An Evolutionary Approach to the Resource-Constrained Project Scheduling Problem

Vicente Valls[*] *Sacramento Quintanilla*[†] *Francisco Ballestín*[*]

[*] Dpto. Estadística e Investigación Operativa, University of Valencia
Dr. Moliner, 50, 46100 Burjasot, Valencia, Spain
Email: {Vicente.Valls, Francisco.Ballestin}@uv.es

[†] Departamento de Economía Financiera y Matemática, University of Valencia
Avda. de los Naranjos, s/n, 46021 Valencia, Spain
Email: Maria.Quintanilla@uv.es

1 Description of the problem

The resource-constrained project-scheduling problem ($RCPSP$) may be stated as follows. A project consists of a set of n activities numbered 1 to n, where each activity has to be processed without interruption to complete the project. The dummy activities 1 and n represent the beginning and end of the project. The duration of an activity j is denoted by d_j where $d_1 = d_n = 0$. There are R renewable resource types. The availability of each resource type k in each time period is R_k units, $k = 1, ..., R$. Each activity j requires r_{jk} units of resource k during each period of its duration where $r_{1k} = r_{nk} = 0$, $k = 1, ..., R$. All parameters are assumed to be non-negative integer valued. There are precedence relations of the finish-start type with a zero parameter value (i.e., $FS = 0$) defined between the activities. In other words, activity i precedes activity j if j cannot start until i has been completed. The structure of a project can be represented by an activity-on-node network $G = (V, A)$, where V is the set of activities and A is the set of precedence relationships. $S_j(P_j)$ is the set of successors (predecessors) of activity j. It is assumed that $1 \in P_j, j = 2, ..., n$, and $n \in S_j, j = 1, ..., n-1$. The objective of the $RCPSP$ is to find a schedule S of the activities, i.e., a set of starting times $(s_1, s_2, ..., s_n)$ where $s_1 = 0$ and the precedence and resource constraints are satisfied, such that the schedule duration $T(S) = s_n$ is minimised. Let T^* be the minimum schedule duration or minimum makespan.

2 Previous works

As a job shop generalisation, the $RCPSP$ is NP-hard in the strong sense (see [1]). In recent years, the applications and difficulties of the $RCPSP$ and its extensions have attracted increasing interest from researchers and practitioners. Although several exact algorithms have been proposed, only small-sized problem instances with up 30 activities can be solved exactly in a satisfactory manner. Therefore, heuristic solution procedures remain as the only feasible methods of handling practical resource-constrained project scheduling problems. Many heuristic approaches have been proposed for $RCPSP$. An overview of heuristics focusing on X-pass methods (single pass methods, multi-pass methods, sampling procedures) and metaheuristics (simulated annealing, genetic algorithms, tabu search) can be found in [4]. An investigation of the performance of those $RCPSP$ heuristics is provided in [3]. The computational results indicate that the best metaheuristics outperform the best sampling

approaches. As stated by the authors, this is mainly because sampling procedures generate each schedule anew without considering any information given by previously visited solutions - as metaheuristics typically do.

A tabu search based heuristic algorithm for a generalisation of the $RCPSP$ has been recently presented in [8]. It is able to handle multi-mode processing, variable availability of renewable and non-renewable resources, and complex objective functions. The code was tested for a number of benchmarks of the jobshop and $RCPSP$, as well as some problems from real applications.

[6] presents an ant colony optimization approach for the $RCPSP$. The algorithm was tested on the standard set $j120$ generated using $ProGen$ and available from the library $PSPLIB$ [5]. The algorithm was compared to various other heuristics that are include in the computational study of [3]. Every heuristic was allowed to construct and evaluate at most 5000 schedules for each problem instance. The computational results obtained with this experimental design show that their algorithm performed best on average. The authors also claim that their algorithm outperforms that of [8].

[7] proposes Lagrangian-based list heuristics to compute feasible solutions for resource-constraint projects. A computational study on the $j60$, $j90$, $j120$ instance sets from $PSPLIB$ shows that the quality of the feasible solutions compares to results obtained with state-of-the-art local search algorithms.

