Assistant Professor Esmaeil MEHDIZADEH, PhD
Faculty of Industrial and Mechanical Engineering, Islamic Azad University, Qazvin Branch, Iran
E-mail: emehdi@qiau.ac.ir (Corresponding author)
Siamak NEZHAD DADGAR, MSc
Faculty of Industrial and Mechanical Engineering, Islamic Azad University, Qazvin Branch, Qazvin, Iran
E-mail: siamak.dadgar@yahoo.com

USING VIBRATION DAMPING OPTIMIZATION ALGORITHM FOR RESOURCE CONSTRAINT PROJECT SCHEDULING PROBLEM WITH WEIGHTED EARLINESS-TARDINESS PENALTIES AND INTERVAL DUE DATES

Abstract. *Resource constrained project scheduling problem with weighted earliness-tardiness penalties is one of the most crucial problems in resource constrained project scheduling problems. This paper focuses on the notable characteristic of the model which is an interval due dates for activities. In this problem setting, activities have an interval activity due date with associated unit earliness and tardiness penalties. Vibration damping optimization (VDO) is proposed to solve this strongly NP-hard model. The proposed VDO algorithm is computationally compared with simulated annealing (SA) algorithm and the results are analyzed and discussed. Response surface methodology (RSM) is employed to tune of the VDO and SA parameters.*

Keywords*: project Scheduling, resource constrained, earliness-tardiness, simulated annealing and vibration damping optimization.*

JEL Classification: M11, C44, C61

1 Introduction

This paper deals with the resource constrained project scheduling problem with weighted earliness-tardiness penalties (RCPSPWET) and interval due dates. It is an extended case of the classical resource constrained project scheduling problem (RCPSP). One measure of performance, which is gaining attention in JIT environments, is the minimization of the weighted earliness–tardiness penalty costs of the project activities. In this problem setting, activities have an interval due date with associated unit earliness and unit tardiness penalty costs. The objective is to schedule the activities to minimize the weighted penalty cost of the project. This problem often occurs in practice since many project

schedulers have to deal with due dates and penalty costs. Costs of earliness include extra storage requirements and idle times and implicitly incur opportunity costs. Tardiness leads to customer complaints, loss of reputation and profits, monetary penalties or goodwill damages. Moreover, tardiness can cause penalties due to delays in the project completion, which is often faced by many firms hiring subcontractors, maintenance crews as well as research teams.

In spite of RCPSP which there are plenty of papers concerning heuristic and exact approaches to solve it, the literature about RCPSPWET is scant. Lawrence and Morton (1993) have studied the due date setting problem of scheduling multiple resource-constrained projects with the objective of minimizing weighted tardiness costs. Kogan and Shtub (1999) have studied the problem of resource-constrained multi-project scheduling with variable-intensity activities. They have developed four dynamic models, based on four types of precedence relations to minimize dynamic earliness and tardiness cost of project activities. Also, they have adopted an efficient time-decomposition approach and an efficient search to solve the models. Vanhoucke et al. (2000) have considered the weighted earliness-tardiness project scheduling problem with the objective of minimizing the net present value in which progress payment occurred. They have presented a branch and bound algorithm to solve the problem. Vanhoucke et al. (2001) have developed an exact recursive search algorithm. They have used the basic idea that the earliness-tardiness costs of a project can be minimized by first scheduling activities at their due dates or at a later time instant if forced so by binding precedence constraints, followed by a recursive search which computes the optimal displacement for those activities for which a shift towards time zero proves to be beneficial. Vanhoucke et al (2001) have presented a branch and bound algorithm that minimizes the weighted earliness-tardiness penalty costs subject to zero-lag finish to start precedence constraints and renewable resource constraint. They have used the mentioned exact recursive procedure as a lower bound in their algorithm. The branching strategy resolved resource conflicts through the addition of extra precedence relations, based on the concept of minimal delaying alternatives developed by Demeulemeester and Herroelen (1992, 1997) and further explored by Icmeli and Erenguc (1996). Vanhoucke et al (2003) have studied the unconstrained project scheduling problem with discounted cash flows where the net cash flows were assumed to be dependent on the completion times of the corresponding activities. The objective was to schedule the activities in order to maximize the net present value of the project subject to the precedence constraints and a fixed deadline. They have introduced a B&B algorithm which computes upper bounds by making piecewise linear overestimations. The algorithm has transformed the problem into a weighted earliness–tardiness project scheduling problem. Since the RCPSPWET is strongly NP-hard as a generalization of RCPSP, thus using metaheuristics to solve it is justified in the literature. Mendes (2003) has presented a genetic algorithm that uses a random key representation and a modified parallel schedule generation scheme (SGS). This genetic algorithm minimizes simultaneously the tardiness, earliness and flow time deviation criteria

Elmaghraby (2005) has demonstrated the significant advantages of adaptive scheduling with a simple example of a project with tardiness penalties and stochastic activity durations. Kwan Woo et al. (2005) have proposed a hybrid genetic algorithm with fuzzy logic controller to solve the resource-constrained multiple project scheduling problem. Objectives were to minimize total project time and to minimize total tardiness penalty. They have proposed a new approach which was based on the design of genetic operators with fuzzy logic controller. Shadrokh and Kianfar (2007) have presented a genetic algorithm, for solving a class of project scheduling problems, called resource investment problem. Tardiness of project was permitted with defined penalty. Afshar nadjafi and shadrokh (2008) presented a new depth-first branch and bound algorithm for the problem, which time value of money is taken into account by discounting the cash flows. Recently, Khoshjahan et al (2013) proposed two meta-heuristics to solve RCPSP with discounted earliness-tardiness penalties.

