Computers & Industrial Engineering 83 (2015) 30–38

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Modeling and solving the project selection and scheduling ☆

Ali Asghar Tofighian, B. Naderi *

Department of Industrial Engineering, Faculty of Engineering, Kharazmi University, Tehran, Iran

ARTICLE INFO

Article history:
Received 31 May 2014
Received in revised form 7 January 2015
Accepted 14 January 2015
Available online 9 February 2015

Keywords:
Project selection and scheduling
Mixed integer linear programming model
Multi-objective ant colony optimization

ABSTRACT

This paper considers the integrated bi-objective problem of projects selection and scheduling to optimize both total expected benefit and resource usage variation. The benefit is time-dependent. Although this integrated problem has become a very active field of research, the available model and algorithms suffer from serious shortcomings. This paper analyzes the available methods and develops a novel mathematical model, in form of a mixed integer linear program, for the problem. Then, it proposes an ant colony optimization algorithm employing four features of ant generation, colonial, Pareto front updating, and pheromone updating mechanisms. To evaluate the proposed algorithm, it is compared with two available genetic algorithm and scatter search. Using comprehensive numerical experiments and statistical tools, it is shown that the proposed ant colony optimization outperforms the two available algorithms.

1. Introduction

Regarding real world competitive markets, project selection is one of the most important strategic decisions that each enterprise may deal with (Ghorbani & Rabbani, 2009; Lin & Hsieh, 2004; Rabbani, Aramoon Bajestani, & Baharian Khoshkhou, 2010; Yu, Wang, Wen, & Lai, 2012). The problem is to select an optimum portfolio of projects among several available projects subject to enterprise's intrinsic constraints of budget, available resources, and other extrinsic and technical limitations of the real world. The project portfolio selection (PPS) problem may consider various objectives but in financial trades, gaining maximum benefit is always considered as a crucial objective. Hence, many studies have considered this objective function (Bhattacharyya, Kumar, & Kar, 2011; Chen & Askin, 2009; Ghorbani & Rabbani, 2009; Liu & Wang, 2011; Tseng & Liu, 2011). The economic nature of many projects causes that more benefit can be achieved if a project is completed earlier, (Chen & Askin, 2009). Thus, after selecting a portfolio of projects, each enterprise requires to schedule them to acquire maximum benefit. The joint problem of project selection and scheduling have recently received significant attentions (Carazo et al., 2010; Chen & Askin, 2009; Ghorbani & Rabbani, 2009; Liu & Wang, 2011). The literature focuses more on the project portfolio selection problem than the joint problem under consideration. For example, see Cruz, Fernandez, Gomez, Rivera, and Perez (2014). In this paper, the authors consider only the project selection part. They formulate the problem using knapsack-based variables and propose an ant colony optimization with the matrix representation.

The problem can be described as follows. Let us suppose a set of projects are available. There are also a set of resources to carry out the selected projects. The model has a planning horizon consisting of multiple time periods, and each project has certain duration. Resource usage of each project and the available amount of resources in each time period are predetermined. Each selected project in each time period releases certain benefit. The objective is to select a subset of projects to both maximize total expected benefit of selected projects and to minimize the summation of the absolute variation of the allotted resource between each pair of two successive time periods. These two objectives are in conflict (Ghorbani & Rabbani, 2009). On one hand, the first objective function tends to select as many projects as available to maximize the earned benefit. On the other hand, selecting more projects needs more resources in each time period; thus, it likely worsens the second objective function (Ghorbani & Rabbani, 2009). Additionally, the following assumptions are characterized. The benefit gained from each project is time dependent. The earlier a project is completed, the more its benefit becomes. All the parameters are deterministic. Projects are interdependent. This is, some projects might be mutually exclusive. Projects are also integrated. That is, a project, if selected, must continue and finish without any interruption. There is a limited time horizon, i.e., all selected projects must be completed within a limited time horizon, called makespan.

☆ This manuscript was processed by Area Editor T.C. Edwin Cheng.
* Corresponding author.

http://dx.doi.org/10.1016/j.cie.2015.01.012
0360-8352/

One objective for this problem is maximization of total benefit. In real world, resources such as specialized labors are highly cost-intensive. Hence, managers face multidimensional decisions considering both benefit maximization and recourse usage minimization. Commonly, papers consider minimizing total required resources or required resources in each time period. Yet, this objective may increase the level of risk. Resource planning is done mostly at beginning of the project implementation, but projects may not be performed as planned. Therefore, if the level of resources is minimized, it may cause many disruptions in project implementation. In such circumstances, recent researches (Ghorbani & Rabbani, 2009) tend to minimize the maximum level of required resources as the second objective.

It is proved that the resource constrained project scheduling problems (RCPSP) are NP-hard (Demeulemeester & Herroelen, 2002). In this case, metaheuristics are utilized since exact methods cannot achieve acceptable solutions within reasonable time for large sized problems. Note that often real world problems are not small sizes. Among different alternatives, ant colony optimization (ACO) algorithm show high performance in many optimization problems (Doerner, Gutjahr, Hartl, Strauss, & Stummer, 2001; Doerner, Gutjahr, Hartl, Strauss, & Stummer, 2006; Liang, Chen, Kao, & Chyu, 2004). Moreover, ACO seems more effective than other evolutionary metaheuristics like genetic algorithms for intensive constrained problems. In evolutionary metaheuristics, solutions are built and then checked feasibility. Afterwards, the infeasible solutions are either discarded or converted. This procedure is time-consuming. Yet, in ACO, solutions are built step by step and at each stage, generation of infeasible solutions is avoided. Consequently, this paper proposes a new multi-objective metaheuristic method based on ACO. To enhance ACO, it is also hybridized with a mimetic search algorithm.

The rest of this paper is organized as follows. Section 2 proposes a new mathematical model for the problem of project selection and scheduling under resources restrictions. The proposed multi-objective evolutionary algorithm is developed in Section 3. Section 4 presents carries out experiments to evaluate the proposed model and algorithm. Finally, the paper is concluded in Section 5.

2. Proposed mathematical formulation

In the joint problem of project selection and scheduling, a small subset of projects is chosen from a much larger set of projects based on a set of objective criteria and a given set of limitations. The first phase of studying such an optimization problem is developing mathematical formulation. The problem under consideration is already modeled by Ghorbani and Rabbani (2009). Unfortunately, this model includes several shortcomings. First, the mixed integer model is non-linear; thus, there is no guarantee of optimality while solving this model. Next, it does not support more than one resource. Last but not the least; the model is ineffective since it suffers from high size complexity. That is, it includes an extensive number of decision variables and constraints. The same decisions can be represented by much fewer variables and constraints in a more intelligent model. This paper develops a new mixed integer linear programming (MILP) model for the problem to resolve all these shortcomings.

