Method of Musical Composition and Static Topologies for Resource Constrained Project Scheduling: A Case Study

Rafaela Blanca Silva López, Rosa Elena Cruz Miguel, Eric Alfredo Rincón García, Roman Anselmo Mora Gutiérrez, Ponsich Antonin

Dpto. de Sistemas, Universidad Autónoma Metropolitana, Unidad Azcapotzalco
México D.F., Mexico
{rbsl,recm,rigaeral,mgra,aspo}@correo.azc.uam.mx

Abstract. Resource constrained project scheduling problems have a computational complexity that makes it difficult to obtain optimal solutions using exact methods. Thus, heuristic techniques have been used to generate feasible solutions in acceptable computational times. In this paper, we analyse static population topologies for an algorithm based on the Method of Musical Composition (MMC) to solve the problem of course scheduling in a university in a minimum number of quarters. We show that the social network topology used within the MMC operating mode has a significant influence on the performance of the algorithm.

Keywords: Resource constrained project scheduling problem, static population topologies, Method of Musical Composition.

1 Introduction

The resource constrained project scheduling problem (RCPSP) consists in scheduling a set of activities with deterministic processing times, resource requirements and precedence relations between activities. The aim is to find a schedule with minimum makespan (total project duration) respecting both precedence relationships and resource limits. The RCPSP has attracted increasing interest in both academic and engineering fields, including medical research [1], software development [2], audit scheduling [3] and market research interviewers scheduling [4]. The methods for solving the RCPSP have ranged from exact methods [5, 6] to heuristics techniques [7]. However only small sized instances of the RCPSP can be solved optimally using exact algorithms within reasonable time, due to its complexity which has been proved to be NP-hard [8]. Thus, in recent years, the need for solving large, realistic project instances has motivated a shift in research trends towards heuristic approaches.

In this paper, we consider a case study of the Autonomous Metropolitan University (UAM, Mexico) where a set of courses must be scheduled, in such a way that a student can achieve his/her degree in the minimum number of quarters. Due to the complexity of this problem we decide to apply a heuristic method to

find solutions within acceptable computing times. However, in recent decades a variety of heuristics have been developed [9], which seek for an improvement of the solution quality or a reduction of the computing time required to generate them. Thus we decided to use a recently proposed population-based heuristic technique known as Method of Musical Composition (MMC). Its performance has been shown to be satisfactory in the context of continuous optimization problems and its application to discrete problems seems a promising alternative. For more details about MMC, its advantages and drawbacks, and differences with other population-based techniques, we refer the reader to [10]. In order to improve the performance of the algorithm we varied the way an individual in the MMC interacts with its neighbors, as proposed in [11, 12]. We tested five commonly used neighborhood configurations and discovered that some of them, for instance, the "star" neighborhood, that performed better than the classical ones on a suite of standard test problems.

In the next section we describe the resource constrained project scheduling problem, while the university project is introduced in section 3. In section 4 we explain the features and operation mode of the MMC. The proposed implementation of the algorithm and result analysis are presented in section 5. Finally conclusions are presented in section 6.

2 The Resource Constrained Project Scheduling Problem

A project consists of a set $\mathcal{J}$ of N activities, $\mathcal{J} = \{1, ..., N\}$, and a set $\mathcal{R}$ of K renewable resources, $\mathcal{R} = \{1, ..., K\}$. Each resource $k \in \mathcal{R}$, is available in limited amounts, R_k, and is renewable from period to period. The duration of activity j is denoted by d_j. Each activity has to be processed without interruption. The precedence relationships are given by sets of immediate predecessors P_j indicating that an activity $j \in \mathcal{J}$ may not be started before each of its predecessors $i \in P_j$ is completed. Activity j requires r_{jk} units of resource k in each period. The activities $j = 1$ and $j = N$ are dummy activities, which represent the start and end of the project, respectively. The dummy activities are added to the project to act as source and sink of the project, and it is assumed that they do not request any resource and their durations are equal to zero.

A schedule can be presented as $\mathcal{S} = \{s_1, ..., s_N\}$, where s_i denotes the starting time of activity i, with $s_1 = 0$. The objective is to determine the starting time of each activity, so that the total project duration, s_N, is minimized, and both the precedence and the resource constraints are respected.

