An integrated hybrid metaheuristic model for the constrained scheduling problem

Bidisha Roy[1], Asim Kumar Sen[2]
[1]Department of Computer Engineering, St. Francis Institute of Technology, Mumbai, India
[2] Visiting Faculty, Institute of Marine Engineers, Mumbai, India

Article Info

ABSTRACT

Several problems in the domains of project management (PM) and operations research (OR) can be classified as optimization problems which are classically non-deterministic polynomial-time hard (NP-hard). One such highly important problem is the resource constrained project scheduling problem (RCPSP). The main aim of this problem is to find a schedule of the lowest and optimum makespan to complete a project, which involves resource as well as precedence constraints. But, being classically NP-hard, the RCPSP requires exponential computational resources as the problem complexity increases. Thus, approximate techniques like computational intelligence (CI) based approaches provide better chances of finding near optimal solutions. This paper presents the usage of a hybrid technique using the phases of teaching learning-based optimization (TLBO) metaheuristic integrated with operators like crossover and mutation from the genetic algorithm (GA). An integrated hybrid using TLBO and 2-point crossover is applied in the teacher and learner phases to the discrete RCPSP problem. Further, to diversify the population, and enhance global search, the mutation operator is applied. The proposed model is extensively tested on well-known benchmark test instances and has been compared with other seminal works. The encouraging results make evident the efficiency of the provided solution for the RCPSP problem of varying magnitudes.

Corresponding Author:

Bidisha Roy
Department of Computer Engineering, St. Francis Institute of Technology
Mumbai, India
Email: bidisha.bhaumik.roy@gmail.com

1. INTRODUCTION

Scheduling activities in a project is among the vital tasks in project management (PM). The scheduling process involves decision-making that aims to find the optimal time of completion of a project, also called the makespan, by arranging the activities in the project in such a manner that the requisite constraints are satisfied. The success of a project and hence the business of an organization, thus largely depends on how effectively and efficiently the activities in a project are scheduled. Most of the tools and techniques used to schedule activities in a project, however, assume that the availability of resources to be used in a project is unlimited over the time horizon of the project. However, in most practical real-time scenarios, resources such as humans, machines, and money. are available in limited capacities when several activities are going on concurrently in a project. Also, as industries are progressing towards Industry 4.0, it is becoming more and more imperative to schedule projects such that the limited resources are used more intelligently, along with optimizing the makespan of the project. Hence, a more representative approach is to take into consideration the limited

availability of resources while scheduling the activities in a project. This creates a new variant of this problem, which is called the resource constrained project scheduling problem (RCPSP) [1].

The RCPSP is a combinatorial optimization problem whose objective is to list and chart out the activities in a project with limited resources in a manner to minimize the makespan while satisfying the usual activity precedence of a project scheduling problem and also the constrained availability of resources. It is a key problem in many industries such as construction, aircraft maintenance, as the availability of resources in a constrained manner affects the scheduling of such projects decisively. As the problem is relatively general with multiple application areas, at the same time exhibiting complex, large-scale and non-linear behaviours, it has been a widely studied and researched area amongst engineers and scientists, both in the field of operations research (OR) as well as combinatorial optimization (CO).

Being a challenging problem both in terms of research and practical applications, RCPSP has received vigorous attention for almost two decades now and is still a very active and continuing area of research. Early research proposed many exact methods based on techniques like branch and bound, dynamic programming, mixed integer, and linear programming [2] to solve the problem. However, Blazewicz *et al.* [3] described RCPSP and other constrained scheduling problems as combinatorial optimization problems which in the strongest sense are NP-hard. Hence, though the exact methods did solve the problem optimally for smaller size instances, they were unable to provide a satisfactory solution in polynomial time as the problem size increased. The research focus, then shifted towards the use of approximate techniques based on greedy methods to solve the problem. In the last two decades, there has been ongoing research towards the use of approaches based on nature inspired swarm intelligence as given in [4], [5] to mention a few, to find solutions to instances of the problem as the size increases. Unlike exact methods which come with a guarantee of an optimal solution, these methods act as a best-effort delivery methods finding the near optimal solution in most cases and suitable feasible solutions almost always as they emulate characteristics found in the natural habitat of species like basic nature, adaptability, and cooperation.

Initial research proposed exact methods to solve the problem using problem-solving techniques like mixed integer programming, branch and bound, and dynamic programming [6]. Though these methods came with the assurance of an optimal solution, they could be used to solve only small-sized instances of the problem since computational efforts and execution time increased exponentially as the size of the problem increased. This limitation motivated researchers to look for approximate algorithms either based on heuristics or Computational Intelligence based metaheuristics to find solutions for large-sized practical problems within an acceptable computational effort.

Researchers have tried to obtain near optimal solutions using several heuristic methods for instances of larger sizes of the RCPSP problem in a rational amount of time. Heuristics can be considered as problem-specific methods beginning with an initial empty solution set which is subsequently filled up with activities iteratively. These heuristics tried to provide good solutions mostly based on either priority rule based heuristics or schedule generation schemes (SGS). In priority based methods, priority values were calculated for every activity based on some rule base and then scheduled so that a good solution would be obtained. On the other hand, SGS encoding generated feasible schedules taking into consideration starting times of the activities based on their precedence. Generally, two SGS schemes were applied: i) serial SGS based on the incrementation of activities in each iteration and ii) parallel SGS which was based on incrementation of time. Hartmann and Kolisch [6]–[9] put forward different heuristics to solve the problem which were based on both the schedule generation schemes, forward backward improvement (FBI), X-pass methods, and also heuristics based on priority rules. Though priority heuristics could solve large-sized RCPSP problems with acceptable computational efforts, their adaptability to the dynamic constraints of the problem was wanting. Hence SGS was preferred as heuristics for the larger instances of the problem.

The performance of heuristics also was largely affected by the problem size and its complexity. Thus, in the last two decades, researchers have explored the use of nature inspired soft computing metaheuristics over heuristics to better the feasible solutions provided by the heuristic methods. Metaheuristics are a class of methods used to solve optimization problems. They seek to attain the optima by emulating successful foraging behaviours and processes found in nature. Starting with an initial set of population constituting initial feasible solutions, they keep evolving and improving over generations due to the application of operations that transform current solutions into better solutions. Kolisch and Hartmann [6], [8] experimented with evolutionary strategies like genetic algorithms (GA), simulated annealing (SA), and tabu search (TS) over the initial schedules found using earlier heuristic methods. The tests were carried out on project scheduling problem library (PSPLIB) [10], which is considered a benchmark test set for RCPSP. The average standard deviation from optimal solutions was found to be experimentally better using this strategy. These experiments also initiated the idea of applying various metaheuristics over feasible solutions generated using heuristic methods to select optimal schedules.

