation (IEEE CEC), 2018 IEEE, pp. 2633–2640 (2018).

43. Mirjalili, S., Mirjalili, S.M., and Lewis, A. “Grey wolf optimizer”, Advances in Engineering Software, 69, pp. 46–61 (2014).

44. Long, W. and Xu, S. “A novel grey wolf optimizer for global optimization problems”, Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), pp. 1266–1270 (2016).

45. Muro, C., Escolado, R., Spector, L., and Coppeinger, R.P. “Wolf-pack (Canis lupus) hunting strategies emerge from simple rules in computational simulations”, Behavioural Processes, 88(3), pp. 192–197 (2011).

46. Eberhart, R. and Kennedy, J. “A new optimizer using particle swarm theory”, In MHS’95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, pp. 39–43 (1995).

47. Taguchi, G., Introduction to Quality Engineering, White Plains, Asian Productivity Organization, NewYork (1986).
48. Eberhart, R. and Kennedy, J. “A new optimizer using particle swarm theory”, In *Micro Machine and Human Science, MHS'95*, Proceedings of the Sixth International Symposium on IEEE, pp. 39–43 (1995).

**Biography**

**Mahdieh Akhbari** received her BS degree in Industrial Engineering from Khajeh Nasir Toosi University of Technology in 2004, an MS degree in Industrial Engineering from Isfahan University of Technology in 2007, and a PhD degree in Industrial Engineering in 2014 from Science and Research Branch of Islamic Azad University. She is currently an Assistant Professor at the Industrial Engineering Department of the Electronic Branch of the Islamic Azad University. Her research interests include soft computing, game theory, and project planning.