In consideration of the above assumptions the resource constrained project scheduling problem can be formulated as follows:

$$\min\{\max\{f_i : i = 1, 2, ..., N\}\} \tag{1}$$

subject to:

$$s_j - s_i \geq d_i \ \forall i \in P_j, \ j = 1, 2, ..., N \tag{2}$$

$$\sum_{A_t} r_{jk} \leq R_k, \ k = 1, 2, ..., K, \ t = s_1, s_2, ..., s_N \tag{3}$$

Where f_i is the completion time of activity i ($i = 1, ..., N$) and A_t is the set of ongoing activities at time t.

3 University Project

The UAM's study program consists of courses that must be completed in a predefined number of quarters. All courses have credits and, in some cases, prerequisites and other requirements. These requirements are divided into two groups:

R1. Enrolment to some courses is subject to the successful completion of other predefined classes (precedence constraints).
R2. Enrolment to some courses is subject to have a minimum credit numbers completed.

Besides, the maximum and minimum amount of credits necessary for a student to be able to enroll without special overload approval from his/her college office is restricted.

R3. First quarter, maximum 46 credits.
R4. The remaining quarters, maximum 60 credits.
R5. For all quarters, minimum 21 credits.

Recently UAM's study program was modified, resulting in the elimination and creation of courses, affecting the order in which courses should be completed. These changes required a revision of the new plan, both to create a new schedule for the courses in different quarters and to determine if the new plan could be completed within the time set by the university, i.e. 13 quarters. This work was carried out manually and required an effort of several days to design a solution able to meet the conditions labeled as **R1-R5**.

We decided to modify this problem in order to find the minimum number of quarters required for a student to complete his/her degree. In this case, the problem can be modelled as a RCPSP, where the objective is to determine the starting quarter of each course, so that the total career duration is minimized, while respecting precedence constraints (**R1-R2**) and resource constraints (**R3-R5**).

As mentioned previously, due to the computational complexity of these problems, most authors have focused on the development of heuristic solution methods to find good quality schedules. Thus, we decide to implement the MMC technique whose application in RCPSP has not been reported in the specialized literature.

4 Method of Musical Composition

The Method of Musical Composition (MMC) is a novel heuristic (see [13, 10]), designed according to the following ideas:

- Musical composition can be viewed as an algorithm, since this process uses rules, principles and a finite number of steps to create original music of some particular style.
- Musical composition is a creativity system, involving interactions among agents.
- Usually, agents learn from their experience and use what they learn for future decisions.

In the MMC algorithm, each solution is called a tune and is represented by an D-dimensional vector:

$$tune = [x_1\ x_2\ \ldots\ x_D] \tag{4}$$

Initially, a composer i in the society creates a set of tunes, namely an artwork, which is registered in a score matrix ($P_{\star,\star,i}$). Subsequently, and while the termination criterion is not met, the following steps are carried out: *(a)* the social network linking the different composers is modified; *(b)* composers exchange information (tunes) among each other; *(c)* each composer i decides whether or not using the received information (by including or not within its artwork a tune from other composer) and therefore builds his acquired knowledge (denoted by matrix $ISC_{\star,\star,i}$); *(d)* the i^{th} composer then builds his knowledge background ($KM_{\star,\star,i}$) as the union of $P_{\star,\star,i}$ and $ISC_{\star,\star,i}$; *(e)* the resulting knowledge matrix $KM_{\star,\star,i}$ is used as a basis for creating a new tune $x_{i,new}$; *(f)* finally, the composer makes a decision on whether updating or not his score matrix $P_{star,\star,i}$: based on the quality of $x_{i,new}$, this latter will replace the worst tune within $P_{\star,\star,i}$. The termination criterion typically consists of reaching a maximum number of arrangements ($max\_arrangement$), or iterations. The basic structure of the *MMC* metaheuristic is presented in Algorithm 1.

We refer the reader to [13] for a more complete description of the MMC algorithm.

5 Implementation and Numerical Results

5.1 Implementation

In this section, the specific implementation of the proposed algorithm, within the RCPSP framework, is described. Each solution is a schedule that can be represented as a vector $\mathcal{S} = (s_1, ..., s_N)$, where s_i denotes the quarter when course i begins.

