International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2015): 6.391

Genetic Algorithm for Resource Constrained Project Scheduling

Vinayak C. Sawant

Mumbai University, Department of Mechanical Engineering, FrCRCE, Bandra, Mumbai, India

Abstract: ***Resource Constrained Project Scheduling (RCPS) can be defined as the project scheduling with limited availability of resources to achieve goals such as minimization of makespan and maximization of Net Present Value (NPV). In this paper we have used Genetic algorithm (GA) to solve RCPS problem to minimize the makespan. By modifying the classical approach using GA, we solved the standard scheduling problems available and compared the results to previous researcher's result and it shows that algorithm gives the optimal schedules and can be used for variable conditions of the resource usage in the problem.***

Keywords: Resource Constrained Project Scheduling Problem (RCPSP), Genetic Algorithm (GA), Optimal Schedules

1. Introduction

Project scheduling involves sequencing, scheduling and execution of certain set of tasks, called as activities, in predefined order on timeline. Scheduling involves the decisions regarding selecting the job for execution at a particular time point whereas the sequencing involves the order in which different tasks or activities are performed. Project scheduling problems ideally specifies the minimization of the make span as primary objective but in reality or practically projects are subjected to multiple and often conflicting objectives because of the various constraints on precedence, resources and resource types. This leads to resource constrained project scheduling problem and the conflicting objectives can be cost minimization, resource utilization, resource efficiency etc

Project scheduling is all about accomplish activities in sequential order and allocate resources over time to perform activities and this part of project management where the available resources and there precedence are restricted is called the resource constrained project scheduling problem (RCPSP). Resource constraint project scheduling problem is a subset of project scheduling in which the constraints are added. In RCPSP a set of activities are to be executed which are interrelated by precedence relationship and the resources available are limited to execute the tasks. These activities are to be performed with these constraints in the effective manner so that the final objective is realized. The goal of a Resource Constrained Project Scheduling (RCPSP) is to find a feasible and possibly optimal schedule for a project. RCPSP involves assigning and managing the resources which has related precision, defining their start and end times to complete the task. Task becomes difficult whenever multiple activities compete for the same resource which has limited quantity at the same time. Resource constrained project scheduling is NP hard combinatorial optimization problem, so exact methods are infeasible to find optimal solution. The heuristic methods helps to find good feasible solution but do not guarantee optimal one. The meta-heuristic methods gives even better feasible solutions, therefore meta-heuristic methods are inevitable to solve this hard problem.

2. Literature

Kolisch and Drexl (1997) presented a heuristic which represents solution with a mode assignment list and a list of activity completion times [1]. Initially the mode assignment list is decided based on the feasibility of non-renewable resource. With this mode assignment a priority rule is applied to generate a solution which is then perturbed for improved mode assignment. Now this improved or so called best mode assigned schedules are subjected to iteration called as intensification for improved objective function. They considered two renewable, two renewable resources, and three modes per activity.

Mori and Tseng (1997) propose a genetic algorithm for the MRCPSP without nonrenewable or doubly constrained resources [3]. A direct representation of a schedule is used. This representation contains information about the activity, the assigned mode, the priority, and the calculated start and completion times of this activity. The initial population is built by setting activities in an ascending order, randomly choosing a mode for each activity. The priority is determined randomly for activity order interval, and start and completion times of each activity are calculated. Two parent chromosomes are used in the crossover operation. One of the parents is chosen randomly from the current population while the second parent is always the solution with the smallest project duration from among all solutions in the current population. The crossover point is chosen randomly, and the offspring solution inherits the head from the second parent and the tail is constructed using the remaining activities from the first parent. Two mutation operators are used. In the first one, a set of activities is chosen and then a mode is randomly chosen for each selected activity. In the second one, a new solution is constructed using the same method as in the initialization phase. A new generation is built of the 20 best solutions from the previous generation, 10 solutions generated using a crossover operator, 7 solutions generated using a mutation operator, and 3 solutions generated randomly.

Özdamar (1999) employed pure and hybrid GA to address the problem [4]. For hybrid GA the solution presentation consists of two lists for mode assignment and the other one is for priority rule indicating the one used for scheduling at that position. A parallel SGS with forward backward scheduling

scheme is used for the construction of the schedule. Two point and uniform crossover together with mutation operator randomly changing mode and priority rule are used for next generation so as to search solution space. Adaptive crossover probabilities are used. They have used two different genetic algorithms. In pure GA instead of priority rule list an activity list is used with Serial SGS. Two-point linear crossover and A mutation operator randomly changes one position in the mode assignment list and pair wise swaps precedence feasible two activities from the activity list, and mode change should feasible with respect to non-renewable resources. 3 modes per activity, two renewable and two nonrenewable resources are considered

