RAIRO-Oper. Res. 51 (2017) 1189–1209
DOI: 10.1051/ro/2017017

RAIRO Operations Research
www.rairo-ro.org

PROJECT SCHEDULING AND EQUIPMENT PLANNING WITH RANDOM BREAKDOWNS

Abbas Shafikhani[1], Amir Abbas Najafi[2] and Seyed Taghi Akhavan Niaki[3]

Abstract. Most of the research works conducted on Project Scheduling Problem (PSP) especially Resource Constrain Project Scheduling Problem (RCPSP) either ignore equipment planning or schedule the activities first, and then plan for the required equipment. Moreover, little works that consider simultaneous PSP and Equipment Planning (EP) are based on the assumption that the equipment is continuously available. However, in reality, equipment is subject to either random breakdowns or deterministic maintenance programs that make it not being available all the time. In this paper, the PSP and EP problems are simultaneously considered in closer to reality situations in which the equipment is not always available. In order to minimize costs and overcome the associated functional and structural complexities, the problem is first mathematically formulated. Then, a system simulator along with a genetic algorithm is utilized to find a near optimum solution. As there are no benchmarks available in the literature, a simulated annealing algorithm is also employed in combination with the simulator to validate the obtained results. In addition, design of experiments is used to set the parameters of the algorithms such that both the running times and the responses are minimized. Computational results on 400 generated test problems of different sizes indicate good performance of the genetic algorithm with respect to the basic parameters of the selected problem.

Mathematics Subject Classification. 90B36, 90C11, 65K05.

Received December 24, 2014. Accepted March 2, 2017.

1. Introduction

Project Scheduling Problem (PSP) is a concept rich in literature due to its wide variants in different usages, various industrial conditions, and several types of activities, resources, and precedence relations. Besides, because of the complexity involved, researchers are constantly seeking efficient solution procedures to solve the problem.

The problem addressed in this paper is a PSP with simultaneous consideration of equipment planning in which the equipment is subject to random breakdowns, a problem that is classified NP-hard, due to its complexity [7]. To do this, the model introduced by Dodin and Elimam [7] is first extended to include equipment with stochastic breakdowns. This makes them not being available all the time; making the model to be applicable to closer

Keywords. Project scheduling, equipment planning, random breakdown, simulation, genetic algorithm, simulated annealing, design of experiments.

[1] Faculty of Industrial and Mechanical Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran. abbas.shafikhani22@gmail.com

[2] Faculty of Industrial Engineering, K.N. Toosi University of Technology, Tehran, Iran. aanajafi@kntu.ac.ir

[3] Department of Industrial Engineering, Sharif University of Technology, P.O. Box 11155-9414 Azadi Ave., 1458889694 Tehran, Iran. Niaki@Sharif.edu

Article published by EDP Sciences

to reality problems that limit usage of the equipment. In this model, there are several costs including activity crashing, equipment setup, transportation, equipment idle time, operator overtime, and penalty for delayed completion. Besides, project worth and rewards for early completion are taken into account.

The equipment considered in this study is an expensive renewable resource that is available in intermittent periods [3,7,10]. Human resources, machinery, tools, equipment and the like, are examples of renewable resources available in limited and specific amounts at any period, to be used on only one project and renewable for the next period.

In the scheduling literature, two actions are usually taken to process an activity after it is stopped because of scheduled preventive maintenance or random breakdowns. Depending on these actions, the activities are so-called resumable if they resume after equipment maintenance and repairs, or non-resumable if they do not [1,2]. In this paper, activities are assumed resumable without any loss of time and monitory penalty at the point where the maintenance or the breakdown has occurred. Moreover, there are two approaches for dealing with scheduled preventive maintenance and random breakdowns. The first approach considers preventive maintenance and random breakdowns as constraints of the production system for which the scheduling is aimed. In the second approach, preventive maintenance and random breakdowns are taken into account in the model with the aim of finding a feasible schedule [11]. In this paper, the first approach is taken. Note that when an equipment breakdown is taken into account as a random constraint (both in time of occurrence and the required repair time), probability appears and the calculation of the project completion times is not possible using exact methods. In other words, the completion time becomes a random variable having a probability distribution. In this case, the project-scheduling and equipment-planning problem is categorized under dynamic scheduling problem, for which the average completion time can be used as a suitable criterion.

The project-scheduling and equipment-planning problem, PS-EP thereafter, with random breakdown at hand possesses two types of complexities: (a) algorithmic complexity; and (b) functional and structural complexity. This problem, regardless of the random breakdowns, is NP-hard, where by increasing the number of activities, complexity increases significantly [5,8,9,12–14,19]. This justifies the use of a meta-heuristic algorithm to find a near-optimum solution. In addition, due to structural and functional complexities such as random breakdown of the equipment, involved in real-world industries the mathematical formulation of the problem is hard to solve; justifying the use of a simulator to obtain a near-optimal solution. Furthermore, we show how a meta-heuristic algorithm can be integrated with a simulation approach for solving the dynamic scheduling problem.

The remainder of the paper is organized as follows. In Section 2, the problem is defined, the assumptions are explicitly made, the notations are shown, and finally the problem is mathematically formulated. The proposed algorithms of a simulator in combination with the meta-heuristics are discussed in Section 3. Parameter tuning, testing, and comparisons come in Section 4. Finally, we summarize the findings and conclude the paper in Section 5.

2. Mathematical formulation

The purpose of this research is to determine the optimal values of activity durations, activity finish times, and equipment scheduling such that the total cost of the PS-EP problem with random breakdowns is minimized while the constraints are satisfied. Before presenting the mathematical formulation, we first define the assumptions used in the formulation.

2.1. Assumptions

The assumptions involved to the model are as follows:

- The project is defined as a network $G(N, A)$, where N and A represente nodes and arc of the network, respectively. Moreover, the first and the last activities are dummies and numbered 1 and N.
- Preemptions are prohibited.
- The project needs some equipment to perform the activities.

- Equipment is not available at any moment due to its random breakdowns. These breaks occur while the equipment is in its working status, not in its idle times.
- Activities are assumed resumable after a random breakdown occurs in equipment.
- Mean Time Between Failure (*MTBF*) is the average time between two breakdowns of an equipment following an exponential distribution. Therefore, the number of failures follows a Poisson distribution.
- Mean Time To Repair (*MTTR*) is the average time needed to repair an equipment.
- Equipment is assumed to have different *MTBF* and *MTTR* values.
- *FEL* (future events list) denotes a set of variables each of which representing the failure time of the equipment after it commences processing. In other words, the *FEL* contains intervals between every two breakdowns of the equipment. This set is updated after any equipment breakdown.
- *LIFE* denotes a set of variables each of which representing the life time of the equipment. At the time the equipment suffers a breakdown, its *LIFE* variable is set to zero. This is due to the memory less property of the exponential distribution.
- Overtime is added to normal working hours.
- Equipment cannot equip more than one activity at a time.
- If an activity needs more than one equipment, it should have all equipment together at the same time. Therefore, if random breakdown occurs in equipment, the processing of the activity is stopped until the equipment is repaired.
- Activity duration times can be crashed by consuming money.
- Equipment works on an activity and then remains idle while it is paid, until other activities are ready for processing in other periods. Thus, this equipment can either be returned or remained idle until it is needed or some activities that do not need equipment are crashed by consuming time and money to reduce the idle time cost.
- Crashing can be done on both critical and non-critical activities.
- Activities are performed without error.

2.2. Alternatives to solve PSEP

The project requires special rented equipment and work on several project activities. This equipment works on one or a number of activities, and then remains idle (while incurring charges in idle mode) until the other activities are prepared for processing in other periods. As a result, the equipment can either be returned or kept idle until it will be needed again.

There are several alternatives for scheduling the equipment to carry out the work. Figure 1 illustrates these alternatives.

Alternative 1. The equipment initially travels the route from the main warehouse to the site to perform the desired activity. Then, it remains idle until activity K that does not need the equipment is completed. Subsequently, the equipment travels the route from site J to site R. Therefore, the setup process will be completed only once. However, the usage period of the equipment could be low and idle period costs may be substantial. Hence, the total cost includes the cost of idle time, setup cost, and the cost of transportation from site J to site R.

