An Annealed Genetic Algorithm for Multi Mode Resource Constrained Project Scheduling Problem

Vijay S Bilolikar
Research scholar
Indian Institute of Technology
Powai Mumbai India 400076

Karuna Jain
Professor
Indian Institute of Technology
Powai Mumbai India 400076

Mahesha R Sharma
Associate professor
Fr. C R college of Engineering
Mumbai India 400050

ABSTRACT

This paper presents a genetic algorithm (SA) for the classical multimode resource-constrained project scheduling problem (MRCPSP). The objective is makespan minimization of the project. A new precedence feasible multi point forward backward crossover operator is presented. SA and GA are used in tandem to ensure balance in exploration and exploitation of search space. SA is employed as local search procedure, due to its stochastic neighborhood selection strategy to escape local optima hence a good exploitation strategy; and GA as exploration strategy due to its large number of population. Computational investigations are carried out on standard data set problems from PSPLIB and it is proved that hybridization of GA with SA gives competitive performance when compared with currently available algorithms

General Terms

Project scheduling; Genetic Algorithm; Simulated Annealing.

Keywords

Resource constraints; multi point forward backward crossover.

1. INTRODUCTION

Project scheduling is concerned with the sequencing of the activities and allocation of resources over a period of time to perform activities. This domain of project management where the restriction on the availability of resources and precedence restriction on activities is called the resource constrained project scheduling problem RCPSP. In RCPSP a set of activities, interrelated by technological or precedence relation, are to be scheduled under limited availability of resources. RCPSP is traditionally considered as NP hard classical combinatorial optimization problem and is faced by construction industry, software industry, airplane and ship-building organizations to realize their goals. RCPSP is NP hard problem hence classical optimization techniques fail to give optimal solution within reasonable computational efforts and time, particularly, when the problem size increases. The purpose of the study is to develop a hybrid metaheuristic solution methodology based on genetic algorithm and simulated annealing for multi mode resource constrained project scheduling (MRCPSP). This involves defining start times and assignment of resources to activities, which are precedence related.

2. LITERATURE REVIEW

The complexity of the problem has drawn the attention of researchers to solve RCPSP. As a result, there are several solution methods suggested by researchers for models with varying degree of complexity and application of different constraints. The exact algorithm ([1]-[7]), like branch and bound, can find the optimal solution to small size problems in RCPSP. However the complexity that comes with increase in the problem size of RCPSP and the amount of computational efforts invested to tackle such complexity prohibits the use of exact methods therefore exact methods are limited to suggestion of bounds and fathoming rules. Therefore the use of heuristics or metaheuristic solution methodology is favoured. These algorithms may not achieve a global optimum but they are faster, requiring less efforts and more capable when the solution space is huge. Moreover, the suitability of using general purpose metaheuristic with ease to solve problems with multi modal space makes these techniques a natural choice and hence are extensively used to solve RCPSP.

The heuristic procedure use problem specific information and give feasible solutions. However the heuristic solution search gets trapped in local minima during search through solution space. So the solution provided by the heuristic procedure may be weak but feasible solution. Metaheuristic solution procedure has ability to drive the search out of the local optima. So the quality of the solution obtained by the metaheuristic solutions is superior. Therefore, use of metaheuristic, to address MRCPSP is on the rise and particularly, Genetic Algorithms has been extensively used to address MRCPSP. In genetic algorithm the focus has been on the development of efficient genetic operators which will affect the efficacy of search through solution space.