Hartmann (2001) proposes a genetic algorithm. Before executing it, preprocessing is applied in order to reduce the search space by adapting the project data [5]. A solution is represented by a precedence-feasible list of activities and a mode assignment list. Such a representation allows generation of schedules infeasible with respect to nonrenewable resource constraints. In such a case, a penalty function is used to calculate fitness of an infeasible solution. The next population is generated using one point crossover (two different cut points are used: one for the activity list and another one for the mode assignment list), and a mutation that swaps two adjacent activities on the activity list (if it is precedence-feasible) and changes randomly a mode on the mode assignment list. The ranking method is used as a selection operator. A schedule is constructed applying the serial SGS, and a single- pass or multi-pass local search based on the multi-mode left shift operation is performed on this schedule. Next, if the schedule is improved, an encoding rule is applied to reversely transform the schedule into activity and mode assignment lists. This operation, called inheritance, unfortunately does not significantly improve the obtained results. The author also shows that repetition approach and island model of GA, as well as other selection operators, do not improve the performance of the proposed approach.

Józefowska et al. (2001) employed a simulated annealing approach to the multimode resource constrained project scheduling [2]. Solution representation is based on the precedence feasible list of activities and a mode assignment. Here SA with and without penalty function is employed. The penalty function takes care of the violation of non-renewable resources which is incorporated in the objective function. Three neighborhood generation or local search mechanisms were applied i.e. random neighborhood shift, mode change and composition of two. 3 modes per activity, two renewable and nonrenewable resources are used. They suggest that SA with penalty function performs better.

Lova et al. (2009) proposed a hybrid genetic algorithm (MMHGA) [8]. A preprocessing procedure as suggested by Sprecher et al. (1997) runs in order to reduce the search space. Solution representation uses an activity list and a mode assignment list, and two additional genes namely: forward/ backward and serial/parallel SGS gene adapted from (Alcaraz et al., 2003) [6] and serial/parallel gene denoting the type of Schedule generation scheme used to build the schedule. A new fitness function is proposed which rectifies the problem of different units by normalizing. It is showed that applying this function in the proposed approach improves the performance compared with two other fitness functions proposed by Hartmann (2001) [5] and by Alcaraz et al. (2003) [6], respectively. Two-point crossover as well as mutation operators are used to obtain the next generation. Mutation is applied to both the activity list and the mode assignment list, as well as to the two additional genes. On the activity list a random shift is performed. Mutation applied on the mode assignment list depends on the feasibility of the solution with respect to nonrenewable resources. For a feasible solution modes are changed randomly with a given probability that is the same for all activities and for infeasible solutions a mode is randomly changed for a randomly chosen activity until either a resource feasibility is attained or all modes are exhausted then next activity is chosen and the procedure is repeated. The forward/ backward and serial/parallel genes are randomly changed with a given mutation probability. A 2-tournament selection operator with elitism is applied to generate the next population. Moreover, two additional mechanisms are used: a random replacement of some solutions from the current population, and a multi-mode forward–backward improvement.

Tseng and Chen (2009) develop 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 [7]. In the first phase, the initial population is generated randomly, and the set of good solutions (so-called elite set) is searched. In the second phase, the initial population contains mainly solutions from the elite set and the purpose of this phase is to search more thoroughly within the regions located by the solutions from the elite set. Similarly to other approaches, preprocessing is executed before the start of the main procedure. A single solution is represented by an activity list and a mode assignment list. Fitness function is calculated in the same way as in Alcaraz et al. (2003), and the proposed forward–backward local search method is used to transform a given solution to the standard representation. After this transformation each schedule has exactly one solution representation. Neighbor solutions are generated using two-point crossover proposed by Alcaraz et al. (2003) and its slightly modified version, as well as two mutation operators which allow to diversify population lightly or heavily, respectively. The first mutation operator is taken from Alcaraz et al. (2003), whereas the second one is a new concept developed by the authors. Selection is made using ranking and 2-tournament methods.

Alcaraz et al. (2003) proposed genetic algorithm in which the modified objective function is proposed [6]. The make span objective includes the penalty function for the infeasible solutions which take part in reproduction. The penalty function accommodates the violation of the nonrenewable resource constraint is and also the increase over of the make span from the critical path length calculated for the shortest duration modes and maximum feasible make span from the current population. This ensures the possibility of infeasible solution in crossover operations. The algorithm starts with preprocessing to exclude infeasible and inefficient modes and redundant nonrenewable resources. The solution is encoded using an activity list, a mode assignment list, and an additional bit (forward/backward gene) denoting the

scheduling generation scheme used to build the schedule: serial forward or backward. A so-called two-point forward–backward crossover is proposed in which an offspring is build either from the head or from the tail depending on the value of the parent's forward/backward gene. The mutation operator consists of two-phases: in the first one activities are reordered in the activity list with a given probability in the way that the activity list after this operation is still precedence-feasible, and in the second phase, modes from the mode assignment list are changed with a given probability. The forward/ backward gene may be changed in the first phase. Finally, a random replacement procedure is applied that replaces with a given probability solutions from the current population with other solutions generated randomly.