Over the years, several other nature inspired and swarm intelligence based metaheuristics that maintained a set of solutions in each iteration were proposed. One of the earliest such works was the usage of ant colony optimization (ACO) suggested by [2]. Many variations and improvements to this were suggested by various researchers [11], [12]. These were based on Max-Min ACO or applying oblivion rate to the pheromone trail after every generation. Another popular metaheuristic that has gained popularity is the particle swarm optimization (PSO) algorithm. The work of Zhang *et al.* [13] put forward a solution based on the PSO with competitive results mostly for the J30 test instances from the PSPLIB [10]. This led to various other research using PSO with modifications [14], [15]. Another popular metaheuristic that has been explored in the study of the RCPSP problem is the bee algorithm (BA) which is based on the foraging behaviour of honeybees. Ziarati *et al.* [16] suggested three different variations of the bee algorithms. Some discretized permutation-based bee algorithm techniques have been proposed by [17], [18]. Research based on hybridization of BA and PSO has also been suggested based on experiments carried out in [19]. Some other prominent works include the usage of nature inspired metaheuristics like cuckoo search algorithm (CSA) [20], [21], flower pollination algorithm (FPA) [22], brain storm algorithm (BSA) [23], and discrete firefly algorithm (DFA) [24] amongst a few. A detailed study on hybrid metaheuristics to solve the RCPSP problem has been given by Pellerin *et.al.* in [25].

As seen in the previous paragraph, metaheuristics are now often used to solve large-sized instances of the RCPSP Problem due to their ability to produce reasonably good results in polynomial time. Teaching learning based optimization algorithm (TLBO) by Rao *et al.* [4] is one such prominent metaheuristic. This algorithm emulates the learning environment in a classroom. It tries to obtain the solution by creating two vital entities in a population: the teacher and a set of learners. The algorithm is designed to use the collective intelligence of the class to obtain optimum results.

In this research, the use of the TLBO [4] in solving the RCPSP problem has been investigated. Many studies have proposed the use of the TLBO algorithm for a variety of optimization problems with constraints. Some of the prominent usages of the TLBO for solving computationally hard problems include flow-shop and job-shop scheduling [26], foundry industry [27], and the travelling salesman problem (TSP) [28] to name a few. In these papers, the TLBO has emerged as a popular and prominent metaheuristic for solving hard decision problems with results quite comparable to other metaheuristic algorithms with competitive results. This motivated us to apply the TLBO algorithm to provide an alternate solution to the RCPSP problem. For efficient solving of the discrete RCPSP problem and to further diversify the population to attain global optima, this research proposes to integrate the phases of the TLBO with the advantages of some operators from the GA [5]. The prominent contributions of our work are:

− We have redesigned the original steps of the teacher and learner phases to integrate a two-point crossover to update the learner when it interacts with the better member of the population.
− To improve the search space to not get stuck in local optima, we have added the mutation operator to a crossover-updated learner based on some mutation probability.
− The crossover and mutation operators of GA have been adapted to the RCPSP problem.
− The integrated hybrid of TLBO with GA is our major novel work.

The results of the investigation are presented by testing them on benchmark RCPSP test instances and comparing them with earlier seminal metaheuristics. The numerically competitive results prove the vantage of the investigated research. In this paper, in section 2, the problem formulation for RCPSP is presented, while briefly deliberating the TLBO algorithm and the proposed TLBO-GA based approach is presented. In section 3, we provide the results of the conducted experiments on the benchmark instances. Section 4 provides the analysis and conclusion of the results along with the probable future directions observed in this research.

2.   METHODOLOGY
2.1.   Problem formulation

To represent the RCPSP problem, the information needed is as [1]: RCPSP can be formally defined by viewing a project as a directed acyclic graph $G (A, E)$, where A = {0, 1, 2, …., n+1} denotes activities in a project having n activities. Activities 0 and n+1 are dummy activities indicating the beginning and the termination of the project. Set E contains all the precedence relations existing amongst activities. A precedence between activity *i* to *j* denotes that activity *j* cannot commence before activity *i* is completed. The set $R=\{1,2,3.... k\}$ denotes the *k* number of renewal resources available over the period of execution of the project.

During processing, r_{jk} denotes the amount of resource k needed by activity j during the execution of its non-preemptive execution duration p_j. The duration of the dummy activities is always zero. For every activity j, S_j denotes its start time and $F_j=S_j+p_j$ denotes its completion or finishing time. Thus, the goal of the problem is to determine the schedule set S for the project, such that:

− At every instance of time t the total demand for resource k does not exceed the availability R_k for k = 1, ..., r,

- The precedence constraints are satisfied, i. e. $S_i + p_i <= S_j$, if $i \rightarrow j$,
- The objective function $f(F_1, ..., F_n)$ is minimized with F denoting the Completion time of the project

With these given constraints, an optimal schedule for the project needs to be constructed. A solution S is called feasible if it complies with both the precedence constraints as well as the resource constraints. The mathematical model of the RCPSP would be thus given as:

$$Min\ (F_{max}) \tag{1}$$

Subject to constraints:

$$S_j - S_i \geq p_i \forall (i, j) \in E \tag{2}$$

$$\sum_{i \in A} r_{ik} \leq R_k, \forall R_k \in R, \forall t \geq 0 \tag{3}$$

$$F_j \geq 0 \tag{4}$$

Thus, the RCPSP consists of finding a schedule S that has the minimum time considering the constraints of precedence of activities and resources available at hand. The additional constraint of resources adds to the complexity of the problem. The RCPSP can be demonstrated with a small example as given in Figure 1.
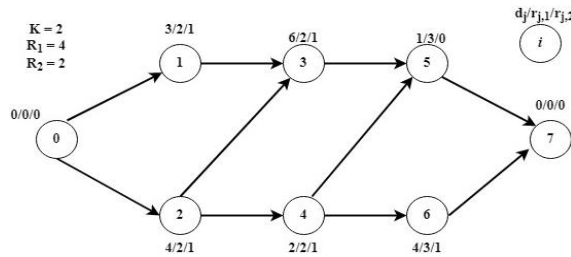