A metaheuristic algorithm, $CARA$, for solving the resource-constrained project-scheduling problem was proposed in [10]. The procedure is a non-standard implementation of fundamental concepts of tabu search without explicitly using memory structures. The authors consider two phases in the application of the algorithm. Phase 1 makes use of temporal information whereas Phase 2 is based on two new multi-pass sampling methods (the β biased random sampling method and the window sampling method) adequately defined to make controlled moves. The algorithm is a robust procedure that can be successfully applied to diversely characterised instance problems without the need for parameter adjustment. Computational experiments on the standard sets $j30$, $j60$, $j90$ and $j120$ for the $RCPSP$ generated using $ProGen$ show that the quality of $CARA$ is quite good and that computational times are short. They also show that $CARA$ can compete with state-of-the-art heuristics for the $RCPSP$.

Population-based strategies, often referred to as evolutionary methods, manipulate a collection of solutions rather than a single solution at each stage. A prominent subclass of these methods is based on strategies for "combining" solutions, as illustrated by genetic algorithms, scatter search and path relinking methods. Another prominent subclass consists of methods that are primarily driven by utilizing multiple heuristics to generate new population members ([2]).

Path relinking generates new solutions by exploring trajectories that connect elite solutions - by starting from one of these solutions and generating a path in the neighborhood space that leads toward the other solutions. This is accomplished by selecting moves that introduce attributes contained in the ending solutions into the current solution.

The scatter search methodology consists of creating a population of solution vectors that "evolves" according to the characteristics of the particular problem instance being solved. It forms linear combinations of subsets of vectors (reference vectors) in the current population to create new solutions that inherit the useful information contained in the selected reference vectors. The linear combinations are chosen to capture information not contained separately in the original vectors. An early application of scatter search to a stochastic project-scheduling problem can be found in [9].

3 The proposed approach

In this paper, we present a population-based approach to the $RCPSP$. The procedure incorporates various strategies for generating and evolving a population of schedules. The method has two phases. The first phase handles the initial construction of a population of schedules and these are then evolved

until high quality solutions are obtained. Random and constructive procedures are used to add quality and diversity during the construction of the initial population. The evolution of the population is driven by the alternative application of an efficient improving procedure for locally improving the use of resources (HIA), and a mechanism for combining schedules that integrates scatter search and path re-linking characteristics (MPA).

HIA searches the space of activity list (AL) representations. Given an AL representation of an active schedule, HIA starts an iterative improving process with backtracking. To move from an AL representation to the next, the position of a subset of activities is advanced and consequently, the position of other activities is put backward. HIA is applied to every schedule in the population. Then MPA explores the "convex hull" of a selection of the best schedules in the improved population. MPA works with the topological order (TO) representations of the selected schedules. MPA generates p TO schedule representations in each of the segments that joint every pair of selected TO representations. The best schedules generated by MPA form the initial population for the next iteration.

The objective of the second phase is to explore in depth those vicinities near the high quality sequences. This exploration consists in: firstly, generating a population by means of a biased random sampling of an vicinity near the sequence; and secondly, applying a variation of the improving procedure that was used in the first phase.

At the end of phase 1, the best solution obtained so far is, hopefully, of a high quality. Experience seems to indicate ([10]) that good candidate schedules are usually to be found "fairly close" to other good schedules. Multi-pass sampling methods can be adequately adapted to explore in a controlled way neighbourhoods of different depths. Specifically, the β *biased random sampling method*([10]) can repeatedly be applied to the best schedule obtained so far to generate a new population of relatively high quality. Usually, these schedules are of lower quality than the current best schedule but are promising starting points for the application of the improving procedure HIA.

Phase 2 starts from the best schedule obtained in phase 1, generates a new population of schedules in the vicinity of it and initiates an improving process from each population element. Each time an improved schedule is generated the whole procedure restarts from it.

