METAHEURISTIC ALGORITHMS FOR RCPS PROBLEM

Dr. G. Sunitha[1], Dr. J. Avanija[2], Dr. K. Reddy Madhavi[3]
[1,2,3]Sree Vidyanikethan Engineering College, A. Rangampet, India

Abstract

Optimization means to make things work in such a manner where all the comprised resources are used as efficient as possible to provide a best outcome. Although the algorithmic space for this is a vast area of discussion, this paper focuses on natural-inspired algorithms. Particle swarm optimization, ant colony optimization and firefly algorithms are the few examples of natural or genetic algorithms, per se. These algorithms that help us most to achieve optimization with profound accuracy in large set of real-time applications. One of such fields is software industry or the corporate industry as we can call it. As this area comprises of huge personnel, management, project cost, time managements and most importantly quality, the projects must be scheduled without compromising the optimization. Therefore, lot many ideas and proposals are made in projects success and risk factors. Among many of these, the major elements of software are task scheduling and project scheduling or simply categorizing as resource constraints that have to be optimized in order to get the best possible outcome. Resource constraint project scheduling problem plays a very important role in the context of project management. The single goal is to find a solution that minimizes the cost, time with efficient usage of resources which may be infrastructure or personnel etc. Recently number of metaheuristic optimization algorithms have been studied and applied to solve the resource constrained project scheduling problem. This paper focuses on application of Bat Algorithm, Tabu Search and Firefly Algorithm for optimization of resource constrained project scheduling. On experimental evaluation of the algorithms, it has been observed that the Firefly algorithm provided better optimization in project scheduling.

Keywords: project scheduling problem; resource constraints; resource management; optimization; firefly algorithm;

I. INTRODUCTION

Resource Constrained Project Scheduling Problem (RCPSCP) is an optimization problem that tries to find the optimum ordering of activities to achieve a set of predefined objectives with the existing resources [1, 2]. Generally, a resource constrained project scheduling contains resources of limited availability, activities with known durations and requests of resources which are linked by precedence relations. The problem mainly concentrates on finding an optimal solution with minimum makespan by assigning a starting time to each activity so that the resource availabilities and precedence relations are correctly managed.

The firefly is one of the meta-heuristics algorithms proposed by Xin-she Yang [3] and inspired by the flashing light behavior of fireflies. The main purpose of a firefly's flash is to act as a signal system to attract other fireflies. The firefly algorithm can be formulated based on the below three rules. They are:

(i) All fireflies are unisexual, so that any individual firefly will be attracted to all other fireflies

(ii) Attractiveness is proportional to their brightness and for any two fireflies, the less bright one will be attracted by the highest brighter one; however the intensity decreases as their mutual distance increases

(iii) If there are no fireflies brighter than a given firefly, it will move randomly.

The brightness must be associated with the objective function. The flashing light of the fireflies resembles the resources that represent the schedule of the problem at hand. Mechanisms of fireflies have been implemented effectively and major areas of wireless networks design, mobile robotics and dynamic market pricing. One of those unexplored areas is RCSCP problem, where firefly algorithm might give one of the optimized solutions. In this paper and effort is made to apply the standard Bat Algorithm, Tabu Search and Firefly algorithms to solve RCPSP problems and compare their efficiency.

II. Related Work

The success of a software project relies on the proper and timely usage of resources. A resource might be any utility that is required to complete the software project like human resources, third-party softwares & tools, financial resources, sequence of activities, durations of activities, durations of availability of resources, demands of activities etc. As such, scheduling the usage of resources is a decisive task to achieve the set milestones.

Resource constrained project scheduling problems are involved in optimized assignment of tasks to available resources. As the resources available may be limited with respect to many factors, efficient utilization of resources is desirable. This is especially crucial in today's market strategies to have first mover advantage with timely launch of the products.

The goal of Resource constrained project scheduling is to engender tangible schedules to meet the predefined objectives and resource constraints. An RCPS problem can be viewed as a combinatorial optimization problem by considering activities, durations and their precedence relationships. A schedule specifies the time duration of the activity along with the start and end times of the

Many different objectives are possible and those are depend on the goals of the project manager, but the most common of those objectives is to find out the minimum makespan i.e., the minimum time to complete the given schedule.