In this study, we develop a new RCPSPWET model with interval due date consideration. In our model, we assume project activities cannot be interrupted and the resources are renewable, where in each period the renewable resources are limited. The aim is to schedule all activities of the project to minimize the weighted earliness-tardiness penalty costs. A metaheuristic algorithm namely vibration damping optimization (VDO) algorithm is developed to solving RCPSPWET which is strongly NP-hard as a generalization of RCPSP. VDO is a new metaheuristic algorithm which was introduced by Mehdizadeh and Tavakkoli-Moghaddam (2009). This stochastic search method is created based on the concept of the vibration damping in mechanical vibration.

The paper is organized as follows: In section 2 the RCPSPWET with interval activity due date is formulated mathematically. Proposed metaheuristic algorithms and application of them to solve the model are described in section 3. In section 4, parameters of the algorithms are tuned and performance of the algorithms is investigated. Finally, the paper is concluded in section 5.

2 Problem formulation

The deterministic RCPSPWET with interval activity due dates involves the scheduling of project activities in order to minimize the weighted earliness-tardiness costs of the project under resource constraints, with consideration of a permissible interval for activities finish time. The project is represented by an AON network where the set of nodes, N, represents activities and the set of arcs, A, represents finish-start precedence constraints with a time-lag of zero. The activities are numbered from the dummy start activity 1 to the dummy end activity n and are topologically ordered, i.e., each successor of an activity has a larger activity number than the activity itself. The fixed duration of an activity is denoted by d_i ($1 \leq i \leq n$), while [h_i^- , h_i^+] denotes its deterministic interval due date. The

completion time of activity *i* is denoted by the nonnegative integer variable f_i ($1 \le i \le n$).
The earliness of activity *i* can be computed as:

$$E_i = max\ (0,\ h_i^- - f_i)$$

The tardiness of activity *i* can be computed as:

$$T_i = max\ (0,\ f_i - h_i^+)$$

If we let e_i and t_i denote the per unit earliness and tardiness cost of activity *i*, respectively, the total earliness-tardiness cost of activity *i* equals:

$$e_i E_i + t_i T_i$$

It is assumed that $h_1^- = h_1^+ = 0$, $h_n^- = h_n^+ = \infty$, $e_1 = t_1 = \infty$ and $e_n = t_n = 0$.

The objective of the RCSPWET with interval activity due dates is to find a schedule such that the total earliness-tardiness penalty cost is minimized. The way of calculating the value of earliness-tardiness penalty cost depends on the payment model considered. We suppose that the cost due to earliness or tardiness of each activity will impose in progress model (e.g., at the end of each month, earliness or tardiness of each activity may impose some cost to the client).

We have the following notation for RCPSPWET with interval activity due dates:

n = Number of activities
N = set of nodes of acyclic digraph representing the project
A = set of arcs of acyclic digraph representing the project
d_i = Duration of activity *i*
$[h_i^-, h_i^+]$ = Interval due date of activity *i*
f_i = Finish time of activity *i* (integer decision variable)
e_i = Per unit earliness cost of activity *i*
t_i = Per unit tardiness cost of activity *i*
K = Number of renewable resources
a_k = Per-period availability of renewable resource type k ($1 \le k \le K$)
r_{ik} = Per-period usage of renewable resource type k required to execute activity *i*. ($1 \le i \le n, 1 \le k \le K$)

$S(t)$ =The set of activities in progress in period [t -1, t]

The basic form of the RCPSPWET with new assumption about due dates can be conceptually formulated as follows:

$$Minimize \quad \sum_{i=2}^{n-1} e_i E_i + t_i T_i \tag{1}$$

$$S.t:$$

$$f_i \le f_j - d_j \qquad \forall (i,j) \in A \tag{2}$$

$$E_i \ge h_i^- - f_i \qquad \forall i \in N \tag{3}$$

$$T_i \ge f_i - h_i^+ \qquad \forall i \in N \tag{4}$$

$$\sum_{i \in S(t)} r_{ik} \le a_k \qquad k = 1, \ldots, m \;,\; t = 1, 2, \ldots, T \tag{5}$$

$$f_1 = 0 \tag{6}$$

$$f_i \in int^+ \;,\; E_i \in int^+ \;,\; T_i \in int^+ \;,\; i = 1,...,n \tag{7}$$

The objective in equation (1) minimizes the weighted earliness-tardiness cost of the project. Equation (2) enforces the finish to start precedence relations between activities. Equation (3) and (4) compute the earliness and tardiness of each activity. Equation (5) represents the renewable resource constraints. Equation (6) forces the dummy start activity to end at time zero. Equation (7) ensures that the activity finish times, earliness and tardiness of activities to be nonnegative integer values.

Figure (1) demonstrates a graphical representation of the RCPSPWET with interval due date [h^- , h^+] for activity A, in which situations A_1 and A_3 denote tardiness and earliness, respectively. Situation A_2 describes a situation that the activity finishes on time with no penalty.