3. Problem Statement

The resource constrained project scheduling problem (RCPSP) can be given as follows.

3.1 Basic Model of Single mode RCPSP:

A single project consists of a set J = {0, 1, n,n+1} of activities which have to be processed with task times as dj where j = 0, 1,2 . . . n+1.

The activities j=0 and j = n+1 correspond to the project start and to the project end, respectively and they do not consume resources and their task times dj are zero units. The activities are interrelated by two kinds of constraints. First, precedence constraints in which activity j cannot start unless its immediate predecessor activity i, has not started or finished. The set of all predecessor activities of j can be represented by P_j. And the set of all successor activities of activity j can be represented by set S_j.

Second, the activities are constrained by resource. Execution of the activities requires resources which are limited in capacities. Suppose from the K resource types, given by the set K = {1,. . . ,k} activity j requires r_{jk} units amount of resource during every period of its non–preemptiable duration d_j. Resource type k has a limited capacity of R_k at any point in time. The parameters dj, rjk, and R_k are assumed to be deterministic. For the project start and end activities $d_j = 0$ and $r_{jk} = 0$ for all k ε K.

The objective of the RCPSP is to find precedence and resource feasible completion times for all activities such that the total make span of the project is minimized.

Let FTj denote the finish time of activity j. A schedule S is given by a vector of finish times (FT_1, FT2, . . . , FTn). Let A (t) = {j ϵ J | $FT_j - d_j < t < FT_j$} be the set of activities which are being processed (active) at time instant t.

$$MinFTn+1 \quad \ldots\ldots (1)$$
$$FTi < Fj - dj \qquad j = 1. \ldots, n+1; I \varepsilon Pj \ \ldots (2)$$
$$\sum_{j \varepsilon A(t)} r_{jk} = R_k \quad for\ k \varepsilon K, t > 0 \ \ldots\ldots (3)$$
$$FTj > 0 \qquad j = 1, 2 \ldots n+1 \ \ldots\ldots\ldots (4)$$

The objective function (1) minimizes the finish time of the project end activity and thus the make span of the project. Constraints (2) enforce the precedence constraints between activities, and constraints (3) limit for each resource type k and each time instant t that the resource demand of the activities which are currently processed does not exceed the capacity. Finally, (4) define the decision variables. Equations (1) to (4) is a conceptual model. SetA (t) is the set of activities which are active at the instant t.

3.2 Basic Model of Multimode RCPSP

Talbot (1982) has introduced 0-1 programming model for the multi-mode problem. We need to determine the execution mode and its starting time. This is expressed by a decision variable

x_{jmt} = 1, if activity j is executed in mode m and started at time t
x_{jmt} = 0, otherwise
The model then can be written as follows

$$\text{Minimize} \sum_{es_j}^{ls_j} t x_{jmt} \quad \ldots\ldots\ldots\ldots (5)$$

Subject to

$$\sum_{m=1}^{M_j} \sum_{t=es_j}^{ls_j} x_{jmt} = 1 \quad j = 1,2, \ldots\ldots, J \qquad (6)$$

$$\sum_{m=1}^{M_j} \sum_{t=es_j}^{ls_j} (t + d_{jm}) x_{jmt} \leq \sum_{m=1}^{M} \sum_{t=es_i}^{ls_i} t x_{imt} \qquad i \in p_j \qquad (7)$$

$$\sum_{j=1}^{J} \sum_{m=1}^{M_j} r_{jmk}^{\rho} \sum_{s=max\{t-d_{jm}\ ,es_j\}}^{\min\{t-1,ls_j\}} x_{jmt} \leq a_k^{\rho} \quad k = 1,2, \ldots, k^{\rho}; t = 1,2, \ldots, T. \qquad (8)$$

$$\sum_{j=1}^{J} \sum_{m=1}^{M_j} r_{jmk}^{v} \sum_{s=es_j}^{ls_j} x_{jmt} \leq a_k^{v} \quad k = 1,2, \ldots, k^{v}; \qquad (9)$$

$$x_{jmt} = \{0,1\} \quad j = 1,2, \ldots\ldots, J; m = 1,2, \ldots, M_j; t = es_j, \ldots, ls_j. \qquad (10)$$

Where,
J = 1,2,. . . J. set of activities in the project.
m is the mode of activity j.
e_{sj} and l_{sj} are earliest and late start times of the activity j.
d_{jm} is the duration of the activity j when executed in mode m.
P_j is the set of immediate predecessors of the activity j.
r_{jmk}^{ρ} The requirement of the activity j for renewable resource type k= 1,2,. . .K_
r_{jmk}^{v} The requirement of the activity j for nonrenewable resource type k= 1,2,. . .K_
a_k^{ρ} Constant availability of renewable resource type k= 1,2,….K^{ρ}
a_k^{v} Availability of non-renewable resource type k.
T is the feasible upper bound of the project duration.

