A Simulated Annealing Algorithm for the Preemptive Multi-Objective Multi-Mode Resource-Constrained Project Scheduling Problem

Zahra Zare
Wright State University
Dayton, OH, 45435, USA

Maysam Ashrafzadeh
Islamic Azad University
Najafabad, Iran

ABSTRACT
This paper proposes a new mathematical model for the preemptive multi-objective multi-mode resource-constrained project scheduling problem (P-MOMRCPSP) that focuses on two objectives: minimizing the makespan and maximizing the net present value (NPV). The model allows multiple execution modes for each project activity and permits activities to be preempted at any appropriate time and resumed later. This problem is classified as NP-hard, necessitating the use of the Simulated Annealing (SA) algorithm to achieve either a global optimal solution or a satisfactory one. The SA algorithm offers significantly shorter computation times compared to exact methods, making it well-suited for solving large-scale problems. Finally, the model is validated through a numerical example.

Keywords
Project scheduling, preemption, multi-objective, multi-mode, Metaheuristic, Simulated Annealing (SA) algorithm.

1. INTRODUCTION
The resource-constrained project scheduling problem (RCPSP) aims to schedule project activities to ensure the shortest possible project duration, subject to precedence and resource constraints. Precedence constraints specify that no activity can start until all its predecessors are completed. An extension of this problem is the multi-mode resource-constrained project scheduling problem (MRCPSP), where each activity can be executed in one of several modes. Each mode is characterized by specific durations and resource requirements. In the non-preemptive case, once an activity starts, it must run to completion without interruption. On the other hand, the preemptive resource-constrained project scheduling problem (PRCPSP) allows activities to be preempted at any integer time point and resumed later at no additional cost [49]. This paper focuses on the preemptive case, where activities can be interrupted multiple times. As a generalization of the RCPSP, the preemptive multi-mode resource-constrained project scheduling problem (P-MRCPSP) integrates multiple execution modes with preemptive scheduling and is classified as NP-hard [3].

Many studies have explored the multi-objective MRCPSP and the use of SA; however, the present study advances this research by incorporating preemptive scheduling with renewable and non-renewable resources to develop a model that more accurately reflects real-world challenges. Furthermore, the SA algorithm is enhanced through the implementation of dynamic cooling schedules and advanced strategies for generating new solutions. These upgrades make it faster and more effective, especially when dealing with large, complex problems.

The objectives for the project scheduling problem can generally be divided into two major types: regular and irregular. The performance objectives of a non-decreasing function with respect to activity start time are regular performance objectives. Some representative objectives that minimize the Makespan, project cost, and project delay time are discussed. All these target values can be improved by advancing the start or completion time of an activity. In contrast, changing the start or completion time of an activity does not affect target values like maximizing the NPV and addressing the resource leveling problem, which are considered irregular performance objectives [16]. The objectives of the model presented in this paper are: (1) minimizing the total makespan of the project and (2) maximizing the NPV.

The rest of this paper is structured as follows: Section 2 reviews the related literature on resource-constrained project scheduling problems. Section 3 describes the problem and introduces the mathematical model for the preemptive multi-mode resource-constrained project scheduling problem (P-MOMRCPSP). Section 4 provides an overview of the SA method, followed by the algorithm's steps for solving P-MOMRCPSP in Section 5. A numerical example is discussed in Section 6, and the results of sensitivity analysis and performance evaluation are presented in Section 7. Finally, Section 8 concludes the paper with key takeaways and suggestions for future research.

2. LITERATURE REVIEW
The basic RCPSP assumes that an activity cannot be interrupted once it has started. However, Bianco et al. [6], Brucker and Knust [10], Debels and Vanhoucke [14], Demeulemeester and Herroelen [15], and Nudtasomboon and Randhawa [34] allow activity preemption at discrete points in time, meaning an activity can be interrupted after each integer unit of its processing time. Franck et al. [20] propose a calendar concept for project scheduling that includes preemptive scheduling. A calendar is defined as a binary function that determines for each period whether activity execution is possible or if a break occurs during which an activity may not be started or continued.

Later, Zare et al. [54, 55] developed a mathematical model for the P-MRCPSP. This model introduces multiple execution modes for each activity and permits activities to be interrupted and restarted at any time without incurring additional costs. This level of flexibility makes the model highly applicable to complex, real-world project management scenarios.

Alcaraz et al. [3], Bouleimen and Lecocq [8], Hartmann [21], Jarboui et al. [25], Jozefowska et al. [26], Ozdamar [35], Pesch [36], and Varma et al. [48] discuss multi-mode problems without nonrenewable resources. Multi-mode problems with

generalized precedence constraints have been considered by Barrios et al. [5], Brucker and Knust [10], Calhoun et al. [11], Reyck and Herroelen [41], Drexl et al. [17], Heilmann [23, 24], Nonobe and Ibaraki [33], and Sabzehparvar and Seyed Hosseini [42]. Zhu et al. [58] employ a multi-mode problem with generalized resource constraints. Salewski et al. [43] extend the multi-mode RCPSP by introducing so-called mode identity constraints. Schultmann and Rentz [44] present a case study that demonstrates how the multi-mode RCPSP can be applied to projects involving the dismantling of buildings.

Voß and Witt [50] utilize the multi-mode RCPSP with an objective function that integrates makespan, weighted tardiness, and setup costs, facilitating activity batching. Słowinski [45] is credited as the first to outline the framework of the multi-objective resource-constrained project scheduling problem (MORCPSP) and list various objectives. Subsequent research on MORCPSP identifies primary objectives such as makespan, activity tardiness, NPV, resource investment (RI), and robustness. Wang and Zheng [53] propose a multi-objective Drosophila optimization algorithm aimed at minimizing makespan and total cost. Tirkolaee et al. [47] solved the multi-objective multi-mode resource-constrained project scheduling problem to maximize NPV while minimizing completion time. Al-Fawzan and Haouari [4] merge makespan minimization with maximization of total free slack into a unified objective. Dealing with multiple objectives, another approach is to generate Pareto-optimal schedules, as has been done by Davis et al. [13], who have demonstrated the minimization of makespan and renewable resource overutilization simultaneously.