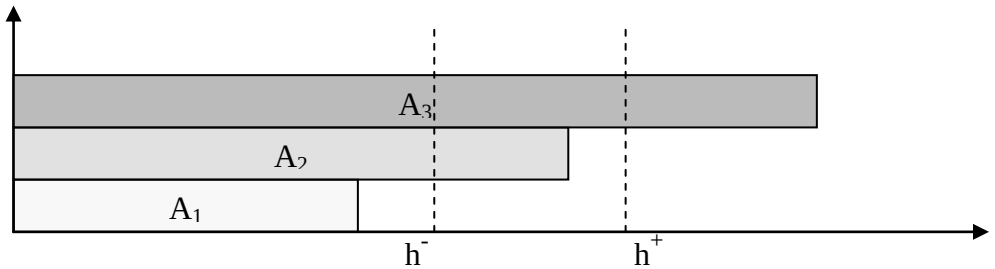

Figure 1. The possible positions for an activity from the view point of due date

3 Proposed Vibration Damping Optimization algorithm

In this section, a vibration damping optimization algorithm is developed for solving RCPSPWET with interval due dates.

3.1 Common features

In this section, we describe the elements of the VDO algorithm.

3.1.1 Solution representation and scheduling scheme

A feasible solution for both approaches is represented by two *n*-element lists. The first one is a precedence feasible permutation of activities, in which each activity i $(1 \le i \le n)$ must occur after all its predecessors and before all its successors. This structure is called the *activity list* .The second one is a list of delays for all activities and is called the *delay list*. The k^{th} element of delay list determines the delay of k^{th} activity in activity list. We use the delay list to create a random delay at start time of activities, after the biggest finish time of their predecessors.

Our scheduling scheme is like the serial scheduling scheme proposed by Kelley (1963) in many features. In serial scheduling, activities are added to the schedule sequentially until a feasible complete schedule is obtained. In each stage the next activity in the priority list is chosen and the first possible starting time is assigned to that activity such that no precedence or resource constraint is violated. Our scheduling scheme some differs. In each stage of our scheduling scheme, when an activity is chosen to schedule, the largest finish time of its predecessors is computed and added to the corresponding number of the activity in the delay list.

The resulted number (time) is the first alternative to survey the resource feasibility to schedule the activity. If the situation is resource feasible, the activity will be scheduled, otherwise, next alternatives will be surveyed. This scheduling scheme is like a situation in serial scheme which we add the number in delay list to the activities duration.

3.1.2 Fitness function

As we mentioned in section 2, the objective is to minimize the total earliness-tardiness penalty of the project's activities. For each solution, we calculate the objective function according to the following formula:

$$\sum_{i=2}^{n-1}(e_i E_i + t_i T_i) \qquad (8)$$

3.1.3 Starting solution

The initial solution for each instance is generated by setting all activities of activity list in an ascending order of activity numbers. All elements of delay list are set to zero. This procedure has been commonly used in local search algorithms before especially in multi-mode scheduling problems.

3.1.4 Neighborhood generation mechanism

Neighborhood solution of a current solution is generated by using one of the following three operators:

- *Activity shift:* it operates only on the activity list in the following ways:
  1. One activity *i* is randomly chosen from the activity list.
  2. The nearest predecessor *p* and the nearest successor *s* of the activity *i*, are found in the activity list.
  3. A position *x* between activities *p* and *s* is randomly chosen.
  4. Activity *i* is moved to position *x* and all activities between *i* and *x+1* (*x-1*) are shifted to the left (right).
- *Delay change:* it operates only on the delay list as follow:
  1. A position *y* is randomly chosen from the delay list.
  2. If the number in position *y* is greater than zero, a discrete number is randomly chosen between -1, 0 and 1, else a number is randomly chosen between 0, 1. It is obvious that the numbers in delay list are nonnegative. Recent chosen number is added to the number in position *y*.
- *Combined move:* it operates simultaneously on both activity list and delay list and is a combination of the activity shift and delay change.

In each iteration of algorithms one of the three mentioned operators is chosen with a certain probability which depends on the size of the considered problem. P_o, P_d and P_c are probability of order change (activity shift), delay change and combined move, respectively.

3.2 Vibration Damping Optimization

There is a useful connection between vibrations damping process and optimization. In the analogy between an optimization problem and the vibration damping process, I) The states of oscillation system represent feasible solutions of the optimization problem, II) The energies of the states correspond to the objective function value computed at those solutions, III) The minimum energy state corresponds to the optimal solution to the problem and rapid quenching can be viewed as local optimization. Vibration damping optimization (VDO) is metaheuristic algorithm that is created based on heuristics from vibration damping

process in physics field. This stochastic search method is similar to SA algorithm in some parts, but it has different base in comparison with SA one. In this section, we describe proposed VDO algorithm that is developed for RCPSPWET to minimize the weighted earliness-tardiness penalty costs.