2.1. Mathematical model

In the proposed model, the following indexes and parameters are used.

n: total number of available competitive projects
m: the number of different types of resources
T: number if time periods (makespan)
i,h: indexes for projects; $i,h \in \{1,2,\ldots,n\}$
k: resources index; $k \in \{1,2,\ldots,m\}$
t,j: time period indexes; $t,j \in \{1,2,\ldots,T\}$
d_i = the duration of project i
H_i = set of projects that are interdependent with project i
$r_{k,t}$ = available amount of resource k, in time period t
$a_{i,k}$ = required amount of resource k should be allocated to project i in each time period
$b_{i,t}$ = the expected benefit of project i if it start in t-th time period
Decision variables
$X_{i,t} = 1$ if project i starts in time period t, 0 otherwise; $1 \leqslant t \leqslant T - d_i + 1$
$W_{t,k}$ = variation of k-th allotted resource between each successive time period

The model formulates the problem as follow.

$$Max\, Z_1 = \sum_{i=1}^{n}\sum_{t=1}^{T-d_i} b_{i,t}X_{i,t} \tag{1}$$

$$Min\, Z_2 = \sum_{t=1}^{T}\sum_{k=1}^{m} W_{t,k} \tag{2}$$

$St:$

$$\sum_{t=1}^{T-d_i+1} X_{i,t} \leqslant 1; \forall i \tag{3}$$

$$\sum_{i=1}^{n}\left(\sum_{j=\max\{1,t-d_i+1\}}^{\min\{t,T-d_i+1\}} X_{i,j}\right) a_{k,i} \leqslant r_{t,k}; \forall_{t,k} \tag{4}$$

$$W_{t,k} \geqslant \sum_{i=1}^{n}\left(\sum_{j=\max\{1,t-d_i+1\}}^{\min\{t,T-d_i+1\}} X_{i,j}\right) a_{i,k} - \sum_{i=1}^{n}\left(\sum_{j=\max\{1,t+2-d_i\}}^{\min\{t+1,T-d_i+1\}} X_{i,j}\right) a_{i,k}; \forall_{k,t<T} \tag{5}$$

$$W_{t,k} \geqslant \sum_{i=1}^{n}\left(\sum_{j=\max\{1,t-d_i+1\}}^{\min\{t,T-d_i+1\}} X_{i,j}\right) a_{i,k} - \sum_{i=1}^{n}\left(\sum_{j=\max\{1,t-d_i\}}^{\min\{t-1,T-d_i+1\}} X_{i,j}\right) a_{i,k}; \forall_{k,t>1} \tag{6}$$

$$\sum_{t=1}^{T-d_i+1}(x_{i,t}) + \sum_{t=1}^{T-d_h+1}(x_{h,t}) \leqslant 1\ \forall i,h \in H_i \tag{7}$$

$$X_{i,t} \in \{0,1\} \forall_{i,t\leqslant T-d_i+1} \tag{8}$$

$$W_{t,k} \geqslant 0\ \forall_{t,k} \tag{9}$$

Eqs. (1) and (2) are the first and second objective functions (i.e., total expected benefit and resource usage variation between each two successive time periods.) Constraint set (3) determines the time the selected project is started. Constraint set (4) ensures that resource limitations are not violated. Constraints sets (5) and (6) linearly specify the resource usage variation between the two successive time periods. Constraint set (7) ensures that the projects' interdependency is held. Constraint sets (8) and (9) define the decision variables.

2.2. Illustrative example

In the following, we provide a small numerical example and one of its feasible solutions to demonstrate different aspects of the problem and the model. Let us assume five projects are available. Table 1 shows durations and resource usage of a single resource. The expected benefits and interdependencies are shown in Tables 2 and 3, respectively. Makespan is equal to 10 time units and the maximum available resource for all time periods is 6 units.

Table 1
Duration of projects.

Factor	Projects				
	1	2	3	4	5
Duration	4	5	3	2	4
Required resource	3	5	1	2	4

As shown in Fig. 1, one solution for this problem, can be $X_{1,1} = 1, X_{2,5} = 1$ and $X_{3,1} = 1$. This means projects 1, 2 and 3 are selected and started at 1, 5, and 1 respectively.

With this solution total expected benefit equals 64 (25 + 19 + 20 = 64) and based on Table 5 second objective is equal 8 (0 + 1+0 + 0+2 + 0 + 0 + 0+5 + 0 = 8). In Tables 4 and 5 we check the validity of constraint (4) and constraints (5) and (6) simultaneously.

Validity of Constraints (3) and (7) are straightforward. If one of the projects which has interdependency with other one is selected, $\sum_{t=1}^{T-d_i+1}(x_{i,t})$ becomes 1 and the other part $\sum_{t=1}^{T-d_h+1}(x_{h,t})$ must be 0.

3. Proposed Pareto ant colony optimization algorithm

In this section, we develop an algorithm, based on ant colony optimization (ACO), to solve the problem. ACO is a metaheuristic, inspired from the ant's behavior in nature. It is firstly introduced by Dorigo, Maniezzo, and Colorni (1991), Dorigo (1992) and Dorigo, Maniezzo, and Colorni (1996). It combines stochastic search with a learning mechanism. One significant advantage of ACO, over other evolutionary metaheuristics such as genetic algorithms, is to construct solutions step-by-step. Thus, ACO likely performs well in solving problems with intensive feasibility constraints.

3.1. Classical ant colony optimization

In classical ACO, search is done by a population of solutions represented by a construction graph $\mathcal{C}$. A complete solution is generated step-by-step by some sequential walks. In case of infeasibility in any of walks, the construction process stops. To specify the next walk, ACO uses pheromone value τ and visibility value η. The pheromone value is a memory that stores the desirability of each walk step through pervious runs, and visibility is a per-defined value according to the nature of the problem and its circumstances. The probability of choosing each walk step between each pair of nods is shown as p_{kl} where k and l are two feasible successor nodes. This probability can be calculated using different formulas, but the most frequently used formula is Eq. (10).

$$P_{k,l} = \tau_{k,l}^{\alpha} \times \eta_{k,l}^{\beta} \tag{10}$$

where α and β are parameters determining the influence of the pheromone value and visibility value, respectively. At the end of each iteration, the pheromone on each path is updated.

Table 2
The expected benefit in each time period.

Projects	Time									
	1	2	3	4	5	6	7	8	9	10
1	25	23	20	17	16	14	10	0	0	0
2	30	28	25	21	19	17	0	0	0	0
3	20	19	17	15	14	12	10	9	0	0
4	15	13	12	11	11	10	7	5	3	0
5	25	24	21	19	17	14	8	0	0	0

Table 3
Project interdependencies.