Various meta-heuristic and heuristic algorithms have been developed for solving RCPSP and MRCPSP. Meta-heuristic algorithms improve upon heuristic algorithms by drawing on concepts from different disciplines and abstracting them to form general algorithms, independent of specific problem structures and frameworks. Heuristics can be used to generate the initial solutions required by meta-heuristic algorithms. Kolisch and Drexl [27] initiated a local search procedure for non-preemptive resource-constrained project scheduling problems, where activity durations are discrete functions of committed renewable and nonrenewable resources. Their methodology begins with an initial solution and iteratively improves it by moving the mode assignments of activities in order to create the best feasible schedule. Hartmann [21] developed a GA that included single and multi-pass improvements in the local search component. Another variant of a GA for the MRCPSP was proposed by Alcaraz et al. [3], where solution representation was based on forward/backward genes and had a new fitness function to improve performance. Peteghem and Vanhoucke [37, 38] introduced GA to preemptive and non-preemptive versions of MRCPSP with considerations of resource availability to improve solution robustness. Słowinski et al. [46], Boctor [7], Jozefowska et al. [26], and Bouleimen and Lecocq [8] have developed several SA algorithms to find the near-optimal solution for MRCPSP using probabilistic search. Zhang et al. [57] and Jarboui et al. [25] proposed the use of particle swarm optimization (PSO) in solving the MRCPSP for global optimization with the help of swarm intelligence. Coelho and Vanhoucke [12] provide a two-step approach to the MRCPSP involving mode assignment and the RCPSP. This basically decomposes the complexity of the problem at hand. Wang and Fang [52] developed a shuffled frog-leaping algorithm for MRCPSP, utilizing virtual frogs and a multi-mode serial schedule generation scheme to efficiently explore and exploit the search space. Elloumi et al. [19] addressed MRCPSP with different execution modes of tasks and proposed an evolutionary algorithm coupled with a reactive multi-objective heuristic in a bid to deal with complex objectives and constraints. Zoraghi et al. [59] considered MRCPSP with material ordering and presented three hybrid meta-heuristic algorithms for near-optimum solutions. Bredael and Vanhoucke [9] propose a new genetic algorithm for solving the resource-constrained multi-project scheduling problem.

Other metaheuristics to solve MRCPSP have been proposed by other authors, such as Elloumi and Fortemps [18], Lova et al. [28], Lova et al. [29], Mori and Tseng [31], Muritiba et al. [32] Nonobe and Ibaraki [33], Zhang [56], Peteghem and Vanhoucke [39], and Voskresenskii et al. [51].

3. PROBLEM DEFINITION

The project is represented as an activity-on-the-node network *G (N, A)*, where *N* is the set of activities and *A* is the set of pairs of activities between which a finish-start precedence relationship with a minimal time lag of *0* exists. A set of activities, numbered from *1* to $/N/$ with a dummy start node *0* and a dummy end node $/N/ + 1$, is to be scheduled on a set *R* resources. Each activity $i \epsilon N$ is performed in a mode m_i, which is chosen out of a set of $/M_i/$ different execution modes $M_i=\{1,\ldots,. /M_i/\}$. The duration of activity *i*, when executed in mode m_i, is d_{im}. Each mode m_i requires $r_{im,k}$, nonrenewable resource units and $r_{im,z}$ renewable resource units. A schedule *s* is defined by a vector of activity start times s_i and a vector denoting their corresponding execution modes m_i. A schedule is said to be feasible if all precedence and resource constraints are satisfied. In the P-MOMRCPSP, activities are allowed to be preempted at any time and restarted later at no additional cost. Therefore, each duration unit *v* of an activity *i* scheduled in mode m_i (with $v \epsilon \{0, \ldots, d_{im} -1\}$) is assigned a starting time s_{iv}. The objectives of the P-MOMRCPSP are to minimize the makespan and maximize the NPV of the project.

3.1 Mathematical Model

In this section, a novel mathematical model is introduced for the preemptive multi-objective multi-mode resource-constrained project scheduling problem (P-MOMRCPSP).

3.1.1 Indices and parameters and variables

$\boldsymbol{T}$: project time window

$\boldsymbol{N}$: number of activities

$\boldsymbol{i}$: index of activity

$\mathbf{0}$: dummy start node

$\boldsymbol{n+1}$: dummy end node

$\boldsymbol{m}$: Index of mode

$\boldsymbol{\alpha}$: Discount Rate

$\boldsymbol{P_i}$: Positive cash flow for activity *i*

$\boldsymbol{U_{im}}$: Negative cash flow for activity *i* in mode *m*

$\boldsymbol{S_{i,lm}}$: start time of lth units of activity *i* in mode *m* where each activity *i* is broken into d_{im}

$\boldsymbol{d_{im}}$: duration of Activity *i* executed mode *m*

$\boldsymbol{t}$: index for a period of time

$\boldsymbol{k}$: index of nonrenewable resource

$\mathbf{Z}$: index of renewable resource

$\boldsymbol{a_k}$: availability of each nonrenewable resource type *k* in each time period

$\boldsymbol{a_z}$: availability of each renewable resource type *z* in each time period

$\boldsymbol{r_{im,k}}$: each activity *i* in mode m requires *k* nonrenewable resource units

$\boldsymbol{r_{im,z}}$: each activity *i* in mode m requires *z* nonrenewable resource units

E: a very large positive number

$\boldsymbol{x_{im}}$	*1* if activity *i* is completed in mode *m* 0 otherwise
$\boldsymbol{y_{iht}}$	*1* if h^{th} units for activity *i* is executed in period *t* 0 otherwise

3.1.2 Proposed mathematical model

The P-MOMRCPSP can be stated as follows:

$$min\ Makespan\ min(S_{n+1,0}) \quad (1)$$

$$max\ NPV = \sum_{i=1}^{N} CF_i(1+\alpha)^{-(S_{n+1,0})} \quad (2)$$

$$subject\ to$$

$$S_{0,0} = 0 \quad (3)$$

$$S_{i,d_{im}m} \le S_{j,1m} + (1 - x_{im})E + (1 - x_{jm})E$$
$$m = 1,\dots M, \qquad i = 1,\dots,N \quad (4)$$

$$S_{i,(v-1)m} + 1 \le S_{i,vm} + (1 - x_{im})E$$
$$m = 1,\dots M, \qquad i = 1,\dots,N, \qquad V = 1,\dots,d_i \quad (5)$$

$$S_{n+1,0} \le T \quad (6)$$

$$\sum_{i=1}^{N}\sum_{m=1}^{M} x_{im} = 1 \quad (7)$$

$$\sum_{i=1}^{N}\sum_{t=0}^{T} y_{iht} \le 1 \quad (8)$$

$$\sum_{t=0}^{T}(t \times y_{iht}) - \sum_{m=1}^{M} S_{ijm} = 0 \quad (9)$$

$$\sum_{i=1}^{N}\sum_{m=1}^{M} r_{im_i,k}x_{im} \le a_k \quad (10)$$

$$\sum_{t=0}^{T}\sum_{i=1}^{N} y_{iht} \times \sum_{m=1}^{M}(x_{im} \times r_{imz}) \le a_Z \quad (11)$$