Alternative 2. Initially, the equipment travels the route from the main warehouse to the site and performs the desired activity. Then again, it will be returned to the main site and remains there until activity K is completed. Afterwards, it will go back from the main site to work on activity R. Therefore, the setup process will occur twice, we have no idle time, and have a policy similar to Lot for Lot policy.

Alternative 3. Following the A1 state, while crashing activity K, we will reduce the machine-idle time in order to decrease the project duration. Therefore, it does not matter that the activity is critical or non-critical. Hence, the cost of idle time and the setup cost will be reduced, but the crashing

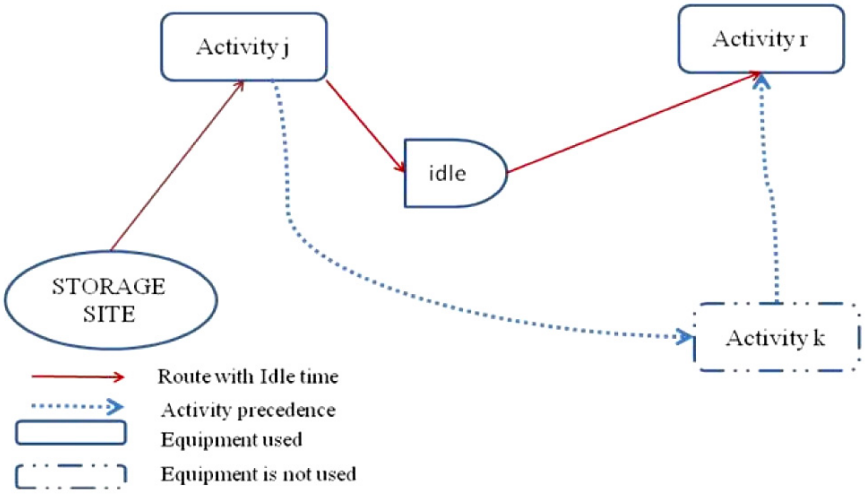

FIGURE 1. Alternatives for scheduling the equipment in a three activities project.

cost will increase. Therefore, we should establish a balance between the costs of crashing, idle time, and setup processing cost [7].

2.3. Introducing the costs associated with PSEP

The costs associated with PSEP are:

(1) Activities' Crashing Cost: this cost is of continuous and linear type during the activity processing (for both critical and non-critical activities).
(2) Holding Cost: this cost will be calculated at the end of each period as a percentage of the project cost.
(3) Rewards and Penalties Cost: the fact that a single equipment cannot do more than one activity at a time will lead to the completion of the project beyond the due date. Thus, we establish and impose rewards and penalties in order to compress and reduce idle time and decrease the number of setup repetition times.
(4) Transition Cost: this is the cost of moving the activity from one activity site to another (like the cost of traveling from one city to another (Travelling Salesman Problem [TSP])).
(5) Setup Cost: this is the cost of preparing and transporting equipment for activities and also returning them to the starting location or the main warehouse.
(6) Idle Time Cost: this is the cost incurred due to keeping the equipment out of work and unused in the site.
(7) Overtime Cost: this cost is associated with the equipment and the operators being used in overtime. Note that the former is calculated on a daily basis and the latter is charged on an hourly basis.

The equipment has two types of motions, one from the main location of the equipment toward the activity location, providing services, and eventually returning to the original location, and the other from one activity to another, in which the equipment may travel from the main warehouse to the site and perform the desired activity. Then again, it returns to the main site and remains there until activity K is completed. Afterward it goes from the main site back to activity R. Therefore, the main site will be considered as a Hub. Then, two virtual activities of N_1 and N_2 will be added to the project activities, where N_1 represents the main site of the equipment and N_2 represents the Hub in the movement of equipment from one activity to another. Thus, the virtual activity of N_2 is the only activity that the equipment can pass more than once [7].

2.4. The model

The indices, parameters, and decision variables of the model are as follows:

Indices

$j = 1, \ldots, N$ Index of project activities

$i = 1, \ldots, M$ Index of equipment used in the project

$t = 0, 1, \ldots, H$ Time index

Parameters

The parameters are grouped into classes related to activity, equipment, and project. The activity-based parameters are:

B_j Set of activities immediately preceding activity j, where B_0 is the set containing activities without predecessor.

b_j Cost of activity j at its crash duration (\$).

c_j Marginal cost of reducing the duration of activity j by one period.

u_j Upper bound on the duration of activity j (normal activity time).

v_j Lower bound on the duration of activity j (crash activity time).

e_j The earliest completion time of activity j (assuming the project starts at time zero and the duration of an activity is equal to its crash time).

l_j The latest completion time of activity j (assuming the project can end at time $H$and the duration of an activity is equal to its crash time).

A_{ij} Set of activities requiring equipment i.

EA_{ij} Set of activities that follow activity j and, along with activity j, require the use of equipment i (This set includes a dummy activity $N2$).

EB_{ij} Set of activities that can precede activity j and, along with activity j, require the use of equipment i (it also includes the dummy activity $N2$).

J_t Set of activities that can be completed in period t.

The equipment-related parameters are:

AE_i Set of activities that require equipment i.

c_i Idle-time cost of equipment i (\$/period, where a period is assumed to be a day).

co_i Overtime cost of crew required to operate equipment i for an extra shift (\$/period).

EE_i Set of activities visited last by equipment i.

g_i Total set-up cost for a round trip of equipment i from the main storage to the project site.

ru_i Utilization rate of equipment i (hour/shift).

SE_i Set of activities based on which equipment $i$can start.

s_{ijk} Change-over/set-up cost of equipment i as it shifts from activity j to activity k without going through the hub (\$).

a_{ij} Amount of overtime needed to operate equipment i to process activity j at its minimum duration (crash duration). This assumes that the work content of the activity is constant.

ci_i Equipment idle time cost.

The project-related parameters are:

d The project due date.

h The absolute due date of the project, and the maximum length of the planning horizon ($h > d$).

p Penalty cost per period to complete the project beyond its due date (\$/period).

r Reward paid per period to complete the project before its due date (\$/period).

s Worth of completed activity representing the holding cost (% per period).

Decision Variables

$$\Delta_{jt} = \begin{cases} 1 & \text{If activity } j \text{ is completed in period } t \\ 0 & \text{Otherwise.} \end{cases}$$

$$Y_{\text{ijk}} = \begin{cases} 1 & \text{If equipment } i \text{ processes activity } k \text{ after activity } j \\ 0 & \text{Otherwise.} \end{cases}$$

IM_{ijk} Idle time of equipment i if it waits to process activity k after processing activity j.
TO_i Total overtime required to operate equipment i.
OT_{ij} Extra labor needed for overtime operation of equipment i on activity j.
W_t Project worth at the end of period t.
W_{jt} Worth of activity j that is completed in period t (\$) (It is the cost of activity j that uses renewable and non-renewable resources).
X_j Completion time of activity j.
Z_j Duration of activity j.