Figure (2) demonstrates the general process of the proposed VDO algorithm. During the initialization step of the presented algorithm, initial amplitude (A_0), number of neighborhood searches in the search loop for each amplitude (L), damping coefficient (γ) and sigma of Rayleigh distribution (σ) are determined. The amplitude in suggested algorithm has a control parameter role. This factor controls the possibility of the acceptance of a worse solution in various steps of the algorithm. At high amplitude (early in the search), there is some flexibility to move to a worse solution; but at lower amplitude (later in the search) less of this flexibility exists. The algorithm escapement from local optimum is reduced in low amplitude and the accessibility to global optimization is increased in higher amplitude. In addition, in the initialization step, an initial solution X is generated randomly and is set as the current solution, X' $(X' \longleftarrow X)$. In each iteration of the algorithm search loop, a new neighboring solution, X'', is generated by implementing neighborhood structure on the current solution, X'. After the neighboring solution, X'', is produced, the difference in objective function value, $(\Delta = E(X'') - E(X'))$, is computed. If the difference is negative or zero i.e., $\Delta \leq 0$ (for a minimization problem), current solution, X', is replaced with neighboring solution, X'' $(X' \leftarrow X'')$. Besides, In the case of positive difference, $(\Delta > 0)$, the algorithm moves from current solution, X', to neighboring solution, X'', only if:

$$1 - \exp(^{-A^2}\!/_{2\sigma^2}) \succ R \qquad (R \sim U(0,1)) \qquad (10)$$

This rule provides acceptance possibility of worse solutions and increases the possibility of finding a global optimal solution out of a local optimum. If none of the conditions above is met, the current solution is preferred. On condition that all iterations of the search loop are considered, an iteration of the VDO algorithm is accomplished. In that case, stop condition is checked. If stop condition is not met, next iteration of the algorithm begins with new amplitude ($A \leftarrow A_0 \exp(-\gamma t / 2)$), otherwise, the algorithm is finished.

3.2.1 Damping scheme

The damping scheme in our adapted VDO algorithm is according to the following formula, which A_0 is a certain fixed parameter:

(11)

$A = A_0 \exp(-\gamma t / 2)$

3.2.2 Stop criterion

There are many rules for the stopping condition in metaheuristic algorithms which depends on the problem at hand. In this paper, the stop criterion is fulfilled when at least one of two stop criterions occur, first stop criterion occurs when the objective function reaches to zero, second criterion is fulfilled when the amplitude of oscillation reaches to zero.

Begin
Determine the values of (A_0) , (L) , (σ) and (γ) ;
Create an initial solution X;
X' ←X;
A ←A_0;
t ←1 ;
Repeat
For i=1 to L do: {Search Loop}
Generate a neighboring solution X" from
The neighborhood of X' by neighborhood structure;
Calculate Δ=E (X") – E(X');
If Δ ≤ 0 then X" ←X';
Else if Δ ≥ 0 then generate random R in the range (0,1) ;
If 1-$e^{\frac{-A^2}{2\sigma^2}}$ >R then X" ←X' ;
End if
Else X' is preferred;
End if
End for {End of Search Loop}
Update A and t (A ← $A_0 e^{\frac{-\gamma t}{2}}$ and t←t+1);
Until the stop condition is reached
End.

Figure 2. The pseudo code of the proposed VDO algorithm

3.3 Simulated annealing

For validation the VDO algorithm especially for large size problems, the simulated annealing (SA) algorithm as a well-known local search metaheuristic algorithm selected and describe in this section. First time a physical annealing process was simulated by Metropolis (1953). Using of cooling process in optimization problems as simulated annealing optimization problem was suggested by Kirkpatrick (1980). Simulated annealing procedure is one of the neighborhood search methods and is used in both continuous and discrete optimization problems.

3.3.1 Cooling scheme

The adaptive cooling scheme described by Bouleimen (2003) and Boctor (1996) is used in our algorithm to control the cooling process. We use a certain fixed parameter, α , for cooling program according to the following formula.

3.3.2 Markov chain

$$T_{k+1} = \alpha T_k \tag{9}$$

A Markov chain is a random process that consists of a finite number of states with the property that the next state depends only on the current state and some known probabilities *p*, where *p* is the probability of moving from a state to another. The length of the Markov chains, L, specifies the number of neighborhoods generated for a fixed value of the temperature. We assume that the length of Markov chains is deterministic and depends on the size of the problem.

3.3.3 Stop criterion

The stop criterion is fulfilled when at least one of two stop criterions occur; first stop criterion occurs when the objective function reaches to zero, second criterion is fulfilled when the temperature in SA algorithm reaches to the final value.

The objective function, starting solution and neighborhood generation mechanism of VDO algorithm are like the simulated annealing algorithm.

4 Tuning and comparison

In this section, the parameters of the two algorithms are first tuned. Then, the computational performance of the VDO and SA on a set of test problems is analyzed.

4.1 Tuning the parameters

In this section response surface methodology (RSM) is used to determine the effective parameters of the algorithms. RSM is a collection of statistical and mathematical techniques useful for developing, improving, and optimizing processes (Raymond et al. 2002). The aim is to find the levels of the algorithms parameters and subsequently the best values for the both of the value of the objective function and run time. Length of the Markov chain (L), Primary temperature (T_o), Final temperature (T_f), Cooling program parameter (α), Probability of activity list shift (P_o) and Probability of delay list change (P_d) are assumed as input variables for SA and Length of the Markov chain (L), σ, γ, Primary amplitude (A_0), Probability of activity list shift (P_o) and Probability of delay list change are considered as input variables for VDO algorithm.