$$p_i - \sum_{i=1}^{N}\sum_{m=1}^{M} u_{im}x_{im} = CF_i \quad (12)$$

$$S_{i,vm} \in int^{+} \quad (13)$$

The objective function (1) minimizes the total makespan of the project, and the objective function (2) maximizes the NPV. Constraint (3) ensures that the project starts at time zero. Constraint set (4) ensures that the earliest start time of an activity j cannot be earlier than the finish time of the last unit of duration of its predecessor i. Constraint set (5) guarantees that the start time for every time instance of an activity is at least one-time unit later than the start time of the previous unit of duration. Constraint set (6) ensures that the makespan does not exceed T. Every activity has to be executed exactly in one mode m, which is ensured by constraint (7). Constraints (8) and (9) determine the processing time of each activity section. Constraints (10) and (11) deal with the nonrenewable and renewable resource constraints, respectively. Constraint (12) guarantees a positive cash flow for every activity i. This net cash flow may be simply expressed as the difference between the income and the expenses related to activity i. It is crucial to note that the income generated from performing each activity is independent of how the activity is carried out. In other words, the modes and resources used to execute an activity do not affect the income derived from the project. However, the expenses of each activity are directly related to how the activity is executed, and the optimal approach should be chosen for each activity to maximize the NPV added. Constraint (13) ensures that the start times of activities are nonnegative integer values.

4. SIMULATED ANNEALING

SA is a local search method inspired by the physical annealing process studied in statistical mechanics [1]. An SA algorithm repeats an iterative neighbor generation procedure and follows search directions that improve the objective function value. While exploring the solution space, the SA method offers the possibility to accept worse neighbor solutions in a controlled manner to escape from local minima. More precisely, in each iteration, for a current solution x characterized by an objective function value $f(x)$, a neighbor x' is selected from the neighborhood of x denoted $N(x)$ and defined as the set of all its immediate neighbors. For each move, the objective difference $\Delta = f(x') - f(x)$ is evaluated. For minimization problems, x' replaces x whenever $\Delta \leqslant 0$. Otherwise, x' could also be accepted with a probability $P = e(-\Delta)/T$. The acceptance probability is compared to a number $y_{random} \in [0,1]$ generated randomly and x' is accepted whenever $P > y_{random}$.

The factors that influence acceptance probability are the degree of objective function value degradation Δ (smaller degradations induce greater acceptance probabilities) and the parameter T called temperature (higher values of T give higher acceptance probabilities). The temperature can be controlled by a cooling scheme specifying how it should be progressively reduced to make the procedure more selective as the search progresses toward neighborhoods of good solutions. There exist theoretical schedules guaranteeing asymptotic convergence toward the optimal solution. However, they require infinite computing time. In practice, much simpler schedules with finite computing times are preferred, even if they do not guarantee an optimal solution.

A common finite time implementation of SA involves decreasing the temperature T in S steps, starting from an initial value T_0 and using an attenuation factor β $(0 < \beta < 1)$. The initial temperature T_0 is supposed to be high enough to allow acceptance of any new neighbor proposed in the first step. In each step S, the procedure generates a fixed number of neighbor solutions N_{sol} and evaluates them using the current temperature value $T_s = \beta^{s} T_0$. The whole process is commonly called "cooling chain". Adapting SA to an optimization problem involves defining its specific components: a solution representation of the problem, a method for calculating the objective function value, a neighbor generation mechanism for exploring the solution space, and a cooling scheme including stopping criteria. These adaptation steps for our new adaptations of SA for P-MOMRCPSP are described below.

5. SIMULATED ANNEALING ALGORITM FOR SOLVING P-MOMRCPSP

Abbasi et al. [2], Bouleimen and Lecocq [8], Mika et al. [30], He et al. [22], Jozefowska et al. [26], and Rahimi et al. [40] have successfully used the SA algorithm for a significant number of project scheduling problems. In this section, the SA algorithm designed to solve the preemptive multi-objective multi-mode resource-constrained project scheduling problem (P-MOMRCPSP) is presented.

5.1 Step One:

In this step, project information is read, and an initial solution is generated to start the SA algorithm. A feasible list of activity sequences is determined as the initial solution to initiate the SA procedure. The choice of an appropriate initial solution significantly affects the convergence speed and the quality of the final solution of the SA algorithm. Various methods exist for generating a feasible schedule (a feasible list of activity sequences) from project activities. In the present study, the following method is applied to generate the initial solution:

5.1.1 Obtaining the initial solution:

In the first stage, the project activities are entered into a table, similar to Table 1, based on their precedence relationships.

Table 1. Sample Table for Determining the Initial Solution

activity	d_{im}	r_{im}	*precedence activities*			
1						
2						
3			*1*	*2*	*0*	*0*
⋮						

In the right column of the table, the precedence activities for each activity are listed. For instance, in the table above, activities 1 and 2 are precedence activities for activity 3. It should be noted that, to obtain a feasible initial solution, only the data related to one of the activities is used.

In the second stage the *early time* matrix is formed. This matrix is $i \times 2$, where the number of rows is equal to the number of existing activities. For each activity, the earliest start time and the earliest finish time are recorded in the matrix.

$$\begin{array}{cc} ES & EF \\ \left| \begin{array}{cc} 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{array} \right| & \end{array}$$

In the third stage, to complete the early time matrix, which currently has all its elements set to zero, the earliest start time and the earliest finish time for each activity is calculated. This task is performed as follows:

$for\ \ i = 1\ to\ size(D, 1)$
$if\ \ pre(i, 1) = 0\ then$
$ES = 0$
$EF = 0 + d_i\ else$
$for\ \ j = 1, \ldots J$
$ES = max(EF_j + 1)$
$EF = ES + d_i$
end

in the fourth stage, the *split matrix* is constructed to account for the preemption of activities in order to obtain a feasible solution. The split matrix is $i \times T$ matrix, where the number of columns is equal to the project's time horizon. The elements of this matrix are defined as follows

$for\ \ i = 1\ to\ size(D, 1)$
$for\ \ j = ES\ to\ EF$
$split\ (i, j) = 1$

In the fifth stage, the *resource consumption matrix* is generated. This matrix is obtained by multiplying each "one" in the row corresponding to activity *i* by the resource consumption of that activity.

In the sixth stage, the R matrix is constructed. This matrix is obtained by summing the rows of the resource consumption matrix. This matrix shows the amount of resource R used each day. To level resources and ensure that the daily consumption of each resource does not exceed the available capacity, the largest element in the matrix is first identified. If this value exceeds the available capacity, the following steps are performed:

1. Identify the day on which the maximum *R* occurs
2. Randomly select activity *X*. If the entry corresponding to variable *x* on that day in the *split* matrix is *1*, change the last written *1* plus 1 to *1* and change the initial *1* to *0*. This effectively delays the activity by one day to level the resources.
3. If variable *x* has value of zero in *split matrix*, select another random activity and repeat all the above steps for that activity.