	Projects				
Projects	1	2	3	4	5
1	–	0	0	1	1
2	0	–	0	1	0
3	0	0	–	0	1
4	1	1	0	–	0
5	1	0	1	0	–

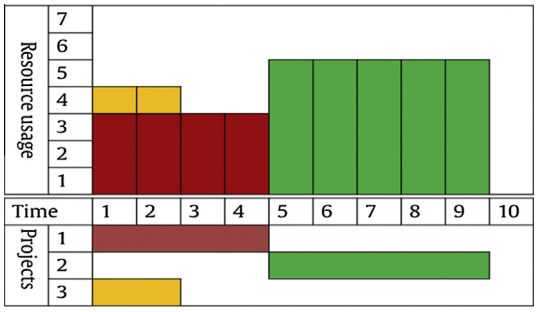

Fig. 1. Project Gantt chart and resource usage.

Table 4
Constraint (4) validity check.

Factor	Time period									
	1	2	3	4	5	6	7	8	9	10
Resource usage	4	4	3	3	5	5	5	5	5	0
Available resource	6	6	6	6	6	6	6	6	6	6
Constraint (4) validity status	√	√[a]	√	√	√	√	√	√	√	√

[a] $3 \times (1) + 5 \times (0) + 1 \times (1) + 2 \times (0) + 4 \times (0) \leqslant 6$.

Table 5
Constraints (5) and (6) validity check.

	Time period									
Factor	1	2	3	4	5	6	7	8	9	10
Constraint (5): $W_{t,1} \geqslant$	0	1	0	−2	0	0	0	0	5	–
Constraint (6): $W_{t,1} \geqslant$	–	0	−1	0	2	0	0	0	0	−5
Constraint (9): $W_{t,1} \geqslant$	0	0	0	0	0	0	0	0	0	0
$W_{t,1} \geqslant$	0	1	0	0	2	0	0	0	5	0
Overall validity status	√	√	√	√	√	√	√	√	√	√

The pheromone updating procedure can be applied either during (local pheromone update rule) or after (global pheromone update rule) solution constructing phase. Local and global pheromone updating procedures enhance exploration and intensification capabilities (Bonabeau, Dorigo, & Theraulaz, 1999). Eqs. (11) and (12) show the first and second pheromone updating rules, respectively.

$$\tau_{k,l} = (1-\rho)\tau_{k,l} + \rho\tau_0 \tag{11}$$

$$\tau_{k,l} = (1-\rho)\tau_{k,l} + \Delta\tau_{k,l} \tag{12}$$

where ρ and $\Delta\tau_{k,l}$ are the pheromone evaporation coefficient and the amount of pheromone deposited by selected ants, respectively.

3.2. Proposed Pareto ant colony optimization

The framework of the proposed metaheuristic relies on a Pareto Ant Colony Optimization (P-ACO). The P-ACO algorithm generalizes the ant colony optimization (ACO) metaheuristic for single-objective problems to the case of multi objective functions,

determining approximations to the set of Pareto-efficient solutions. P-ACO is introduced by Doerner et al. (2001). While metaheuristics in general provide an attractive compromise between the computational effort necessary and the quality of an approximated solution space, Pareto ant colony optimization (P-ACO) has been shown to perform particularly well for this class of problems (Doerner et al., 2006).

There are several strategies to solve multi-objective problems by an ant colony optimization. One strategy is to have K colonies (K is the number of objectives) one for each objective. Then, colonies are merged to build an approximation Pareto set (Aljanaby, Mahamud, & Norwawi, 2010; Delévacq, Delisle, Gravel, & Krajecki, 2013; Ellabib, Calamai, & Basir, 2007). This strategy suffers from two major disadvantages. First, due to being time-consuming, it is slow. Second, determining and tuning effective parameters and specifying merging time and exchange strategies are also difficult. Another strategy is to consider all objectives in one colony (see Doerner, Gutjahr, Hartl, Strauss, & Stummer, 2004). In this case, procedure of weighting the objective functions, selecting pheromone updating strategy and calculating the probability of choosing each walk step are not straightforward (Bakk, 2010) and find proper parameters of algorithm is very hard and challenging; hence, it is hard to code. The other strategy is to search by the most important objective onlyor treat the multiple objectives in a lexicographic order (Gravel, Price, & Gagne, 2002). Although this strategy is much faster than two other strategies, it does not provide divergence solutions.

All these strategies have serious disadvantages. The two first strategies are slow and hard to implement and the other one does not generate diverse solutions. To obviate these disadvantages, we use a mimetic based local search to gain more effective and efficient solutions. The proposed mimetic Pareto ant colony optimization (MPACO) can be described as follows. Initially, the pheromone trails are set to one. Then main loop of the algorithm starts. At each iteration, ants are generated from the current pheromone trails using ant generation mechanism. Each ant is a feasible solution encoded by a representation mechanism. Then, each original is proliferated by a procedure called colonial mechanism. All original and proliferated ants are put in a pool. Then, the approximation Pareto front is updated using a fast non-dominated sorting mechanism. Finally, the pheromone trails are updated using a pheromone updating mechanism. The pseudo code of the proposed MPACO algorithm is outlined in Fig. 2.

3.2.1. Representation, initialization and termination mechanisms

Representation schemes are used to make recognizable solutions for algorithms and computers. The available representation scheme in the literature proposed by Ghorbani and Rabbani (2009) is a matrix representation. It includes a $n \times T$ binary matrix, each row for one project and each column for time units. If the element of row i and column t is one, it means that project i is selected and started at time t. For example, for a problem with 5 projects and makespan of 7, we need a matrix with (5×7). One possible solution is as follows. In this solution, Projects 1, 2 and 3 are selected and started at times 1, 1 and 5, respectively.

Procedure: *The proposed ant colony optimization*

Initialization mechanism
repeat until the termination criterion is not met
{
  Ant generation mechanism
  Colonial mechanism
  Pareto front updating mechanism
  Pheromones updating mechanism
}

Fig. 2. The proposed algorithm's pseudo code.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}_{5\times 7}$$

This encoding scheme suffers from several drawbacks. For example, this matrix becomes very large when n and/or T is a large number. In this case, it requires a lot of computational effort. This matrix includes many unnecessary elements with value of zero. We propose another encoding scheme to overcome the mentioned issues. In this scheme, there is a string of n integer numbers each of which shows the staring time of a project. For example, we can represent the mentioned solution as follows.

$$[1 \quad 5 \quad 1 \quad 0 \quad 0]$$

This scheme needs much less computational efforts comparing with the available scheme. The value zero means that the project is not selected.