The objective function (5) is to minimize the project duration. It is assumed that the dummy start node and dummy end node can only be processed in a single mode with duration equal to zero. Equations (6) to (9) represents the constraints of the problem. Equation (6) assure that each

activity is assigned exactly one mode and exactly one start time. Equation (7) 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 p of j. equation (8) checks the per period renewable resource violation by the activity which are in progress at time t. equation (9) represent the constraints on the nonrenewable resources. It ensures that total requirement of nonrenewable resources by all the activities is less than or equal to available. Finally, Equation (10) impose binary values on the decision variables.

4. Methodology

As we all know RCPS problem is NP hard combinatorial optimization problem, so exact methods are infeasible to find optimal solution. The heuristic methods helps to find good feasible solution but do not guarantee optimal one. The meta-heuristic methods gives even better feasible solutions, therefore so we are using Genetic Algorithm (GA) to solve this hard problem.

The resource constraint project scheduling problem is to solve by developed genetic algorithm with objective as make span. We will be using the model formulation by Talbot (1982) as given in section 3.2 on this page.

Before the start of the genetic algorithm, we apply a procedure of preprocessing data over the project data which is proposed by Sprecher and later used by Hartmann. In this procedure, from the input data of the problem the modes of the activities and some non-renewable resources are filtered. The mode is considered incapable which take more resources but the activity time is same or more than any other mode for the same activity. These incapable modes are removed from the problem data. Likewise those modes which consume more non-renewable resources than available are in executable modes that is they will surely violate resources are eliminated from the data. Finally the non-renewable resources, for which the resource violation will not take place due to any combination of activities with any of their mode combinations, which are redundant are excluded from the problem input data. This procedure of preprocessing data reduces the volume of the data and the search space leads to fast execution of the algorithm.

After applying the pre-processing data procedure, the genetic algorithm starts execution. Below is the flow chart shown in the fig 1 gives the steps to be followed.

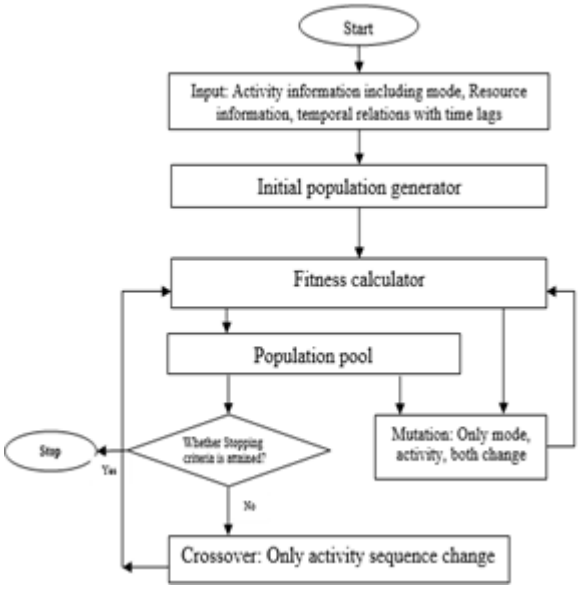

Figure 1: Flowchart of Genetic Algorithm

4.1 Fitness Function

It must be evaluated, once the initial population has been generated, that is, according to how good a solution to the problem is, each individual is assigned a fitness score or fitness value. The make span of the schedule related to an individual is a good measure of how good that individual is, since our objective is to minimize the project total duration. However, we must take into account that in the initial population, and subsequent generations, individuals which are not feasible with respect to non-renewable resource constraints may survive. This is because of certain combinations of mode for the activities that may result in excess demand for the non-renewable resource than actually available. In respect of non-renewable resources such individual solutions then become infeasible. But they may contain high quality genes which can be rather should be captured during the crossover and mutation process so that the better qualities can be transferred to the next generation. For this reason they should be allowed in the population. Although they do not represent a feasible solution, such individuals must be also assigned a value, indicating their fitness. Therefore, we must define a function that assigns these individuals a fitness value. Moreover, the fitness value of infeasible individuals should notbe better than that of the feasible ones, because they will displace the feasible schedules in the population, so they must be penalized. Jozefowska et al. (2001) examined the performance differences between a fitness function with penalty function and without penalty function and discovered that the fitness function with penalty function clearly performs better. The way in which they are penalized, in the definition of the fitness/penalty function, is very important in the performance of the algorithm. Different evaluation or fitness functions have been proposed in the literature taking into account penalty.