The PSEP problem is formulated as a Mixed Integer Programming (MIP) model as follows:

$$\begin{aligned} Minimize \quad & \left[\sum_{j=1}^{N}\left[b_j - c_j\left(Z_j - v_j\right)\right] + \sum_{i=1}^{M} g_i \left(\sum_{j\in SE_i} Y_{ijN1} + \sum_{j\in AE_i} Y_{ijN2}\right)\right] \\ & + \sum_{i=1}^{M}\sum_{j\in AE_i}\sum_{k\in A_{ij}} \left(s_{\text{ijk}}\right) Y_{\text{ijk}} + \sum_{i=1}^{M}\sum_{j\in AE_i}\sum_{k\in A_{ij}} \left(ci_i\right) IM_{\text{ijk}} + \sum_{i=1}^{M}\left(co_i\right) TO_i \\ & + \sum_{t=1}^{h-1} sW_t - \sum_{t=e_n}^{d-1} r\left(d-t\right)\Delta_{Nt} + \sum_{t=d+1}^{l_n} p\left(t-d\right)\Delta_{Nt} \end{aligned}$$

Subject to:

$$X_j - X_k \geqslant Z_j; \quad \forall \quad k \in B_j \tag{2.1}$$

$$X_j \geqslant Z_j; \quad \forall\, j \in B_0 \tag{2.2}$$

$$Z_j \leqslant u_j; \quad \text{for} \quad j = 1, \ldots, N \tag{2.3}$$

$$Z_j \geqslant v_j; \quad \text{for} \quad j = 1, \ldots, N \tag{2.4}$$

$$X_j \leqslant l_j; \quad \text{for} \quad j = 1, \ldots, N \tag{2.5}$$

$$X_j \geqslant e_j; \quad \text{for} \quad j = 1, \ldots, N \tag{2.6}$$

$$\sum_{t=e_j}^{l_j} \Delta_{jt} = 1; \quad \text{for } j = 1, \ldots, N \tag{2.7}$$

$$\sum_{t=e_j}^{l_j} t\left(\Delta_{jt}\right) \geqslant X_j; \quad \text{for} \quad j = 1, \ldots, N \tag{2.8}$$

$$\sum_{j\in SE_i} Y_{ijN1} = 1; \quad \text{for} \quad j = 1, \ldots, N \tag{2.9}$$

$$\sum_{j\in EE_i} Y_{ijN1} = 1; \quad \text{for } j = 1, \ldots, N \tag{2.10}$$

$$\sum_{k\in EB_{ij}} Y_{\text{ijk}} - \sum_{k\in EA_{ij}} Y_{\text{ijk}} = 0; \quad j \in AE_i \cup N_2 \tag{2.11}$$

$$\sum_{k\in EA_j} Y_{\text{ijk}} = 1; \quad \text{for} \quad i = 1, \ldots, M \quad \text{and} \quad j \in AE_i \tag{2.12}$$

$$\sum_{k\in EB_{ik}} Y_{\mathrm{ijk}} = 1; \quad \text{for } i = 1,\ldots,M \quad \text{and } k \in AE_i \tag{2.13}$$

$$IM_{\mathrm{ijk}} \geqslant X_k - X_j - Z_k - h\,(1 - Y_{\mathrm{ijk}}) \tag{2.14}$$

$$IM_{\mathrm{ijk}} \geqslant 0; \quad \text{for } i = 1,2,\ldots M, \quad j \in AE_i, \quad k \in A_{ij} \tag{2.15}$$

$$OT_{ij} \geqslant a_{ij} - ru_i\,(Z_j - v_j)\,; \quad \text{for } i = 1,2,\ldots M, \quad j \in AE_i \tag{2.16}$$

$$OT_{ij} \geqslant 0 \tag{2.17}$$

$$TO_i = \sum_{j\in AE_i} OT_{ij} \tag{2.18}$$

$$W_{jt} \geqslant (b_j - c_j\,(Z_j - v_j)) - \left(\sum_{j=1}^{N} b_j\right)(1 - \Delta_{jt})\,; \quad \text{for } j = 1,2,\ldots N \quad \& \; t \in [e_j, l_j] \tag{2.19}$$

$$W_t \geqslant W_{t-1} + \sum_{j\in J_t} W_{jt}; \quad \text{for } t = 1,2,\ldots,e_N \tag{2.20}$$

$$W_t \geqslant W_{t-1} + \sum_{j\in J_t} W_{jt} - \left(\sum_{j=1}^{N} b_j\right)\Delta_{Nt}; \quad \text{for } t = e_N + 1,\ldots,h \tag{2.21}$$

$$W_t \geqslant 0; \quad \text{for } t = 1,2,\ldots,h \tag{2.22}$$

$$W_0 = 0. \tag{2.23}$$

The total cost in the objective function consists of eight terms. These terms are (1) the activity crashing cost. This is the cost of reducing the activity duration from its normal time. It is assumed to be linear and continuous over the duration of the activity. While such cost is normally required to crash critical activities, it shows that crashing non-critical activities cannot be of merit. (2) The set-up cost for shipping equipment to and from the main storage. (3) The cost for equipment transition from one activity to another. The transition cost might affect the equipment sequence between various activities requiring its services and their durations. This cost resembles the cost of moving from one city to another for the well-known Traveling Salesman Problem (TSP). (4) The equipment idle time. The cost incurred due to keeping the equipment unused on project site, rather than returning it to the main storage. Typically, the equipment stays idle on the project site after completing an activity and waiting until another activity to use it. (5) The crew cost for operating equipment during overtime. It is assumed that the equipment cost is charged on a daily basis, while the operators' cost is incurred per shift. Moreover, operators' overtime is incurred on hourly basis. In order to reduce the equipment setup and idle time costs, the crew might operate the equipment more than 8 h per day leading to higher operators overtime cost and lower equipment idle and setup costs. (6) The project worth. This is the cost of the project worth. It is accumulated as the project goes on. This is usually computed as a percentage of the project value at the end of each period. This cost is similar to the material inventory holding cost. However, in our case it is the cost of holding the completed work of the project before the project is ready for final delivery. (7) The rewards for early completion, and (8) the penalty for delayed completion. It is a well-known fact that resource scheduling can lead into project delays. In order to counteract the impact of equipment sequencing on the project schedule and its ability to meet the project due date, a penalty is introduced for delays beyond such due dates. Conversely, one might be willing to complete the project ahead of its due date in order to be able to claim an offered reward. A penalty or a reward is typically computed in monetary amount per period of early or late completion, respectively. Constraints (2.1) and (2.2) take into consideration the precedence relation between each pair of activities (i, j), where i immediately precedes j. Constraints (2.3) and (2.4) limit activity durations between their normal and crash time. Constraints (2.5) and (2.6) set bounds on the activity completion times. Constraints (2.7) and (2.8) indicate activity completion time and guarantee that each activity must be completed within the interval $[e_j, l_j]$. Constraints (2.9)–(2.13) are of a flow-conservation type that limit transitions of the equipment because each equipment initially starts working on the project, then, it must eventually leave the

project, and finally, if it works on an activity must leave it to another activity requiring its service or go to the hub, represented by dummy activity $N2$. The rest of the constraints indicate the limitations associated with equipment idle time, operator overtime cost, and project worth [7].

Taking into account that expressing limitation of industrial problems such as equipment random failure is difficult to model due to its structural and functional complexities, a hybrid simulation and meta-heuristic algorithm is proposed in the next section to find a near-optimum schedule.

3. The proposed hybrid algorithm

The proposed hybrid algorithm is a combination of a simulator and a meta-heuristic algorithm. The simulator is used to model random failures of the equipment. When a failure occurs in equipment, not only is the duration of the repair time needed, but also its lifetime, which is the time the equipment will work after the repair until its next failure, should be calculated. Moreover, as equipment failures occur only during the processing time of activities, the equipment idle and set up time do not contribute to its lifetime, and hence, only the activity processing times between two failures are considered. Furthermore, the repair time of equipment is added to the activity completion time. We must note that since the simulation output is random due to its stochastic input, independent replications in which all parameters remain constant are required to estimate the mean completion time of the network activities. Moreover, *FEL* and *LIFE*, which are obtained based on independent exponential random generations, are used to simulate the system. Note that after the equipment suffers a breakdown and is repaired, its lifetime is considered equal to when it had started processing the activities (due to the memory-less property of the exponential distribution.) In the next subsection, a genetic algorithm is developed to accompany the simulator in finding a near optimum schedule.

3.1. Genetic algorithm

While the general characteristics of GA follow, we refer the reader to the numerical illustration given in Section 3.6 for more details:

- A chromosome is coded as a $2 \times N$matrix. The first row contains durations and the second row shows the completion times of the activities.
- In the first generation, a uniformly distributed random duration between the normal and crashing times is initially generated for each activity. Then, the duration is randomly selected for each activity between its low and high value chromosomes as follows:
  - The low value chromosome is a 2-row matrix. The first row is the crash duration v_j and the second row is the early completion time e_j. The early completion time, e_j is determined using v_j starting at time zero.
  - The high value chromosome is also a 2-row matrix. The first row is the normal duration u_j and the second row is the late completion time l_j. The late completion time, l_j, is determined using u_j ending the schedule at the deadline.
- A specific percentage is used for elitism.
- A specific percentage is used for a one-point crossover operation.
- A specific percentage is used for a random multi-point mutation operation. The maximum number of mutation points comes from the number of activities divided by five. For instance, if the project consists of 10 activities, then a maximum of two random activities are selected in a chromosome to be mutated.