Algorithm 1: MMC algorithm

```
1  Create an artificial society with rules of interaction among agents.
2  for each composer i in society do
3  |  Randomly initialize an artwork.
4  end
5  repeat
6  |  Update the artificial society of composers.
7  |  Exchange information between agents.
8  |  for each composer i in the society do
9  |  |  Update the knowledge matrix.
10 |  |  Generate and evaluate a new tune (x_{i,new}).
11 |  |  if x_{i,new} is better than the worst tune in the artwork of composer i
   |  |  (x_{i,worst}) then
12 |  |  |  Replace x_{i-worst} by x_{i,new} in the artwork.
13 |  |  end
14 |  end
15 |  Update the solution set.
16 until termination criterion is met;
```

1. First the algorithm generates for each composer a social network, i.e. a set of arcs representing the relationships between composers. In this case, we tested five commonly used topology networks: random, ring, linear, tree, and star, see next section for a detailed definition of these topologies. The social network remains fixed throughout the algorithm, so the process of society links updating (row 6 of the above-mentioned algorithm) is omitted.
2. Each composer creates a set of solutions using a uniform random number generator that provides integer numbers in the range [0,$Max_{Quarter}$].
3. Each composer i compares his worst tune, $\mathcal{S}_{i-worst}$, with composers' solutions in his/her social network. Then, he/she updates his/her acquired knowledge, $ISC_{\star,\star,i}$, with those solutions that are better than $\mathcal{S}_{i-worst}$.
4. Each composer i creates a new tune, $\mathcal{S}_{i,new}$, using three solutions. The first solution is the best tune in his/her knowledge background , $\mathcal{S}_{best} \in KM_{\star,\star,i}$. The second solution, $\mathcal{S}_{KM,rnd}$, is randomly selected in $KM_{\star,\star,i} - \{\mathcal{S}_{best}\}$. The third solution is randomly selected from his/her artwork, $\mathcal{S}_{i,rnd} \in P_{\star,\star,i}$. To create $\mathcal{S}_{i,new}$ the starting quarter of the $j-th$ course is randomly selected from the periods assigned in $\mathcal{S}_{best}$, $\mathcal{S}_{KM,rnd}$ and $\mathcal{S}_{i,rnd}$. The cost of $\mathcal{S}_{i,new}$ is evaluated in terms of the objective function and the number of unsatisfied restrictions.
5. Each composer i makes a greedy decision on whether updating or not his/her score matrix. If the new solution $\mathcal{S}_{i,new}$ has a lower cost than $\mathcal{S}_{i,rnd}$, $\mathcal{S}_{i,new}$ replaces $\mathcal{S}_{i,rnd}$.

5.2 Numerical Experiments

In this case we used an instance of 66 courses that must be scheduled in the minimum number of quarters, in such a way that restrictions **R1-R5** are satisfied.

Note that the size of this instance cannot be reduced since it would not result in a realistic challenging problem. Moreover, an exact solver, GUSEK[1], was applied to the instance proposed in this section. Neverthelees, the method could not converge after a 48 hours run, therefore confirming the NP-hard charateristics of the problem.

In Table 1 we present the 66 courses considered in this case. The first and fourth columns provide an identifying label for each course. The second and fifth columns provide the number of credits corresponding to each course. The third and sixth columns provide the precedence list for each course. In Table 2 we present seven courses where students are required to have completed a minimum number of credits to be enrolled.

Table 1. Courses, credits and precedence

Course	Credits	Precedence	Course	Credits	Precedence
1	4		34	9	19
2	9		35	9	27
3	3		36	3	29
4	4		37	9	34
5	3		38	9	8; 20
6	7		39	9	22; 32; 33
7	9	1	40	6	
8	6	2	41	6	
9	3	3	42	12	31; 36
10	9	6	43	8	31; 36
11	6	6	44	9	32
12	9	7	45	7	36
13	3	7	46	9	27; 37
14	6		47	9	35; 46
15	3	12; 13	48	12	46
16	9	11	49	9	22; 30
17	7	10; 11	50	6	
18	6	14	51	6	
19	9	16	52	9	27; 44
20	9	12	53	9	42; 47
21	7	17	54	6	50
22	8	17	55	7	42; 43; 46
23	12	17	56	8	43
24	6		57	6	42; 43; 46
25	6	16; 12	58	8	42; 43
26	9	19	59	3	54
27	9	23; 22	60	8	42; 53
28	3	24	61	6	
29	8	17	62	6	
30	9	17	63	7	55; 56; 57
31	8	22; 26	64	18	59
32	9	17	65	6	54
33	9	21	66	9	24; 43