Figure 1. An example

The RCPSP is represented here as a directed acyclic activity on node (AoN) graph. Activities are represented as nodes and the precedences as directed paths between activities. There are 6 non-dummy activities represented numbered as nodes 1-6 which have to be scheduled. They would be utilizing K=2 renewable resources which have a capacity of 4 and 2 units respectively. In (4) gives the constraint of the completion time decision variable. The primary goal or objective is thus to identify a schedule of minimum makespan (1), considering both the precedence constraints (2) and resource constraints (3). The constraint in (4) depicts the constraint of the completion time decision variable. A feasible schedule for the given example is shown in Figure 2, which has an optimal makespan of 15 units.
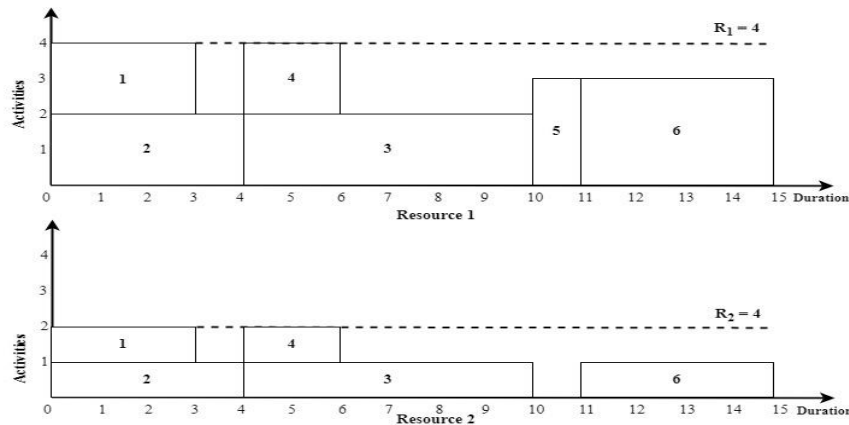

Figure 2. A feasible solution to example of Figure 1

2.2. Teaching learning based optimization

Like most evolutionary algorithms in the field of computational intelligence, TLBO [4] also employs a population-based optimization method. It approaches the global optima using a population of solutions. It takes inspiration from the fact that teachers as well as peers amongst learners influence the outcome of learners and hence the entire class. The algorithm consists of two vital entities in the system; The teacher and the learners. The algorithm works on two modes of learning: firstly, the teacher teaching or interacting with the learners, known as the teacher phase followed by the interaction amongst the learners known as the learner phase.

2.2.1. Teacher phase

This is the first phase of interaction where the learners try to gain knowledge from the teacher where as the teacher through their knowledge attempts to improve the overall result of the class. The individual i.e. the solution with the least finish time (best fitness value) is chosen as the teacher of the class. Considering a d-dimensional variable objective function $f(x)$, the ith learner would be a feasible solution set $L_i = [l_{i1}, l_{i2}, ... l_{id}]$. In a class of m learners, the mean of the class would be given by $L_{mean} = 1/m[\sum_{i=1}^{m} l_{i1}, \sum_{i=1}^{m} l_{i2} .... \sum_{i=1}^{m} l_{id}]$. If the teacher is represented as $L_{Teacher}$, the position of learners' are updated in each iteration as (5):

$$L_{i,new} = L_{i,old} + r_i (L_{Teacher} + T_F L_{mean}) \qquad (5)$$

Where in $L_{i,new}$ and $L_{i,old}$ are the updated and initial positions of the ith learner. r_i is any random number ranging from 0 to 1. The teaching factor, T_F denotes the mean value to be changed. Calculation of T_F is a heuristic step generating values either 1 or 2, to keep the decision random with equal probability. T_F takes values 1 or 2 with 1 indicating no transfer of knowledge to the learner and 2 denoting transfer of knowledge from teacher to student. The new solution replaces the older solution if found to be better.

2.2.2. Learner phase

Here, the learners learn from each other due to peer interaction. If another learner has more knowledge than the current learner, the current learner also benefits from it. An individual learner L_k randomly selects another learner $L_{j,} j \neq k$, and the learning happens as (6) and (7):

$$L_{j,new} = L_{j,old} + r_j(L_j - L_k), if \, f(L_j) < f(L_k) \qquad (6)$$

$$L_{j,new} = L_{j,old} + r_j(L_k - L_j), if \, f(L_j) > f(L_k) \qquad (7)$$

The steps in (6) and (7) bring the potential of a learner towards a better learner. The new value is accepted if it is better. Since it is a minimization problem, the lesser value of the objective function is considered the fitter value.

2.3. TLBO for RCPSP

As mentioned in the previous sections, the TLBO has been effectively adapted to solve numerous constrained optimization problems in the field of engineering and operations research with relative success [26]–[28]. However, as seen in the previous section, the equations for classic TLBO are mostly designed for continuous optimization problems. RCPSP, though, is a discrete optimization problem with every activity in the schedule list being distinct. Hence, some modifications needed to be done to make the TLBO suitable to be applied to the problem. These modifications were inspired by the use of integrated hybrid algorithms using some features of the GA [7], [9] and also from the detailed study of hybrid algorithms mentioned in [25]. The framework for the proposed approach is given by the flowchart as shown in Figure 3.

2.3.1. Solution representation and schedule generation

The representation and encoding of the population in the RCPSP is an essential step for better algorithmic performance. Though the solution can be represented in many forms, Kolisch and Hartmann [7] in their research, have mentioned that RCPSP can be solved with heuristics using an activity list (permutation-based) or random key list (priority-key) representation. In our investigations, a permutation-based activity list (AL) encoding scheme has been used to generate the initial solutions, as it was proven to be more suitable for single-mode RCPSP problems due to its ease of implementation and also quick decoding. Also, there always is an AL schedule that induces an optimal schedule [29].

Every member or learner of the population in this framework is an activity vector representing a feasible solution in the n-dimensional parameter space, n representing the number of activities in the project.

Ordering of the activity is represented by the index of the activity in the sequence. The encoding scheme lists the activities such that the precedence constraints have been maintained i.e. the predecessor should be indexed before its successors. As RCPSP is a discrete and deterministic problem, all the activities are integers and are distinct, the search space required by this permutation-based encoding scheme is drastically reduced.