Hartmann (2001) defined the following fitness function.

$$f(i) = \begin{cases} make\ span(i) & if\ SFT(i) = 0; \\ T + SFT(i) & otherwise \end{cases}$$

where make span is the make span of the individual i and T is the upper bound on the project's make span that is given by the sum of the maximal durations of the activities. SFT (i) is the excess of non-renewable resources defined by

$$SFT(i) = \sum_{k \in NR} max\left\{0, \sum_{i=1}^{J} (r_{jmk} - R_k)\right\}$$

Where i= 1,…,J represent the number of activities. R_k represents the total availability of the non-renewable resources k = 1,…,K. and r_{jmk} represents requirement of the non-renewable resource of type k by the activity j performed in mode m =1,…M. Therefore an individual is a feasible schedule if and only if SFT (i) is 0. Alcaraz et al. (2003) points out that this formulation while calculating the fitness value of an infeasible individual does not consider its duration and only depends on the excess of non-renewable resources and also the upper bound T is so poor that an infeasible individual will have a probability close to zero and will not participate in the genetic process. He rectified this problem and proposed his formulation as given below

$$f(x) = \begin{cases} mak\ (i)\ SFT(i) = 0 \\ maxmak(p)\ + (mak(i) - mincp) + SFT(i) \\ otherwise \end{cases}$$

where mak(i) is the make span of the individual i and maxmak(p) gives the maximal make span of feasible solutions related to individuals of the current generation which represents an upper bound of the project's make- span. To this bound is added the excess of non-renewable resources SFT (i) and the increase over the make span given by the minimal critical path, mincp, using the minimal duration of activities. This fitness function gives reasonable probabilities to participate in the genetic process as two solutions with identical SFT but different make span will have a different fitness value and the penalty of a non-feasible individual.

However, this fitness function is built by adding units of time from the make span and units of resources from the excess of non-renewable resources. The magnitude of both aspects of the solution can disturb the meaning of the fitness function. To solve this weak point, Lova (2009) *et. al.* proposed a new fitness function where both aspects of the solution are jointly considered but normalized in order to eliminate their magnitudes. The new fitness function that is computed for each individual according to the following expressions depending on whether individual I is feasible (SFT (i) = 0) or non-feasible.

$$f(x) = \begin{cases} 1 - \dfrac{maxmak(p) - mak(i)}{maxmak(p)}, if\ SFT(i) = 0 \\ 1 + \dfrac{mak(i) - mincp}{mak(i)} + \sum\limits_{k \in NR}^{k} \max\left\{0, \dfrac{\sum_{j=1}^{J} r_{jmk} - R_k}{R_K}\right\}, \\ otherwise \end{cases}$$

The feasible individual with the greatest make span will have a fitness value equal to 1 while the best one will have a fitness value close to zero. The fitness function value of a non-feasible individual always is greater than 1.

Then, all non-feasible individuals will have a fitness value greater than that of the feasible ones (and will have fewer opportunities for survival in the selection process). In addition, the sum of the normalized deviation of the make span from the minimal critical path and the normalized excess of non-renewable resources are added. Lova (2009) *et. al.*Claim that this fitness function solves weak points of fitness computations for the multi-mode RCPSP that appeared in the literature. We decided to use the fitness function given by Lova (2009) *et. al.* so our fitness function will be as given below

$$f(x) = \begin{cases} 1 - \dfrac{maxmak(p) - mak(i)}{maxmak(p)}, if\ SFT(i) = 0 \\ 1 + \dfrac{mak(i) - mincp}{mak(i)} + \sum\limits_{k \in NR}^{k} \max\left\{0, \dfrac{\sum_{j=1}^{J} r_{jmk} - R_k}{R_K}\right\}, \\ otherwise \end{cases}$$

After calculating the fitness function
Selection is the process of selecting two parents from the population for crossing. The aim of selection is to assure that only fitter individuals in the population participate in crossover or reproduction with better chance of getting fitter offspring. An individual has more chance to be selected if the fitness function is higher. The degree to which the better individuals are favored is defined as the selection pressure. The higher the selection pressured, the more the better individuals are favored. This selection pressure drives the GA to improve the population fitness over the successive generations but there is risk of converging to suboptimal solutions. Too weak selection will result in too slow evolution. To adjust its selective pressure and population diversity so as to fine-tune GA search performance should be an ideal selection strategy.

Rank selection and the tournament selection is the favoured selection strategy for the resource constrained project scheduling. For the resource constrained project scheduling problem, rank selection and tournament selection has been used in most of the literature on genetic algorithm application for RCPSP and it produced competitive performance. In some papers random selection method is also employed Apart from Roulette wheel selection method random selection method is also employed in some papers.