In order to generalize the statistical results, a set of 27 test problems that include problems with 10, 20 and 30 non dummy activities with 3, 4 and 5 resources are generated using RanGen project scheduling instances generator developed by Demeulemeester et al. (2003). RanGen has three project characteristics including OS, RF and RC. The values 0.25, 0.5 and 0.75 are considered for the order strength, the density of the network. The value of 1 is used to resource factor, the average number of resource types used by an activity and finally the value of 0.25 is considered for resource constrained-ness, the average portion of the resource availability. The earliness and tardiness unit costs are randomly chosen from the [0, 10]. The due date generating method developed by vanhoucke et al. (2001) is used to calculate the due dates of activities with some differences. First, a maximum due date is obtained for each project by multiplying the critical path length with one of the 1, 1.25, 1.5, 1.75, 2, 2.25 and 2.5 factors. Then 2n random numbers are generated between 1 and the maximum due date. Finally, these numbers are sorted and assigned to the activities interval due date limits in increasing order. Tables 1 and 2 present the levels of the input variables for SA and VDO, respectively.

Table 1: The levels of the input variables for SA algorithm

Size	Levels	L	T_o	T_f	α	P_d	P_o
Small problems (10 activity)	Low	10	8	0.3	0.8	0.2	0.15
	High	16	10	0.5	0.9	0.6	0.35
Medium problems (20 activity)	Low	20	12	0.2	0.85	0.2	0.15
	High	26	16	0.4	0.93	0.6	0.35
Large problems (30 activity)	Low	24	14	0.15	0.85	0.2	0.15
	High	28	18	0.35	0.93	0.6	0.35

Table 2: The levels of the input variables for VDO algorithm

Size	Levels	L	σ	γ	A_0	P_d	P_o
Small problems (10 activity)	Low	23	0.7	0.25	0.03	0.2	0.15
	High	29	0.9	0.35	0.05	0.6	0.35
Medium problems (20 activity)	Low	23	0.4	0.35	0.03	0.2	0.15
	High	29	0.5	0.45	0.05	0.6	0.35
Large problems (30 activity)	Low	30	0.1	0.4	0.03	0.2	0.15
	High	40	0.2	0.6	0.05	0.6	0.35

The fractional two-level factorial design, 2^{6-2} with four central points is chosen for the experiments. The algorithms are run according to the design. The results obtained from the analysis of variance test showed a statistically significant curvature on the two response surfaces. Therefore, CCF (Central Composite Face Centered) as second order model, which consists of a fractional two-level model 2^{6-1} with 9 central points and 12 axial points, is employed to perform the experiments in three problem sizes.

The fitted second order models for three problem sizes and two algorithms are shown in the equations (12)-(23). The value of objective function and run time are denoted by Y and Y′, respectively. For example Y_{SA1} in the equation (12), denotes the value of the objective function in small size problems that is obtained from the SA algorithm.

$$Y_{SA1} = 106.262 - 1.808\ L + 0.629\ T_f - 2.535\ \alpha + 0.818\ P_d + 0.877\ P_O + 0.593\ L.\alpha - 0.874\ \alpha.P_d - 0.647\ \alpha.P_O \quad (12)$$

$$Y_{SA1}' = 2.128 + 0.57\ L + 0.067\ T_O - 0.181\ T_f + 0.867\ \alpha - 0.049\ P_d + 0.28\ \alpha^2 - 0.052\ L.T_f + 0.215\ L.\alpha - 0.053\ L.P_d - 0.0887\ T_f.\alpha - 0.051\ \alpha.P_d \quad (13)$$

$$Y_{SA2} = 685.779 + 5.326\ T_f - 10.461\ \alpha \quad (14)$$

$$Y_{SA2}' = 25.636 + 4.091\ L + 1.036\ T_O - 2.822\ T_f + 10.87\ \alpha + 0.021\ P_d + 4.448\ \alpha^2 - 0.553\ L.T_f + 1.176\ L.\alpha + 0.407\ T_O.\alpha - 0.746\ T_f.\alpha - 0.479\ \alpha.P_d \quad (15)$$

$$Y_{SA3} = 1006.21 - 9.157\ L + 7.234\ T_f - 20.606\ \alpha + 17.463\ P_d - 7.635\ P_O \quad (16)$$

$$Y_{SA3}' = 66.765 + 5.827\ L + 2.551\ T_O - 7.632 T_f + 29.298\ \alpha + 1.438\ T_f^2 + 10.948\ \alpha^2 - 0.510\ L.T_f + 2.147\ L.\alpha + 0.907\ T_O.\alpha - 3.049\ T_f.\alpha \quad (17)$$

$$Y_{VDO1} = 105.342 - 1.171\ L + 2.021\ L^2 \quad (18)$$

$$Y_{VDO1}' = 1.849 + 0.198\ L - 0.2\ \gamma - 0.019\ \gamma.L \quad (19)$$

$$Y_{VDO2} = 690.328 - 0.762\ L + 3.507\ \gamma + 2.075\ A_0 + 3.236\ P_d + 5.135\ \gamma^2 - 4.261\ L.\gamma - 3.733\ L.A_0 \quad (20)$$

$$Y_{VDO2}' = 27.01 + 3.235\ L - 3.187\ \gamma + 1.292\ A_0 - 0.311\ L.\gamma - 0.337\ \gamma.A_0 \quad (21)$$

$$Y_{VDO3} = 1009.33 - 4.028\ L - 2.482\ \sigma - 2.771\ A_0 + 13.861\ P_d + 18.33\ L^2 - 10.34\ L.P_d - 7.416\ \sigma.A_0 \quad (22)$$

$$Y_{VDO3}' = 61.445 + 9.257\ L - 12.622\ \gamma + 2.648\ A_0 + 3.722\ \gamma^2 - 1.97\ L.\gamma - 1.358\ \gamma.A_0 \quad (23)$$