2.1 Metaheuristic approach.

The metaheuristic procedures like GA, SA, ACO, Tabu serach are extensively used to address MRCPSP. simulated annealing approach is used in [8] along with penalty for resource violation. Another implementation of simulated annealing is proposed by [9]. Tabu search is proposed by [10] while considering the schedule dependent set up times to take the problem closer to practical situations. [11] proposed a genetic algorithm for the MRCPSP without nonrenewable or doubly constrained resources. Two different genetic algorithms named pure and hybrid GA are proposed by [12]. They employed a parallel SGS with forward backward scheduling scheme for the construction of the schedule with adaptive crossover probabilities. [13] proposed a genetic algorithm with preprocessed data and suggested very effective single-pass or multi-pass local search based on the multi-mode left shift operation. A modified objective function is presented in GA by [14] and so-called two-point forward–backward crossover is proposed in which an offspring is build either from the front or from the end depending on the characteristics of parent’s gene. The effect of activity splitting due to resource vacation is studied by [15]. A hybrid metaheuristic algorithm based on scatter search and path relinking methods for resource tradeoff problem is proposed by [16]. [17] developed a two-phase genetic local search algorithm where the same genetic local search algorithm runs with different initial populations for both phases for different search purposes. A hybrid genetic algorithm (MMHGA) is proposed by [18] in which a new fitness function is proposed which rectifies the problem of different units by normalizing. A bi population version of GA is presented by [19] in which two different populations of the right justified schedules and left justified schedules are used. A new metaheuristic technique called particle swarm optimization (PSO) is applied to MRCPSP by [20] and is followed by [21]with two phase particle swarm optimization (PSO). In the first phase the modes of the activities are decided based on relative resource requirements by all modes and in the second stage local search is employed to find the suboptimal solution using the criteria for selection of activity based on the successor activity cardinality ratio. [22] proposed a four phase ant colony optimization algorithm which uses the feedback for the best solution for pheromone concentration on the links of specific paths in the construction graphs. A multi objective GA for software project planning is presented by [26]

In this paper, a genetic algorithm approach is introduced to address the multimode resource constrained project scheduling problem with makespan minimization as objective function. A multi point forward backward crossover operator is proposed and simulated annealing is employed as local search procedure. The remainder of the paper is organized as follows. Section 3 describes the multimode resource constrained project scheduling problem. Section 4 describes the genetic operators and simulated annealing local search procedure. Section 5 details the results of the computational experiments as well as the comparison with other heuristics. Finally, conclusion of the work is presented at the end.

3. PROBLEM DESCRIPTION

Resource constrained project scheduling problem can be described by considering a project which consists of J activities. These activities have precedence relation due to technological reasons. The set of all predecessors of activity j is represented by the set $\mathcal{P}_j$ and the set of all successor activities is represented by the set $\mathcal{S}_j$. Precedence relation requires that activity j cannot be started unless all of its predecessor activities ($i \in \mathcal{P}_j$) are finished processing. Additional activities j = 0 and j = J+1 representing the only source and the unique sink activity respectively of the network, are considered. Fig1 shows an example for the project network.

All the activities except source and sink activity need resources for processing. The activities may use renewable as well as non renewable resource. The set of renewable resources is denoted by set $\mathcal{K}^r = \{1,2, \ldots k\}$ and non renewable resources are denoted by $\mathcal{K}^{nr} = \{1,2, \ldots l\}$. Each activity j may be processed in more than one way, referred to as mode of the activity, depending upon the amount of resources it consumes. Activity j may be executed in $\mathcal{M}j$ modes given by the set $\mathcal{M}j = \{1,, . . . , m\}$. The processing time or duration of job j being performed in mode m ∈ Mj is given by $d_{jm.}$ It consumes r^r_{jmk} units of renewable resource of type $k \in \mathcal{K}^r$ during each period it is processed and uses r^{nr}_{jmk} amount of non renewable resource of type $k \in \mathcal{K}^{nr}$.

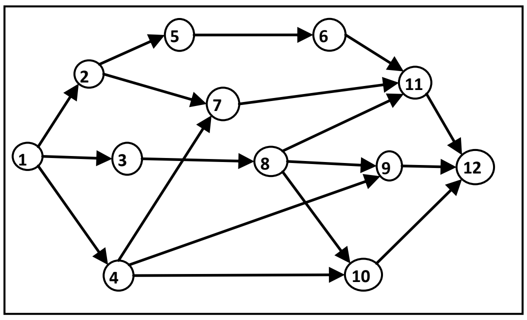

Figure 1 Project network example

For each renewable resource $k \in \mathcal{K}^r$ the per-period-availability is constant and given by a_{rk}. For each nonrenewable resource of type $k \in \mathcal{K}^{nr}$ the overall availability for the entire project is given by a_{nrk}. The activities once started may not be preempted and a mode once selected may not change, i.e., a job j once started in mode m has to be completed in mode m without interruption. The dummy source and the dummy sink activity have only one mode each with duration of zero periods and do not request any resource. The objective is to minimize the makespan of the project that meets the constraints imposed by the precedence relations and the limited resource availabilities. The 0-1 programming model [1] for the multi mode problem is modeled to determine, for every activity, the execution mode and its starting time. The model then can be written as follows