Rank Selection ranks the population and every chromosome receives fitness from the ranking. The worst has fitness 1 and the best has fitness N. It is shown that it results in slow convergence and prevents too quick convergence. When the fitness variance is low it also keeps up selection pressure. It preserves diversity and hence leads to a successful search. Tournament selection strategy provides selective pressure by holding a tournament competition among individuals. So we go with these two selection strategies of rank selection and tournament selection.

4.2 Crossover Operator

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. Cross over combines the features of two parent chromosomes to form two

offspring which inherit their characteristics. A poorly designed crossover becomes a sort of mutation.

4.2.1 Crossover Type

Literature reveals single point, two point crossover, partially matched crossover, uniform crossover methods. In most of the literature two point crossover method has been used and found to be very effective. In two point cross over technique it will not destroy entire chromosome structure which is useful for fitter solution. It is possible to capture the strongest portion of the chromosomes. This will improve the quality of the solutions in the next generations. So we have decided to use two point cross over technique. We will use uniform or multi point cross over technique for weaker solutions.

The choice of the crossover sites or points is the next important thing. In literature of genetic algorithm application for RCPSP, it is found that the authors are using random selection for the cross over points, some authors used resource information to decide the cross over point e.g. Debels and vanhouke (2007) peak cross over, Ranjbar et al (2007)resource utilization ratio RUR [9] [10].

4.2.2 Crossover Probability

The crossover probability controls the rate at which the solutions are subjected to crossover. If the value of crossover probability is high, then more solutions will introduced into the population. This will disrupt the solutions more and diversity will increase and convergence rate will be slowed down. If probability rate is too less, the convergence will be quicker due to homogeneity in the population. Depending upon other genetic operator, crossover probability suggested is in range 0.7 to 0.9 for the RCPSP. We will consider the fixed crossover probability as 0.7, 0.8 and 0.9.

4.2.3 Procedure for Two-point Crossover Method

Draw two cross over points based on the modified resource utilization ratio. Let these two points be p and q such that 1_ p < q _ J (total number of activities)

A Generation of Son

By copying genes from position p+1 to q from the father's schedule, the corresponding modes will be transferred, the son will be generated and the scheduling way gene is taken from the father. Remaining gene positions, from 1 to p and q+1 to J, are copied from mother. All the genes or activities from mother, except the genes copied from father, are copied by scanning from left to right. The corresponding modes are transferred as it is from mother to son.

B Generation of Daughter

By copying genes from position p+1 to q from the mother's schedule, the corresponding modes will be transferred, the daughter will be generated and the scheduling way gene is taken from the mother. Remaining gene positions, from 1 to p and q+1 to J, are copied from father. All the genes or activities from father, except the genes copied from mother, are copied by scanning from left to right. The corresponding modes are transferred as itis from father to son. This procedure will ensure precedence feasible schedule.

4.3 Mutation Operator

To maintain genetic diversity in the population mutation is viewed as a background operator. It introduces new genetic structures in the population by randomly modifying some of its building blocks. Mutation helps escape from local minima's trap and maintains diversity in the population.

4.3.1 Mutation Probability

Depending upon other genetic operators the mutation probability for the RCPSP, suggested is in the range 0.02 to 0.05. We will consider the fixed mutation probability as 0.03 and 0.05.

A Termination criteria

As no of generation and no of schedules generated we will use termination criteria. In literature most of the authors have used the 1000 schedules generation as termination criteria and gives the required solutions, so we plan to go for 2500 schedules generation as termination criteria.

4.4 LibRCPS

LibRCPS uses genetic algorithms to find solutions to project scheduling problems under limited resources. LibRCPS that is open source resource constrained project scheduling library uses a model of project that can only reflect a given set of properties, and can only schedule projects that fit into this category [11]. Most notably it will do project scheduling under limited resources, but not resource levelling.

4.4.1 Usage

It is important to understand the basic usage pattern for LibRCPS, which is the same for any language. Using the library is done in a number of distinct steps:

1) Setting up the problem structure, resources, jobs, modes and alternatives
2) Declaring relationships between jobs
3) Setting up the solver
4) Running the Solver
5) Retrieving the results

5. Result and Analysis

We took the problem no j1011_1 from the Kolisch Standard problem library, PSBLIB [12] and apply the new developed genetic algorithm (GA) program on the above problem and come out with ten different solution as shown in the table 1. GA program has set to population of 700 with 2500 iterations. This program selects two parents which has higher fitness function from the population for crossing and mutated. Program is run for the problem with mutation probability 0.01 and crossover probability 0.5. So the problem is undergoing the population of 700 solutions, 2500 iterations, 0.01 mutation probability and 0.5 crossover probability. With this setup we got the one optimal solution which is similar to or close to standard solution available in the literature. From these solution we come out with ten solution for one problem as shown in table 1 below and it shows that we can get different solution with different sequence and variable resource consumption that shows the flexibility of the program.