The solution of the previous six stages provides an initial feasible solution that satisfies all constraints. It should be noted that after every stage, the early time matrix, the split matrix, the resource consumption matrix, and the R matrix are updated. The stages to obtain the initial solution are illustrated in figure 1.

5.2 Step Two:

The list of activities generated in step one is considered the best list of activity sequences (best-list), and the completion time of the n^{th} activity is regarded as the best value of the objective function (best-obj). In addition to minimizing the project completion time, this study also aims to maximize the net present value (NPV). To combine these two objective functions, the daily project cost is calculated. It is assumed that a penalty of *n* dollars is incurred for each day of project delay; therefore, the monetary value of each day is set to *n* dollars. Furthermore, the second objective function is multiplied by a negative value to transform it into a minimization problem. This allows the two objective functions to be combined into a

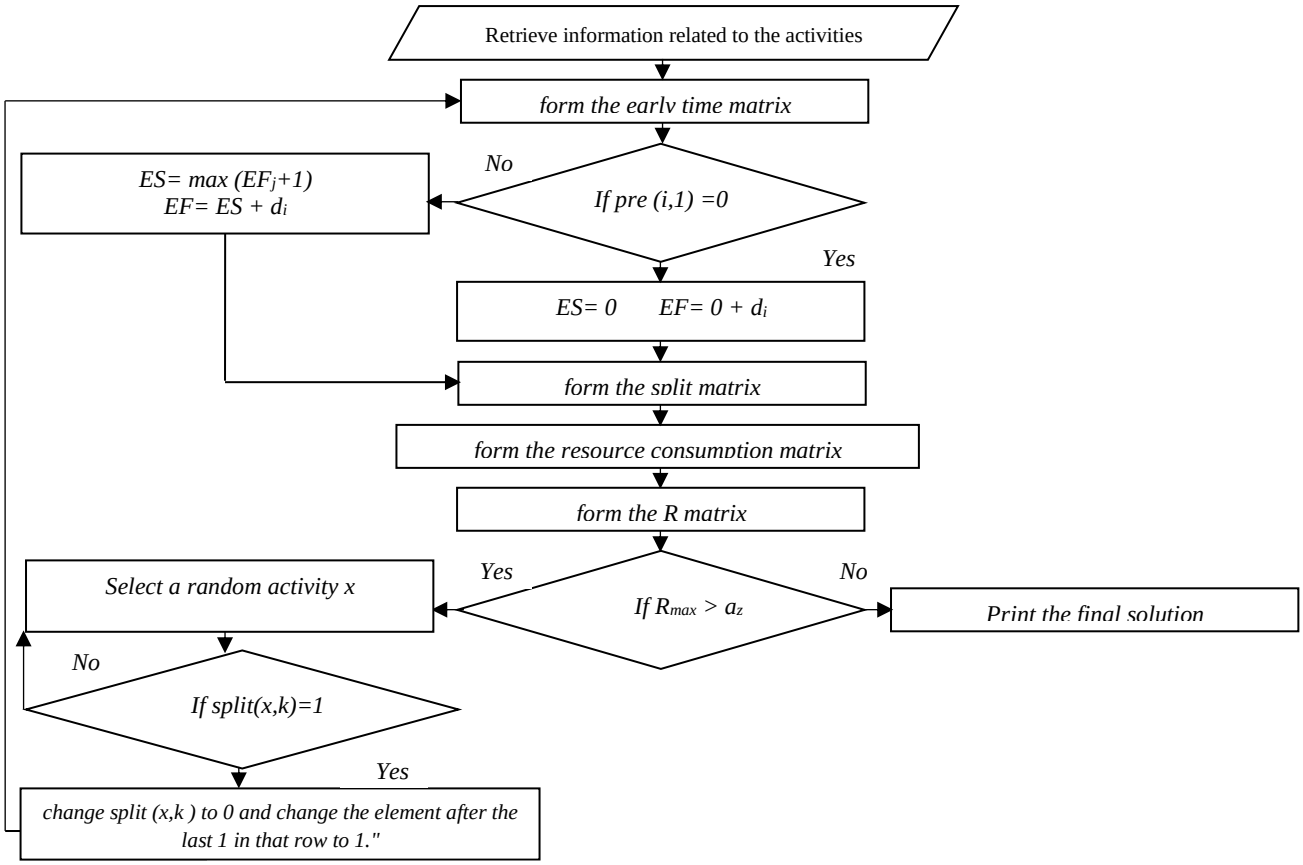


Fig 1: Initial solution generation algorithm

single objective. The list of activity sequences obtained at the end of each iteration is referred to as the current list (current-list), and the corresponding value of the objective function for this list is referred to as the current objective (current-obj). In the first iteration, the current list is assigned as the best list, after which the procedure proceeds to step three.

5.3 Step Three:

In this step, the parameters of the SA algorithm are determined.

1. The initial temperature is considered a multiple of the objective function value ($T_s = k * best - obj$), which in this project is set to *100.*
2. The maximum number of iterations for stopping the algorithm is set to *10,000.*
3. Boltzmann's constant k_B is assumed to be *1*.
4. The final temperature for stopping the algorithm is set to *0.01 (T_f = 0.01).*
5. The iteration counter is set to *0 (iterate = 0).*
6. The temperature is set to the starting temperature *($T = T_S$).*

5.4 Step Four:

In this step, the optimal solution is sought using the SA algorithm. the iteration counter is increased by setting *interat = interat + 1*, then a new neighborhood is generated for the current point. To move to a neighboring point in this study, the state of an activity is modified and all feasibility steps as explained in the initial solution determination section are applied. This will result in a feasible neighboring point. From the neighborhood list, calculate the objective function value and assume the obtained value as the objective function value of the neighborhood *(nbr - obj).* The procedure then proceeds to step five.

5.5 Step Five:

In this step, the changes in the objective function is calculated. The changes in the objective function in each iteration are equal to the difference between the objective function in this iteration and the previous iteration, defined as follows:

$$\Delta obj = (nbr - obj) - (current - obj)$$

If *$\Delta obj<0$,* it means an improvement in the objective function, and the procedure proceeds to step seven.

If *$\Delta obj=>0$,* it means there is no improvement in the objective function, and the procedure proceeds to step six.

5.6 Step Six:

A random number between *0* and *1* is generated and considered as p'_r. The value of *exp(-Δobj/T)* is calculated and denoted by p_r. If $p_r > p'_r$, the procedure then proceeds to Step Seven; otherwise, it advances to step eight.

