Flex Serv Manuf J
DOI 10.1007/s10696-016-9260-8

MIP formulations for an application of project scheduling in human resource management

Tom Rihm[1] · **Norbert Trautmann**[1] · **Adrian Zimmermann**[1]

Abstract In the literature, various discrete-time and continuous-time mixed-integer linear programming (MIP) formulations for project scheduling problems have been proposed. The performance of these formulations has been analyzed based on generic test instances. The objective of this study is to analyze the performance of discrete-time and continuous-time MIP formulations for a real-life application of project scheduling in human resource management. We consider the problem of scheduling assessment centers. In an assessment center, candidates for job positions perform different tasks while being observed and evaluated by assessors. Because these assessors are highly qualified and expensive personnel, the duration of the assessment center should be minimized. Complex rules for assigning assessors to candidates distinguish this problem from other scheduling problems discussed in the literature. We develop two discrete-time and three continuous-time MIP formulations, and we present problem-specific lower bounds. In a comparative study, we analyze the performance of the five MIP formulations on four real-life instances and a set of 240 instances derived from real-life data. The results indicate that good or optimal solutions are obtained for all instances within short computational time. In particular, one of the real-life instances is solved to optimality. Surprisingly, the continuous-time formulations outperform the discrete-time formulations in terms of solution quality.

✉ Norbert Trautmann
norbert.trautmann@pqm.unibe.ch

Tom Rihm
tom.rihm@pqm.unibe.ch

Adrian Zimmermann
adrian.zimmermann@pqm.unibe.ch

[1] University of Bern, Bern, Switzerland

Published online: 28 October 2016

Keywords OR in the service industries · Human resource management · Project scheduling · Mixed-integer programming · Comparative analysis

1 Introduction

Over the past decades, mixed-integer linear programming (MIP) methods have been significantly improved (cf., e.g., Koch et al. 2011; Bixby 2012) and successfully applied to a large variety of real-life scheduling problems in manufacturing and services. Two major advantages of MIP methods are the flexibility to account for changes in the problem setting and the possibility to obtain upper or lower bounds on the solutions. In general, different formulations can be used to model the same planning problem. Because the performance of MIP approaches is determined by the underlying formulation (cf., e.g., Vielma 2015), alternative formulations should be considered for each planning problem.

In this paper, we investigate an assessment center planning problem (ACP). This problem was reported to us by a human resource management service provider that organizes assessment centers (AC) for firms. The goal of an AC is to evaluate some candidates' job-related skills and abilities for one or several open positions (cf., e.g., Collins et al. 2003). In an AC, each candidate performs multiple tasks, and for each task, a prescribed number of assessors (i.e., psychologists or managers) is required. Some tasks involve role play and additionally require a prescribed number of actors. For example, the actors might represent unhappy customers with whom the candidate must interact. Tasks sometimes require a preparation time during which only the candidate is present. During the execution of the task, the candidate is joined by the assessors and the actor. Some tasks include a subsequent evaluation during which the assessors and the actors discuss their observations. This evaluation time can differ between assessors and actors. Each candidate takes a lunch break within a prescribed time window. When assigning assessors to tasks, the following rules must be considered: each candidate should be observed by approximately half the number of assessors; if a candidate and an assessor know each other personally, no observation is allowed, which is called a no-go relationship. Assessors are expensive, and hence, their total waiting time should be minimized. Because the assessors meet before the start and after the completion of all tasks and lunch breaks, this objective corresponds to minimizing the total duration of the AC (in what follows the AC duration for short). The planning problem consists of (1) scheduling all tasks and a lunch break for each candidate and (2) determining which assessors are assigned to which candidate during which task such that the AC duration is minimized.

The ACP can be interpreted as an extension of the resource-constrained project scheduling problem (RCPSP). The RCPSP consists of scheduling a set of activities subject to completion-start precedence and renewable-resource constraints such that the project duration is minimized. For the ACP, each candidate's tasks and lunch break correspond to project activities, and the candidates, assessors, and actors represent renewable resources. However, the ACP does not involve precedence relationships among the activities, but the above-described additional constraints. In

the literature, different MIP formulations have been proposed for the RCPSP. In discrete-time (DT) formulations, the planning horizon is divided into a set of time intervals of equal length, and the activities can only start or end at the endpoints of these intervals. Conversely, in continuous-time (CT) formulations, the activities can start at any point in time. The DT formulations usually involve binary time-indexed variables. However, the meaning of these variables differ between the formulations, e.g., so-called pulse variables indicate whether an activity starts or ends at a specific point in time (cf. Pritsker et al. 1969; Christofides et al. 1987; Kopanos et al. 2014), and on/off variables specify whether an activity is in progress at a given time (cf. Kaplan 1988; Mingozzi et al. 1998; Kopanos et al. 2014). The CT formulations differ with regard to the modeling of the resource constraints, e.g., Artigues et al. (2003) use resource-flow variables, and Kopanos et al. (2014) use overlapping variables. For a comprehensive overview of different MIP formulations for the RCPSP, we refer to Artigues et al. (2015).