After the initial population generation, the next populations are randomly created using the genetic operators, where ranking chromosomes using their fitness is employed. The pseudo code of this algorithm follows.

3.1.1. The pseudo code of GA

1. Read inputs including problem data and genetic parameters;
2. Let the current schedule be set based on ECT (early completion time);

3. Do from the first to the last generation;
   3.1. Save the start solving time;
   3.2. If first loop, then create initial population, otherwise create population using genetic parameters;
   3.3. Do for all chromosomes with fitness less than the number of activities (N).
      3.3.1. Evaluate chromosome using the "eval_psep.m" function described in Section 3.3.
      3.3.2. Run the failure simulation function, "failure_sim_psep.m," for each chromosome and obtain the failure chromosome and the relative objective value.
      3.3.3. Make sure the failure chromosome does not exceed the deadline. If the completion time of the chromosome is greater than the deadline, let the fitness be equal to zero. Otherwise, let the fitness be equal to the number of activities.
      3.3.4. End of Loop;
   3.4. Calculate chromosome ranks of this generation: For each chromosome divide its objective function by its fitness. Since the fitness is either zero or N, the objective function for chromosomes with no fitness is infinite. Hence, chromosome with fitness will be sorted increasingly.
   3.5. Save the best chromosome with its solving time;
   3.6. End of loop;
4. Calculate the best schedule using the best chromosome and its related failure chromosome.
5. Finish.

As there are no benchmarks available in the literature, in the next subsection a simulated annealing algorithm is developed to validate the results obtained.

3.2. The simulated annealing algorithm

Similar to GA, Simulated Annealing (SA) is a search algorithm with the advantage of not being trapped in the local minima. In this algorithm, the initial population is generated similar to the GA and the pseudo code follows:

1. Input data.
   1.1. N_x: Number of initial solutions (it is equal to one).
   1.2. $T_{0\max}$: The initial maximum temperature.
   1.3. α_{SA}: The cooling parameter.
   1.4. *Chain*: Number of Markov chains.
   1.5. *Step*: Number of steps in each chain.
   1.6. N_0: Number of neighbors in the first step.
2. Save the initial solution as the current and the best solution.
3. Do from the first to the last chain.
   3.1. $T = T_{0\max}$, where T is the current temperature.
   3.2. $N_s = N_0$, where N_s is the number of neighbors in the current step.
   3.3. Do from the first to the last step.
      3.3.1. Do from 1 to N_s
         3.3.1.1. Create a neighbor with the following innovative method.
         3.3.1.2. Choose some activities randomly so that the number of selected activities decreases while the number of neighbors increases.
         3.3.1.3. Select a portion from these activities.
         3.3.1.4. Change the durations and finish times of the activities in random.
         3.3.1.5. Evaluate the result using the "eval_psep.m" function.
         3.3.1.6. Run failure_sim_psep.m for each solution and obtain the failure solution and the relative objective value.
         3.3.1.7. Make sure the failure solution does not exceed the deadline.

3.3.1.8. If the completion time of the solution is greater than the deadline, let the objective value be infinite (to not to be chosen for the probability algorithm of selecting a neighbor).
3.3.1.9. Run the probability algorithm to select the neighbor
3.3.1.9.1. Calculate Δ: the difference between the objective values of the current and the neighbor solution.
3.3.1.9.2. If $\Delta < 0$ (the neighbor is better)
3.3.1.9.2.1. Let the current solution be the neighbor solution.
3.3.1.9.2.2. If neighbor is better than the best solution, let the best solution be the neighbor solution.
3.3.1.9.3. If $\Delta > 0$ (The neighbor is worse).
3.3.1.9.4. Calculate P: the probability of selecting neighbor using $P = e^{-\Delta/T}$
3.3.1.9.5. If $P < Y$(Y is a uniform random number between 0 and 1)
3.3.1.9.5.1. Select the worse neighbor as the current solution.
3.3.2. End of loop neighbors N_s
3.3.3. Decrease the temperature using $T = T\,(\alpha_{SA})$
3.4. End of loop Steps.
3.5. Save the solution time for the current Markov chain.
3.6. Find the best solution in the current chain.
4. End of loop of the Markov chain.
5. Return to the best solution.
6. End of the SA algorithm.

In the next three subsections, the two common functions used in the GA and SA algorithms and the simulator responsible for modelling equipment failures are described.

3.3. The "eval_psep.m" function

This function initially evaluates a two-row schedule, selects the activity with the lowest start time (the most important activity has the lowest start time), and matches it with the problem conditions. Then, if at any stage the schedule is not matched, this function will change the schedule and provide a feasible schedule as output. The input of this function includes the schedules that have not been evaluated yet, the number of activities, the precedence relations, the earliest start times, and the equipment. The output is the evaluated feasible schedule.

3.4. The "failure_psep.m" function

Before describing this function, it is essential to notice some points.

- After a breakdown occurs, it is required to calculate the repair time, *i.e.* the duration of the repair.
- The lifetime of the equipment, the time which has been consumed on the equipment from the last breakdown event, is necessary to be saved.
- As a breakdown occurs while an activity is being processed, the lifetime of the equipment at a particular time is defined as the total processing times of the activities which the equipment has processed from the last breakdown event up to that time. In addition, neither idle times nor setup times are taken into account for the lifetimes of the equipment.
- We add the repair time to the completion time of the considered activity j.
- On account of the fact that the problem has probabilistic nature, it is essential to replicate the computation of simulator several times (no.simu) for each sequence when all the features of the problem remain constant.
- We calculate the mean of the project duration obtained in the previous step as the fitness value for each sequence used in the genetic algorithm.

This function initially evaluates working equipment and adds the repair time to the duration of the activities being processed in the case of a failure. Then, it returns the total durations. The inputs of this function are

MTBF, *MTTR*, *Life*, *FEL*, and active equipment. The output is the activity duration time, the updated *FEL*, and *Life* of equipment.

3.5. Pseudo code of the simulator

The failure simulator is implemented to account for stochastic failures. This algorithm first simulates the equipment failure on one chromosome with the number of replications equal to *no.rep*. Then, it returns the related failure chromosome and the value of the objective function as the output. The steps involved in the simulator follow

1. Save the first row of the chromosome (durations).
2. Set the finish times and the value of the objective function to zero.
3. Do the following for *no.rep* times.
   3.1. Initialize the *FEL* for all equipment; *FEL*(equipment) = exp-rand (*MTBF*).
   3.2. Set the lifetime equal to zero for all equipment.
   3.3. Do following for activity $j = 1$ to N.
      3.3.1. Find equipment required to process activity j.
      3.3.2. Run the "failure_psep.m" function and find the duration of activity j after failure implementation.
   3.3. Finish.
   3.4. Create failure chromosome based on the duration of all activities after failure implementation.
   3.5. Evaluate failure chromosome using the "eval_psep.m" function.
   3.6. Calculate the objective function value for the failure chromosome.
   3.7. End of simulation.
4. Calculate the average value of the objective functions.
5. Find the nearest chromosome to the average.
6. Return the failure chromosome and its objective function value.

In the next section, an illustrative example is given to demonstrate the application of the proposed methodology.

3.6. An illustrative example

Consider a project consisting of 10 activities and three equipment with initial data given in Table 1.

Table 1. Data for the illustrative example.