5.7 Step Seven:

The best value of the objective function is updated to match the objective function value of the generated neighborhood. he following changes are applied before proceeding to step nine.

$$best - obj = nbr - obj$$
$$best - list = nbr - list$$
$$current - obj = nbr - obj$$
$$current - list = nbr - list$$

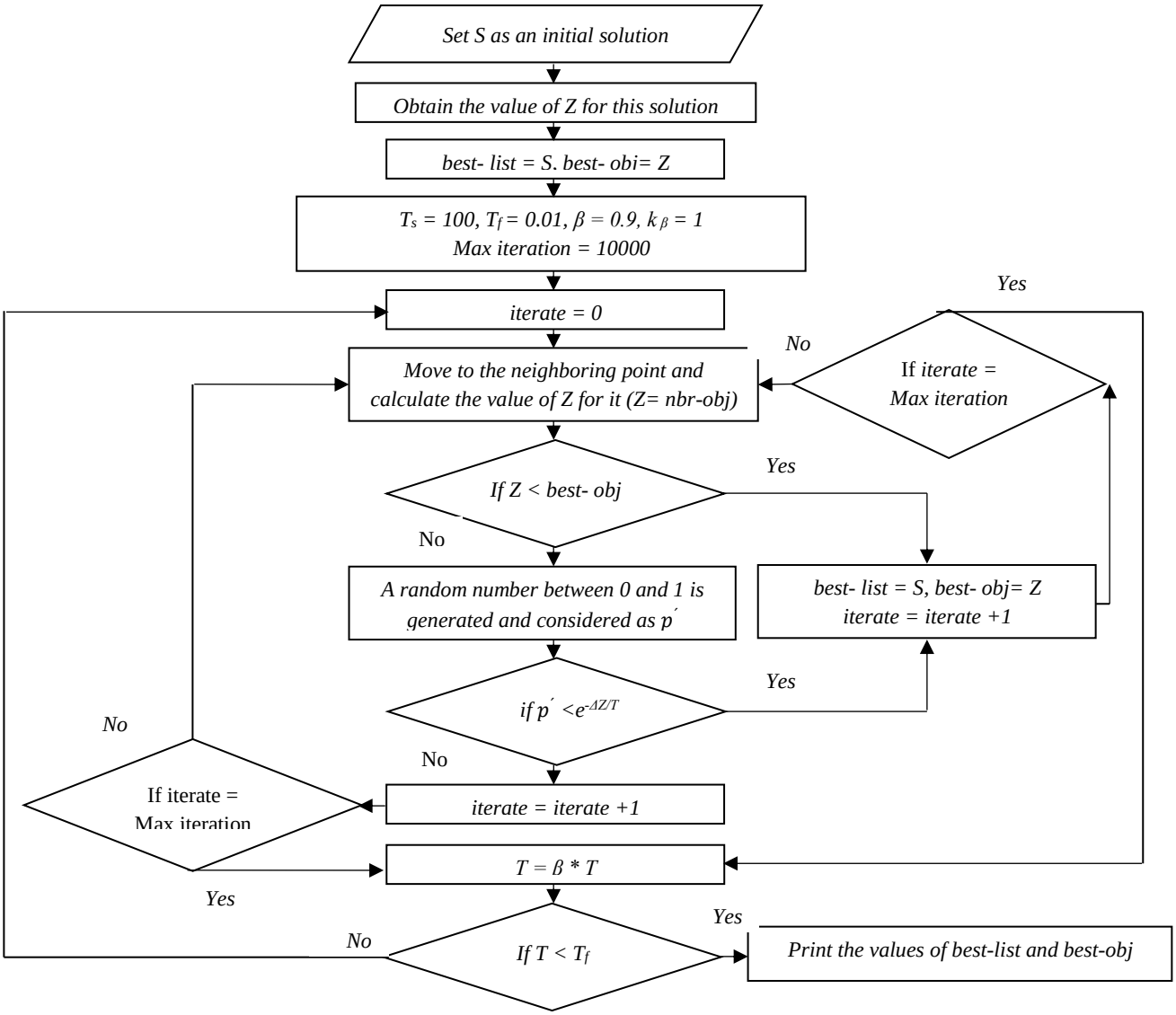


Fig 2: Simulated annealing algorithm for solving the P-MOMRCPSP

5.8 Step Eight:

The following changes are applied, and the procedure proceeds to Step Nine.

$current - obj = nbr - obj$
$current - list = nbr - list$

5.9 Step Nine: Cooling chain

At this stage, the temperature is reduced according to the cooling schedule, and the procedure proceeds to step ten. In this study, a cooling schedule is applied in which the temperature decreases according to the equation $T = \beta * T$.

5.10 Step Ten: Re-annealing

To avoid being trapped in a local optimum after accepting some new points, the temperature is increased back to the initial temperature, and an attempt is made again at higher temperatures by use of search algorithm. In this research, if the best value for the objective function remains constant for 50 consecutive iterations, re-annealing is performed, and the temperature is set to the initial temperature $(T=T_S)$, Then, the procedure proceeds to step eleven.

5.11 Step Eleven: Algorithm Stopping Conditions

Here, two conditions for stopping the algorithm have been defined:

- Condition One: If the number of iterations equals the maximum determined iterations.
- Condition Two: If the temperature equals the final temperature (T_f).

If any one of these conditions holds, then the algorithm stops otherwise it goes back to step four.

The stages of the SA algorithm for solving the project scheduling problem are shown in figure (2)

6. NUMERICAL EXAMPLE

In this section of the research to verify the accuracy of the proposed model, a numerical example is presented. This problem involves a network with six activities numbered from 0 to 5, where activities 0 and 5 are dummy activities (Figure (3)). In this problem, one renewable resource with 11 available

units and one non-renewable resource with 10 available units are considered.

The activities in this project are preemptive and have two execution modes. The relevant information, including resource consumption and precedence relationships of the activities for mode *m=1*, is provided in table *2*, and for mode *m=2*, in table *3*. This is a simple example of a project scheduling problem, with its network depicted in the following figure:

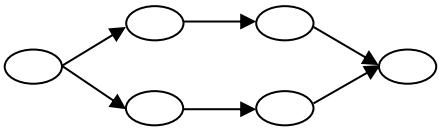


Fig 3: Project Network

Table 2. information of project for m=1

Activities	d_i	$r_{i1,k}$	$r_{i1,z}$	***Prerequisites***
0	*0*	*0*	*0*	*1, 2*
1	*3*	*3*	*3*	*4*
2	*4*	*4*	*5*	*3*
3	*2*	*7*	*5*	*5*
4	*1*	*1*	*7*	*5*
5	*0*	*0*	*0*	-

Table 3. information of project for m=2

Activities	d_i	$r_{i2,k}$	$r_{i2,z}$	***Prerequisites***
0	*0*	*0*	*0*	*1, 2*
1	*2*	*5*	*4*	*4*
2	*3*	*2*	*4*	*3*
3	*1*	*4*	*4*	*5*
4	*2*	*3*	*2*	*5*
5	*0*	*0*	*0*	-

The project time horizon is equal to *10* time units, and the discount rate is *15%*. If the project is delayed, the contractor is fined 7000 monetary units per day. The cost of using resources per unit time is *100* monetary units with both types costing similar amount. Table *4* presents the revenue generated by activities.