$$minimise \sum_{es_j}^{ls_j} t x_{jmt} \qquad (1)$$

Subject to

$$\sum_{m=1}^{M_j} \sum_{t=es_j}^{ls_j} x_{jmt} = 1 \quad j = 1,2,\ldots\ldots,J \qquad (2)$$

$$\sum_{m=1}^{M_j} \sum_{t=es_j}^{ls_j} (t + d_{jm}) x_{jmt} \le \sum_{m=1}^{M} \sum_{t=es_i}^{ls_i} t x_{imt} \quad (3)$$

$$i \in p_j$$

$$\sum_{j=1}^{J} \sum_{m=1}^{M_j} r^r_{jmk} \sum_{s=max\{t-d_{jm},es_j\}}^{\min\{t-1,ls_j\}} x_{jmt} \le a^r_k \quad (4)$$

$$k = 1,2,\ldots,k^r; t = 1,2,\ldots.,T.$$

$$\sum_{j=1}^{J} \sum_{m=1}^{M_j} r^{nr}_{jmk} \sum_{s=es_j}^{ls_j} x_{jmt} \le a^{nr}_k$$

$$k = 1,2,\ldots,k^{nr}; \qquad (5)$$

$$x_{jmt} = \begin{cases} 1, & if\ activity\ j\ is\ executed \\ & in\ mode\ m\ and\ started\ at\ time\ t \\ 0, & otherwise \end{cases} \quad (7)$$

$$j = 1,2,\ldots\ldots,J; m = 1,2,\ldots,M_j; t = es_j,\ldots,ls_j.$$

Where J = 1,2,….J. set of activities in the project. es_j and ls_j are earliest and late start times of the activity j. T is the feasible upper bound of the project duration.

The objective function (1) is to minimize the project duration. Equations (2) to (5) represent the constraints of the problem. Equation (2) assures that each activity is assigned exactly one mode and exactly one start time. Equation (3) represents the precedence constraints i.e. the start time of the j is always greater than or equal to the finish time of its predecessor activity i which belongs to predecessor set $\mathcal{P}$j. Equation (4) checks the per period renewable resource violation by the activity which are in progress at time t. Equation (5) represent the constraints on the nonrenewable resources. It ensures that total requirement of non renewable resources by all the activities does not exceed availability. Finally, Equation (6) imposes binary values on the decision variables.

4. ALGORITHM

4.1 Genetic algorithm

Genetic algorithm is nature inspired computational optimization technique which adapts natural evolution phenomenon of species for solving optimization problem. The problem data is captured in the form of chromosome code and a population of such chromosomes is subjected to the genetic operators like crossover and mutation for evolution purpose. The genetic operators transmit critical or desirable features to next generation during evolution process.

4.2 Solution representation

The problem characteristics are captured by a solution representation in the form of an activity list, mode list as shown in Fig 2. In activity list representation the solution is encoded as a precedence feasible list of the activities. Each activity can appear in the list in any position after all its predecessors.

Activity	J_1	J_2	-	-	-	J_n
Mode	m_1	m_2	-	-	-	M_j

Figure 2 Solution representation

4.3 Preprocessing

Prior to the start of the genetic algorithm, the data is processed to filter out unwanted and redundant data. Sprecher (1997) has proposed a data preprocessing procedure to remove redundant data i.e. inefficient, in-executable modes of activities and redundant non renewable resources from the problem input data. An inefficient mode is mode of an activity which consumes more resources for the same duration or vice versa than any other mode of the activity. An in-excutable mode consumes more renewable resources than available. A non renewable resource is redundant if its availability is more than the total maximum demand by activities. This preprocessing procedure reduces the search space.