Since MPACO is a population-based algorithm, it requires a set of initial solutions. We generate an empty portfolio to serve as the initial population. Then, the main colony and Pareto front is filled with these empty ants. Initial pheromones for all walk steps are considered to be equal. Because of using standardized decision rules, function, values and procedures, initial pheromones considered all to be equal one. The termination criterion is a time limit depending on problem characterizations. It is introduced later in Section 4.3.

3.2.2. Ant generation mechanism

At each iteration, for each ant, we define a set R including projects still possible to be selected. At the beginning, set R includes all possible projects. Each ant randomly selects the first feasible project, and then schedules it. In scheduling phase, due to nature of economic projects, the algorithm preferably starts the selected projects in earlier times. This means that earlier starting times have greater chance to be selected. To determine the probability of each time unit as starting time, Eq. (13) is used.

$$\mathcal{S}_t = \frac{2(T+1-t)}{T(T+1)}, 1 \leqslant t \leqslant T \tag{13}$$

After selecting the first feasible project and its starting time, the selected project and all its interdependent projects are excluded from set R. This is to assure that projects with interdependency are not selected.

The next phase is to select the next project from set R with regard to feasibility rules. In this phase, if the new walk is feasible, probability of including each selected project and its starting time (new walk step) in pervious walk steps is calculated. The selected project and all interdependent projects are then excluded from R. If the walk step is not feasible, only the selected project is excluded from R. These steps continue until there is no feasible walk step or R is empty. Fig. 3 shows walk steps chain.

Consider a problem with 5 projects and makespan of 10. The ant randomly selects project 1 as its first project and then randomly starts it at time period 1 with regard to $\mathcal{S}_t$. After this selection, R is updated with exclusion of project 1 and all interdependent projects (for example, projects 4 and 5). In the next step, the ant selects a project from available one (i.e., projects 2, 3). It randomly selects project 2 with starting time of 5. R in this step includes project 3. Thus, it selects project 3 and schedules it at time period

```
Procedure: Ant generation mechanism
For each ant
{
    Put all projects in set ℝ.
    Select the first project randomly
    Update ℝ
    repeat until set ℝ is not empty
    {
        Test selecting all the projects in set ℝ
        Select the next project using the selection mechanism
        Update ℝ
        Calculate fitness values
    }
}
```

Fig. 3. The ant generation mechanism.

1. In this step, R is empty and the chain of walk step finishes (see Fig. 4).

Regarding the first objective (total benefit), ants try to select as many projects as possible. Yet, this may result in violating the second objective (resource usage variation). In this case, one or some projects are randomly excluded from R to generate each ant.

As earlier discussed, the first objective is more important than the second objective. Therefore, the pheromone updating criterion is based on the first objective. The visibility or heuristic information is calculated by Eq. (14).

$$\eta_{i,t} = \frac{b_{i,t}}{\max(b_{i,t})} \tag{14}$$

Thus, we have $0 \leqslant \eta_{i,t} \leqslant 1$. This assures that if project i with starting time t, has bigger return, then, it has greater change to be selected in portfolio. The probability of selecting project i with starting time t in pervious walk steps is calculated as follow.

$$p_{i,t} = \begin{cases} \tau^{\alpha}_{lp,lt,i,t} \times \eta^{\beta}_{i,t} & \text{if walk is feasible} \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

where lp and lt are the last selected project and its start time respectively. $\tau_{lp,lt,i,t}$ stores the desirability (deposited pheromone) of the last selected project and its start time (the last walk step) to any feasible project i and its start time t(next walk step). Finally, the overall probability of including project i with start time t in portfolio ($\mathcal{P}_{i,t}$) is calculated using Eq. (16).

$$\mathcal{P}_{i,t} = \frac{p_{i,t}}{\sum_{i,t} p_{i,t}} \tag{16}$$

After calculating $\mathcal{P}_{i,t}$, a roulette wheel selection is applied to determine solutions. In roulette wheel selection, projects with greater $\mathcal{P}_{i,t}$ have more chance for being selected. Note that $\mathcal{S}$ (Eq. (13)) is used by ants to schedule first selected project. But after that to find next steps (selecting next projects and scheduling them), the algorithm uses $p_{i,t}$ (Eq. (16)). When all ants find their routes, two objective functions are calculated for each ant.

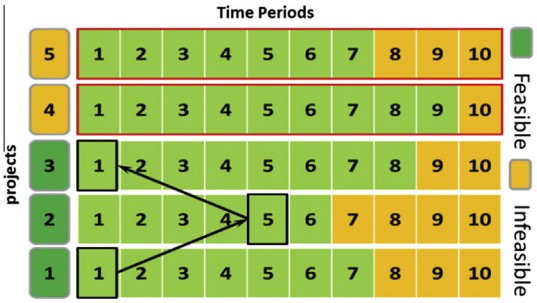

Fig. 4. Example of walk steps.

3.2.3. Colonial mechanism

To enhance algorithm's speed and diversity, after any ant in the main colony finds its solution, a parallel local search is applied. This local search has three operators. The first operator is for feasibility checking, and it assures that generated solutions be acceptable and feasible. The second is mutation operator. This operator randomly chooses an element of solution matrix and randomly changes it. This operator applied with probability P_o. And the third one is the move operator. This operator stochastically chooses two ants, then, it randomly selects an element and exchanges the corresponding element between two individuals. This operator is applied randomly with probability P_m. More information can be found in Ghorbani and Rabbani (2009).

The parallel local search yields five individuals per each individual. After applying parallel local search, a new colony, called parallel colony, is constructed. Size of this colony is 5× Size of main colony. Then, two objectives are calculated for these parallel ants. All parallel and main ants are used to update Pareto front ants.

3.2.4. Pareto front updating mechanism

To update Pareto front, we need to add non-dominated ants to Pareto front set and also remove ants in Pareto front dominated by new ants added to the front. To compare new added ant with existing ants in Pareto front and Because the size of Pareto front is limited and equals to 3× Size of main colony, we use a fast non dominated sorting and crowding distance procedures which first introduced by Deb, Pratap, Agarwal, and Meyarivan (2002). With these two procedures rank and crowding distance of each ant are calculated. Ants with minimum ranking number dominate other ants. Crowding distance is assures that when ants in Pareto front are more than the size of Pareto front, ants with higher diversity are chosen.

3.2.5. Pheromone updating mechanism

As mentioned earlier, local pheromone updating aims at diversifying solutions. In this paper, instead of using local pheromone updating, we apply a local search based on mimetic algorithms. Overall pheromone updating policy is that all elite (ants in Pareto front set) and non-elite ants (ants generated at each iteration) deposit pheromones with pre-determined coefficients $Q1$ and $Q2$, respectively. This can enhance both diversity and quality of solutions. Pheromone updating formulas are as follows.