Table 4. Income generated from activities

Activities	*1*	*2*	*3*	*4*
Income	*5000*	*2000*	*3500*	*1800*

The model was coded in MATLAB to be solved by the SA algorithm. Tables *5* and *6* show the results obtained from running the numerical example by the SA algorithm with max iteration =1000, $T_S = 100$, $T_f = 0.001, \beta = 0.95$.

Table 5. Initial results

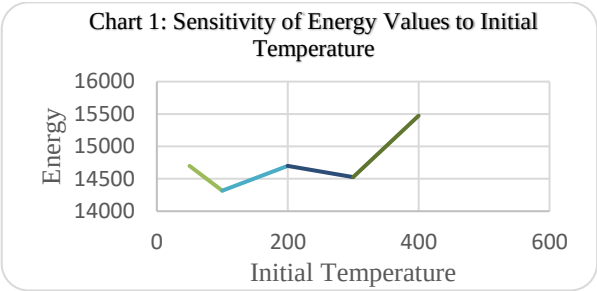
Makespan =5			NPV= 14281		
$x_{11}=1$	$x_{12}=0$	$x_{21}=1$	$x_{22}=0$	$x_{31}=1$	$x_{32}=0$
$x_{41}=1$	$x_{42}=0$	Initial Energy =14526			

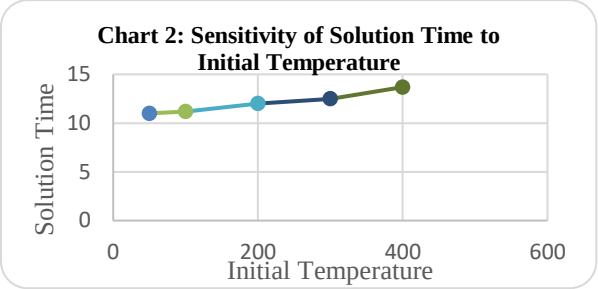
Table 6. Final results

Makespan =5			NPV= 14281		
$x_{11}=0$	$x_{12}=1$	$x_{21}=0$	$x_{22}=1$	$x_{31}=0$	$x_{32}=1$
$x_{41}=1$	$x_{42}=0$	Final Energy =10349			

7. SENSITIVITY ANALYSIS

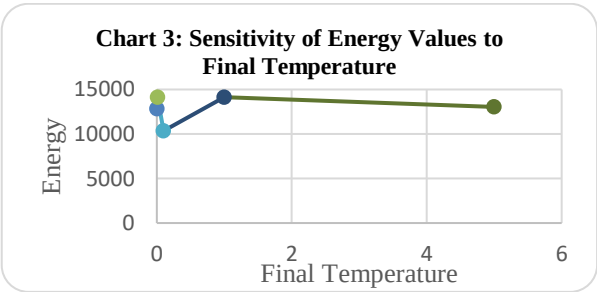
To examine the sensitivity of the solution obtained from this algorithm to T_s, the temperature values are set to 50, *100*, *200*, *300* and *400*. The graph showing the changes in energy is similar to the graph in Chart *1*, and the graph showing the changes in the solution time of the problem is similar to the graph in Chart *2*.

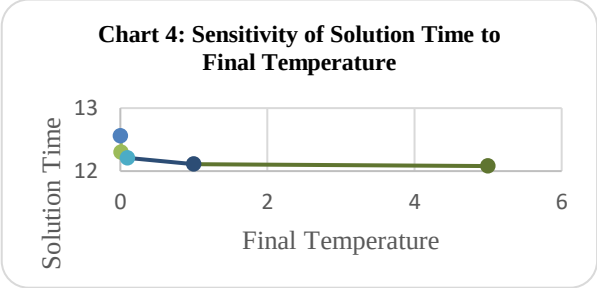




As seen in charts *1* and *2*, with an increase in the value of T_S, the *Z* values either decrease (better) or increase (worse). This means that the change in the T_S parameter has a slight effect on the objective function value. However, changing the parameter T_S significantly affects the solution time, indicating a direct relationship.

Next, to examine the sensitivity of the solution obtained from this algorithm to T_f, the T_f values are sets to *0.001, 0.01, 0.1, 1* and *5*. The graph showing the changes in energy is similar to the graph in chart *3*, and the graph showing the changes in the solution time of the problem is similar to the graph in chart *4*.





As seen in charts 3 and 4, with an increase in the value of T_f, the Z values either decrease (better) or increase (worse). This indicates that changing the parameter T_f has a slight effect on the objective function value. However, changing the parameter T_f significantly affects the solution time of the problem, indicating an inverse relationship.

Finally, to examine the sensitivity of the solution obtained from this algorithm to β, the β values are set to *0.5*, *0.6*, *0.7*, *0.8* and *0.9*. The graph showing the changes in energy is similar to the graph in chart *5*, and the graph showing the changes in the solution time of the problem is similar to the graph in chart *6*.

Chart 5: Sensitivity of Energy Values to β

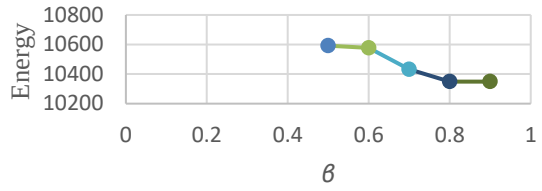
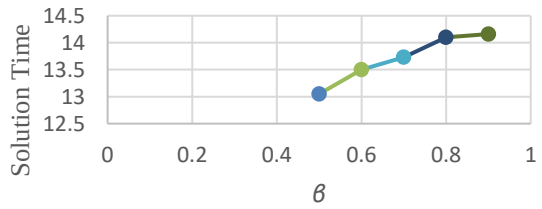


Chart 6: Sensitivity of Solution Time to β



As seen in charts *5* and 6, with an increase in the value of β, the Z values decrease (better), and the solution time of the problem generally increases (worse).
Based on the results, it can be concluded that the solution values are highly sensitive to the β parameter, which has a direct relationship with the objective function value and an inverse relationship with the solution time. These findings indicate that the performance of the SA algorithm is strongly influenced by its initial parameters. Adjustments to these parameters can lead to either improved or suboptimal solutions.

