A Novel Matheuristic for the Multi-Site Resource-Constrained Project Scheduling Problem

Tamara Bigler, Mario Gnägi and Norbert Trautmann

University of Bern, Switzerland
tamara.bigler@pqm.unibe.ch, mario.gnaegi@pqm.unibe.ch,
norbert.trautmann@pqm.unibe.ch

Keywords: Multi-site resource-constrained project scheduling problem, Matheuristic.

1 Introduction

In the well-known resource-constrained project scheduling problem (RCPSP), a set of precedence-related project activities must be scheduled such that the project makespan is minimized subject to limited resource availabilities. We consider a planning problem that extends the RCPSP by involving different sites. Some of the renewable resource units are mobile and can be transferred between sites while others are permanently located at one site. It is assumed that the mobile resource units are available at the site at which they are used for the first time. Transportation times apply a) when a mobile resource unit is transferred from one site to another or b) when the output of an activity is transferred to another site where one of its successor activities will be processed. In the latter case, the successor activity can only start once the outputs of all predecessor activities have arrived at the respective site. These transfers enable the sharing of resources among sites, e.g., the sharing of medical staff among hospitals. As activities from multiple sites are scheduled simultaneously in the multi-site RCPSP, the number of activities to be scheduled is often greater than in the single-site RCPSP.

The literature comprises some exact and heuristic approaches for this planning problem. Laurent et al. (2017) introduced a discrete-time mathematical model that they applied to instances involving 5 to 30 activities. Because their model did not seem to scale well, they developed four metaheuristics, which are based on an activity list and a site list representation of the solution. One metaheuristic conducts a local search, one metaheuristic is based on simulated annealing, and two metaheuristics perform an iterated local search. They applied the four metaheuristics to instances comprising 30 to 120 activities. It turned out that the iterated local search and the simulated annealing metaheuristics perform best. Gnägi and Trautmann (2019; 2021) formulated a continuous-time mathematical model that they applied to the same instances as Laurent et al. (2017) comprising 30 activities. Their model was able to derive a large number of new best known solutions for these instances.

In this paper, we propose a novel matheuristic for the multi-site RCPSP. In the matheuristic, the activities are scheduled by performing the following two steps in an iterative manner. First, a subset of activities that will be scheduled in the next iteration is selected based on standard priority rules from the literature. Second, the selected activities are scheduled by solving a relaxation of the model of Gnägi and Trautmann (2019; 2021). The matheuristic obtains high-quality solutions for instances comprising 30 activities and 2 or 3 sites. Among the 960 tested instances from the literature, it derives new best known solutions for 164 instances.

The remainder is structured as follows. In Section 2, we illustrate the planning problem with an example. In Section 3, we outline the novel matheuristic. In Section 4, we report the computational results. In Section 5, we conclude and give an outlook on future research.

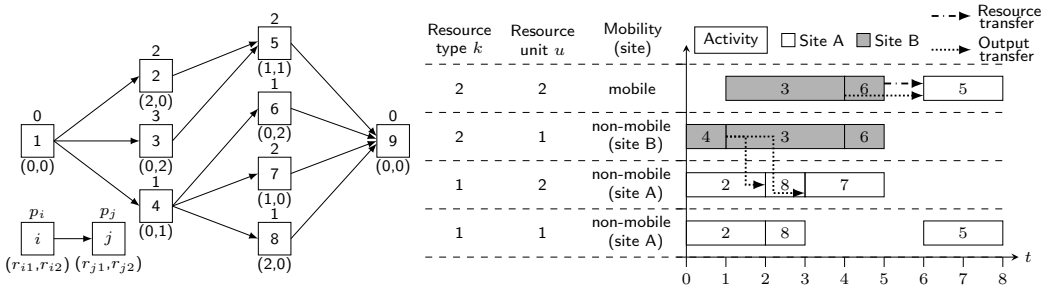

Fig. 1. Example: activity-on-node network (left) and an optimal solution (right)