4.4 Initial population

A highly diversified initial population is generated using different priority rules, as detailed in Table 1. The rule selection fairly captures activity time information, resource information and randomness. To guarantee a diversified population of solutions, biased random sampling method is used. In biased random sampling the probabilities for the activities to be selected for scheduling is a function of the priority value v_j of the activity and sum of the priority values of the other activities in the eligible set ES. If the objective of the priority rule is to select the activity with the highest priority value v_j then the probability of choosing activity j from the eligible set ES is given by equation (2)

$$p_j = \frac{v_j}{\sum_{j \in ES} v_j} \qquad (2)\,.$$

For the initial population the modes for the activities are randomly selected. Once the initial population has been generated, every individual is then send to serial schedule generation scheme which schedules i. e. assigns start times to activities considering the resource and precedence feasibility and generates the schedule for every solution.

Table 1. The priority rules used for the generation of initial population.

Minimum late finish time(LFT)	$\min\{t_j^{LF}\}$, where t_j^{LF} is the latest finish time of activity j from the eligible set.
Minimum slack (MINSLK)	$\min\{t_j^{LS} - t_j^{ES}\}$ where t_j^{ES} and t_j^{LS} is the early start and late start time resp. of the activity j from the eligible set
Shortest proc. time (SPT)	$\min\{p_j\}$, where $p_j$is the processing time of activity j from the eligible set.
Greatest resource utilization (GRU)	$\mathrm{Max}_j \left\{\sum_{k=1}^{K} {}^{r_{jk}}/_{a_k} \mid j \in J\right\}$ where r_{jk} & $a_k$are resource requirement and availablity of k type resource by the activity j
Greatest resource demand (GRD)	$P_j \sum_{k=1}^{K} r_{jk}$ where p_j is the processing time of the activity j and r_{jk} is the per period requirement of k typ resource by activity j
Random (RAN)	select the activity randomly.

4.5 Serial schedule generation scheme

The entire schedule is constructed in J stages and each stage corresponds to scheduling of one activity based on precedence and renewable resource feasibility. During scheduling, at every stage, the activity set is divided into two mutually exclusive sets of activities as scheduled set and unscheduled activities set. The set of scheduled activities is called partial schedule or scheduled set. The decision set, a subset of unscheduled activities set, is comprised of set of unscheduled activities whose predecessors are in scheduled set. At every stage, activities from decision set are given priorities using priority rules. The activity with highest priority is removed from decision set and transferred to partial schedule by scheduling it at its earliest precedence and resource feasible start time. The ties are arbitrarily broken. The algorithm terminates once all activities are transferred to scheduled set. The finish time of terminal activity is makespan of the schedule.

4.6 Fitness computation

The schedule makespan is indirectly used as fitness score of every chromosome in the population. By relaxing non renewable resource constraint, solutions with infeasibility with respect non renewable resources, are allowed to exist in the initial population and in subsequent generations so that high quality genes from such schedules are captured during the crossover and mutation process for fitter offspring. Such infeasible individuals are penalized in fitness function. The fitness function proposed by [18] as given by equation (2) is used for fitness computation. Fitness is computed for each individual depending on whether an individual i is feasible or infeasible.

$$f(x) = \begin{cases} \mathbf{1} - \frac{maxmak(p) - mak(i)}{maxmak(p)} & \textbf{\textit{if feasible}} \\ \mathbf{1} + \frac{mak(i) - mincp}{mak(i)} + \sum_{h \in NR}^{H} \max\left\{\mathbf{0}, \frac{\sum_{j=1}^{J} r_{jmh}^{nr} - NR_h}{NR_h}\right\}, & \textbf{\textit{otherwise}} \end{cases} \quad (2)$$

$maxmak(p)$ is makespan of the worst schedule. $mak(i)$ is makespan of the schedule i and $mincp$ is makespan of the best schedule in current generation. The fitness function value of a feasible individual is always less than one and that of infeasible individuals will have a fitness value greater than one.

4.7 Cross over

The rank selection strategy is used for selecting chromosomes for crossover operation. Crossover is one of the most important genetic operators, and the performance of the algorithm greatly depends on how the crossover operator has been designed since features of parent chromosomes are captured and inherited to offspring during crossover operation. The precedence feasible multi point forward backward crossover method is used for crossover operation and Fig 3 describes the procedure for generation of son and daughter for the example network shown in Fig 1.