8. CONCLUSION

The project scheduling problem with resource constraints is diverse due to the possibility of considering various constraints and different solution methods. Therefore, it has always attracted the attention of researchers, and numerous studies have been conducted to present various models and solve the models of previous researchers. This study proposed a new mathematical model for the P-MOMRCPSP. In this project scheduling problem, two objectives were to be optimized. The first objective, considered in most existing studies, was to minimize project completion time. However, in this study, another goal, which is of particular importance in today's world, was examined: maximizing the NPV. Objectives that discuss the project's cash flow were very important and had not been considered in many studies. By incorporating various constraints, the problem was formulated to better reflect real-world conditions and enhance its practical applicability.

Additionally, since it is feasible in practice to assign more than one execution mode to activities, the activities of this study are multi-mode. In the above example, two execution modes were assigned to each activity. Besides being multi-mode, the activities of the proposed model are interruptible. Although this restriction has been addressed less frequently in previous studies, it is not uncommon to encounter in practical projects, where, for certain activities, it might be useful to preempt the activity. Preemption of activities introduces problem complexity to the scheduling problem but can result in improved solutions. In most of the existing models, resource constraints are renewable or non-renewable. In real projects, however, both resources are typically available. Thus, the proposed model considers both renewable and non-renewable resources in the development of the resource constraint.

As the model proposed is a hard problem, a metaheuristic algorithm was used to find solutions within a reasonable computation time, especially for large instances. The SA algorithm was used to solve the problem and implemented on the model. One of the primary reasons for choosing this algorithm is that it can find solutions close to the optimal solution.

For future research, the activity durations can be modeled as probabilistic rather than deterministic. Further, while this study optimized two objective functions with equal importance, future models can assign higher importance to one objective function compared to the other using weighting methods. Since the proposed model is entirely novel, it has not yet been solved by any other algorithms. Consequently, future work can investigate using other metaheuristic algorithms and measuring performance on the basis of gained results during the progress of present work.

9. REFERENCES

[1] Aarts, E.H.L., & Korst, J.H.M. 1989. Simulated annealing and boltzmann machines: A stochastic approach to combinatorial optimization and neural computing, Wiley, Chichester.

[2] Abbasi, B., Shadrokh, S., & Arkat, J. 2006. Bi-objective resource constrained project scheduling with robustness and makespan criteria, Applied Mathematics and Computation. 180, 146–152.

[3] Alcaraz, J., Maroto, C., & Ruiz, R. 2003. Solving the multi-mode resource constrained project scheduling problem with genetic algorithms. Journal of the Operational Research Society, 54(6), 614-626.

[4] Al-Fawzan, M., & Haouari, M. 2005. A bi-objective model for robust resource-constrained project scheduling. International Journal of Production Economics, 96(2), 175-187.

[5] Barrios, A., Ballestin, F., & Valls, V. 2011. A double genetic algorithm for the MRCPSP/max. Computers and Operations Research, 38(1), 33-43.

[6] Bianco, L., Caramia, M., & DellOlmo, P. 1999. Solving a preemptive project scheduling problem with coloring techniques. In Weglarz, 193, 135– 146.

[7] Boctor, F. 1996. An adaptation of the simulated annealing for solving resource-constrained project scheduling problems. International Journal of Production Research, 34, 2335–2351.

[8] Bouleimen, K., & 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(2), 268–281.

[9] Bredael, d., & Vanhoucke, M. 2024. A genetic algorithm with resource buffers for the resource-constrained multi-project scheduling problem. European Journal of Operational Research, 315, 19–34.

[10] Brucker, P., & Knust, S. 2001. Resource-constrained project scheduling and timetabling. Lecture Notes in Computer Science, 2079, 277-293.

[11] Calhoun, K. M., Deckro, R. F., Moore, J. T., Chrissis, J. W., & Hove, J. C. V. 2002. Planning and replanning in project and production scheduling. Omega – The international Journal of Management Science, 30(3), 155-170.

[12] Coelho, J., & Vanhoucke, M. 2011. Multi-mode resource-constrained project scheduling using RCPSP and SAT solvers. European Journal of Operational Research, 213(1), 73–82.

[13] Davis, K. R., Stam, A., & Grzybowski, R. A. 1992. Resource constrained project scheduling with multiple objectives: A decision support approach. Computers and Operations Research, 19(7), 657-669.

[14] Debels, D., & Vanhoucke, M. 2006. Pre-emptive resource constrained project scheduling with setup times. Faculteit Economie En Bedrijfskunde, 1-24.

[15] Demeulemeester, E. L., & Herroelen, W. S. 1996. An efficient optimal solution procedure for the preemptive resource-constrained project scheduling problem. European Journal of Operational Research, 90(2), 334-348.

[16] Ding, H., Zhuang, C., & Liu, J. 2023. Extensions of the resource-constrained project scheduling problem. Automation in Construction, 153, Article 104958.

[17] Drexl, A., Nissen, R., Patterson, J. H., & Salewski, F. 2000. Progen/px - an instance generator for resource-constrained project scheduling problems with partially renewable resources and further extensions. European Journal of Operational Research. 125(1), 59-72.

[18] Elloumi, S., & Fortemps, P. 2010. A hybrid rank-based evolutionary algorithm applied to multi-mode resource-constrained project scheduling problem. European Journal of Operational Research, 205(1), 31–41

[19] Elloumi, S., Fortemps, P., & Loukil, T. 2017. Multi-objective algorithms to multi-mode resource-constrained projects under mode change disruption. Computers and Industrial Engineering, 106, 161–173.

[20] Franck, B., Neumann, K., & Schwindt, C. 2001. Project scheduling with calendars. Or Spektrum, 23, 325- 334.

[21] Hartmann, S. 2001. Project scheduling with multiple modes: A genetic algorithm. Annals of Operations Research, 102(1–4), 111–135.

[22] He, Z., Wang, N., Jia, T., & Xu, Y. 2009. Simulated annealing and tabu search for multimode project payment scheduling, European Journal of Operational Research, 198, 688–696.

[23] Heilmann, R. 2001. Resource-constrained project scheduling: a heuristic for the multi-mode case. Or Spektrum, 23(3), 335-357.

[24] Heilmann, R. 2003. A branch-and-bound procedure for the multi-mode resource-constrained project scheduling problem with minimum and maximum time lags. European Journal of Operational Research, 144(2), 348-365.

[25] Jarboui, B., Damak, N., Siarry, P., & Rebai, A. 2008. A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems. Applied Mathematics and Computation, 195(1), 299–308.

[26] Jozefowska, J., Mika, M., Rozycki, R., Waligora, G., & Weglarz, J. 2001. Simulated annealing for multi-mode resource-constrained project scheduling. Annals of Operations Research, 102(1–4), 137–155.