[1] http://gusek.sourceforge.net/gusek.html

Table 2. Courses and minimum number of credits required

Course	Minimum credits
14	50
40	150
41	150
50	280
59	360
61	150
62	150

As mentioned previously, five social networks, based on different topologies, which have been commonly used to improve the performance of social algorithms [12], were considered for the implementation of the MMC-based algorithm:

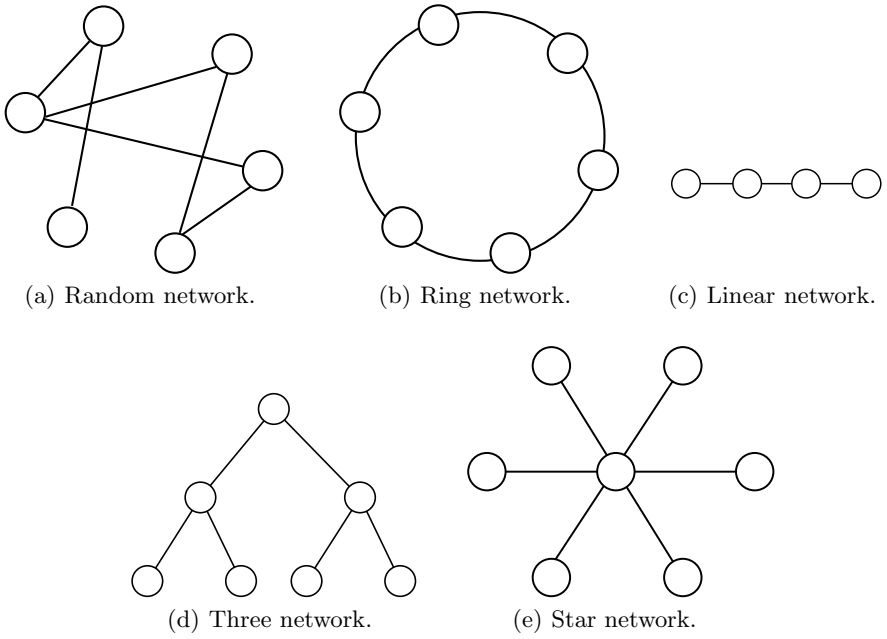
1. **Random**: Is composed by randomly generated arcs.
2. **Ring**: A circular list of composers (i.e. a permutation) is created, such that each composer sends (receives) information to his/her successor (from his predecessor).
3. **Linear**: Composers are arranged over a line, where most composers are connected to two other composers. However, the first and last composers are not connected as they would be in the ring structure.
4. **Tree**: One "root" composer connects to other composers, which in turn connect to other composers, forming a tree structure.
5. **Star**: One central composer is connected to all the remaining composers.

We applied a brute force calibration procedure by several executions of the algorithm using a random social network. This way, the number of composers and tunes for each composer were set to 25 and 3, respectively. We decided to use these values for all social networks in order to provide a fair comparison basis between different topologies. Finally, the number of generations was set to 90,000.

Besides, in order to deal with the stochastic effect inherent to heuristic techniques, 100 independent executions were performed for each topology. Each run produced a single solution and the resulting 100 solutions were subsequently divided into two groups according to the number of quarters required to schedule the 66 courses: solutions that required 11 quarters and solutions that required 12 or more quarters. We must remark that all generated solutions satisfied **R1-R5**. The results for all topologies are presented in Table 3.

Table 3. Results for different topologies

Topology	11 quarters	12 or more quarters
Random	9	91
Ring	12	88
Linear	5	95
Tree	6	94
Star	61	30



(a) Random network. (b) Ring network. (c) Linear network.

(d) Three network. (e) Star network.