Since the aim is to find the parameter values of the algorithms such that both the value of the objective function and run time are simultaneously optimized, we have to solve a bi-objective decision making problem with conflicting objectives. We can combine two objectives as following model:

$$Min \left[w \left(\frac{f_1(x) - f_1^*}{f_1^*} \right)^p + (1-w) \left(\frac{f_2(x) - f_2^*}{f_2^*} \right)^p \right]^{1/p} \quad (24)$$

$$s.t:$$

$$L_i \le X_i \le U_i \quad (25)$$

Where $f_1(x)$ and $f_2(x)$ denote estimated values for two objectives (the value of objective function and run time). The optimum value for two objectives are denoted by f_1^* and f_2^* regardless to the bi-objective model. X_i, L_i and U_i denote factor i, low level of factor i and high level of factor i, respectively. Regarding the importance of the value of the objective function in comparison with run time, the value 0.75 is chosen for w and subsequently 0.25 for 1-w, weights of objective function and run time, respectively. Finally, the optimum values of the parameters are obtained using LINGO 8 software. The optimum parameter values are presented in tables 3 and 4.

Table 3: Optimum values of parameters for SA

Size	L	T_o	T_f	α	P_d	P_o
Small problems	10	8	0.5	0.8	0.2	0.15
Medium problems	20	12	0.4	0.8523	0.2	0.25
Large problems	24	14	0.35	0.8524	0.2	0.35

Table 4: Optimum values of parameters for VDO

Size	L	σ	γ	A_0	P_d	P_o
Small problems	24	0.8	0.9	0.04	0.4	0.25
Medium problems	23	0.45	0.45	0.03	0.2	0.25
Large problems	30	0.2	0.2	0.05	0.2	0.25

4.2 Comparison of the VDO algorithm vs. SA algorithm

In order to compare the performance of VDO algorithm, we have coded the VDO and SA procedures in the MATLAB version 7.4. We have generated a problem set, consisting of 90 problem instances, by the project generator RroGen, using the parameters setting as subsection 4.1. Algorithms are run 5 times for each problem. The experiments are performed on a laptop with 1.6 GHz CPU (Genuine Intel(R)) and 1 GB RAM, limiting the solution times to be less than or equal to 3.5, 22 and 60 CPU seconds for small, medium and large size problems, respectively. Regarding the importance of the value of the objective function, the comparisons are done based upon the value of the objective function criterion. Also, for robustness comparison of the algorithms, we consider the relative deviation percent (RDP) criterion. Relative deviation percent for problem instance *i* at iteration *j* (RDP_{ij}) is computed as follows:

$$RDP_{ij} = \frac{\text{Objective function value at iteration } j \text{ - Best found objective function value}}{\text{Best found objective function value}} \quad (26)$$

4.2.1 Comparison of the algorithms in the small size problems

For the comparison of the algorithms in the small problems, 30 problems including 10 activities with 3, 4 and 5 resources are considered. Table 5 shows the computational results of the proposed algorithms in small size problems. The results of table 5 show that the solutions quality of SA is better than the solutions quality of VDO in small size problems. In addition it seems that the best solution values of SA are better than VDO, overall.

Table 5: Computational results of the proposed algorithms in the small size problems

No. of activities	No. of resources	No. Of problems	Best solutions average		Solutions Average	
			SA	VDO	SA	VDO
10	3	10	254.7	253.8	263.4	260.5
	4	10	235.8	237.4	241.1	242.7
	5	10	185.8	188.2	190.0	192.8

A Paired t-test is used to perform a hypothesis test of the mean difference between paired observations in the algorithms outputs. Analysis of variance results in table 6 show that at 95% confidence level, Null hypothesis is accepted. It means that there is no significant difference between the mean solutions of the two methods.

Table 6: Analysis of variance results for small size problems

	N	Mean	St. Dev	SE Mean
SA	30	231.5	105.4	19.2
VDO	30	232.0	103.4	18.9
Difference	30	-0.47	6.67	1.22

95% CI for mean difference: (-2.96; 2.03)
T-Test of mean difference = 0 (vs. ~ = 0):
T-Value = -0.38, P-Value = 0.705

For robustness comparison of the algorithms, we test the equality of relative deviation percent of two algorithms, statistically. Table 7 shows that at 95% confidence level, equality of relative deviation percent of two algorithms is accepted statistically.

Table 7: Robustness results for small size problems (two way test)

	N	Mean	St. Dev	SE Mean
SA	30	0.3106	0.2445	0.0446
VDO	30	0.2852	0.1809	0.0330
Difference	30	0.0254	0.2330	0.0425

95% CI for mean difference: (-0.0616; 0.1124)
T-Test of mean difference = 0 (vs. ~ = 0)
T-Value = 0.60, P-Value = 0.555

4.2.2 Comparison of the algorithms in the medium size problems

For performance comparison of the algorithms in the medium size problems, 30 problems including 10 activities with 3, 4 and 5 resources are considered. Table 8 shows the computational results of the proposed algorithms in medium size problems.