A. *RCPSP based on random keys*

The chromosome representation of the resource constrained project scheduling problem is based on the random keys [4]. Here the schedules are constructed using a heuristic priority rule in which the properties of the activities are defined. The heuristic priority rule generated parameterized active schedules. This approach was tested on a set of standard problems taken from the literature and compared with, other approaches. The computational results validate the effectiveness of the proposed algorithm.

B. *Ant colony optimization for RCPSP*

In this algorithm [4, 5], the solution is often attempted in a sequence of iterative steps. Candidate solutions are identified from a sample of solutions using a parametric probability distribution. A group of ants are selected and a set of decision of variables are defined in this problem. The ants select the design variables for creating the candidate solutions. As the ants creating the candidate solutions, a local updation of the solution is done based on its suitability. Hence, the main idea of using ant colony optimization (ACO) approach is for deciding which activity from the set of eligible activities should be scheduled next by the Schedule Generation Scheme (SGS). The general principle of ant colony algorithm is similar to an ant algorithm called AS-TSP for travelling salesman problem.

C. *PSO for RCPSP*

Particle swarm optimization [6, 7] is based on the collective behavior or organisms such as fish schooling, insect swarming or birds flocking. A swarm is a large number of decentralized, homogeneous agents that interact locally among themselves and with their environment, whereby a global is to be achieved. Particle swarm optimizations are used for problems where the optimized functions must be discontinuous, non-differentiable with too many non-linear related parameters. Each particle or member in the swarm tries to sense a potential solution at any point of time. It communicates with a signal proportional to the suitability of the candidate solution to the other particles in the swarm. Each swarm particle or member can therefore sense the strength of the signal communicated by the other swarm particle or members, thus the suitability of the candidate solution is based on a fitness function. When a particle tries to concentrates on a more suitable candidate solution from among the locally available candidate solutions, based on different learning mechanisms, a new movement direction is identified to guide the particles towards an optimal solution.

D. *Other Approaches*

There are different other approaches for solving resource constrained project scheduling problem which includes genetic algorithms, artificial bee colony, cuckoo algorithm, bat algorithm, tabu search etc [8-13]. The hybridization of ant colony optimization and particle swarm optimization of these algorithms will improve the efficiency and can give best optimal solutions for given schedules.

III. FIREFLY ALGORITHM

The basic firefly approach [14, 15] starts by generating initial population of N objects as seed. The initial values of the parameters of the objective functions are defined by the domain expert. The process proceeds as follows. The objects are evaluated with respect to the pre-defined objective functions and are targeted towards achieving a global optimal solution. The parameters of the objective functions are then recomputed by evaluating the solution space. These two steps are repeated until the best possible solution is found.

By natural instincts, the flocks of fireflies move towards the brighter and highly intensified regions of light source. With the same idea, the firefly algorithm works by directing the objects towards finding the optimal solution. This helps in efficiently finding the best possible solution in a huge candidate solution space. Hence, the algorithm provides or finds a way to find high intensity locations in such a huge search space.

The movement of the objects towards each other is based on their level of attractiveness. A formidable object attracts other objects towards it. Highly clustered regions of objects attract remaining objects towards them, thus increasing the density of the clustered regions. This will lead to the fast propagation of the flock of objects towards optimal solution. The convergence of solution is based on selection of initial seed objects, the context parameters and the objective functions.

Algorithm Firefly
Objective function f(x), $x = (x_1, x_2, \dots x_d)^T$
Generate an initial population of *n* fireflies $x_i = (i = 1, 2, \dots n)$
Light Intensity I_i at x_i is determined by $f(x_i)$
Define light absorption coefficient γ

While (t< MaxGeneration)
    For i=1:n all *n* fireflies
        For j=1:n all *n* fireflies
            If($I_i < I_j$)
                Move firefly i towards j
            End if
            Vary attractiveness with distance *r* via $\exp[-\gamma r^2]$
            Evaluate new solutions and update light intensity
        End for
    End for
    Rank the fireflies and find the current global best g*
End while
Post-process results and visualization

The pseudo code of Firefly Algorithm [14] is as shown above.
The method will start a set of seed objects being placed randomly in the huge problem space and then updating the solutions in iterations, until the termination condition is met. By the time, the algorithm gets terminated, the algorithm returns the best solution found in the lattice of solution space.