For each ant in Pareto front:

$$\tau^{new}_{i,t,h,j} = \tau^{old}_{i,t,h,j} + Q1 \times \frac{Z_{1_{Pa_v}}}{min(Z_{1_{Pa}})} \qquad \forall i,t,h,j \tag{19}$$

For each non elite ant:

$$\tau^{new}_{i,t,h,j} = \tau^{old}_{i,t,h,j} + Q2 \times \frac{Z_{1_{neA_v}}}{min(Z_{1_{Pa}})};\ \forall\ non\ elite\ ant;\ \forall i,t,h,j \tag{20}$$

where $Q1$ and $Q2$ have values between zero and one, Pa_v is v th ant in Pareto front, neA_v v th non elite ant. $\frac{Z_{1_{Pa_v}}}{min(Z_{1_P})}$ and $\frac{Z_{1_{neA_v}}}{min(Z_{1_{Pa}})}$ are used to standardized τ value. At the end of each iteration, pheromones are evaporate with evaporation rate ρ, where ρ is determined and $0 \leqslant \rho \leqslant 1$. Evaporation effect on pheromones is shown in Eq. (21).

$$\tau^{new}_{i,t,h,j} = \tau^{old}_{i,t,h,j} \times (1-\rho) \tag{21}$$

4. Computational evaluation

This section describes the computational evaluations. It also compares proposed MPACO with two other well-known algorithms, the fast non-dominated sorting genetic algorithm (NSGA-II) and multi-objective scatter search (MOSS). NSGAII is first introduced

by Deb et al. (2002) and used for the joint problem of project selection and scheduling by Ghorbani and Rabbani (2009). MOSS is also proposed by Ghorbani and Rabbani (2009). All these three algorithms are coded in Matlab 7.12 and executed on a laptop computer with Core i7, and Windows 7 using 4 GB of RAM.

4.1. Performance measures and experimental data

We use some prominent metrics to compare these algorithms for both quality and diversity points of view. A good multi-objective algorithm for a specific problem is expected to satisfy two characteristics: (1) approximated Pareto front convergence to real optimal Pareto front and (2) diversity maintaining of approximated Pareto front. To measure and examine these two characteristics, we use three types of indicators. (1) Hyper volume indicator. (2) Epsilon indicator, and (3) Spacing metrics. The first two indicators measure quality of approximated Pareto front, and the third gauges diversity of solutions.

The hypervolume I_H is first introduced by Zitzler, Thiele, Laumanns, Fonseca, and Da Fonseca (2003). It is defined as the volume of the objective space dominated by approximated Pareto front. In order to calculate I_H, objective space must be bounded. If real Pareto optimal front is not available, bounding reference point is used instead. Among different alternatives to calculate hypervolume, this paper uses inclusion–exclusion algorithm proposed by Wu and Azarm (2000). This algorithm adds volumes of rectangular polytopes dominated by each point individually, then subtracts volumes dominated by intersections of pairs of points, finally adds back in volumes dominated by intersections of three points, and so on. Also the reference point used in I_H calculation is the worst possible solution both objectives. Thus, bigger a hypervolume indicator, the better is the approximated Pareto front.

The epsilon indicator I_ε is introduced by Zitzler et al. (2003). It returns minimum value of ε by which the approximated Pareto front dominates the reference point. In this paper, we use epsilon indicator with summation of component-wise. On the other hand, the best possible solution is used as the reference point. Therefore, the smaller an epsilon indicator is, the better the approximated Pareto front is. Geometrical interoperation of the indicator is the level of convexity of the approximated Pareto front to reference point.

Spacing metric is to measure the uniformity of the approximated Pareto front. We use the metric used in Ghorbani and Rabbani (2009) which is as follow.

$$S = \frac{1}{(|APF| - 1)\overline{\mathcal{D}}}\left[\sum_{l=1}^{|APF|}|\mathcal{D}_l - \overline{\mathcal{D}}|\right] \tag{22}$$

where APF represents the approximated Pareto front and $\mathcal{D}_l$ is the Euclidean distance between two consecutive solutions l and $l+1$ in approximated Pareto Front. $\overline{\mathcal{D}}$ is the average of these distances.

We compare the performance of the three algorithms on four different data sets: (1) low, moderate, high complexity levels and also large sized problems. In following, we describe each of these data sets. Projects in low complexity level have very little interdependency so that selecting one specific project causes deselect of other projects. In this level, all projects have short-term durations and compared to this durations, makespan is very long. The number of different types of resources is small and the available resources are much more than required resources. The methodology of generating this type of test data is very similar to methodology that used in Ghorbani and Rabbani (2009) and Rabbani et al. (2010). Table 6 shows levels considered in these data sets.

In moderate and high complexity levels, durations, number of resources and probability of interdependency between projects are increased. On the other hand, makespans and portion of available amount of resource to required resources are decreased. Levels considered in moderate and high complexity level are shown in Tables 7 and 8, respectively.

Table 6
Strategy of generating low complexity level test data.

Factor	Generating rule
Number of available projects	$n = \{3, 4, \ldots, 10\}$
Durations	$d = U(1, 3)$
Number of different types of resources	$m = \{1, 2\}$
Required resources for each projects	$a = U(10, 15)$
Available amount of resource	$r = 4U(10, 15)$
Makespan	$T = max\{\max(d_i), \sum_i d_i \times U(0.8, 1)\}$
Probability of interdependency	10%

Although in real cases, we rarely face a problem with more than 14 available projects, we generate larger sized problems to gain more accurate assessment of proposed algorithm. Levels considered in large sized problems are shown in Table 9.

In all these data sets, minimum makespan is $\max(d_i)$. This assures that at least one project can be selected and scheduled. The benefit matrix in all data sets are generated with regard to the following rules.

(1) Benefits for all projects and time periods are randomly generated by uniform distribution over U(100,999).
(2) With regard to the nature of problem, each row of benefit matrix is generated in descending order.
(3) To be more close to real-world problems, it is better that projects with longer makespans and durations release more benefit. (Optional)

4.2. Calibration

Appropriate design of the parameters and operators significantly impact on the effectiveness of metaheuristics. This section studies the behavior of the proposed MPACO regarding different operators and parameters. There are several approaches to tune and calibrate algorithms (e.g. full factorial, fractional factorial experiments and etc.). Because of large number of factors, the fractional factorial experiment is more efficient than the full factorial experiment. Among different designs, Taguchi approach is the most suitable one. In this approach, orthogonal arrays are used to study a large number of decision variables with a small number of trials. The response variable is converted to the signal-to-noise (S/N) ratio. In minimization problems, the following definition for S/N is used.

$$SN_l = -10\log\left(\frac{1}{N_l}\sum_{u=1}^{N_i}\frac{1}{y_u^2}\right) \tag{23}$$

where y_u is the hypervolume indicator for a given experiment and N_l is number of trials for trial l.