In this paper, we provide two DT formulations and three CT formulations for the ACP. The two DT formulations are based on pulse variables (DT-P) and on/off variables (DT-O), respectively. The three CT formulations use assessor-assignment variables (CT-A), resource-flow variables (CT-F), and overlapping variables (CT-O), respectively, to model the resource constraints. Moreover, we provide problem-specific lower bounds. The different MIP formulations are tested on four real-life instances and 240 test instances based on real-life data. For all instances, good or optimal solutions are obtained within short computational time. In detail, formulation CT-A consistently outperforms the other four formulations in terms of solution quality. However, using DT-P, the best MIP-based lower bounds are obtained. Furthermore, only with DT-P, optimality is proven for one of the real-life instances within the prescribed time limit. Nevertheless, in contrast to the RCPSP, the CT formulations provide better solutions than the DT formulations.

The remainder of this paper is structured as follows. In Sect. 2, we describe the ACP using an illustrative example and relate the ACP to the RCPSP. In Sect. 3, we provide an overview of the related literature. In Sect. 4, we present the MIP formulations for the ACP. In Sect. 5, we derive the problem-specific lower bounds. In Sect. 6, we discuss the design and the results of our comparative analysis. In Sect. 7, we provide some concluding remarks and an outlook on future research.

2 Planning problem

In Sect. 2.1, we describe the problem features of the ACP in detail and illustrate them through an example. In Sect. 2.2, we discuss the relation between the ACP and the RCPSP.

2.1 Illustration of the planning problem

In our illustrative example, the participants of the AC are as follows: there are three candidates, C_1, C_2, and C_3; four assessors, A_1, A_2, A_3, and A_4; and an actor, P_1. A

no-go relationship exists between candidate C_3 and assessor A_2. Each of the three candidates must perform the three tasks E_1, E_2, and E_3, and take a lunch break.

The tasks of the illustrative example are listed in Table 1. The durations of the tasks are stated in 5-min time units. Tasks E_1 and E_3 require two assessors, and task E_2 requires one assessor. Task E_1 involves role play and requires one actor. Tasks E_1 and E_2 include a preparation time, and tasks E_1 and E_3 include an evaluation time. Figure 1 shows at which time during the execution of task E_1 the candidate, the assessors, and the actor are required. The evaluation time differs between the assessors and the actor. Due to fairness and objectivity considerations, no waiting times are allowed between the preparation, the execution, and the evaluation. A waiting time for a candidate would increase the preparation time, whereas a waiting time for the assessors and actors could bias their evaluations of the candidate.

The earliest and latest possible start times for the lunch break are 20 and 30, respectively. The duration of the lunch break is 6 time units. Because each candidate has a lunch break and performs each of the three tasks exactly once, a total of 12 activities are considered. Table 2 shows the indices of these activities.

The rules for assigning assessors to candidates are as follows: each candidate should be observed by at least half of the total number of assessors rounded down and by at most half of the total number of assessors rounded up plus one. The lower limit ensures an objective overall evaluation for each candidate, and the upper limit is motivated by fairness considerations. The difference between the upper and lower limits facilitates the assessor assignment without affecting fairness. The number of times that an assessor can observe the same candidate is not limited. In the illustrative example, each candidate must be observed by 2–3 different assessors. Additionally, because a no-go relationship exists, candidate C_3 can never be observed by assessor A_2.

An optimal schedule for the illustrative example is presented in Fig. 2. The dotted lines indicate the earliest and latest start times for the lunch breaks, and the solid line indicates the AC duration. Whether an assessor has been assigned to a candidate at least once is indicated by a checkmark (✔).