2 Illustrative example

In this section, we illustrate the planning problem with an example that comprises seven real activities $\{2, \ldots, 8\}$. The fictitious activities 1 and 9 represent the project start and completion, respectively. The left part of Figure 1 shows an activity-on-node network. Each node in the network represents one activity and each arrow represents one precedence-relationship. Moreover, the example includes two sites A and B between which we assume a transportation time of one time unit. Each of the two resource types $k = 1$ and $k = 2$ comprises two resource units $u = 1$ and $u = 2$. Both resource units of resource type $k = 1$ are non-mobile and permanently located at site A. One unit ($u = 2$) of resource type $k = 2$ is mobile, and the other unit ($u = 1$) is permanently located at site B. The activity-on-node network further shows for each activity i its resource requirement r_{ik} for the two resource types and its duration p_i. The right part of Figure 1 shows an optimal solution for the illustrative example. Each line represents a resource unit u of a resource type k, and the rectangles represent the activities. Each real activity is assigned to at least one resource unit and exactly one site. The activities $\{2, 5, 7, 8\}$ are executed at site A while the activities $\{3, 4, 6\}$ are executed at site B. The resource transfers are indicated by a dash-dotted arrow, e.g., between activities 6 and 5 which take place at a different site; thus, the commonly used resource unit $u = 2$ of resource type $k = 2$ must be transferred from site B to site A. The output transfers are indicated by a dotted arrow, e.g., between activities 4 and 8, which are precedence-related and take place at a different site; thus, the output of activity 4 must be transferred from site B to site A before activity 8 can start.

3 Novel matheuristic

In this section, we describe the novel matheuristic in more detail and illustrate it with the example from Section 2. The matheuristic is based on the continuous-time sequencing/ natural-date model of Gnägi and Trautmann (2019), subsequently referred to as GT19. The model involves continuous start-time variables that indicate the start time of an activity, and binary site-selection variables that represent the execution site for each activity. Moreover, it includes binary resource-assignment variables that assign the activities to the resource units, and binary sequencing variables y_{ij} that indicate the sequence between pairs of activities i and j. Hence, $y_{ij} = 1$ means that activity i is scheduled before activity j.

Our matheuristic is based on a variant of GT19, in which some sequencing variables are relaxed, i.e., they can take any fractional value between 0 and 1, and some activities are locked, i.e., the site-selection and resource-assignment variables of these activities as well as the sequencing variables between all pairs of these activities are fixed to their values in the current solution of the relaxation. Before the first iteration, the matheuristic derives promising initial values for the site-selection variables by solving a relaxation of GT19,

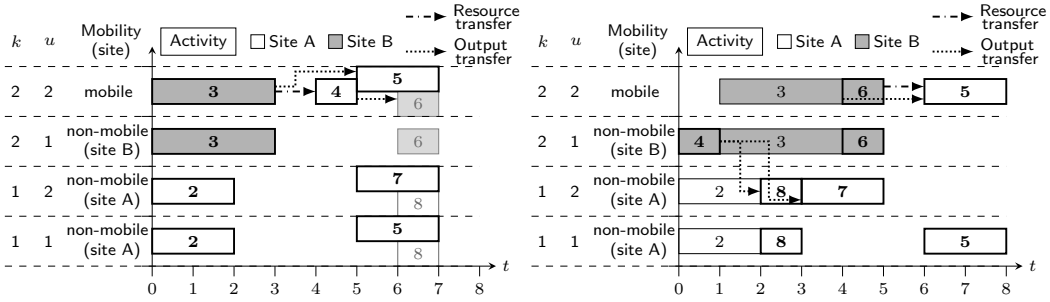

Fig. 2. Example: schedule after Iteration 1 (left) and schedule after Iteration 2 (right)

in which all sequencing variables are relaxed. Then, the matheuristic iteratively schedules changing subsets of activities in a rolling-horizon manner as follows. First, $2b$ activities are selected based on the latest starting time (LST) priority rule and the latest finishing time (LFT) priority rule as a tie breaker. Second, a relaxation of GT19 is solved, in which only the sequencing variables among and between the $2b$ selected and the locked activities are defined as binary; all remaining sequencing variables are relaxed. Moreover, the initial values for the site-selection variables of the $2b$ selected activities are provided. The solution of this relaxation provides a schedule in which the sequence of the $2b$ selected and the locked activities is determined. The remaining activities (subsequently referred to as eligible activities), however, may overlap among each other as well as with the selected and the locked activities. Finally, the initial values for the site-selection variables are updated based on the current solution of the relaxation, and the b activities with the highest priority (according to the combined LST and LFT priority rule) among the $2b$ selected activities are locked. Consequently, b activities remain selected. Then, the next iteration starts by selecting b additional activities from the eligible activities based on the combined LST and LFT priority rule. If there are no eligible activities to select in this step, the matheuristic stops.