In this study, we have 8 control factors: *nAnt* (number of ants), α (pheromone coefficient), β (heuristic coefficient), ρ (evaporation coefficient), $Q1$ (contribution of current ants in pheromone depositing), $Q2$ (contribution of Pareto front ants in pheromone depositing), P_m (probability move operator) and P_o (probability of using mutation operator). Table 10 shows considered levels of these factors. The orthogonal array L_{18} is chosen since it meets all minimum requirements. After some modifications, orthogonal array L_{18} is presented in Table 11.

After performing the experiment, hyper-Volume values are individually transformed into S/N rations. Fig. 5 shows the average S/N ration obtained at each level. The selected level for each parameter is as follows.

Table 7
Strategy of generating moderate complexity level test data.

Factor	Generating rule
Number of available projects	$n = \{7, \ldots, 14\}$
Durations	$d = U(3, 7)$
Number of different types of resources	$m = \{2, 3, 4\}$
Required resources for each projects	$a = U(10, 15)$
Available amount of resource	$r = 3U(10, 15)$
Makespan	$T = max\{\max(d_i), 0.8 \times U(\max(d_i), \sum_i d_i)\}$
Probability of interdependency	20%

Table 8
Strategy of generating high complexity level test data.

Factor	Generating rule
Number of aavailable projects	$n = \{7, \ldots, 14\}$
Durations	$d = U(7, 10)$
Number of different types of resources	$m = \{4, 5\}$
Required resources for each projects	$a = U(10, 15)$
Available amount of resource	$r = 2U(10, 15)$
Makespan	$T = \max\{\max(d_i), 0.5 \times U(\max(d_i), \sum_i d_i)\}$
Probability of interdependency	30%

Table 9
Strategy of generating large size test data.

Factor	Generating rule
Number of available projects	$n = \{15, \ldots, 22\}$
Durations	$d = U(10, 15)$
Number of different types of resources	$m = \{4, 5\}$
Required resources for each projects	$a = U(10, 15)$
Available amount of resource	$r = 2U(10, 15)$
Makespan	$T = \max\{\max(d_i), 0.5 \times U(\max(d_i), \sum_i d_i)\}$
Probability of interdependency	30%

Table 10
Factors and levels.

Factor	Symbol	Level	Values		
Number of ants	*nAnt*	3	10	20	30
Pheromone coefficient	α	3	1	2	3
Heuristic coefficient	β	3	2	3	4
Contribution of current ants in pheromone depositing	$Q1$	3	0.7	0.8	0.9
Evaporation coefficient	ρ	2	0.6	0.7	
Contribution of Pareto front Ants in pheromone depositing	$Q2$	3	0.3	0.2	0.1
Probability of move operator	P_m	3	0.7	0.8	0.9
Probability of mutation operator	P_o	3	0.1	0.15	0.2

$$\rho = 0.7, \alpha = 2, \quad \beta = 2, \quad Q_1 = 0.9, \quad nAnt = 10, \quad Q_2 = 0.2, \quad P_m = 0.8, \quad P_o = 0.15$$

To assess the impact of each factor on the performance of MPACO, we use delta test. Table 12 shows delta values obtained by each factor. The most effective factors are $Q1, Q2$ and heuristic coefficient with 0.46, 0.45 and 0.339, respectively. P_m has the least impact with 0.075.

Table 11
The modified orthogonal array L_{18}.

# Trials	ρ	α	β	$Q1$	*nAnt*	$Q2$	P_m	P_o
1	0.6	1	2	0.7	10	0.3	0.7	0.1
2	0.6	1	3	0.8	20	0.2	0.8	0.15
3	0.6	1	4	0.9	30	0.1	0.9	0.2
4	0.6	2	2	0.7	20	0.2	0.9	0.2
5	0.6	2	3	0.8	30	0.1	0.7	0.1
6	0.6	2	4	0.9	10	0.3	0.8	0.15
7	0.6	3	2	0.8	10	0.1	0.8	0.2
8	0.6	3	3	0.9	20	0.3	0.9	0.1
9	0.6	3	4	0.7	30	0.2	0.7	0.15
10	0.7	1	2	0.9	30	0.2	0.8	0.1
11	0.7	1	3	0.7	10	0.1	0.9	0.15
12	0.7	1	4	0.8	20	0.3	0.7	0.2
13	0.7	2	2	0.8	30	0.3	0.9	0.15
14	0.7	2	3	0.9	10	0.2	0.7	0.2
15	0.7	2	4	0.7	20	0.1	0.8	0.1
16	0.7	3	2	0.9	20	0.1	0.7	0.15
17	0.7	3	3	0.7	30	0.3	0.8	0.2
18	0.7	3	4	0.8	10	0.2	0.9	0.1

4.3. Results

This section compares MPACO against two available algorithms in the literature, NSGA II by Deb et al. (2002) and MOSS by Ghorbani and Rabbani (2009). The four data sets are used to evaluate the algorithms. Let us remind that each of these sets include 80 instances. Table 13 shows the results for both hypervolume and epsilon indicators.

In all the levels, the proposed algorithm has better performance than the two others. Another notable point is that in low level MOSS has better hypervolume value than NSGAII. This is consistent with the obtained results by Ghorbani and Rabbani (2009). Table 16 also shows that MPACO on average is almost 10% and 12% better than NSGAII and MOSS respectively. This superiority is also seen in Epsilon indicator. Hence, the proposed method is almost 6% and 8% better than two other algorithms. To further analyze the results, Figs. 6 and 7 show performance of the algorithms versus the time for hypervolume and epsilon indicators, respectively.

Fig. 6 clearly shows the superiority of proposed algorithm versus the time. It also shows that the proposed algorithm is the best algorithm after 20 s. All the three algorithms continue improving significantly in the first 100 s. Table 14 shows ANOVA result for all level of instances. It shows that in all instances, supremacy of the proposed algorithm is significant in 95% confidence level.

Fig. 7 shows proposed algorithm has better epsilon values in all times. This figure also reveals that after 90 s NSGA-II does not cause any significant improvement in epsilon value. This time is 110 and 100 for MPACO and MOSS, respectively.

Table 15 shows the results of ANOVA test. It clears that the supremacy of proposed algorithm is significant in 95% confidence level for low, moderate and large instances. However, the obtained epsilon mean by the proposed algorithm is better than two other algorithms, this supremacy is not significant.

Table 16 shows the results of spacing metric. In all four data sets, the proposed algorithm has the best value. ANOVA results show that in low, moderate and high level of complexity, the supremacy of MPACO is significant at 95% CL. But, in large size problems, this supremacy is not significant.