1	3	8	2	4	10	5	9	6	7	11	12
1	1	3	1	2	2	2	3	3	1	1	1

Father

1	2	5	3	6	8	4	10	7	9	11	12
1	2	1	1	3	2	2	1	3	3	1	1

Mother

1	3	8	2	4	5	6	10	7	9	11	12
1	1	3	1	2	2	3	1	3	3	1	1

Son

1	2	5	3	8	4	10	9	6	7	11	12
1	2	1	1	2	2	2	3	3	1	1	1

Daughter

Figure 3 Multipoint forward backward Crossover

For the RCPSP the cross over probability suggested in literature is in the range 0.7 to 0.9 depending on other genetic operators. Present study considered the fixed crossover probability as 0.7, 0.8 and 0.9. Fig 4 describes pseudo algorithm for the precedence feasible multipoint forward backward crossover.

4.8 Mutation

Mutation introduces new genetic structures in the population by randomly modifying some of its building blocks. Mutation helps escape from local minima's trap. The mutation technique used here chooses one schedule randomly and an activity from the schedule is selected randomly. Mutation is carried out by activity swapping or shift or insertion. As shown in Fig 5 for the example network shown in Fig 1, randomly select one schedule or solution and select randomly one activity.

```
Generation of son
  $for\ i = 1\ to\ n\ do$
  $J_i^s = J_i^f$
  $\mathrm{m}_{\mathrm{im_k}}^{\mathrm{s}} = \mathrm{m}_{\mathrm{im_k}}^{\mathrm{f}}$ such that $l = lowest\ index\ / 1 \le l \le J\ j_l^f \notin \{j_1^s, \ldots\ldots, j_n^s\}$
  $J_{n-(i-1)}^s = J_{n-(i-1)}^m$
  $\mathrm{m}_{(n-(i-1))\mathrm{m_k}}^{\mathrm{s}} = \mathrm{m}_{(n-(i-1))\mathrm{m_k}}^{\mathrm{m}}$ such that $l = highest\ index\ / 1 \le l \le J\ j_l^m \notin \{j_1^s, \ldots\ldots, j_n^s\}$
Generation of daughter
  $for\ i = 1\ to\ n\ do$
  $J_i^d = J_i^m$
  $\mathrm{m}_{\mathrm{im_k}}^{\mathrm{d}} = \mathrm{m}_{\mathrm{im_k}}^{\mathrm{m}}$ such that $l = lowest\ index\ / 1 \le l \le J\ j_l^m \notin \{j_1^d, \ldots\ldots, j_n^d\}$
  $J_{n-(i-1)}^d = J_{n-(i-1)}^f$
  $\mathrm{m}_{(n-(i-1))\mathrm{m_k}}^{\mathrm{d}} = \mathrm{m}_{(n-(i-1))\mathrm{m_k}}^{\mathrm{f}}$ such that $l = highest\ index\ / 1 \le l \le J\ j_l^f \notin \{j_1^d, \ldots\ldots, j_n^d\}$
```

Figure 4 Pseudo code for Multi point forward backward crossover

The selected activity 7 has predecessor set $P_7 = \{2,4\}$ and successor activity set $S_7 = \{11\}$. The nearest immediate predecessor activity 4 in sequence list is limiting case for the left direction and immediate successor 11 is limiting case for right direction for the activity 7 to shift. Within these limiting positions on the sequence list, activities 9, 7, 6, 10 lie. Therefore activity 7 can be swapped with any of the activities 9,7,6,10 without violating precedence constraint. If fitness function improves retain the change else take next activity or else change the schedule. The probabilities for mutation are fixed and are set to 0.03 and 0.05

1	2	5	3	8	4	9	7	6	10	11	12
1	3	3	3	1	2	3	2	3	2	3	1

1	2	5	3	8	4	9	10	6	7	11	12
1	3	3	3	1	2	3	2	3	2	3	1

Figure 5 Mutation by activity swapping

4.9 Termination and performance evaluation criteria

The termination criteria used here is in terms of number of schedules as 1000 and 5000 no of schedules generated and the performance of the algorithm is evaluated using percentage average deviation from the optimal solution.

$$\%\ Average\ deviation = \frac{(makespan_i - optimal\ makespan_i) * 100}{(optimal\ makespan_i)}$$