These AL need to be made feasible so that they can be evaluated to find the best solution. Hence, SGS are used to produce active non-delay schedules from the AL. SGS is the fundamental decoding procedure for the RCPSP that takes into consideration both the precedence constraints as well as the availability of the constrained resources at hand [7]. It is a heuristic that starts from a schedule of zero activities and adds activities to be scheduled through iterative improvements. Of the two types of SGS as discussed in section 2 SSGS, which generates schedules using activity incrementation and PSGS, which performs time incrementation to add activities iteratively in each time interval, in this research, SSGS decoding has been adopted for generating active schedules. The SSGS is the most oft-applied heuristic by most models for the RCPSP [8] as it is found to be most suitable for permutation-based encoding due to its ability to always generate active non-delay (feasible) schedules. Hence searching the solution space using a metaheuristic would have a greater probability of returning optimal results. SSGS generates the entire generation of a feasible schedule in n iterations. An activity is taken iteratively and is scheduled at its earliest feasible completion time satisfying both the precedence as well as resource constraints. On the nth iteration, the schedule terminates with all the non-dummy activities duly scheduled. The makespan is indicated by the maximum completion time amongst all the activities preceding dummy activity n+1. The SSGS operates in time $O(n^2R)$, R being the resources available for the project. The next immediate pending activity to construct the schedule is selected on a latest start time (LST) priority basis. The makespan determines the fitness value of every learner i.e. every feasible schedule.

Consider the example shown in Figure 1. If we only consider a typical feasible schedule based on the precedence relations overlooking the resource constraints, a random AL would be represented as shown in Figure 4. Figure 4(a) represents the duration if only a random AL is generated on the basis of precedence relations which is 12 units. The corresponding AL is as given in Figure 4(b) with 0 and 7 representing the dummy start and sink activities respectively.
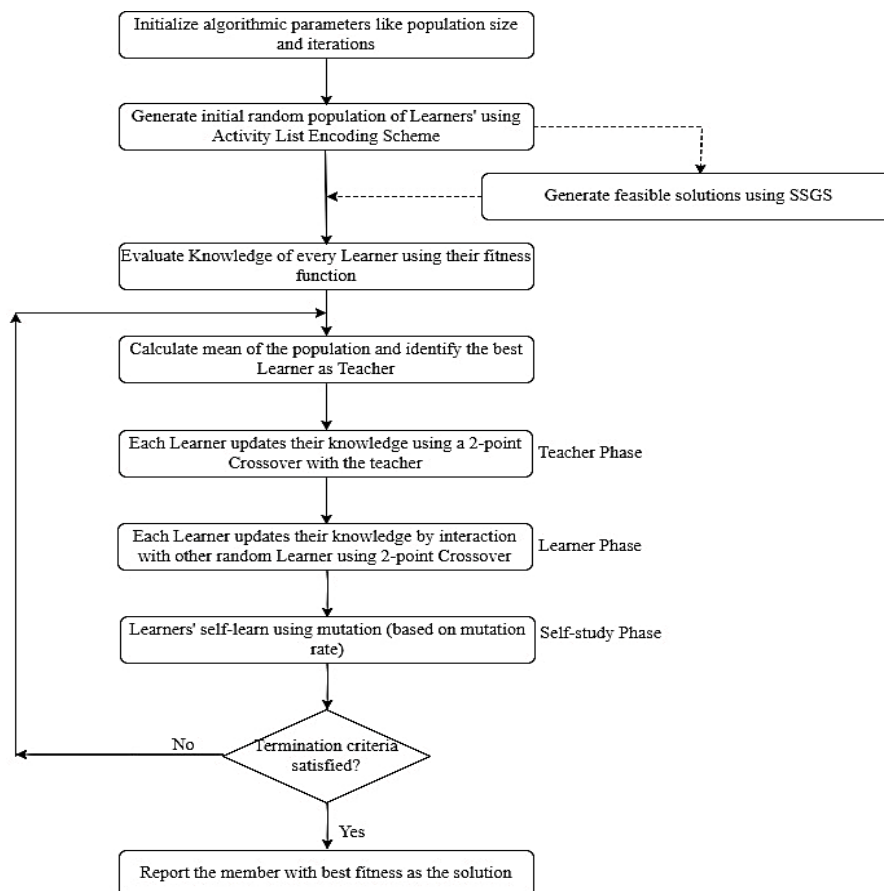

Figure 3. A framework for the proposed model
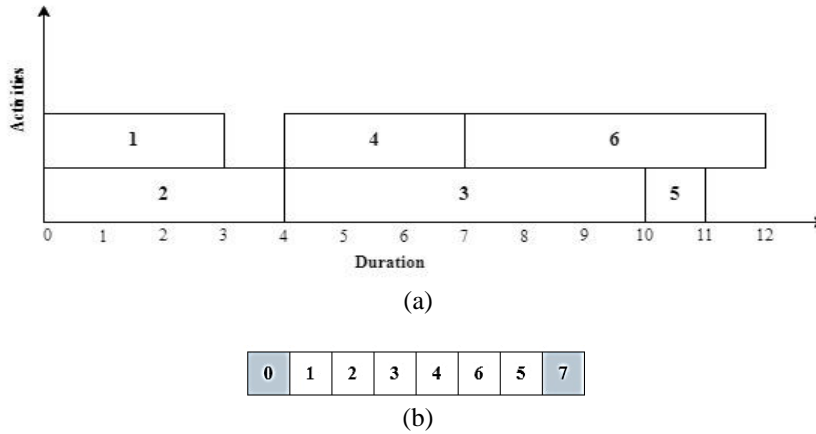
(a)

(b)

Figure 4. A precedent feasible AL where (a) schedule generated only on basis of precedence constraints and (b) corresponding AL generated

However, both resources R_1 and R_2 and their utilization by the activities also need to be taken into consideration. For example, though activities 5 and 6 are independent precedence-wise, they have a common requirement for R_1. To solve this, the SSGS is used to generate both resource and precedence feasible schedules which generate the solution as shown in Figure 2 where the duration is now 15 units. Such an AL taking into consideration both the constraints is shown in Figure 5.
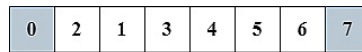

Figure 5. AL generated using SSGS

2.3.2. Teacher and learner using 2-point crossover

The schedule with the minimum makespan i.e. maximum fitness in the entire population generated using SSGS is selected as the teacher. The mean for the entire population is calculated. Every learner is updated if the updated value has a better fitness calculated using SSGS. However, the equations for classic TLBO are mostly designed for continuous optimization problems. To modify the algorithm to adapt to the discrete and deterministic RCPSP problem, the 2-point crossover of the GA inspired by [5] is used to bring the learners fitness/ability closer to the teacher. A crossover operation is used as it reserved the precedence feasibility of the schedules generated. As an example, consider two individuals, I^1 and I^2 as the teacher and a learner Respectively. Let their feasible schedules be depicted as shown in Figure 6 with 0 and 7 being the dummy activities. These schedules are for the example illustrated in Figure 2.
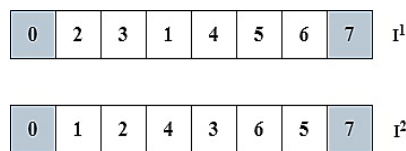

Figure 6. Feasible schedules of teacher and learner

Two integers u_1 and u_2 in the range [1, n] are randomly generated. Let the values generated by random number generation be $u_1=2$ and $u_2=4$. A new individual I^{new} is generated using the following crossover operations:

$$I_j^{new} = I_j^2, 1 \le j \le u_1 \tag{8}$$

$$I_j^{new} = I_k^1, k = \min\{k|I_k^1 \notin I_k^{new}, \dots, I_{u1}^{new}\}, u_1 + 1 \le j \le u_2 \tag{9}$$

$$I_j^{new} = I_k^2, k = \min\{k | I_k^2 \notin I_k^{new}, \dots, I_{u2}^{new}\}, u_2 + 1 \le j \le n \tag{10}$$

Hence, the new Individual based I^1 and I^2 would be as illustrated in Figure 7. The new individual replaces the current learner if found to be having better fitness. Though the random number and the T_F factor is removed, u_1 and u_2 are generated randomly which helps to keep the decision random.