After solving several problems in different data sets, it was understood that the required time for solving problems depends on the number of projects (n) and makespan (T), number of resources (m) and complexity level. Thus, we propose the best termination time for each level of tested problems using a quadratic regression function. Eqs. (24)–(27) show this termination time for low, moderate, high level complexity and large size problems.

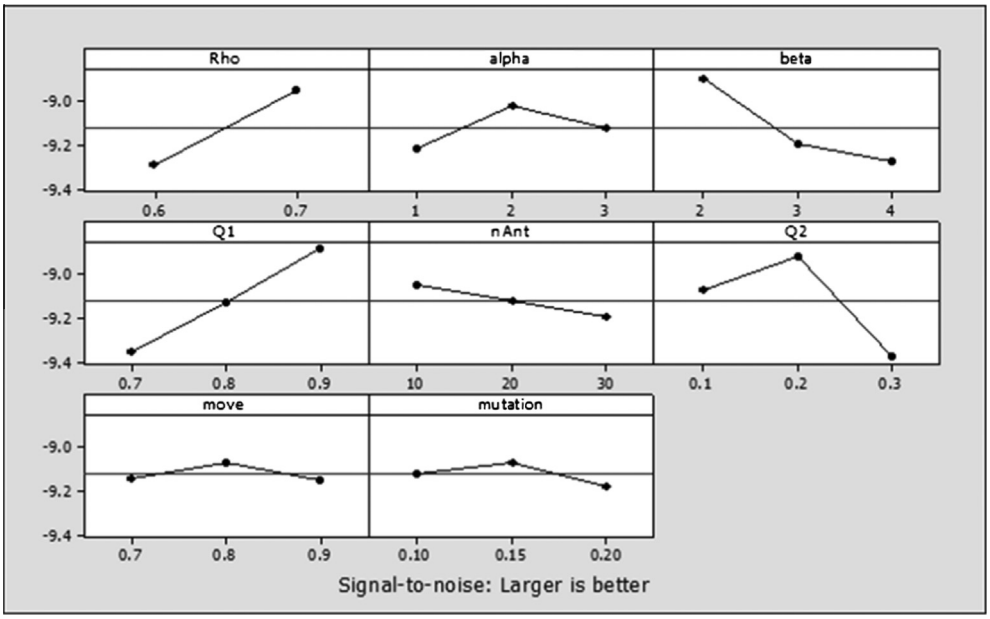

Fig. 5. The mean S/N ratio plot for each level of factors.

Table 12
Rank of each factor.

Factors	ρ	α	β	$Q1$	$nAnt$	$Q2$	P_m	P_o
Level 1	−9.29	−9.217	−8.899	−9.352	−9.047	−9.069	−9.141	−9.117
Level 2	−8.951	−9.02	−9.191	−9.126	−9.122	−8.921	−9.072	−9.068
Level 3		−9.27	−9.270	−9.192	−9.371	−9.147	−9.175	−9.175
Δ	0.339	0.197	0.371	0.468	0.144	0.45	0.075	0.107
Rank	4	5	3	1	6	2	8	7

Table 13
The results obtained by the algorithms.

Indicator	Instance	Algorithm		
		NSGAII	MOSS	MPACO
Hypervolume				
	Low	0.63	0.72	0.73
	Moderate	0.74	0.70	0.81
	High	0.60	0.56	0.73
	Large	0.75	0.67	0.86
	Average	0.68	0.66	0.78
Epsilon				
	Low	0.45	0.40	0.35
	Moderate	0.35	0.40	0.30
	High	0.56	0.58	0.49
	Large	0.32	0.40	0.28
	Average	0.42	0.44	0.35

$$time = -317.69n^2 + 351.85nT + 48.71nm - 92.68T^2 - 71.48Tm + 288m^2 + 64.09n - 17.42T - 256m + 138 \quad (24)$$

$$time = 1.41n^2 + 4.32 \times 10^{-3}nT - 1.22 \times 10^{-1}nm - 7.69 \times 10^{-2}T^2 - 1.39 \times 10^{-1}Tm + 1.50m^2 - 27.94n + 5.14T - 7.01m + 191.59 \quad (25)$$

$$time = -5.73n^2 + 2.10 \times 10^{-1}nT + 21.66nm - 1.19 \times 10^{-1}T^2 - 7.65 \times 10^{-1}Tm - 26m^2 + 14.56n + 9.83T + 24m - 198 \quad (26)$$

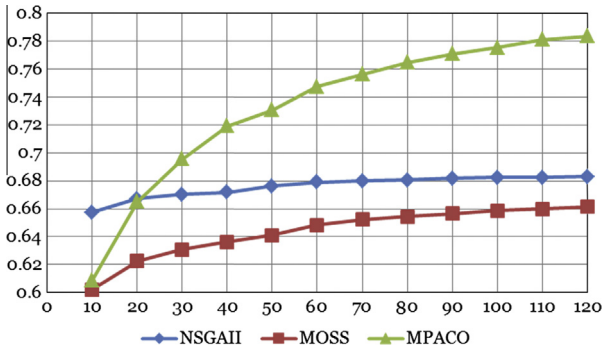

Fig. 6. Hypervolume mean values versus time.

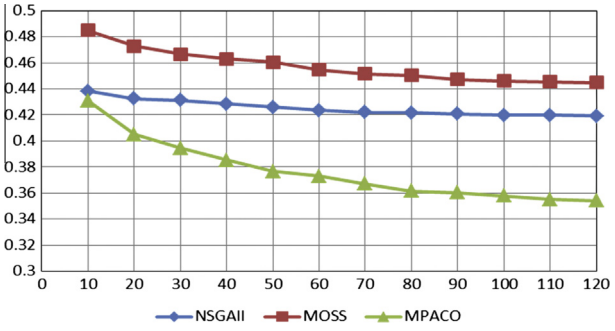

Fig. 7. Epsilon mean values versus time.

Table 14
Hypervolume mean value for algorithms and related *p*-value for each level.

Level	Source	DF	SS	MS	*F*	*P*
Low	Algorithm	2	0.4733	0.2367	16.96	0
Moderate	Algorithm	2	0.4584	0.2292	9.45	0
High	Algorithm	2	1.0473	0.5237	5.28	0.006
Large	Algorithm	2	1.5566	0.7783	16.29	0

Table 15
Epsilon mean value for algorithms and related *p*-value for each level.

Level	Source	DF	SS	MS	*F*	*P*
Low	Algorithm	2	0.413	0.2065	18.48	0
Moderate	Algorithm	2	0.3377	0.1688	7.66	0.001
High	Algorithm	2	0.299	0.15	0.98	0.377
Large	Algorithm	2	0.6631	0.3316	6.91	0.001

Table 16
Spacing metric mean value for algorithms and related *p*-value for each level.