A. *Light Intensity and attractiveness initialization*

The parameters of the objective function are to define the intensity and brightness of the objects. The intensity and brightness of an object are defined by its fitness value. The fitness value of an object is proportionate to its distance from the optimal solution. The higher the fitness of the object, the higher the number of other objects that it attracts forming clustered regions. Thus, higher is its attractiveness quotient. The initial fitness values for the seed objects are defined based on the intensity of the solution.

B. *Location Update*

The distance between any of the two objects of solution space at their respective position is always a Cartesian distance. In every iteration, the locations of the objects of solution space are updated by recomputing their fitness values. The fitness values of the objects are considered as the makespan for the resource constrained project scheduling problem. On recomputing the fitness values of all the objects, the objects are prioritized with respect to their fitness value. A random vector of fitness values is then generated by using permutation-based representation to permute the process. The fitness value of each object is compared with the respective value in the random vector. The location of the object is updated only when better fitness value is obtained on relocating it. The location of each object is updated by moving objects with lower fitness values towards objects with higher fitness values. At the end of every iteration, the locations of all objects are updated, and they occupy their new locations for continuing with the next iteration.

IV. Experimentation and Results

The Library PSPLIB [16, 17] contains various benchmark data sets for resource constrained project scheduling problem. The library has solution sets for evaluating single-mode as well as multi-mode resource constrained project scheduling problems. Also, ProGen, which is a standard data set generator, can be used for generating synthetic project data sets.

The benchmark datasets from PSPLIB have been used to evaluate the Bat Algorithm, Tabu Search and Firefly Algorithm for solving the resource constrained project scheduling problems. The evaluation of the considered problems is done on single-mode datasets. The single-mode datasets J60hrs.sm, J90hrs.sm and J120hrs.sm are used for experimentation. Each of them contains heuristic solutions for 60 jobs, 90 jobs and 120 jobs respectively. As the number of jobs increases, the complexity of the resource constrained project scheduling problem increases. Given, the datasets and the respective optimal solution, the algorithms under consideration are evaluated by comparing their provided solution with the optimal solution. The parameter settings are: the population size is taken to be 30, termination conditions are varied and the number of schedules was set to evaluate the success rate which is obtained over the above mentioned case studies from PSPLIB.

Table I. Evaluation Results of Case Studies

Case Study	*Average Success Rate (%)*		
	Bat Algorithm	*Tabu Search*	*Firefly Algorithm*
J60hrs.sm	89.58	93.75	98.33
J90hrs.sm	72.91	74.37	78.75
J120hrs.sm	72.3	75.1	77.08

The success rate illustrates the number of instances in a case study that are successfully solved by the methods under consideration. For example, the method under consideration is called successful if it finds the optimal solution with the minimum makespan [18]. The average deviations are considered as an important measure to compare the performance of investigated methods. Here, the average deviation is used to compare the proposed method with other methods. Table I shows the experimentation results of the Bat algorithm, Tabu Search and Firefly Algorithms evaluated on single-mode datasets J60hrs.sm, J90hrs.sm and J120hrs.sm.

Considering the resource constrained project scheduling problem in single-mode datasets, the metaheuristic algorithms - Bat algorithm, Tabu Search and Firefly Algorithm have all showed the capability of providing near to the optimal solutions, with evident efficiency. The experiments showed the following insights into the Firefly algorithm. The success rate of the Firefly algorithm increases with the number of schedules and Firefly algorithm is not suitable for solving case studies with more number of activities.

V. Conclusion and Future Work

The metaheuristic algorithms - Bat algorithm, Tabu Search and Firefly Algorithm are studied and experimented for solving the resource constrained project scheduling problem in order to achieve an optimized scheduling solution. The basic firefly algorithm is proved to be efficient to solve project scheduling problems for efficient utilization of resources. Although the Firefly algorithm is proved to be superior to the compared metaheuristic algorithms, further tuning of local and global search is required to ensure that the obtained solution is optimum and not premature. Further research can be extended, by applying Firefly algorithm to high-dimensional and non-linear scheduling problems.