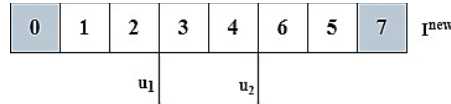| 0 | 1 | 2 | 3 | 4 | 6 | 5 | 7 | I^{new} |
|---|---|---|---|---|---|---|---|

Figure 7. New learner generated after 2-point crossover

2.3.3. Self-study using mutation

In a general learning scenario, learners after learning from Teacher and peers engage in self-study to enhance their knowledge. Also, with the above teaching-learning steps as described in section 2.2 and 2.3.2, there is a tendency of the solution getting stuck in the local optima and missing out the global solution at times. Hence, inspired by the works of [30]–[32], we have implemented the self-study phase as a mutation operation to create diversity in the population. We have used a swap operator in which activities are swapped for a certain population selected based on the mutation probability. The position of swap for an activity maintains the precedence i.e. higher than the predecessors and lower than the successors.

Again, the new mutated learner is retained if its fitness is found to be better. This process is carried out for all learners that are selected on the basis of mutation probability till the termination criterion is satisfied. The algorithm finally terminates upon which the updated teacher is returned as the best solution.

2.   RESULTS AND DISCUSSION

The results of the computational experiments carried out to inspect the performance of the proposed model mentioned in the previous section are reported and discussed in this section. The proposed model has been investigated using the benchmark data sets from the PSPLIB [10] which have become a standard for the RCPSP. The PSPLIB consists of 4 different types of datasets viz; J30, J60, J90 and J120. The first three are sets of projects having 30, 60 and 90 activities (non-dummy) respectively. These sets contain 480 such instances that can be comprehensively tested. Similarly, the J120 set has instances of 600 projects containing 120 non-dummy activities each. ProGen generator was used to generate these test instances. Each instance is generated using 3 main parameters: i) network complexity (NC), ii) resource strength (RS), and iii) resource factor (RF). More details about the instances and their generation can be found in [10]. Since the RCPSP is such a complex problem, optimal solutions are known only for the J30 instances. For the other instances, the critical path lower bound solutions are provided. To be able to provide a reasonable comparison with other seminal models, the instances were tested over 1,000 and 5,000 schedules. The solutions were measured on the basis of average standard deviation, $Dev\_Avg,$ from the optimal solutions for the J30 instances and average deviation from best solutions obtained from lower bound critical path methods for the other instances. Dev_Avg is the deviation of our proposed model's solution with respect to the best solutions stored in the data sets.

$$Dev\_Avg = \frac{\sum_{i \in BI} \frac{(Obtained_i - Best_i) * 100}{Best_i}}{instances} \tag{11}$$

Where *BI* denotes Benchmark Instances.

3.1. Parameter settings

The framework was implemented using Matlab R2014a on a CORE i5 10th generation 16 GHz machine. The major advantage of the proposed model is that it does not require setting of parameters. The only factor that is calculated i.e. the TF takes value between 1 and 2 and are calculated randomly with every population member, which has been now replaced by the 2-point crossover.

3.1.1. Population size

The TLBO is a fast-converging algorithm. Hence, even if the population size is large, it does not slow down the algorithm. However, we have tuned the population size so as to provide a decent search space for the

algorithm to find reach the optima. Keeping the mutation rate constant at 0.5 initially, the model was tested on increasing population sizes. The benchmark test instances used was from J30 of the PSPLIB [10] as these instances have their optimal solutions known. 10 independent runs for each value of population sizes were considered and the average standard deviations were recorded over 1,000 generated schedules in each generation. The behaviour with changing population size is as shown in Figure 8.
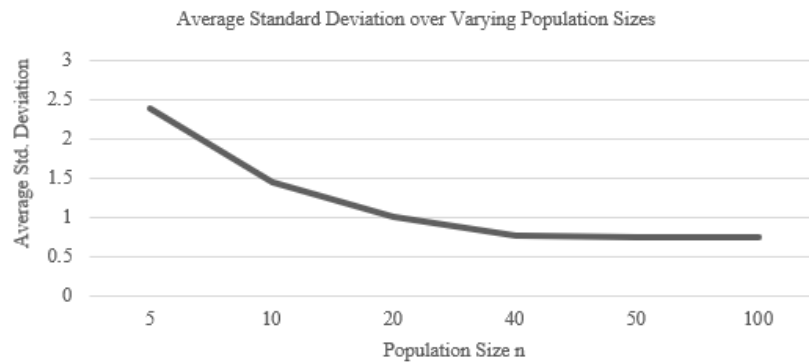

Figure 8. Average standard deviation over varying population size

The increments in population size were done considering the 1,000 schedules to be generated. The number of schedules is taken as the product of the population size *n* and the number of generations. As is clearly demonstrated, a population size closer to the number of activities in a project gives better standard deviation which then stabilizes. Hence, to maintain a decent search space as well as adequate number of generations to reach the global optimal solution, by experimentation, we have kept the population size for this model at 50.

3.1.2. Mutation rate

Once the population size was fixed, the mutation rate was tuned to the population size of 50. Again, the model was tested on J30, as the optimal solutions are known, with mutation probabilities 0.5, 0.6, 0.7, 0.8 and 0.9 respectively for 1,000 schedules with the rates indicating the percentage of students engaging in self-study post peer interaction. We conducted 10 independent runs for 10% of the test instances of all the data sets without any repetition. Table 1 tabulates the results of these independent experiments. It depicts the results of 10 independent experiments on the J30 test set for different mutation rates. The deviation improves as more and more learners indulge in self-study i.e. the rate of mutation increases.