In the illustrative example, we set $b = 3$. Figure 2 illustrates the resulting two iterations. The selected activities are marked in bold, the eligible activities are transparent, and the locked activities are not highlighted. In Iteration 1, the activities $\{1, 3, 2, 4, 5, 7\}$ are selected. Figure 2 (left) shows the schedule obtained in Iteration 1, in which some of the eligible activities $\{6, 8, 9\}$ overlap with some of the selected activities. This is feasible in this iteration because all sequencing variables between the eligible activities and the selected activities are relaxed. Next, the activities $\{1, 3, 2\}$ are locked and the activities $\{6, 8, 9\}$ are selected in addition to the already selected activities $\{4, 5, 7\}$. As all activities are either selected or locked in Iteration 2, the sequencing variables between all activities are defined as binary and the conflicts between the activities $\{5, 6\}$, $\{5, 8\}$, and $\{7, 8\}$ must be resolved. Figure 2 (right) illustrates the schedule obtained in Iteration 2. Compared to Iteration 1, activity 4 is scheduled at a different time, at a different site, and on a different resource unit. Without the site change of activity 4 in Iteration 2, an additional transportation time would apply, which would delay the project makespan by one time unit. After performing Iteration 2, there are no eligible activities and the matheuristic stops.

4 Computational results

In this section, we present the computational results. The matheuristic was tested on 960 instances comprising 30 activities and 2 or 3 sites that belong to the test set MSj30. This set has been adapted to the multi-site context by Laurent et al. (2017) from the instances of the PSPLIB by Kolisch and Sprecher (1996). We implemented the matheuristic

Table 1. Computational results

	Overall	Laurent et al. (2017)		Gnägi and Trautmann (2021)	
# Sites	# New BKS	# Better	∅ Gap [%]	# Better	∅ Gap [%]
2	63	110	0.70	94	0.38
3	101	143	0.53	144	-0.57

in Python 3.7 and used the Gurobi 9.1 solver. We prescribed a time limit of 300s to the Gurobi solver in each iteration. Moreover, we set $b = 5$.

Table 1 summarizes the results obtained. The solutions of the matheuristic are compared to the best known solutions that Laurent et al. (2017) published for their metaheuristics on their website and to the solutions Gnägi and Trautmann (2021) reported for their continuous-time mathematical model. We group the results by the number of sites (2 or 3). The column # New BKS corresponds to the number of instances for which the matheuristic found a new best known solution. The columns # Better report the number of instances for which the matheuristic obtained a better solution than the approaches of Laurent et al. (2017) or Gnägi and Trautmann (2021), respectively, and the columns ∅ Gap report the average gap of the matheuristic solutions to the solutions of the approaches of Laurent et al. (2017) or Gnägi and Trautmann (2021), respectively. Even though the average gap to the benchmark approaches is overall slightly positive, our matheuristic is able to derive 164 new best known solutions in a shorter average running time than the benchmark approaches.

5 Conclusions and outlook

In this paper, we studied a variant of the RCPSP that involves multiple sites. This extension allows for resource pooling among sites and introduces two types of transportation times that must be considered. We developed a matheuristic for this problem that derives high-quality solutions for a standard test set of instances with 30 activities and 2 or 3 sites.

In future research, the matheuristic could be extended by an LP-based improvement step involving the so-called justification technique. This technique has been shown to improve schedules considerably while running times increase only slightly (cf. Valls et al., 2005). Also, the planning problem could be extended to take into account resources that are required for the transfer of the output of an activity to another site (Krüger and Scholl, 2010).

References

Gnägi, M., and Trautmann, N., 2019, "A continuous-time mixed-binary linear programming formulation for the multi-site resource-constrained project scheduling problem.", In: Wang, M., Li, J., Tsung, F., and Yeung, A. (eds.): *Proceedings of the 2019 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, Macau, pp. 382–365.

Gnägi, M., and Trautmann, N., 2021, "A continuous-time model for the multi-site resource-constrained project scheduling problem", In: *Proceedings of the 17th International Conference on Project Management and Scheduling (PMS)*, Toulouse, to appear.

Kolisch, R., and Sprecher, A., 1996, "PSPLIB-a project scheduling problem library.", *European Journal of Operational Research*, Vol. 96(1), pp. 205–216.

Krüger, D., and Scholl, A., 2010, "Managing and modelling general resource transfers in (multi-)project scheduling", *OR Spectrum*, Vol. 32(2), pp. 369–394.

Laurent, A., Deroussi, L., Grangeon, N., and Norre, S., 2017, "A new extension of the RCPSP in a multi-site context: Mathematical model and metaheuristics.", *Computers & Industrial Engineering*, Vol. 112, pp. 634–644.

Valls, V., Ballestin, F., and Quintanilla, S., 2005, "Justification and RCPSP: A technique that pays.", *European Journal of Operational Research*, Vol. 165(2), pp. 375–386.