4.10 Simulated annealing as local search.

Simulated Annealing (SA) is motivated by physical annealing process of solids in which cooling of solid is carried out in controlled manner to manipulate physical properties of metals. It was first introduced by [23] later [24] applied it to optimisation problems. The algorithm simulates the cooling process by lowering the temperature in steps. With every temperature reduction algorithm performs a predefined number of neighbourhood moves. Simulated annealing accepts an inferior neighbourhood move probabilistically. The decision parameters are defined by cooling schedule which includes temperature settings and iterations at each temperature. For the present study the value of starting temperature is chosen as the worst fitness function value of generation of GA and the geometric temperature decrement with decrement coefficient or attenuation factor 'k' equal to 0.9 as given in equation (4), is used.

$$T_{(i+1)} = k\, T_{(i)} \qquad (4)$$

Where $T_{(i)}$ is temperature at iteration i and temperature $T_{(i+1)}$ at iteration (i+1). The number of iterations is dynamically changed as algorithm progresses so that at lower temperatures large number of iteration is done to intensify the search at local optimum. The increase in the number of iterations, with every temperature decrement, is equal to the number of activities in the project.

4.11 Neighbourhood moves

The schedules from every GA generation are provided to simulated annealing local search procedure. The local search around these solutions is done in SA by generating neighbouring solutions. These nneighboring solutions or moves are generated from the current solution by activity insertion and mode change simultaneously.

Select an activity j at random from the schedule, locate the nearest positions of immediate predecessor and successor of activity j. Select a location randomly within these two positions on the chromosome. Insert activity J at that position shifting activities to the right maintaining precedence feasibility. Change mode m_j of that activity to the next mode m_{j+1}. If mode of the activity is shortest duration mode M_j; change it to longest duration mode m_1.

For a move if change in the objective function is Δ; probability of acceptance p is $p = e^{(-\Delta/T)}$ where T is the temperature. The acceptance probability is compared to a randomly generated number $r \in [0, 1]$. Whenever $P > r$ the move is accepted. Hence with reduction in temperature the probability of accepting the move goes on reducing. In order to reduce the solution search space, only feasible solutions are supplied to the simulated annealing search procedure.

5. COMPUTATIONAL EXPERIMENT

The experiments have been performed on Intel Pentium desktop machine with frequency of 2.60GHz and 512 MB RAM. The GA has been coded in C++ compiled with Microsoft Visual C++ v.6.0 compiler and tested under Linux. A set of standard test problems, systematically constructed by the project generator ProGen which has been developed by [25], are used to test the performance of algorithm. They are available in the project scheduling problem library PSPLIB from the University of Kiel. Detailed information is available at www.psplib.com

The standard multi-mode problem data sets containing instances with 10, 12, 14, 16, 18, 20, and 30 non-dummy activities are used to test the performance of algorithm. Each of the non-dummy activities may be performed in one out of three modes. The duration of a mode varies between 1 and 10 periods due to usage of two renewable and two nonrenewable resources. For each problem size, a set of instances was generated by systematically varying four parameters, that is, the resource factor and the resource strength of each resource category. The resource factor is a measure of the average portion of resources requested per job. The resource strength reflects the scarceness of the resources. For each project size, 640 instances are available and Table 2 shows the number of instances for which feasible solutions are available.

Table 2 PSPLIB instances

Number of activities	10	12	14	16	18	20	30
Total Number of instances	640	640	640	640	640	640	640
Number of instances with feasible solution	536	547	551	550	552	554	552

5.1 Configuration of algorithm.

For the numerical investigation, the best configuration for GA parameter is arrived at after experimentation. The population size is an important parameter to be determined as it significantly impacts the solution quality when combined with number of generations. It is observed that as the population size increases the performance of the algorithm improves for a particular generation number. It is contemplated that by keeping population size larger, for the crossover operator, a huge pool of very diverse combinations are possible for the algorithm to further the search. It is also observed that the algorithm performance is superior when crossover probability is set to 0.7 and mutational probability is 0.03.