Table 1. Average deviation on 10% training instances

Expt. No.	Mutation rate				
	0.5	0.6	0.7	0.8	0.9
1	0.77	0.66	0.61	0.65	0.62
2	0.68	0.55	0.55	0.67	0.70
3	0.87	0.80	0.69	0.82	0.81
4	0.63	0.61	0.54	0.56	0.60
5	0.65	0.56	0.65	0.75	0.77
6	0.92	0.8	0.77	0.89	0.94
7	0.8	0.69	0.8	0.8	0.82
8	0.73	0.60	0.5	0.64	0.65
9	0.97	0.88	0.81	0.89	0.93
10	0.77	0.73	0.64	0.78	0.76

However, as the rate of mutation becomes very high, from 80% on wards, the randomness of the search process becomes high thus leading to a degradation of the results. Mutation in the self-study phase had been introduced to dissipate the effects of crossover operator's lack of generation of new type of genes, here learner, which causes the solution to get trapped in local optima if the solutions generated were poor. Mutation as a concept alters learners after the feasible solutions are created. However, if a mutation rate is set to be very high, the population gets converted into a random search population [33], thus reducing the positive impact of the crossover operator, which is also seen in the graph of Figure 9.
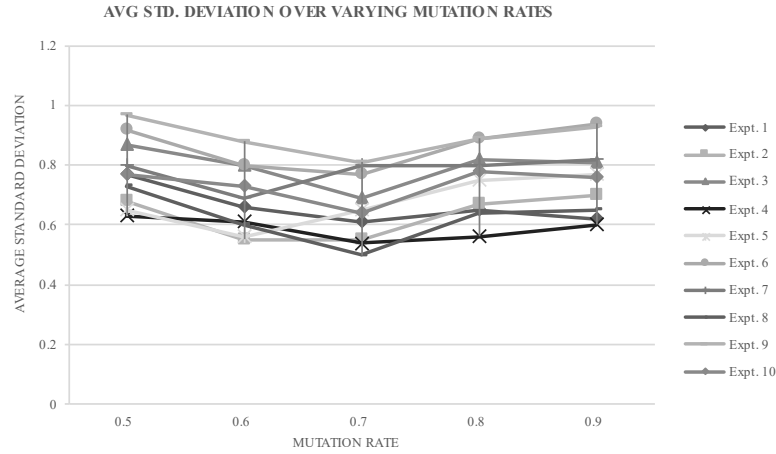
Figure 9. Standard deviation trajectory over various mutation rates

In TLBO, though every individual learner is an individual population member, they are viewed as a class unit. Hence, diversifying the entire population to a random space is not desirable. In our research, we thus assume that 70% of the students in a class in general indulge in self-study and for future experiments we have kept the mutation rate to 0.7.

An exhaustive comparison of our model has been done with the best available metaheuristics in literature. These include models involving genetic algorithms [5], ACO [2], [11], [12], PSO [15], [19], BA [16], [18], and other such metaheuristics. To be able to present a fair comparison, the TLBO model too has been tested over 1,000 and 5,000 schedules for 30 independent runs for lower and medium size test instances i.e. J30 and J60 and 20 independent runs for the J120 data set for its computational complexity.

Table 2 presents the average standard deviation for the J120 test instances where all the instances have been tested for 1,000 as well as 5,000 schedules respectively. Similarly, Tables 3 to 4 show the average standard deviations from the lower bound critical path solutions for the J60 and the optimal known solutions for J30 case studies. There have not been many tests conducted on the J90 datasets to provide a substantial comparison.

Table 2. Average deviation (%) on j120 instance case study

Algorithm or Model	SGS	Schedules	
		1000	5000
GA [5]	Serial	39.37	36.74
GA [5]	Priority rule	39.93	38.49
MMAS [12]	Serial	--	31.5
ACO-DOR [11]	--	26.58	26.57
A-PSO [15]	Random key	34.93	32.47
ABC [16]	Serial	43.24	39.87
BSO [16]	Serial	41.18	37.86
BA [16]	Serial	40.38	38.12
PBA [17]	Serial	42.85	38.45
ABC-PSO [19]	Serial	36.15	35.28
IDCS [20]	Serial	33.43	32.69
CRO-GA [34]	Random	33.85	32.42
This study	Serial	30.64	25.62

From the tables, we find that the proposed model using TLBO integrated with the power of crossover and mutation of GA is very much competitive and comparable algorithm to solve the RCPSP problem. The deviations for J120 and J60 which are more indicative of medium and large sized projects show a great improvement in the standard deviation due to the TLBO algorithm. The deviation of the J30 test instances is also comparable. The usage of the 2-point crossover enhances the exploration search as well as and helps generate more potential learners for the next generation. Also, the swap mutation operator helps create a diverse population thus reducing the possibility of the standard algorithm getting stuck in the local search space and not reaching the global optima.

The crossover method used in both the phases generates the new population based on a current and historical random learner, allowing the learner in the new population to be modified with better probability.

Both the teacher and the learner phases are adapted suitably to accommodate the discrete and deterministic nature of the RCPSP problem for single-mode variant, thus enhancing the capability to find the global optima balancing both the exploration and exploitation capability. Also, the need to modify the standard formulae for the discrete nature of the problem is eliminated. The self-study phase which incorporates the swap mutation with probability randomly modifies some percentage of the learners thereby enhancing the search-ability in the global space.

Table 3. Average deviation (%) on J60 instance case study

Algorithm or model	SGS	Schedules	
		1000	5000
GA [5]	Serial	12.68	11.89
GA [5]	Priority rule	13.30	12.74
ACO-DOR [11]	--	11.51	11.51
A-PSO [15]	Random key	11.94	11.12
ABC [16]	Serial	14.57	13.12
BSO [16]	Serial	13.67	12.70
BA [16]	Serial	13.35	12.83
PBA [17]	Serial	13.39	12.10
ABC-PSO [19]	Serial	12.14	11.90
IDCS [20]	Serial	11.78	10.99
CRO-GA [34]	Random	11.64	10.80
This study	Serial	12.18	10.82

Table 4. Average deviation (%) on J30 instance case study

Algorithm or model	SGS	Schedules	
		1000	5000
GA [5]	Serial	1.03	0.56
GA [5]	Priority rule	1.38	1.12
A-PSO [15]	Random key	0.28	0.06
PSO-ICA [14]	--	0.57	0.46
ABC [16]	Serial	0.98	0.57
BSO [16]	Serial	0.65	0.36
BA [16]	Serial	0.63	0.33
PBA [17]	Serial	0.62	0.30
ABC-PSO [19]	Serial	0.28	0.15
IDCS [20]	Serial	0.44	0.25
CRO-GA	Random	0.15	0.05
This study	Serial	0.48	0.36