[27] Kolisch, R., & Drexl, A. 1997. Local search for non-preemptive multi-mode resource constrained project scheduling. IIE Transactions, 25(5), 74–81.

[28] Lova, A., Tormos, P., & Barber, F. 2006. Multi-mode resource constrained project scheduling: Scheduling schemes, priority rules and mode selection rules. Inteligencia Artificial Revista Iberoamericana De Inteligencia Artificial, 10(30), 69–86.

[29] Lova, A., Tormos, P., Cervantes, M., & Barber, F. 2009. An efficient hybrid genetic algorithm for scheduling projects with resource constraints and multiple execution modes. International Journal of Production Economics, 117(2), 302–316.

[30] Mika, M., Waligo´ra, G., & Weglarz, 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.

[31] Mori, M., & Tseng, C. C. 1997. A genetic algorithm for multi-mode resource constrained project scheduling problem. European Journal of Operational Research, 100(1), 134–141.

[32] Muritiba, A. E. F., Rodrigues, C. D., & Costa, F. A. D. 2018. A path-relinking algorithm for the multi-mode resource-constrained project scheduling problem. Computers and Operations Research. 92, 145-154.

[33] Nonobe, K., & Ibaraki, T. 2002. Formulation and tabu search algorithm for the resource constrained project scheduling problem. In C. C. Ribeiro and P. Hansen, editors, Essays and Surveys in Metaheuristics, pages 557–588. Kluwer Academic Publishers.

[34] Nudtasomboon, N., & Randhawa, S. U. 1997. Resource constrained project scheduling with renewable and non-renewable resources and time-resource tradeoffs. Computers and Industrial Engineering, 32(1), 227-242.

[35] Ozdamar, L. 1999. A genetic algorithm approach to a general category project scheduling problem. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 29(1), 44-59.

[36] Pesch, E. 1999. Lower bounds in different problem classes of project schedules with resource constraints. In Weglarz , 53–76.

[37] Peteghem, V. V., & Vanhoucke, M. 2010. A genetic algorithm for the preemptive and non-preemptive multi-mode resource-constrained project scheduling problem. European Journal of Operational Research, 201(2), 409–418.

[38] Peteghem, V. V., & Vanhoucke, M. 2011. Using resource scarceness characteristics to solve the multi-mode resource-constrained project scheduling problem. Journal of Heuristics, 17(6), 705–728.

[39] Peteghem, V. V., & Vanhoucke, M. 2014. An experimental investigation of metaheuristics for the multi-mode resource-constrained project scheduling problem on new dataset instances. European Journal of Operational Research, 235(1), 62–72.

[40] Rahimi, A., Karimi, H., & Afshar-Nadjafi, B. 2013. Using meta-heuristics for project scheduling under mode identity constraints, Applied Soft Computing. 13(4), 2124–2135.

[41] Reyck, B., & Herroelen, W. S. 1999. The multi-mode resource-constrained project scheduling problem with generalized precedence relations. European Journal of Operational Research, 119(2), 538-556.

[42] Sabzehparvar, M., & Seyed Hosseini, S. M. 2008. A mathematical model for the multi-mode resource constrained project scheduling problem with mode dependent time lags. Journal of Supercomputing, 44(3), 257-273.

[43] Salewski, F., Schirmer, A., & Drexl, A. 1997. Project scheduling under resource and mode identity constraints: Model, complexity, methods, and application. European Journal of Operational Research, 102(1), 88-110.

[44] Schultmann, F., & Rentz, O. 2001. Environment-oriented project scheduling for the dismantling of buildings. Or Spectrum, 23(1), 51-78.

[45] Słowinski, R. 1981. Multi-objective network scheduling with efficient use of renewable and nonrenewable resources. European Journal of Operational Research. 7(3), 265–273.

[46] Słowinski, R., Soniewicki, B., & Weglarz, J. 1994. DSS for multi-objective project scheduling. European Journal of Operational Research, 79, 220–229.

[47] Tirkolaee, E.B., Goli, A., Hematian, M., Sangaiah, A.K., & Han, T. 2019. Multi-objective multi-mode resource constrained project scheduling problem using Pareto-based algorithms, Computing, 101(6), 547–570.

[48] Varma, V. A., Uzsoy, R., Pekny, J., & Blau, G. 2007. Lagrangian heuristics for scheduling new product development projects in the pharmaceutical industry. Journal of Heuristics, 13(5), 403-433.

[49] Voß, S., & Witt, A. 2007. Hybrid flow shop scheduling as a multi-mode multi-project scheduling problem with batching requirements: A real-world application. International Journal of Production Economics, Vol. 105(2), 445-458.

[50] Voskresenskii, A., Kovalchuk, M., Filatova, A., Nasonov, D., & Lutsenko, A. 2023. Hybrid Algorithm for Multi-Contractor, Multi-Resource Project Scheduling in the Industrial Field. Procedia Computer Science, 229, 28-38.

[51] Wang, L., & Fang, C. 2012. An effective shuffled frog-leaping algorithm for multi-mode resource-constrained project scheduling problem. Information Sciences, 39(5), 890–901.

[52] Wang, L., & Zheng, X. 2018. A knowledge-guided multi-objective fruit fly optimization algorithm for the multi-skill resource constrained project scheduling problem. Swarm and Evolutionary Computation, 38, 54–63.

[53] Zare, Z., Naddaf, A., & Salehi, M.R., 2012. Proposing a Model on Preemptive Multi-Mode Resource-constrained Project Scheduling Problem. International Journal of Business and Social Science, 3(4), 126-130.

[54] Zare, Z., Naddaf, A., Ashrafzadeh, M., & Salehi, M.R. 2012. Preemption in multi-mode Resource-constrained project scheduling problem. Interdisciplinary Journal of Contemporary Research in Business, 3(9), 636-642.

[55] Zhang, H. 2012. Ant colony optimization for multimode resource-constrained project scheduling. Journal of Management in Engineering, 28(2), 150–159.

[56] Zhang, H., Tam, C. M., & Li, H. 2010. Multimode project scheduling based on particle swarm optimization. Computer-Aided Civil and Infrastructure Engineering, 21(2), 93–103.

[57] Zhu, G., Bard, J. F., & Yu, G. 2006. A branch-and-cut procedure for the multimode resource-constrained project-scheduling problem. INFORMS Journal on Computing, 18(3), 377-390.

[58] Zoraghi, N., Shahsavar, A., Abbasi, B., & Peteghem, V. V. 2017. Multi-mode resource-constrained project scheduling problem with material ordering under bonus-penalty policies. TOP: An Official Journal of the Spanish Society of Statistics and Operations Research, 25(1), 1–31.