Normal duration	Crash duration	Preceding activities	Activity j
3	1	–	1
3	1	1	2
4	2	1	3
3	1	2	4
3	2	2	5
3	1	3	6
3	2	3	7
3	1	4,6	8
3	2	5,7	9
2	1	8,9	10

Activities requiring equipment	Equipment i
1, 5, 7, 9	1
5, 6, 7, 9, 10	2
1, 7, 8, 9	3

Based on the data given in Table 1, the first row of the chromosome matrix indicates the activity durations and the second row shows their completion times as:

$$\text{Chromosome} = \begin{matrix} 1 & 1 & 3 & 2 & 3 & 2 & 2 & 2 & 3 & 2 \\ 8 & 4 & 11 & 13 & 8 & 4 & 9 & 13 & 11 & 9 \end{matrix}$$

Then, the prior start time of the activities becomes

$$\text{prior\_start} = 7 \;\; 3 \;\; 8 \;\; 11 \;\; 5 \;\; 2 \;\; 7 \;\; 11 \;\; 8 \;\; 7$$

First iteration:

Activity 1 with the lowest time is selected from the list of activities that can be started. The start time vector of activity is updated to:

$$\text{Start} = \mathbf{0} \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0$$

Read the required equipments of the selected Activity 1. Since the assigned equipments 1 and 3 are dedicated to this activity, set the release time of all other activities that need these equipment and still are not selected (activities 5, 7, 8, and 9), equal to the finish time (2.1) of the selected Activity 1. The activities 5, 7, 8, and 9 should wait until processing of Activity 1 is finished. Then, the release time vector becomes

$$\text{Release} = 0 \;\; 0 \;\; 0 \;\; 0 \;\; \mathbf{1} \;\; 0 \;\; \mathbf{1} \;\; \mathbf{1} \;\; \mathbf{1} \;\; 0$$

Second iteration:

Find feasible activities based on the precedence relations. Between activities 2 and 3, which can be processed after Activity 1, the activity with the lowest time in the prior_start vector is chosen (Activity 2), and the start and release time vectors become:

$$\begin{aligned} \text{Start} &= 0 \;\; \mathbf{1} \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \\ \text{Release} &= 0 \;\; 0 \;\; 0 \;\; 0 \;\; 1 \;\; 0 \;\; 1 \;\; 1 \;\; 1 \;\; 0 \end{aligned}$$

Third iteration:

Again, according to precedence relations, one activity among activities 3, 4, and 5 is selected (Activity 5). Then, the start and release time vectors become:

$$\begin{aligned} \text{Start} &= 0 \;\; 1 \;\; 0 \;\; 0 \;\; \mathbf{2} \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \\ \text{Release} &= 0 \;\; 0 \;\; 0 \;\; 0 \;\; 1 \;\; \mathbf{5} \;\; \mathbf{5} \;\; 1 \;\; \mathbf{5} \;\; \mathbf{5} \end{aligned}$$

Thus, the resulting feasible schedule is

$$\text{Finish time} = 1 \;\; 2 \;\; 4 \;\; 4 \;\; 5 \;\; 7 \;\; 9 \;\; 14 \;\; 12 \;\; 16$$

To implement the equipment failure, the simulator is used here to find both the failure chromosome and its objective function value. The procedure repeats until a near optimum schedule is found.

4. Parameter calibration, testing, and comparison

Several algorithms with different scenarios that have been developed to solve various problems forced researchers to create a group of standard problems having various scenarios to provide comparisons of the offered algorithms in solving them. Davis and Patterson [4] were the first who created 83 standard test problems for PSP. Then, other researchers including [20–22] added more problems adding up to 1100 test problems. In addition, Kolisch *et al.* [15] introduced a software package called Progen to generate standard scheduling problems with resources. A standard instance set was also built as PSPLIB and is used today for testing the algorithms. However, since the PSP-EP with random failure has initially been introduced in this study, there is no ready instance library available to test the proposed methodology. Therefore, Rangen2 [6, 24] is employed based on the control parameters given in Table 2 to produce test problems.

TABLE 2. Parameter levels of the PS-EP problem.

Control parameter	Value			
Number of activities	7	10	20	30
Number of equipment types	0;1;3	0;1;3	0;1;3	0;1;3
Number of activities requiring equipment	0;1;7	0;1;7	0;1;12	0;1;12
Activity crash time	[1,10]	[1,10]	[1,10]	[1,10]
Activity normal time	v_j+[1,3]	v_j+[1,3]	v_j+[1,3]	v_j+[1,3]
Reward paid	[0,50]	[0,50]	[0,50]	[0,50]
Penalty cost	[0,50]	[0,50]	[0,50]	[0,50]
Worth of completed activity	[0.01,0.05]	[0.01,0.05]	[0.01,0.05]	[0.01,0.05]
Set-up cost	[1500,3000]	[1500,3000]	[1500,3000]	[1500,3000]
Idle time cost	[300,600]	[300,600]	[300,600]	[300,600]
Operator overtime	[60,100]	[60,100]	[60,100]	[60,100]
Reduction cost ($/pd.)	[200,600]	[200,600]	[200,600]	[200,600]
Crash cost ($)	[2000,5000]	[2000,5000]	[2000,5000]	[2000,5000]
S_{ijk}	[1000,2500]	[1000,2500]	[1000,2500]	[1000,2500]
d	Randomly selected with factor 1.5, 1.75, 2.00, 2.25, 2.5			
H	Randomly selected with factor 1.5, 1.75, 2.00, 2.25, 2.5			

TABLE 3. Search range and the levels of GA parameters.

Parameter	range	Low (–1)	Medium (0)	High (+1)
population size (x_1)	$n-3n$	n	$2n$	$3n$
crossover probability (x_2)	0.6–1	0.6	0.8	1
mutation probability (x_3)	0–0.3	0	0.15	0.3
max iteration (x_4)	50–150	50	100	150
no.rep ($x_{5)}$	20–40	20	30	40

TABLE 4. Search range and the levels of SA parameters.

Parameter	range	Low (–1)	Medium (0)	High (+1)
Temp. decrease (z_1)	0.8–0.99	0.8	0.895	1
Max init. temp (z_2)	40–80	40	60	80
Max iterations (z_3)	20–100	20	60	100
no.rep (z_4)	20–40	20	30	40

4.1. Tuning the parameters

For genetic algorithm, the population size, crossover probability, mutation probability, maximum number of iterations, and number of simulation replications *(no.simulator)*, denoted by x_1, x_2, x_3, x_4, x_5, respectively, are the parameters needing calibration. The parameters of the SA to be tuned are the rate of temperature reduction, maximum initial temperature, maximum number of iterations, and *no.simulator*, denoted by x_1, x_2, x_3, x_4, respectively. These parameters are calibrated by utilizing an experimental design based on the generated problem instances. The goal is to find optimum levels of the significant parameters that optimize two responses; (1) the objective function value of a near-optimum schedule as an accuracy measure, and (2) the CPU time required to find the solution. Tables 3 and 4, determine the search range and the levels of the parameters of GA and SA, respectively. Here, each parameter is coded as –1, 0, and 1 for low, medium, and high levels, respectively.

To analyze the results statistically, 400 problems were selected with 7, 10, 20, and 30 activities along with 1 to 3 equipments for testing. A fractional factorial design with four central points, called the central composite design (CCD), is used for experiments. Then the developed GA and SA algorithms run these test problems based on the design. The result of the analysis of variance indicates that there is a significant curvature for both

responses (accuracy performance Y_1 and CPU time performance Y_2) in GA, for which second order models are required to be estimated. However, only the CPU time response (Z_2) has a significant curvature and hence a second order model is needed.