3. CONCLUSION

It is a well-proven fact that NP-hard Problems like the scheduling problems are the few of the most complicated combinatorial optimization problems in literature, RCPSP being one of them. Also, due to its essential use in industry, an optimal solution to the problem is evidently needed. In this research, we have integrated the phases of the TLBO algorithm with the power of 2-point crossover from the GA to solve the RCPSP problem. In addition, swap mutation is applied in form of a self-study phase to generate a diverse population. This model was tested on various benchmark test instances from the PSPLIB. An exhaustive study and comparison of these techniques' vis a vis other prominent researches have also been presented. The computational experiments show that this model produces consistently good solutions for the problem. The operators of the GA introduced improve the performance of the classic TLBO algorithm demonstrating that hybrid metaheuristics show better promise in reaching global optima for the RCPSP problem. This improved model also requires less parameters, as we only define the number of learners i.e. population size and the iterations indicating the generations. Thus, hybrids with less parameters to be set can be future direction of investigation towards solving this problem optimally. The results also show that a hybrid approach consisting of the best features of various soft computing and nature inspired algorithms provides a better standard deviation from the optimal values. Hence, a hybrid approach with well-known nature inspired swarm intelligence techniques can be a way of approaching this problem in the future. This approach can be extended towards other variants of the RCPSP problem like multi-mode or multi-skill situations, pre-emptive precedence or stochastic time durations by way of future study. This model can also be adapted to find solutions to other hard combinatorial optimization problems in future. Other GA operations like multi-point or position-based crossover or mutation operators like uniform mutation or power mutation can also be investigated by way of future study.

REFERENCES

[1] M. Vanhoucke, *Project management with dynamic scheduling*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[2] D. Merkle, M. Middendorf, and H. Schmeck, "Ant colony optimization for resource-constrained project scheduling," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 4, pp. 333–346, Aug. 2002, doi: 10.1109/TEVC.2002.802450.

[3] J. Blazewicz, J. K. Lenstra, and A. H. G. R. Kan, "Scheduling subject to resource constraints: classification and complexity," *Discrete Applied Mathematics*, vol. 5, no. 1, pp. 11–24, 1983, doi: 10.1016/0166-218X(83)90012-4.

[4] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems," *CAD Computer Aided Design*, vol. 43, no. 3, pp. 303–315, Mar. 2011, doi: 10.1016/j.cad.2010.12.015.

[5] S. Hartmann, "A competitive genetic algorithm for resource-constrained project scheduling," *Naval Research Logistics*, vol. 45, no. 7, pp. 733–750, Oct. 1998, doi: 10.1002/(SICI)1520-6750(199810)45:7<733::AID-NAV5>3.0.CO;2-C.

[6] S. Hartmann and R. Kolisch, "Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 127, no. 2, pp. 394–407, Dec. 2000, doi: 10.1016/S0377-2217(99)00485-3.

[7] R. Kolisch and S. Hartmann, *Heuristic algorithms for the resource-constrained project scheduling problem: classification and computational analysis*. Springer {US}, 1999.

[8] R. Kolisch and S. Hartmann, "Experimental investigation of heuristics for resource-constrained project scheduling: An update," *European Journal of Operational Research*, vol. 174, no. 1, pp. 23–37, Oct. 2006, doi: 10.1016/j.ejor.2005.01.065.

[9] R. Kolisch and R. Padman, "An integrated survey of deterministic project scheduling," *Omega*, vol. 29, no. 3, pp. 249–272, Jun. 2001, doi: 10.1016/S0305-0483(00)00046-3.

[10] R. Kolisch and A. Sprecher, "PSPLIB - A project scheduling problem library," *European Journal of Operational Research*, vol. 96, no. 1, pp. 205–216, Jan. 1997, doi: 10.1016/S0377-2217(96)00170-1.

[11] A. Gonzalez-Pardo, J. Del Ser, and D. Camacho, "Comparative study of pheromone control heuristics in ACO algorithms for solving RCPSP problems," *Applied Soft Computing Journal*, vol. 60, pp. 241–255, Nov. 2017, doi: 10.1016/j.asoc.2017.06.042.

[12] Y. Zhou, Q. Guo, and R. Gan, "Improved ACO algorithm for resource-constrained project scheduling problem," in *2009 International Conference on Artificial Intelligence and Computational Intelligence (AICI)*, 2009, vol. 3, pp. 358–365, doi: 10.1109/AICI.2009.461.

[13] H. Zhang, X. Li, H. Li, and F. Huang, "Particle swarm optimization-based schemes for resource-constrained project scheduling," *Automation in Construction*, vol. 14, no. 3, pp. 393–404, Jun. 2005, doi: 10.1016/j.autcon.2004.08.006.

[14] M. Kasravi, A. Mahmoudi, and M. R. Feylizadeh, "A novel algorithm for solving resource-constrained project scheduling problems: a case study," *Journal of Advances in Management Research*, vol. 16, no. 2, pp. 194–215, Apr. 2019, doi: 10.1108/JAMR-03-2018-0033.

[15] N. Kumar and D. P. Vidyarthi, "A model for resource-constrained project scheduling using adaptive PSO," *Soft Computing*, vol. 20, no. 4, pp. 1565–1580, Feb. 2016, doi: 10.1007/s00500-015-1606-8.

[16] K. Ziarati, R. Akbari, and V. Zeighami, "On the performance of bee algorithms for resource-constrained project scheduling problem," *Applied Soft Computing Journal*, vol. 11, no. 4, pp. 3720–3733, Jun. 2011, doi: 10.1016/j.asoc.2011.02.002.

[17] M. A. Nemmich, F. Debbat, and M. Slimane, "A permutation-based bees algorithm for solving resource-constrained project scheduling problem," *International Journal of Swarm Intelligence Research*, vol. 10, no. 4, pp. 1–24, Oct. 2019, doi: 10.4018/IJSIR.2019100101.

[18] M. A. Nemmich, F. Debbat, and M. Slimane, "An enhanced discrete bees algorithms for resource constrained optimization problems," *Inteligencia Artificial*, vol. 22, no. 64, pp. 123–134, Dec. 2019, doi: 10.4114/intartif.vol22iss64pp123-134.

[19] Q. Jia and Y. Guo, "Hybridization of ABC and PSO algorithms for improved solutions of RCPSP," *Journal of the Chinese Institute of Engineers, Transactions of the Chinese Institute of Engineers, Series A*, vol. 39, no. 6, pp. 727–734, Aug. 2016, doi: 10.1080/02533839.2016.1176866.