Table 8: Computational results of the proposed algorithms in the medium problems

No. of activities	No. of resources	No. Of problems	Best solutions average		Solutions Average	
			SA	VDO	SA	VDO
20	3	10	750	763.9	795.16	801.2
	4	10	802.7	784.5	847.44	838.16
	5	10	1113.7	1119.4	1171.66	1178.36

The results of the paired t-test in table 9 show that at 95% confidence level, Null hypothesis is accepted. It means that there is no significant difference between the mean solutions of the two methods.

Table 9: Analysis of variance results for medium size problems

	N	Mean	StDev	SE Mean
SA	30	938	660	121
VDO	30	939	657	120
Difference	30	-1.15	31.81	5.81

95% CI for mean difference: (-13.03; 10.72)
T-Test of mean difference = 0 (vs. ~ = 0):
T-Value = -0.20, P-Value = 0.844

Table 10 shows that at 95% confidence level, equality of relative deviation percent of two algorithms is accepted statistically.

Table 10: Robustness results for medium size problems (two way test)

	N	Mean	St. Dev	SE Mean
SA	30	0.3102	0.1982	0.0362
VDO	30	0.3711	0.2790	0.0509
Difference	30	-0.0609	0.2580	0.0471

95% CI for mean difference: (-0.1573; 0.0354)
T-Test of mean difference = 0 (vs. ~ = 0)
T-Value = -1.29, P-Value = 0.206

4.2.3 Comparison of the algorithms in the large size problems

For comparison of two algorithms in the large size problems 30 problems including 10 activities with 3, 4 and 5 resources are considered. Table 11 shows the computational results of the proposed algorithms in large size problems. In this problem size the situation seems to be unlike previous problem sizes. The results of table 11 show that the solutions quality of VDO is better than the solutions quality of SA in large size problems. Furthermore, the best solution values of VDO is better than SA overall.

Table 11: Computational results of the proposed algorithms in the large problems

No. of activities	No. of resources	No. of problems	Best solutions average		Solutions Average	
			SA	VDO	SA	VDO
30	3	10	2378.8	2383.2	2459.12	2481.08
	4	10	2357.1	2338.6	2427.43	2408.33
	5	10	2328.7	2296.4	2437.16	2407.07

A paired t-test is applied to perform a hypothesis test of the mean difference between paired observations in the algorithms outputs. The results of the statistical test in table 12 show that at 95% confidence level, Null hypothesis is accepted. It means that there is no significant difference between the mean solutions of the two methods.

Table 12: Analysis of variance results for large size problems

	N	Mean	StDev	SE Mean
SA	30	2441	1344	245
VDO	30	2432	1338	244
Difference	30	9.1	84.6	15.4

95% CI for mean difference: (-22.5; 40.7)
T-Test of mean difference = 0 (vs. ~ = 0):
T-Value = 0.59, P-Value = 0.561

Table 13 shows that at 95% confidence level, there is significant difference between the robustness of the two methods, Namely, VDO has more robustness than SA.

Table 13: Robustness results for large size problems (two way test)

	N	Mean	St. Dev	SE Mean
SA	30	0.2459	0.1991	0.0363
VDO	30	0.1792	0.1952	0.0356
Difference	30	0.0667	0.1772	0.0324

95% CI for mean RDP difference: (0.0005; 0.1328)
T-Test of mean RDP difference = 0 (vs. ~ = 0)
T-Value = 2.06, P-Value = 0.048

5 Conclusions

In this paper the resource constrained project scheduling problem with weighted earliness-tardiness penalty costs (RCPSPWET) and interval due dates has been considered. A mathematical model with new assumption about the due dates has been developed. A novel metaheuristic algorithm namely vibration damping optimization (VDO) algorithms have been applied to solve proposed model. A comprehensive computational experiment has been described which has been performed on a set of standard test problems constructed by the RanGen project generator and VDO algorithm was compared with simulated annealing (SA) algorithm and the parameters of two algorithms have been fine-tuned by using response surface methodology (RSM) and central composite face centered (CCF) design on 27 test problems. The consequences of the computational experiment showed that vibration damping optimization algorithm has more robustness than simulated annealing at large size problems.

REFERENCES

[1] **Afshar-Nadjafi, B. and Shadrok, Sh. (2008),** ***An Algorithm for the Weighted Earliness- Tardiness Unconstrained Project scheduling Problem****; Journal of Applied Sciences: 1651-1659;*

[2] **Boctor, F. F. (1996)**, ***Resource-constrained Project Scheduling by Simulated Annealin;*** *International Journal of Production Research 34 2335–2351;*

[3] **Bouleimen, K. and Lecocq, H. (2003)**, ***A New Efficient Simulated Annealing Algorithm for the Resource Constrained Project Scheduling Problem and its Multiple Mode Version****; European Journal of Operational Research: 149, 268–281;*

[4] **Demeulemeester, E., Vanhoucke, M. and Herroelen, W. (2003),** ***Rangen: A Random Network Generator for Activity-on-the Node Networks****; Journal of Scheduling;*