Further, capabilities of metaheuristic algorithms can be studied for handling huge candidate solution space, to handle multi-objective resource constrained project scheduling problems.

VI. REFERENCES

[1] Brucker, Peter, Andreas Drexl, Rolf Möhring, Klaus Neumann, and Erwin Pesch. "Resource-constrained project scheduling: Notation, classification, models, and methods." European journal of operational research 112, no. 1 (1999): 3-41.

[2] Hartmann, Sönke, and Dirk Briskorn. "A survey of variants and extensions of the resource-constrained project scheduling problem." *European Journal of operational research* 207, no. 1 (2010): 1-14.

[3] Yang, Xin-She. "Firefly algorithms for multimodal optimization." In International symposium on stochastic algorithms, pp. 169-178. Springer, Berlin, Heidelberg, 2009.

[4] Mendes, Jorge Jose de Magalhaes, José Fernando Gonçalves, and Mauricio GC Resende. "A random key based genetic algorithm for the resource constrained project scheduling problem." Computers & Operations Research 36, no. 1 (2009): 92-109.

[5] Merkle, Daniel, Martin Middendorf, and Hartmut Schmeck. "Ant colony optimization for resource-constrained project scheduling." IEEE transactions on evolutionary computation 6, no. 4 (2002): 333-346.

[6] Zhang, Hong, Heng Li, and C. M. Tam. "Particle swarm optimization for resource-constrained project scheduling." International Journal of Project Management 24, no. 1 (2006): 83-92.

[7] Jarboui, Bassem, Najeh Damak, Patrick Siarry, and A. Rebai. "A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems." Applied Mathematics and Computation 195, no. 1 (2008): 299-308.

[8] Yamashita, Denise Sato, Vinícius Amaral Armentano, and Manuel Laguna. "Scatter search for project scheduling with resource availability cost." European Journal of Operational Research 169, no. 2 (2006): 623-637.

[9] Mika, Marek, Grzegorz Waligora, and Jan Węglarz. "Tabu search for multi-mode resource-constrained project scheduling with schedule-dependent setup times." European Journal of Operational Research 187, no. 3 (2008): 1238-1250.

[10] Ziarati, Koorush, Reza Akbari, and Vahid Zeighami. "On the performance of bee algorithms for resource-constrained project scheduling problem." Applied Soft Computing 11, no. 4 (2011): 3720-3733.

[11] Hartmann, Sönke, and Dirk Briskorn. "A survey of variants and extensions of the resource-constrained project scheduling problem." European Journal of operational research 207, no. 1 (2010): 1-14.

[12] Chen, Wang, Yan-jun Shi, Hong-fei Teng, Xiao-ping Lan, and Li-chen Hu. "An efficient hybrid algorithm for resource-constrained project scheduling." Information Sciences 180, no. 6 (2010): 1031-1039.

[13] X. S. Yang, Nature-Inspired Metaheuristic Algorithms, Luniver Press, 2008

[14] Sanaei, Pejman, Reza Akbari, Vahid Zeighami, and Sheida Shams. "Using firefly algorithm to solve resource constrained project scheduling problem." In Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications, Springer, India, 2013: 417-428

[15] Sebt, M.H. and Alipouri, Y. "Solving resource-constrained project scheduling problem with evolutionary programming." Journal of the Operational Research Society 64, no. 9(2013):1327-1335.

[16]http://www.omdb.wi.tum.de/psplib/library.html.

[17] Kolisch, Rainer, and Arno Sprecher. "PSPLIB-a project scheduling problem library: OR software-ORSEP operations research software exchange program." European journal of operational research 96, no. 1 (1997): 205-216.

[18] Kolisch, Rainer, and Sönke Hartmann. "Experimental investigation of heuristics for resource-constrained project scheduling: An update." European journal of operational research 174, no. 1 (2006): 23-37.