4 Numerical Experiments

To assess the merit of the proposed solution method we have performed the following computational experiment. For test instances, we have used the standard sets $j30$, $j60$, $j90$ and $j120$ for the $RCPSP$. These were generated using *ProGen* and are available in the Project Scheduling Problem Library ($PSPLIB$) along with their optimum or best-known values ([5]). The sets $j30$, $j60$ and $j90$ consist of 480 projects with four resource types and 30, 60 and 90 non_dummy activities, respectively. The set $j120$ consists of 600 projects with four resource types and 120 activities. There are 2040 instances in total.

Table 1 summarises the results of the experiment. The first column indicates the instance set referred to in the results shown in each row. The second column, labelled $\sum EVO$, consists of the sum of the values obtained by our algorithm whereas the third column, $\sum PSPLIB$, shows the sum of the best values in $PSPLIB$ as of March 25, 2001. The fourth column, av_dev , consists of the average percentage deviations from the best solutions in $PSPLIB$. The number of instances for which the proposed algorithm obtains the best known values in $PSPLIB$ is reported in the fifth column ($best\_sol$). It is worth noting that these known values include those found by our algorithm. The average computation time in seconds is shown in column six, labelled av_t. Finally, the average percentage deviation from the critical path makespan is reported in the seventh column, labelled CPM_dev.

	$\sum EVO$	$\sum PSPLIB$	av_dev	$best\_sol$	av_t	CPM_dev
$j120$	74876	74263	0.75	301	35	31.94
$j90$	45911	45833	0.16	426	4.90	10.31
$j60$	38493	38405	0.17	415	1.87	10.94
$j30$	28352	28316	0.10	449	0.59	13.52

Table 1: Computational Results for $j30$, $j60$, $j90$ and $j120$

5 Concluding remarks

The computational experiments indicate that our procedure is effective. In all instance sets, our algorithm generated high quality solutions that offered clear improvements over the results obtained by the state-of-the-art heuristics for the RCPSP. The improvement is particularly significant for the set $j120$. Furthermore, the computing times required by our algorithm can be considered as quite reasonable.

References

[1] Blazewicz, J., Lenstra, J. K. and Rinooy Kan, A. H. G. Scheduling subject to resource constraints: Classification and complexity. *Discr. Appl. Math.*, 5, 11–24, 1983.

[2] Glover, F. And Laguna, M. *Tabu Search*. Kluwer Academic Publishers, 1997.

[3] Hartmann, S., Kolisch, R. Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem. *European J. Oper. Res.*, 127(2), 394–407, 2000.

[4] Kolisch, R. and Hartmann, S. *Heuristic algorithms for solving the resource-constrained project scheduling problem: Classification and computational analysis. In Weglarz, J. (Ed.), Project Scheduling. Recent Models, Algorithms and Applications*. Kluwer Academic Publishers, Boston, pp. 147-178, 1999.

[5] Kolisch, R. and Sprecher, A. PSPLIB - A project scheduling library. *European J. Oper. Res.*, 96, 205–216,(downlable from website http://www.bwl.uni-kiel.de/Prod/PSPLIB/index.html),1997.

[6] Merkle, D., Middendorf, M. And Schmeck, H. Ant Colony Optimization for Resource-Constrained Project Scheduling. *In Proceedings of the Genetic and Evolutionary Computation Conference*, Las Vegas, Nevada, 2000.

[7] Möhring, R. H., Schulz, A. S., Stork, F. And Uetz, M. Solving Project Scheduling Problems by Minimum Cut Computations. Technical Report 680-2000. *Fachbereich Mathematik. Technishe Universität Berlin.*, 2000.

[8] Nonobe, K. and Ibaraki, T. Formulation and Tabu Search Algorithm for the Resource Constrained Project Scheduling Problem (RCPSP), Technical report 99010, 1999

[9] Valls, V., Laguna. M., Lino, P., Pérez, A., and Quintanilla, S. *Project Scheduling with Stochastic Activity Interruptions. In Weglarz, J. (Ed.), Project Scheduling. Recent Models, Algorithms and Applications*. Kluwer Academic Publishers, Boston, pp. 333-354, 1999.

[10] Valls, V., Quintanilla, S. and Ballestín F. Resource-constrained Project Scheduling: A Critical Activity Reordering Heuristic. *European J. Oper. Res.*, 2001, to appear.