[5] **Demeulemeester, E. and Herroelen, W. (2002),** ***Project Scheduling - A Research Hand Book;*** *Katholieke University Leuven;*

[6] **Demeulemeester, E. L. (1992),** ***Optimal Algorithms for Various Classes of Multiple Resource-Constrained Project Scheduling Problem****; Ph.D thesis, Department of Applied Economics, Katholieke Universiteit Leuven, Belgium;*

[7] **Elmaghraby, S. E. (2005),** ***On the Fallacy of Averages in Project Risk Management****; European Journal of Operational Research, 165 (2), 307–313;*

[8] **Herroelen W. S, Van Dommelen, P. and Demeulemeester, E. L. (1997),** ***Project Network Models with Discounted Cash Flows- A Guide Tour through Recent Developments****; European journal of Operational Research, 100(1), 97-121;*

[9] **Icemeli, O. and Erenguc, S. S. (1996),** ***A Branch & Bound Procedure for the Resource-Constrained Project Scheduling Problem with Discounted Cash Flows;*** *Management science, 42, 1395-1408;*

[10] **Kelley, J. E. (1963),** ***The Critical-path Method: Resources planning and scheduling****; In: Industrial scheduling, Muth, J. F. and G. L. Thompson (eds.), Prentice-Hall, Englewood Cliffs, NJ, 347-365;*

[11] **Kirk Patrick, S., Gellat, C. D, and Vecchi, M. P. (1983),** ***Optimization by Simulated Annealing****; Science: 671 – 680;*

[12] **Kogan, K. and Shtub, A. (1999),** ***Theory and Methodology Scheduling Projects with Variable-intensity Activities: The case of dynamic earliness and tardiness costs****; European Journal of Operational Research, 118, 65-80;*

[13] **Khoshjahan, Y., Najafi, A. A. and Afshar-Nadjafi, B. (2013),** ***Resource Constrained Project Scheduling Problem with Discounted Earliness–tardiness Penalties: Mathematical modeling and solving procedure.*** *Computers & Industrial Engineering, 66(2), 293-300;*

[14] **Lawrence, S. R., Morton, T. E. (1993),** ***Resource-constrained Multi-project Scheduling with Tardy Costs: Comparing Myopic Bottleneck and Resource Pricing Heuristics***; *European Journal of Operational Research,* 64, 168–187;

[15] **Mehdizadeh E. and Tavakkoli-Moghaddam R. (2009),** ***Vibration Damping Optimization Algorithm for an Identical Parallel-machines Scheduling Problem****; In: The 2nd International Conference of Iranian Operations Research Society, Babolsar, Iran, PP. 216-219;*

[16] **Mendes, J. J. M. (2003),** ***Sistema de apoio a` decisa˜o para planeamento de sistemas de produc¸a˜o do tipo project,.*** *PhD thesis, Departamento de Engenharia Meca^nica e Gesta˜o Industrial, Faculdade de Engenharia da Universidade do Porto, Porto, Portugal;*

[17] **Metropoli, N., Rosenbluth, A., Rosenbluth M., Teller, A. and Teller, E. (1953),** ***Equation of State Calculations by fast Computing Machines****; Journal of Chemical Physics 21, 1087–1092;*

[18] **Mika, M., Walig_ora, G. and Wezglarz, J. (2005),** ***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, 639–668;*

[19] **Myers Raymond, H. and Montgomery, D. C. (2002),** ***Response Surface Methodology: Process and Product Optimization Using Designed Experiment;*** *A Wiley-Interscience Publication;*

[20] **Shadrokh, Sh. and Kianfar, F. (2007),** ***A Genetic Algorithm for Resource Investment Project Scheduling Problem, Tardiness Permitted with Penalty****, European Journal of Operational Research, 181, 86–101;*

[21] **Vanhoucke, M. (2001),** ***Exact Algorithms for Various Types of Project Scheduling Problem Non-regular Objectives and Time/Cost tradeoffs;*** *Unpublished Ph.D . Dissertation, Katholieke University Leuven;*

[22] **Vanhoucke, M., Demeulemeester, E. and Herroelen, W. (2000),** ***Maximizing the Net Present Value of a Project with Progress Payments****; Research Report 0028. Department of Applied Economics, Katholieke Universiteit Leuven;*

[23] **Vanhoucke, M., Demeulemeester, E. and Herroelen, W. (1999),** ***An Exact Procedure for the Resource-Constrained Weighted Earliness-Tardiness Project Scheduling Problem****; Operations Management Group, Department of Applied Economics;*

[24] **Vanhoucke, M , Demeulemeester , E and Herroelen,W , (2001),** ***An Exact Procedure for the Resource-Constrained Weighted Earliness-Tardiness Project Scheduling Problem .*** Operations Management Group, Department of Applied Economics, Katholieke Universiteit Leuven, Naamsestraat 69, B-3000 Leuven, Belgium;

[25] **Vanhoucke, M., Demeulemeester, E. and Herroelen, W. (2003*), Discrete Optimization Progress Payments in Project Scheduling Problems****; European Journal of Operational Research, 148, 604–620;*

[26] **WooK, K., Yunb, Y., S, Yoonc J. M., Gend, M., Yamazakia, G. (2005),** ***Hybrid Genetic Algorithm with Adaptive Abilities for Resource-constrained Multiple Project Scheduling****, Computers in Industry, 56, 143–160.*