[20] K. Bibiks, Y. F. Hu, J. P. Li, P. Pillai, and A. Smith, "Improved discrete cuckoo search for the resource-constrained project scheduling problem," *Applied Soft Computing Journal*, vol. 69, pp. 493–503, Aug. 2018, doi: 10.1016/j.asoc.2018.04.047.

[21] H. Dang Quoc, L. Nguyen The, C. Nguyen Doan, and T. Phan Thanh, "New cuckoo search algorithm for the resource constrained project scheduling problem," Oct. 2020, doi: 10.1109/RIVF48685.2020.9140728.

[22] K. Bibiks, Y. F. Hu, and J. P. Li, "Discrete flower pollination algorithm for resource constrained project scheduling problem," *International Journal of Computer Science and Information Security*, vol. 13, no. 7, pp. 8–19, 2015.

[23] Y. Wu, X. P. Wang, G. T. Li, and At. Lu, "Brain storm optimization algorithm based on adaptive inertial selection strategy for the RCPSP," in *Proceedings - 2019 Chinese Automation Congress (CAC)*, Nov. 2019, pp. 2610–2615, doi: 10.1109/CAC48633.2019.8996409.

[24] B. Roy and A. K. Sen, "A novel metaheuristic approach for resource constrained project scheduling problem," in *Advances in Intelligent Systems and Computing*, vol. 1154, Springer Singapore, 2020, pp. 535–544.

[25] R. Pellerin, N. Perrier, and F. Berthaut, "A survey of hybrid metaheuristics for the resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 280, no. 2, pp. 395–416, Jan. 2020, doi: 10.1016/j.ejor.2019.01.063.

[26] A. Baykasoğlu, A. Hamzadayi, and S. Y. Köse, "Testing the performance of teaching-learning based optimization (TLBO) algorithm on combinatorial problems: flow shop and job shop scheduling cases," *Information Sciences*, vol. 276, pp. 204–218, Aug. 2014, doi: 10.1016/j.ins.2014.02.056.

[27] X. Ji, H. Ye, J. Zhou, Y. Yin, and X. Shen, "An improved teaching-learning-based optimization algorithm and its application to a combinatorial optimization problem in foundry industry," *Applied Soft Computing Journal*, vol. 57, pp. 504–516, Aug. 2017, doi: 10.1016/j.asoc.2017.04.029.

[28] S. Saharan, J. S. Lather, and R. Radhakrishnan, "Combinatorial problem optimization using TLBO," in *4th IEEE International Conference on Signal Processing, Computing and Control (ISPCC)*, Sep. 2017, vol. 2017-Janua, pp. 559–563, doi: 10.1109/ISPCC.2017.8269741.

[29] J.-L. Kim and R. D. Ellis, "Comparing schedule generation schemes in resource-constrained project scheduling using elitist genetic algorithm," *Journal of Construction Engineering and Management*, vol. 136, no. 2, pp. 160–169, Feb. 2010, doi: 10.1061/(asce)0733-9364(2010)136:2(160).

[30] H. Y. Zheng and L. Wang, "An effective teaching-learning-based optimisation algorithm for RCPSP with ordinal interval numbers," *International Journal of Production Research*, vol. 53, no. 6, pp. 1777–1790, Sep. 2015, doi: 10.1080/00207543.2014.961205.

[31] H. yu Zheng, L. Wang, and X. long Zheng, "Teaching–learning-based optimization algorithm for multi-skill resource constrained project scheduling problem," *Soft Computing*, vol. 21, no. 6, pp. 1537–1548, Sep. 2017, doi: 10.1007/s00500-015-1866-3.

[32] D. Morillo, F. Barber, and M. A. Salido, "Chromosome Mutation vs. Gene Mutation in evolutive approaches for solving the resource-constrained project scheduling problem (RCPSP)," in *Lecture Notes in Computer Science (including subseries Lecture*

Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2018, vol. 10868 LNAI, pp. 601–612, doi: 10.1007/978-3-319-92058-0_58.

[33] S. Mirjalili, "Genetic algorithm," *Evolutionary Algorithms and Neural Networks. Studies in Computational Intelligence*, vol. 780, pp. 43–55, Jun. 2019, doi: 10.1007/978-3-319-93025-1_4.

[34] O. Shuvo, S. Golder, and M. R. Islam, "A hybrid metaheuristic method for solving resource constrained project scheduling problem," *Evolutionary Intelligence*, pp. 1–19, Nov. 2021, doi: 10.1007/s12065-021-00675-x.

BIOGRAPHIES OF AUTHORS

Bidisha Roy ⓘ 🔍 SC ↻ has done her ME in Computer Engineering University of Mumbai, India. She is currently working as Associate Professor in St. Francis Institute of Technology, India. She is pursuing PhD from University of Mumbai. Her research interests are algorithms and complexity, computational intelligence, optimization using evolutionary and metaheuristic techniques, data and web mining, semantic and social web and advanced internet technologies. She has published more than 22 papers in international conferences and journals. She can be contacted at email: Bidisha.bhaumik.roy@gmail.com.

Dr. Asim Kumar Sen ⓘ 🔍 SC ↻ received his B.E degree in Electrical Engineering from Indian Institute of Engineering Science and Technology (IIEST), formerly known as B.E.College, Sibpur, M.Tech Degree in Instrumentation Engineering and Ph.D degree in Industrial Engineering & Management (IE&M) from I.I.T, Kharagpur. He did his P.G.D.Ed.M from Narsee Monjee Institute of Management Studies (NMIMS), Mumbai. He has a work experience of 03 years in different divisions of industry such as project, design and erection & commissioning of instruments and system. He has been into teaching, training and research for the last 33 years in different institutes including N.I.T, Rourkela, MERI, Kolkata, TMI,Pune, TEC, Nerul, SFIT, Borivali (as Principal) and YTCEM (as Principal), Chandhai, Karjat under University of Mumbai. He also worked at OERC, Mumbai and HMI Kolkata as a Visiting Faculty member where he taught electrical, electronics, automation and management subjects related to maritime fields. At present he is working as a Visiting Faculty member at OERC, Mumbai and IME, Mumbai. He has around 106 publications (both Technical and Management related subjects) in National and International Journals and Conferences. He has co-authored some books related to computer engineering, electrical engineering, information technology and management. He has guided around 22 projects in B.E and M.E level, and one in Ph.D (Maritime Field) level. At present he is guiding 2 Ph.D. scholars. He can be contacted at email: asim_sen@linuxmail.org or asimsen1956@gmail.com.