Fig. 1. Example of the networks used in this article

It is clear, from the observation of Table 3, that the star and ring topologies outperform the other ones, since they produce 11-quarters solutions for more than 60% and 10% of the runs, respectively. Using this information we deduced that the social network is an important parameter for the performance of the algorithm. We decided to modify the algorithm to apply a variant of the best topologies. Thus two rings and two stars topologies were used while other parameters remain unchanged. In this case, two circles (stars) with 13 and 12 composers were considered, and a link between two composers, one in each circle (star), was used to exchange knowledge between both "sub-societies". As can be seen in table 4, these variants improved the performance of the proposed algorithm, where the two star topologies clearly provided the best results.

Table 4. Results for two circles and two stars topologies

Topology	11 quarters	12 or more quarters
Two rings	43	57
Two stars	78	22

6 Conclusion

In this paper we presented a resource constrained project scheduling problem, which consists of placing a set of courses in the minimum number of quarters, subject to precedence relationships and limited resources. The example presented in this work includes 66 courses and was created using real-world information from the Mexican Autonomous Metropolitan University's study program. 55 courses has at least one precedence restriction, and 7 courses requires a minimum number of completed credits to be available for students. Quarters with limited resources, 46 and 60 credits, are used to schedule the courses, and an additional requirement, at least 21 credits for each quarter, is also involved.

A discrete implementation of the a heuristic technique, namely the Method of Musical Composition, was proposed for the solution of this problem. To analyze the performance of the algorithm different static social networks, based on five topologies, were used. After several executions we observed that star or ring networks were acceptable options. In order to improve the performance of the algorithm we decided to apply a variant of these topologies. Thus two circles and two stars topologies were implemented. The computational experiments proved that a two stars topology produces better solutions than its counterparts. These results indicate that the MMC based algorithm proposed is able to find feasible solutions for the RCPSP described in this work, but its performance highly depends on the social network topology. Our future work includes the study of dynamic topologies within our algorithm.

References

1. Hartmann, S.: Scheduling medical research experiments – an application of project scheduling methods (1997)
2. Alba, E., Francisco Chicano, J.: Software project management with gas. Inf. Sci. **177**(11) (June 2007) 2380–2401
3. Dodin, B., Elimam, A., Rolland, E.: Tabu search in audit scheduling. The a. gary anderson graduate school of management, The A. Gary Anderson Graduate School of Management. University of California Riverside (1996)
4. Hartmann, S.: PROJECT SCHEDULING:. Lecture notes in economics and mathematical systems. Springer (1999)
5. Sprecher, A.: Resource-constrained project scheduling: exact methods for the multi-mode case. Lecture notes in economics and mathematical systems. Springer-Verlag (1994)
6. Zamani, M.R.: A high-performance exact method for the resource-constrained project scheduling problem. Computers & OR **28**(14) (2001) 1387–1401
7. Fang, C., Wang, L.: An effective shuffled frog-leaping algorithm for resource-constrained project scheduling problem. Computers & OR **39**(5) (2012) 890–901
8. Blazewicz, J., Lenstra, J., Kan, A.: Scheduling subject to resource constraints: classification and complexity. Discrete Applied Mathematics **5**(1) (1983) 11 – 24

9. Brownlee, J.: Clever Algorithms: Nature-Inspired Programming Recipes. Lulu Enterprises Incorporated (2011)
10. Mora-Gutiérrez, R.A., Ramírez-Rodríguez, J., Rincón-García, E.A., Ponsich, A., Herrera, O.: An optimization algorithm inspired by social creativity systems. Computing **94**(11) (2012) 887–914
11. Kennedy, J., Mendes, R.: Population structure and particle swarm performance. In: Proceedings of the Congress on Evolutionary Computation (CEC 2002). Volume 2., IEEE Press (2002) 1671–1676
12. Kennedy, J., Mendes, R.: R.: Neighborhood topologies in fully-informed and best-ofneighborhood particle swarms. In: In: Proceedings of the 2003 IEEE SMC Workshop on Soft Computing in Industrial Applications (SMCia03, IEEE Computer Society (2003) 45–50
13. Mora-Gutiérrez, R., Ramírez-Rodrguez, J., Rincón-García, E.: An optimization algorithm inspired by musical composition. Artificial Intelligence Review (2012) 1–15