Level	Spacing metric mean			*p*-Value
	NSGA-II	MOSS	MPACO	
Low	1.06	0.78	0.74	0
Moderate	0.67	0.84	0.65	0
High	0.81	0.90	0.72	0.024
Large	0.78	0.86	0.75	0.069

$$time = 2.61n^2 - 2.82 \times 10^{-1} nT - 21.11nm + 1.47 \times 10^{-2} T^2 + 0.96Tm + 28m^2 - 6.67n - 2.87 \times 10^{-1} T + 112m - 86 \quad (27)$$

5. Conclusions

This paper studied bi-objective project section and scheduling problems. The objectives were to maximize total benefit of selected projects where the benefit of each project is time dependent and to minimize resource usage variation. We first analyzed the drawbacks of available model and algorithms for the problem. Then, we proposed a mixed integer linear programming model. This model solves all shortcomings. We also proposed a multi objective ant colony optimization including four features of ant generation, colonial, Pareto front updating and pheromone updating mechanisms. A comprehensive experiment was designed and the proposed algorithm was compared with two available genetic algorithm and scatter search. The results showed that the proposed algorithm outperformed the two available algorithms.

Future research that can be done on the basis of this research can be divided into two categories: modeling and solving method. In modeling part, we recommend some considerations and developments as follow.

(1) Developing model base on the possibility of stop and resume the projects.
(2) In this paper all sources are considered equal in weight and value which can be prioritized according to their importance.
(3) Considering the time dependency and relations between projects. For example, considering that some projects cannot be started simultaneously, or relations like FS, FF, SF and FF.

Future researches in solving method, can focus on finding better formulation for $p_{i,t}$ and more efficient formulation of heuristic information $(\eta_{i,t})$. Also it may include applying MPACO to continuous problems and other kind of problems.

References

Aljanaby, A., Mahamud, K. R. K., & Norwawi, N. M. d. (2010). Interacted multiple ant colonies optimization framework: An experimental study of the evaluation and the exploration techniques to control the search stagnation. *IJACT, 2*(1), 78–85.

Bakk, M. P. R. (2010). *Metaheuristic algorithms for solving multi-objective/stochastic scheduling and routing problems*. Wien University, pp. 26–27.

Bhattacharyya, R., Kumar, P., & Kar, S. (2011). Fuzzy R&D portfolio selection of interdependent projects. *Computers & Mathematics with Applications, 62*(10), 3857–3870.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). Swarm intelligence from natural to artificial systems. In *A volume in the Santa Fe Institute studies in the science of complexity*. Oxford University Press.

Carazo, A. F., Gómez, T., Molina, J., Hernández-Díaz, A. G., Guerrero, F. M., & Caballero, R. (2010). Solving a comprehensive model for multi objective project portfolio selection. *Computers and Operations Research, 37*(4), 630–639.

Chen, J., & Askin, R. G. (2009). Project selection, scheduling and resource allocation with time dependent returns. *European Journal of Operational Research, 193*(1), 23–34.

Cruz, L., Fernandez, E., Gomez, C., Rivera, G., & Perez, F. (2014). Many-objective portfolio optimization of interdependent projects with 'a priori' incorporation of decision-maker preferences. *Applied Mathematics and Information Sciences, 8*(4), 1517–1531.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*(2), 182–197.

Delévacq, A., Delisle, P., Gravel, M., & Krajecki, M. (2013). Parallel ant colony optimization on graphics processing units. *Journal of Parallel and Distributed Computing, 73*(1), 52–61.

Demeulemeester, E. L., & Herroelen, W. S. (2002). *Project scheduling: A research handbook*. Springer.

Doerner, K. F., Gutjahr, W. J., Hartl, R. F., Strauss, C., & Stummer, C., (2001). Ant colony optimization in multiobjective portfolio optimization. In *4th methaheuristic international conference*.

Doerner, K. F., Gutjahr, W. J., Hartl, R. F., Strauss, C., & Stummer, C. (2004). Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection. *Annals of Operations Research, 131*(1), 79–99.

Doerner, K. F., Gutjahr, W. J., Hartl, R. F., Strauss, C., & Stummer, C. (2006). Pareto ant colony optimization with ILP preprocessing in multiobjective project portfolio selection. *European Journal of Operational Research, 171*(3), 830–841.

Dorigo, M., Maniezzo, V., Colorni, A., (1991). Positive feedback as a search strategy. Technical Report No. 91-016, Politecnico di Milano, Italy.

Dorigo, M., (1992). Optimization, learning and natural algorithms, Ph.D. thesis, Dip. Elettronica e Informazione, Politecnico di Milano, Italy.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), 26*(1), 29–41.

Ellabib, I., Calamai, P., & Basir, O. (2007). Exchange strategies for multiple Ant Colony System. *Information Sciences, 177*, 1248–1264.

Ghorbani, S., & Rabbani, M. (2009). A new multi-objective algorithm for a project selection problem. *Advances in Engineering Software, 40*(1), 9–14.

Gravel, M., Price, W. L., & Gagne, C. (2002). Scheduling continuous casting of aluminum using a multiple objective ant colony optimization metaheuristic. *European Journal of Operational Research, 143*(1), 218–229.

Liang, Y. C., Chen, A. H. L., Kao, W. C., & Chyu, C. C., (2004). An ant colony approach to resource-constrained project scheduling problems. In *Proceedings of the fifth Asia Pacific industrial engineering and management systems conference*.

Lin, C., & Hsieh, P. J. (2004). A fuzzy decision support system for strategic portfolio management. *Decision Support Systems, 38*(3), 383–398.

Liu, S. S., & Wang, C. J. (2011). Optimizing project selection and scheduling problems with time-dependent resource constraints. *Automation in Construction, 20*(8), 1110–1119.

Rabbani, M., Aramoon Bajestani, M., & Baharian Khoshkhou, G. (2010). A multi-objective particle swarm optimization for project selection problem. *Expert Systems with Applications, 37*(1), 315–321.

Tseng, C. C., & Liu, B. S. (2011). Hybrid Taguchi-genetic algorithm for selecting and scheduling a balanced project portfolio. *Journal of Science and Engineering Technology, 7*(1), 11–18.

Wu, J., & Azarm, S. (2000). Metrics for quality assessment of a multiobjective design optimization solution set. *Journal of Mechanical Design, 123*(1), 18–25.

Yu, L., Wang, S., Wen, F., & Lai, K. K. (2012). Genetic algorithm-based multi-criteria project portfolio selection. *Annals of Operations Research, 197*(1), 71–86.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., & Da Fonseca, V. G. (2003). Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation, 7*(2), 117–132.