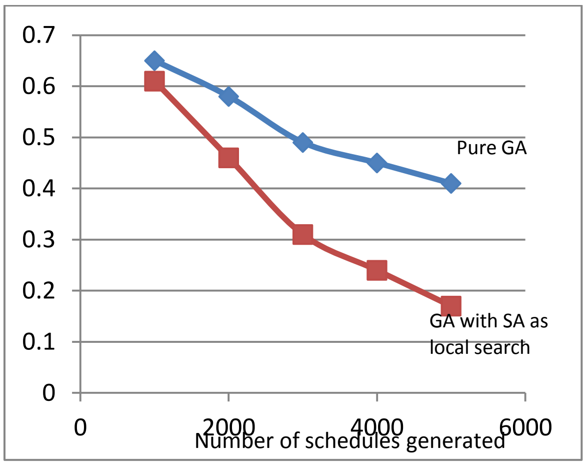

Figure 6 % Average deviation for J10 dataset

The simulative experiments are carried out with standard J10 data set which contains total 640 instances out of which only 536 instances are optimally solved and their optimum values are available. As stated earlier fitness function by [18] is used in conjunction with proposed crossover, mutation strategies and local search by simulated annealing. The experimentation was done for pure GA and GA with simulated annealing as local search procedure. The results of the experimentation are depicted in the Fig 6 and 7.

From the results it is clear that multi point forward backward crossover with simulated annealing as local search performs better on all the criteria when compared with pure GA. This clearly shows that exploitation ability of pure GA is weak, particularly for the problems with multimodal solution space like MRCPSP and it is enhanced when it is hybridized with simulated annealing indicating that there is now balance in exploration and exploitation abilities of the algorithm. The percentage of optimal solutions found is greater right from the beginning of the algorithm and continues to improve at a faster rate.

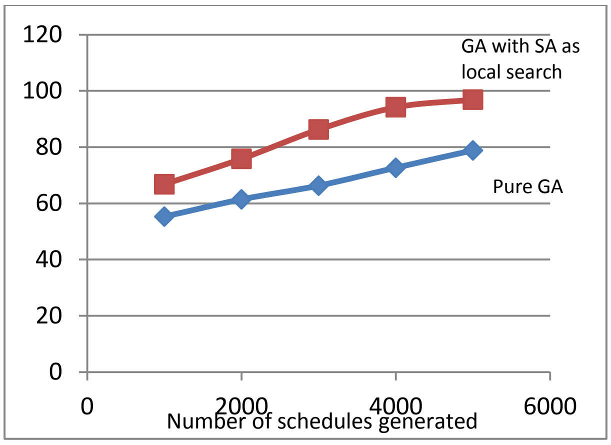

Figure 7 % Optima found for J10 dataset

Similarly the percentage average deviation values are quite competitive. The higher percentage of optimal solutions signifies that algorithm is able to come out of local optima because of SA local search i.e. exploitation of search space around optima has been effective. Low values of deviation indicated that search trajectory is passing through nearby region of optimal solutions

Table 3 Results with final experimental set up for all datasets

Data set	% optima found	% avg deviation	Avg proc time sec.
J10	97.39	0.16	0.12
J12	93.95	0.31	0.46
J14	83.48	0.63	0.62
J16	82.36	0.49	0.77
J18	74.23	0.52	0.95
J20	68.53	1.04	1.12
J30	48.91	11.86	-

The other datasets available on psplibrary i e J12, J14, J16, J18, J20, J30 datasets are solved with proposed algorithm. The results for the experimentation for different databases are shown in the Table 3 and compared our results with other heuristic presented in the literature in Table 4. Our algorithm gives competitive results when compared to existing algorithms particularly when the problem size increases. The clone solutions are encountered and are removed to save on to computational wastage and to improve genetic variety of the population

Table 4 Comparison with other heuristics

Data set	[8]	[13]	[14]	[18]	Our study
J10	1.16	0.06	0.24	0.06	0.16
J12	1.73	0.14	0.73	0.17	0.31
J14	2.6	0.44	1.00	0.32	0.63
J16	4.07	0.59	1.12	0.44	0.49
J18	5.52	0.99	1.43	0.63	0.52
J20	6.74	1.21	1.91	0.87	1.04
J30				16.65	11.86

.