The CCD is the most popular and efficient class of designs used for fitting a second order model. Generally, CCD consists of a 2^k factorial (or fractional factorial of resolution V) with $n_F$factorial runs, $2k$ axial points, and n_c center runs. There are two parameters in the CCD that must be specified; the distance α of the axial points from the design center and the number of center runs. The choice of α depends on the rotatability of the design; that is the variance of the predicted response is constant on spheres. Since the purpose of RSM is optimization and the location of the optimum is unknown prior to running the experiments, it makes sense to use a design that provides equal precision of estimation in all directions. For a spherical region of interest, the best choice of α from a prediction variance viewpoint for the CCD is to set $\alpha = \sqrt{k}$ [16]. When the region of interest is a sphere, the design must include center runs to provide reasonably stable variance of the predicted response. Generally, 3 to 5 center runs are recommended [16]. However, since the region of interest in this study is cuboidal, a useful variation of the CCD is the central composite face-centered (CCF) design in which $\alpha = 1$. The CCF does not require as many center points as the spherical CCD does. In practice, while n_c is 2 or 3, it is sufficient to provide good variance of prediction throughout the experimental region. Sometimes more center runs are employed to give a reasonable estimation of the experimental error [17]. In this research, a 2^{5-2} fractional factorial CCF design with 4 central runs and 10 axial points ($\pm$1,0,0,0,0), (0, $\pm$1,0,0,0), ...,(0,0,0,0,$\pm$1), is selected to estimate the second order models for GA algorithm. Besides, a 2^{4-1} fractional factorial CCF design with 4 central runs and 8 axial points, ($\pm$1,0,0,0), (0,$\pm$1,0,0), ...,(0,0,0,$\pm$1), is selected to estimate the second order model for SA algorithm.

Tables 5 and 6 show the levels of the parameters and the responses for GA and SA, respectively. The results in these tables are used to estimate the quadratic model of each response. Equations (4.1) and (4.2) are related to the responses of the genetic algorithm and equations (4.3) and (4.4) correspond to the responses of SA.

$$\begin{aligned} Y_1 = {} & (0.957734 - 0.009571_* x_1 - 0.015029_* x_2 - 0.001041_* x_3 - 0.008210_* x_4 + 0.002736_* x_5 + 0.001633_* x_1^2 \\ & + 0.007483_* x_2^2 - 0.000511_* x_3^2 + 0.006121_* x_4^2 - 0.004848_* x_5^2 + 0.006214_* x_{1*} x_2 + 0.001450_* x_{1*} x_3 \\ & + 0.003895_* x_{1*} x_4 - 0.001238_* x_{1*} x_5 - 0.003403_* x_{2*} x_3 + 0.003339_* x_{2*} x_4 + 0.004828_* x_{2*} x_5) \end{aligned} \tag{4.1}$$

$$\begin{aligned} Y_2 = {} & (0.358603 + 0.205341_* x_1 + 0.035704_* x_2 - 0.015180_* x_3 + 0.201672_* x_4 + 0.040695_* x_5 - 0.004364_* x_1^2 \\ & - 0.042739_* x_2^2 + 0.026942_* x_3^2 + 0.043027_* x_4^2 + 0.000570_* x_5^2 + 0.022838_* x_{1*} x_2 - 0.052839_* x_{1*} x_3 \\ & + 0.090150_* x_{1*} x_4 + 0.020177_* x_{1*} x_5 - 0.013520_* x_{2*} x_3 - 0.009460_* x_{2*} x_4 + 0.005054_* x_{2*} x_5); \end{aligned} \tag{4.2}$$

$$\begin{aligned} Z_1 = {} & (0.981222 - 0.000193_* z_1 - 0.000827_* z_2 - 0.002738_* z_3 - 0.000848_* z_4 - 0.001789_* z_1^2 + 0.001275_* z_{1*} z_3 \\ & - 0.000300_* z_{1*} z_4) \end{aligned} \tag{4.3}$$

$$\begin{aligned} Z_2 = {} & (0.488047 + 0.002503_* z_1 - 0.000249_* z_2 + 0.352963_* z_3 + 0.051416_* z_4 + 0.011136_* z_1^2 + 0.015041_* z_2^2 \\ & + 0.005301_* z_3^2 + 0.022077_* z_4^2 + 0.034735_* z_{1*} z_2 + 0.001084_* z_{1*} z_3 + 0.002355_* z_{1*} z_4). \end{aligned} \tag{4.4}$$

Tables 7–10 comprehend the results of analysis of variance for the normalized fitness and normalized CPU time of GA and SA. Note that although there is no curvature in the fitness response surface of SA, a quadratic model is used.

Since the goal is to find the GA parameter values such that both the solution accuracy and the corresponding solution time of the algorithm are simultaneously optimized, a bi-objective decision-making problem with conflicting objectives is needed to be solved. Goal planning is one of the powerful methods for solving multi-objective decision-making problems that attempts to minimize deviation from a goal. To accomplish this, the goal and the constraints must be presented accurately. The fuzzy set theory can be used in planning the goal programming, and a group weight ideal planning model is offered with the degree of importance appropriate to the goals [18, 23]. To solve this problem, the payoff tables should first be created (Tabs. 11 and 12).

TABLE 5. Parameter levels and the responses of the GA.

Runs	Input variables					Response variables	
						Fitness	CPU Time
	x_1	x_2	x_3	x_4	x_5	Y_1	Y_2
1	0	0	0	0	0	0.9558	0.3366
2	–1	–1	1	1	–1	0.9844	0.3174
3	–1	1	1	–1	–1	0.9497	0.0830
4	1	–1	–1	–1	–1	0.9737	0.2232
5	0	0	0	0	0	0.9534	0.3338
6	–1	1	–1	–1	1	0.9791	0.0858
7	1	1	1	1	1	0.9513	0.9096
8	–1	–1	–1	1	1	0.9809	0.2460
9	0	0	0	0	0	0.9589	0.3361
10	0	0	0	0	0	0.9529	0.3290
11	1	1	–1	1	–1	0.9446	0.9408
12	1	–1	1	–1	1	0.9747	0.2258
13	1	0	0	0	0	0.9510	0.5719
14	–1	0	0	0	0	0.9702	0.1613
15	0	1	0	0	0	0.9514	0.3639
16	0	–1	0	0	0	0.9815	0.2925
17	0	0	1	0	0	0.9574	0.3827
18	0	0	–1	0	0	0.9595	0.4131
19	0	0	0	1	0	0.9569	0.6157
20	0	0	0	–1	0	0.9733	0.2123
21	0	0	0	0	1	0.9569	0.4122
22	0	0	0	0	–1	0.9514	0.3308

TABLE 6. Parameter levels and the responses of the SA.

Runs	Input variables				Response variables	
					Fitness	CPU Time
	z_1	z_2	z_3	z_4	z_1	z_2
1	1	–1	1	–1	0.9560	0.8144
2	–1	1	1	–1	0.9516	0.8148
3	–1	–1	1	1	0.9491	0.9822
4	–1	–1	–1	–1	0.9577	0.1635
5	–1	1	–1	1	0.9584	0.1989
6	1	1	1	1	0.9495	0.9982
7	1	–1	–1	1	0.9567	0.2036
8	0	0	0	0	0.9506	0.4936
9	0	0	0	0	0.9564	0.4929
10	1	1	–1	–1	0.9539	0.1657
11	0	0	0	0	0.9542	0.4939
12	0	0	0	0	0.9576	0.4971
13	1	0	0	0	0.9501	0.4962
14	–1	0	0	0	0.9498	0.4937
15	0	1	0	0	0.9481	0.4907
16	0	–1	0	0	0.9756	0.5071
17	0	0	1	0	0.9469	0.8150
18	0	0	–1	0	0.9558	0.1633
19	0	0	0	1	0.9478	0.5508
20	0	0	0	–1	0.9555	0.4610

TABLE 7. Analysis of variance for normalized fitness of GA.

Source	DF	Seq SS	Adj SS	Adj MS	F	P-value
Regression	17	0.003159	0.003159	0.000186	11.70	0.014
Linear	5	0.002193	0.000787	0.000157	9.91	0.023
Square	5	0.000577	0.000577	0.000115	7.26	0.039
Interaction	7	0.000389	0.000389	0.000056	3.50	0.122
Residual Error	4	0.000064	0.000064	0.000016		
Lack-of-Fit	1	0.000041	0.000041	0.000041	5.46	0.102
Pure Error	3	0.000023	0.000023	0.000008		
Total	21	0.003222				
S = 0.003986	R-Sq = 98.0%		R-Sq(adj) = 89.6%			

TABLE 8. Analysis of variance for normalized CPU time of GA.