Table 1: Sequence table for RCPSP

Job No.	Job Sequence	Starting Time	Duration
j1011_1	0,1,2,3,4,5,6,8,9,7,10,11	0,0,0,0,5,6,8,8,8,16,20,21	21
	0,1,2,3,4,5,6,9,8,7,10,11	0,0,0,0,6,6,9,9,10,17,21,23	23
	0,1,2,3,4,5,8,9,6,7,10,11	0,0,0,0,7,7,8,8,9,16,22,22	22
	0,1,2,3,4,5,6,9,8,7,10,11	0,0,0,0,6,7,8,8,9,18,22,22	22
	0,1,2,3,4,5,8,9,6,7,10,11	0,0,0,0,7,8,9,9,10,18,21,21	21
	0,1,2,3,5,4,6,9,8,7,10,11	0,0,0,0,6,7,8,8,10,16,21,21	21
	0,1,2,3,4,5,8,9,6,7,10,11	0,0,0,0,6,6,8,8,10,16,22,22	22
	0,1,2,3,4,5,6,8,9,7,10,11	0,0,0,0,6,7,8,8,8,17,20,21	21
	0,1,2,3,4,5,8,6,9,7,10,11	0,0,0,0,7,7,8,9,9,17,20,22	22
	0,1,2,3,4,5,6,8,9,7,10,11	0,0,0,0,6,8,8,9,9,18,20,21	21

From the above table 1 we got optimal duration of 21 units with five time different sequence with respect to different start time. According to this data we have five different optimal solution with five different resource consumption and it shows in the graph below,