6. CONCLUSION

In this paper a novel approach of hybridizing simulated annealing and genetic algorithm is proposed to solve multi mode resource constrained project scheduling problem. Paper also presents a new precedence feasible multi point forward backward crossover operator technique and implemented genetic algorithm with this crossover technique to solve NP hard MRCPSP. The hybridization GA with SA is done in order to have balance in exploration and exploitation capabilities of the algorithm which is proved through the computational experiments. Moreover, different priority rules are used for initial population generation using biased sampling. Performance of algorithm is tested for different crossover probabilities and mutation probabilities in order to determine the most appropriate configuration. Computational experiments confirm competitive performance of proposed algorithm when compared with other heuristics.

7. ACKNOWLEDGMENTS

Authors express their sincere thanks to unknown reviewers and experts for their valuable reviews and inputs for refinement of this work.

8. REFERENCES

[1] Talbot, F.B., 1982. Resource-constrained project scheduling with time-resource trade-offs: the nonpreemptive case. Management Science 28 (10), 1197–1210

[2] Patterson, J.H., Słowin´ ski, R., Talbot, F.B., We˛glarz, J., 1989. An algorithm for a general class of precedence and resource constrained scheduling problems. In: Słowin´ ski, R., Weglarz, J. (Eds.), Advances in Project Scheduling. Elsevier, Amsterdam, pp. 3–28.

[3] Sprecher, A., 1994. Resource-constrained project scheduling: exact methods for the multi-mode case. Springer, Berlin.

[4] Sprecher, A., Hartmann, S., Drexl, A., 1997. An exact algorithm for project scheduling with multiple modes. OR Spektrum 19 (3), 195–203.

[5] Hartmann, S., Drexl, A., 1998. Project scheduling with multiple modes: a comparison of exact algorithms. Networks 32 (4), 283–297.

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

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

[8] Józefowska, J., Mika, M., Rozycki, R., Waligóra, G., We˛glarz, J., 2001. Simulated annealing for multi-mode resource-constrained project scheduling. Annals of Operations Research 102 (1–4), 137–155.

[9] Bouleimen, K., Lecocq, H., 2003. A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version. European Journal of Operational Research 149 (2), 268–281.

[10] Mika, M., Waligóra, G., Wȩglarz, J., 2008. Tabu search for multi-mode resource constrained project scheduling with schedule-dependent setup times. European Journal of Operational Research 187 (3), 1238–1250.

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

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

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

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

[15] Buddhakulsomsiri, J., Kim, D.S., 2006. Properties of multi-mode resource constrained project scheduling problems with resource vacations and activity splitting. European Journal of Operational Research 175 (1), 279–295.

[16] Ranjbar, M., De Reyck, B., Kianfar, F., 2009. A hybrid scatter search for the discrete time/resource trade-off problem in project scheduling. European Journal of Operational Research 193 (1), 35–48.

[17] Tseng, L.Y., Chen, S.C., 2009. Two-phase genetic local search algorithm for the multimode resource-constrained project scheduling problem. IEEE Transactions on Evolutionary Computation 13 (4), 848–857.

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

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

[20] Zhang, H., Tam, C.M., Li, H., 2006. Multi-mode project scheduling based on particle swarm optimization. Computer-Aided Civil and Infrastructure Engineering 21 (2), 93–103

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

[22] Chiang, C.W., Huang, Y.Q., Wang, W.Y., 2008. Ant colony optimization with parameter adaptation for multi-mode resource-constrained project scheduling. Journal of Intelligent and Fuzzy Systems 29 (4–5), 345–358.

[23] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E. 1953. Equation of State Calculation by Fast Computing Machines. J. of Chem. Phys., 21, 1087-1091.

[24] Kirkpatrick, S , Gelatt, C.D., Vecchi, M.P. 1983. Optimization by Simulated Annealing. Science, vol 220, No. 4598, pp671-680

[25] Kolisch, R., Sprecher, A., 1997. PSPLIB – A project scheduling problem library. European Journal of Operational Research 96 (1), 205–216.

[26] D. Sundar, B. Umadevi, K. Alagarasamy, "Multi Objective Genetic Algorithm for optimized Resources usage and the Prioritization of the Constraints in the Software Project Planning", *International Journal of Computer Applications*, vol. 3, 2010, 0975-8887.

.