Source	DF	Seq SS	Adj SS	Adj MS	F	P-value
Regression	17	1.02270	1.022695	0.060159	58.98	0.001
Linear	5 0.99119	0.171996	0.034399	33.72	0.002	
Square	5 0.01074	0.010744	0.002149	2.11	0.245	
Interaction	7	0.02077	0.020766	0.002967	2.91	0.159
Residual Error	4	0.00408	0.004080	0.001020		
Lack-of-Fit	1	0.00404	0.004045	0.004045	341.11	0.000
Pure Error	3	0.00004	0.000036	0.000012		
Total	21	1.02678				
S = 0.03194	R-Sq = 99.6%		R-Sq(adj) = 97.9%			

TABLE 9. Analysis of variance for normalized fitness of SA.

Source	DF	Seq SS	Adj SS	Adj MS	F	P-value
Main Effects	4	0.00007150	0.00007150	0.00001787	1.88	0.315
2-Way Interactions	3	0.00003932	0.00003932	0.00001311	1.38	0.399
Curvature	1	0.00000041	0.00000041	0.00000041	0.04	0.848
Residual Error	3	0.00002850	0.00002850	0.00000950		
Pure Error	3	0.00002850	0.00002850	0.00000950		
Total	11	0.00013973				
S = 0.00308227	R-Sq = 79.60%		R-Sq(adj) = 25.21%			

TABLE 10. Analysis of variance for normalized CPU time of SA.

Source	DF	Seq SS	Adj SS	Adj MS	F	P-value
Regression	11	1.29237	1.29237	0.117488	387.44	0.000
Linear	4	1.27233	1.27233	0.318083	1048.94	0.000
Square	4	0.01034	0.01034	0.002584	8.52	0.006
Interaction	3	0.00971	0.00971	0.003235	10.67	0.004
Residual Error	8	0.00243	0.00243	0.000303		
Lack-of-Fit	5	0.00242	0.00242	0.000483	138.87	0.001
Pure Error	3	0.00001	0.00001	0.000003		
Total	19	1.29480				
S = 0.01741	R-Sq = 99.8%		R-Sq(adj) = 99.6%			

TABLE 11. Payoff table for GA.

Y_2	Y_1	Pay off table
0.66275	0.94421	Min Y_1
0.343925	0.947676	Min Y_2

TABLE 12. Payoff table for SA.

Z_2	Z_1	Pay off table
0.986409	0.975802	Min Z_1
0.488046	0.980538	Min Z_2

The membership function of these two objectives can be achieved according to equations (4.5) and (4.6) for GA and equations (4.7) and (4.8) for SA.

$$\mu(Y_1) = \begin{cases} 1 & Y_1 < 0.94421 \\ \frac{0.947676 - Y_1}{0.947676 - 0.94421} & 0.94421 \leqslant Y_1 \leqslant 0.947676 \\ 0 & Y_1 > 0.947676 \end{cases} \tag{4.5}$$

$$\mu(Y_2) = \begin{cases} 1 & Y_2 < 0.343925 \\ \frac{0.66275 - Y_2}{0.66275 - 0.343925} & 0.343925 \leqslant Y_2 \leqslant 0.66275 \\ 0 & Y_2 > 0.66275 \end{cases} \tag{4.6}$$

$$\mu(Z_1) = \begin{cases} 1 & Z_1 < 0.975802 \\ \frac{0.980538 - Z_1}{0.980538 - 0.975802} & 0.975802 \leqslant Z_1 \leqslant 0.980538 \\ 0 & Z_1 > 0.980538 \end{cases} \tag{4.7}$$

$$\mu(Z_2) = \begin{cases} 1 & Z_2 < 0.488046 \\ \frac{0.986409 - Z_2}{0.986409 - 0.488046} & 0.488046 \leqslant Z_2 \leqslant 0.986409 \\ 0 & Z_2 > 0.986409. \end{cases} \tag{4.8}$$

Therefore, we have:

$$\begin{aligned} &\text{Min } Z = \sum_{j=1}^{2} w_j \alpha_j \\ &\text{Subject to:} \\ &\qquad \alpha_j \leqslant \mu(Y_j); \quad j = 1, 2 \\ &-1 \leqslant x_i \leqslant 1 \;\; ; \;\; i = 1, 2, \ldots, 5 \quad \text{for GA} \\ &\qquad\qquad\qquad i = 1, 2, \ldots, 4 \quad \text{for SA} \\ &\alpha_j \in [0, 1] \;\; ; \;\; j = 1, 2 \end{aligned} \tag{4.9}$$

where α_j is the achievement degree and w_j is the weight of the jth goal. Moreover, since the solution accuracy is more important than the CPU time performance, $w_1 = 0.75$ and $w_2 = 0.25$ are considered.

Solving (4.9) finally leads us to the optimal values of the GA and SA parameters given in Tables 13 and 14, respectively.

4.2. Experiments and comparisons

In order to validate the results obtained by the hybrid simulator and GA, and to evaluate the performance of the proposed algorithms, the computer program is coded using MATLAB and then applied on 400 instance problems. The programs run on a PC with Pentium 2.8 GHz processor with 512 MB RAM. The problems are

TABLE 13. Optimal parameters of GA.

parameters	Optimal value
population size	$2N$
Crossover	0.9
Mutation	0.3
max iteration	138
no.rep	30

TABLE 14. Optimal parameters of GA.

Parameters	Optimal value
Temp. decrease	0.99
Max initial Temp	80
Max iterations	100
no.rep	30

TABLE 15. T-Paired test with 0.95 confidence level.

	H_0	Group	Sample Standard Deviation	t-Statistic	p-value	Decision	Result
Fitness	$\mu_{1t} = \mu_{2t}$	7activities	225.776	–1.89	0.085	Accept H_0	Not Significant
	$\mu_{1S} = \mu_{2S}$	10activities	1059.38	–2.17	0.053	Accept H_0	Not Significant
	$\mu_{1M} = \mu_{2M}$	20activities	1417.35	–8.33	0+	Reject H_0	Significant
	$\mu_{1L} = \mu_{2L}$	30activities	3008.89	–8.8	0+	Reject H_0	Significant
	$\mu_{1A} = \mu_{2A}$	All	3452.4	–5.94	0+	Reject H_0	Significant
CPU Time	$\mu_{1t} = \mu_{2t}$	7activities	1.12711	–13.3	0+	Reject H_0	Significant
	$\mu_{1S} = \mu_{2S}$	10activities	0.52489	–12.93	0+	Reject H_0	Significant
	$\mu_{1M} = \mu_{2M}$	20activities	6.2453	8.34	0+	Reject H_0	Significant
	$\mu_{1L} = \mu_{2L}$	30activities	16.5112	9.09	0+	Reject H_0	Significant
	$\mu_{1A} = \mu_{2A}$	All	21.0443	4.29	0+	Reject H_0	Significant

categorized in four classes of 7, 10, 20, and 30 activities, and the t-paired test with confidence level of 0.95 was used to analyze the results shown in Table 15. In this table, μ_{ij}; $i = 1, 2$, $j = t, S, M, L, T$ stands for the mean response of either fitness or CPU time on 7-activity (t), small (S), medium (M), large (L) size problems and all problems (A), for the GA and SA, respectively ($i = 1, 2$).

The results in Table 15 not only validate the results obtained by the combined simulator and GA algorithm, but also show that GA performs better than SA, especially in larger size problems.

4.3. Sensitivity analysis

The results of sensitivity analysis on the values of the rewards paid per period to finish the project before the due date, r, and the penalty cost per period to complete the project beyond its due date, p, shows that when r increases, more activities would crash. Meanwhile, when p and r are both positive, the same scheduling of when only r itself is positive will be obtained. However, when only p is positive, the project will progress in the direction ending in d. The reason behind this is to prevent penalty. Furthermore, by increasing the holding costs, the value of the target function will increase, resulting in reduction of the project time.

Figures 2 and 3 depict the converging diagram of problem A30410 (30 activities, 4 equipment, and 10 activities requiring equipment) in random breakdown and non-random breakdown states, respectively.