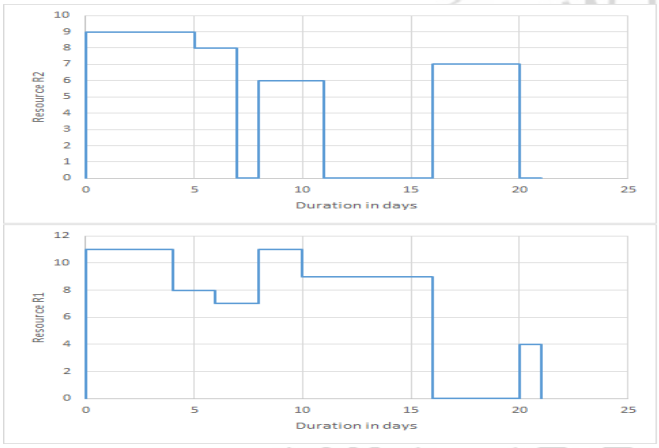

Figure 2: Resource consumption profile for solution 1

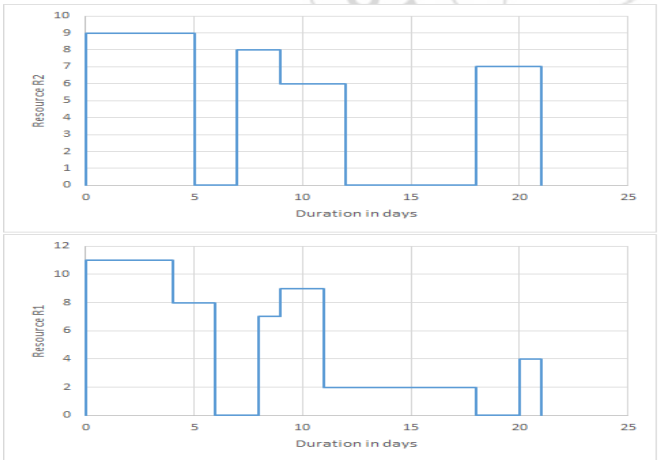

Figure 3: Resource consumption profile for solution 2

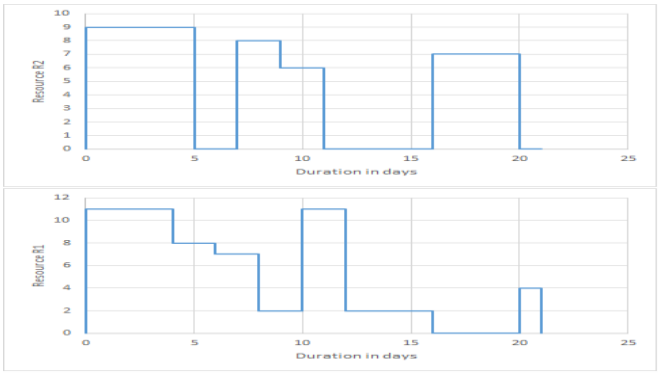

Figure 4: Resource consumption profile for solution 3

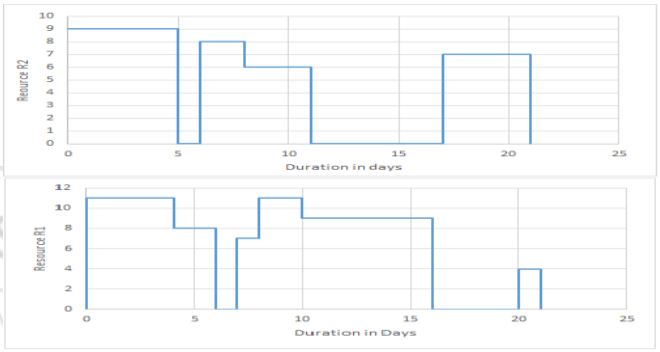

Figure 5: Resource consumption profile for solution 4

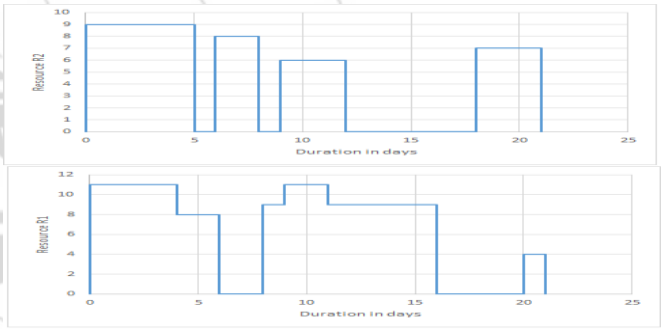

Figure 6: Resource consumption profile for solution 5

From the above graphs we can say that one can choose any optimal solution that meets the final requirement to solution to the problem.

As explained above we run the program on some data sets available on Kolisch Library standard problem data sets with 10, 12, 14, 16, 18, 20, 30 number of jobs [12] and got the average deviation and compared the results with previous one as shown in the table below

Table 2: Reference Results

Data Sets	Lacocq (SA) 2003	jozefowska (SA) 2001	Hartmann (GA) 2001	Alcaraz (GA) 2003	Zang (pso) 2006	Lova et al (GA)2009	Van Peteghem (GA) 2010	Present Work
j10	0.21	1.16	0.06	0.24	0.11	0.06	0.01	1.7
j12	0.19	1.73	0.14	0.73	0.17	0.17	0.09	1.9
j14	0.92	2.6	0.44	1.00	0.41	0.32	0.22	2.0
j16	1.43	4.07	0.59	1.12	0.83	0.44	0.32	2.4
j18	1.85	5.52	0.99	1.43	1.33	0.63	0.42	2.5
j20	2.10	6.74	1.21	1.91	1.79	0.87	0.57	2.6
j30	-	-	-	-	-	16.65	1.08	9.0

6. Conclusion

So from the above results, we can conclude that developed Genetic algorithm for Resource Constrained Project Scheduling problem gives number of optimal solution with different resource consumption, so one can follow any solution that suitable to problem conditions.

References

[1] Kolisch, R., Drexl, A., 1997. Local search for nonpreemptive multi-mode resource constrainedproject scheduling. IIE Transactions 29 (11), 987–999.
[2] Józefowska, J., Mika, M., Rozycki, R., Waligora, G., Weglarz, J., 2001. Simulated annealing for multi-mode resource-constrained project scheduling. Annals of Operations Research 102 (1–4), 137–155.
[3] 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.
[4] Ozdamar, L., 1999. A genetic algorithm approach to a general category project scheduling problem. IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews 29 (1), 44–59.
[5] Hartmann, S., 2001. Project scheduling with multiple modes: a genetic algorithm. Annals of Operations Research 102 (1–4), 111–135.
[6] 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.
[7] 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.
[8] 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.
[9] Dieter Debels Mario Vanhoucke , May–June 2007A Decomposition–Based Genetic Algorithm for the Resource-Constrained Project-Scheduling Problem OPERATIONSRESEARCH Vol. 55, No. 3, pp. 457–469.
[10] Ranjbar, M., Kianfar, F., 2007. Solving the discrete time/resource trade-off problem in project scheduling with genetic algorithms. Applied Mathematics and Computation191 (2), 451–456.
[11] www.librcps.org - LibRCPS - Open Source Resource Constrained Project SchedulingLibrary
[12] www.om-db.wi.tum.de/psplib/library.html - The Library PSBLIB

Author Profile

Mr. Vinayak Chandrakant Sawant is Student in Fr. Conceicao Rodrigues College of Engineering, Bandra, Mumbai. His research interests include Resource Constrained Project scheduling using genetic algorithm.