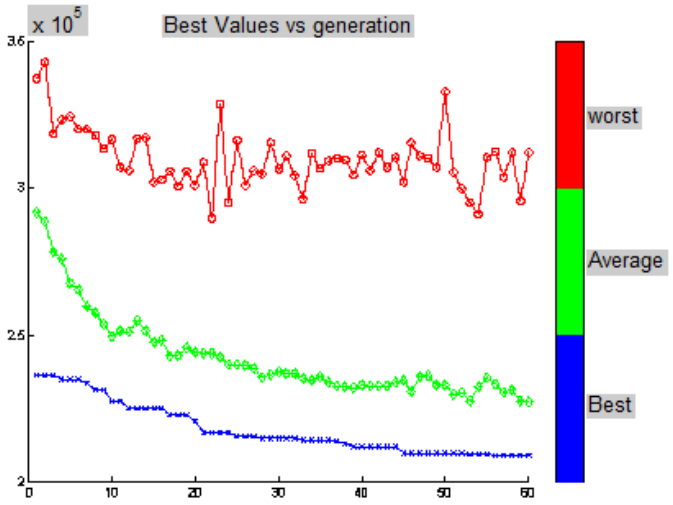

FIGURE 2. A30410 Without failure.

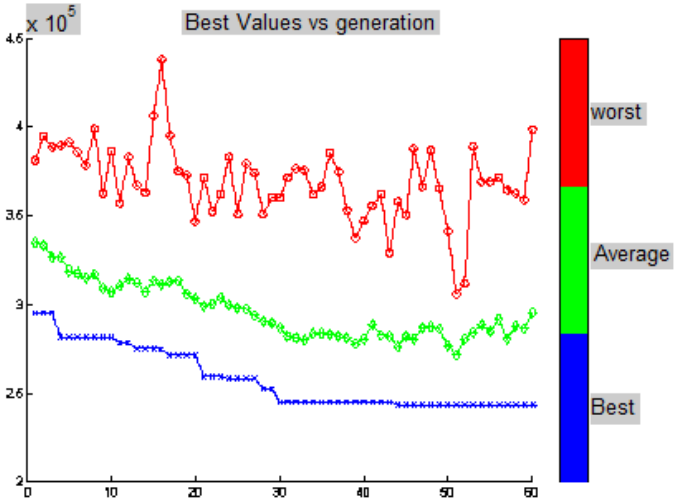

FIGURE 3. A30410 Without failure.

5. Conclusions and recommendations for future research

The contributions of this paper were mainly threefold: first, a new closer to reality problem that had not been addressed in the literature was introduced in this paper. This problem, which was the project-scheduling problem with simultaneous equipment planning, had two types of complexities: algorithmic complexity, and structural and functional complexity. Second, due to the structural and functional complexity of the potential problem, a simulator was used to simulate the problematic space of the problem. To solve the problem, some characteristics of the proposed algorithm were used including the new coding method, employed operators, hybrid algorithm, and high convergence speed of the algorithm in finding a near-optimal solution. Third, by using an experimental design and an ideal fuzzy scheduling model, the fitness and the CPU time performances were optimized simultaneously.

Future research in this area can be divided into two categories: The first is related to the problem formulation, and the other is involves the solving approach. Future research works related to the problem formulation are recommended as follows:

- Focusing on other characteristics of the failure event like non-resumable mode to continue processing after stopping due to random breakdowns
- Using other probability distributions to determine the time between two failures and repair time (such as geometric, binomial, etc.)
- Considering more than one type of equipment
- Using GPR instead of AON network
- Considering fuzzy parameters
- Considering preventive repair and maintenance

The main reasons for using meta-heuristic approaches were the efficiency and flexibility in practical situations. Future works about the solving approach are recommended as follows:

- Using other competing algorithms such as imperialist competitive algorithm, Tabu search, ant colony, etc
- Using another hybrid algorithm such as the resolution recovery

Acknowledgements. The authors are thankful for constructive comments of the anonymous reviewers. Taking care of the comments significantly improved the presentation.

References

[1] H. Allaoui and A. Artiba, Integrating simulation and optimization to schedule a hybrid flow shop with maintenance constraints. *Comput. Ind. Eng.* **47** (2004) 431–450.
[2] H. Allaoui and A. Artiba, Scheduling two-stage hybrid flow shop with variability constraints. *Comput. Oper. Res.* **33** (2006) 1399–1419.
[3] U. Beşkci, U. Bilgea and G. Ulusoyb, Multi-mode resource constrained multi-project scheduling and resource portfolio problem. *Eur. J. Oper. Res.* **240** (2015) 22–31.
[4] E.V. Davis and J.H. Patterson, *An exact algorithm for the multiple constrained project scheduling problem*. Ph.D. thesis, Yale University (1969).
[5] E. Demeulemeester and W. Herroelen, Project Scheduling: A Research Handbook. Kluwer Academic Publishers (2002).
[6] E. Demeulemeester, M. Vanhoucke and W. Herroelen, A random network generator for activity-on-the-node networks. *J. Sched.* **6** (2003) 13–34.
[7] B. Dodin, A.A. Elimam, Integration of equipment planning and project scheduling. *Eur. J. Oper. Res.* **184** (2008) 962–980.
[8] S.E. Elmaghraby, Activity networks: Project planning and control by network models. Wiley and Sons, New York (1977).
[9] S.E. Elmaghraby, Activity nets: A guided tour through some recent developments. *Eur. J. Oper. Res.* **82** (1995) 383–408.
[10] A. Fahmy, T.M. Hassan and H. Bassioni, Improving RCPSP solutions quality with Stacking Justification – Application with particle swarm optimization. *Expert Syst. Appl.* **41** (2014) 5870–5881.
[11] M. Gholami, M. Zandieh and A. Alem-Tabriz, Scheduling hybrid flow shop with sequence-dependent setup times and machines with random breakdowns. *Int. J. Adv. Manufact. Technol.* **42** (2009) 189–201.
[12] W. Herroelen, B. DeReyck and E. Demeulemeester, Resource-constrained project scheduling: A survey of recent developments. *Comput. Oper. Res.* **25** (1998) 279–302.
[13] W. Herroelen, P. Van Dommelen and E.L. Demeulemeester, Project network models with discounted cash flows – A guided tour through recent developments. *Eur. J. Oper. Res.* **100** (1997) 97–121.
[14] O. Icmeli, S.S. Erenguc and C.J. Zappe, Project scheduling problems: A survey. *Int. J. Oper. Prod. Manage.* **13** (1993) 80–91.
[15] R. Kolish, C. Schwindt and A. Sprecher, Benchmark instances for project scheduling problems. Handbook on recent advanced in project scheduling, edited by J. Weglarz. Kluwer Academic Publishers, Dordrecht (1998).
[16] R.H. Myers and D.C. Montgomery, Response surface methodology: process and product optimization using designed experiments. Wiley, New York (1995).
[17] A.A. Najafi, S.T.A. Niaki and M. Shahsavar, A parameter-tuned genetic algorithm for the resource investment problem with discounted cash flows and generalized precedence relations. *Comput. Oper. Res.* **36** (2009) 2994–3001.
[18] R. Narasimhan, Goal programming in a fuzzy environment. *Decis. Sci.* **11** (1980) 325–336.
[19] L. Ozdamar and G. Ulusoy, A survey on the resource constrained project scheduling problem. *IIE Trans.* **27** (1995) 574–586.

[20] J.H. Patterson, A comparison of exact approach for solving the multiple constrained resource, project scheduling problem. *Manage. Sci.* (1984) 854–867.

[21] J.H. Patterson and W.D. Huber, A horizon varying, zero one approach to project scheduling. *Manage. Sci.* **20** (1974) 990–998.

[22] F.B. Talbot and J.H. Patterson, An efficient integer programing algorithm with network cuts for solving resource constrained scheduling. *Manage. Sci.* (1992) 1163–1174.

[23] R.N. Tiwari, S. Dharmar and J.R. Rao, Fuzzy goal programming an additive model. *Fuzzy Sets Syst.* **24** (1987) 27–34.

[24] M. Vanhoucke, J. Coelho, L. Tavares and D. Debels, An evaluation of the adequacy of network generators with systematically sampled networks. Ghent University (2004).