Table 1 Tasks of illustrative example

Task	E_1	E_2	E_3
Required number of assessors	2	1	2
Required number of actors	1	–	–
Duration	20	10	12
Duration of preparation time (candidates)	8	3	–
Duration of execution time	8	7	8
Duration of evaluation time (assessors)	4	–	4
Duration of evaluation time (actors)	2	–	–

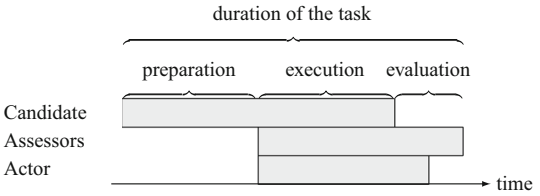
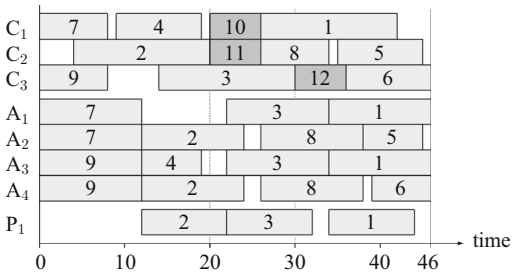


Fig. 1 Varying requirements for candidate, assessors, and actor during task E_1

Table 2 Activity indices of the illustrative example

Candidate	Task			Lunch break
	E_1	E_2	E_3	
C_1	1	4	7	10
C_2	2	5	8	11
C_3	3	6	9	12



Assignments			
	Candidate		
Assessor	C_1	C_2	C_3
A_1	✓		✓
A_2	✓	✓	
A_3	✓		✓
A_4		✓	✓

Fig. 2 Optimal schedule of the illustrative example (*left*) and corresponding assessor assignment (*right*)

2.2 Relation to the RCPSP

The ACP includes many problem features of the well-known RCPSP. Both planning problems consider activities that require prescribed amounts of some renewable resources during their execution. In the case of the ACP, the execution of each task and the lunch break for each candidate correspond to a project activity, and the candidates, assessors, and actors can be interpreted as renewable resources. The ACP does not involve precedence relationships among the activities.

In the RCPSP, only the capacities and not the individual units of the renewable resources are considered. However, in the ACP, the assessor-assignment rules require that all activities that use a particular resource unit can be identified. Therefore, the assessor-assignment rules cannot be formulated in the RCPSP.

If each assessor is interpreted as a renewable resource with unit capacity, then alternative execution modes must be defined in order to represent the alternative assessor assignments. This corresponds to the multi-mode extension of the RCPSP (MRCPSP). Because each candidate must be observed by approximately half the number of the assessors, the assessor assignments interdepend. Such interdependencies between modes are not considered in the MRCPSP. Before assigning any

assessors to a candidate, all modes are feasible. However, selecting the modes for some activities causes several of the modes of the other activities to be infeasible.

3 Literature review

In Sect. 3.1, we provide an overview of different MIP formulations for the RCPSP which can be used as the basis for MIP formulations of the ACP. In Sect. 3.2, we discuss recent works that focus on comparing MIP formulations for extensions of the RCPSP and for specific real-life problems.

3.1 MIP formulations for the RCPSP

In DT formulations, binary time-indexed variables are used that indicate the start, end, or the state (e.g., in progress) of an activity at a specific time. For DT formulations, three types of binary variables can be distinguished (cf. Artigues et al. 2015). Beside the pulse and on/off variables described in the Introduction, there are step variables that indicate whether an activity starts at or before a specific point in time (cf. Klein 2000; Bianco and Caramia 2013). Furthermore, Bianco and Caramia (2013) introduce continuous variables that specify the percentage of completion of the activities at each point in time.

In CT formulations, the activities can start or finish at any time rather than at predefined time points such as in DT. Artigues et al. (2003) present a CT formulation based on resource flows. Besides the continuous start-time variables, this formulation requires two additional sets of variables. The first set consists of binary sequencing variables that determine for each pair of activities whether one precedes the other or whether both are executed in parallel. The second set consists of continuous resource-flow variables for modeling the resource constraints. Kopanos et al. (2014) present another CT formulation with continuous start-time variables, binary sequencing variables, and binary overlapping variables. In combination with the sequencing variables, the overlapping variables are used to model the resource constraints. Other CT formulations are based on events (e.g. Koné et al. 2011) or on minimal forbidden sets (e.g. Alvarez-Valdes and Tamarit 1993).

For the RCPSP, the performances of these different MIP formulations are compared in Bianco and Caramia (2013), Koné et al. (2011), and Kopanos et al. (2014). They all use generic test instances, which are provided in, e.g., Kolisch and Sprecher (1997) and Vanhoucke et al. (2008). For these test instances, Koné et al. (2011) and Kopanos et al. (2014) show that the performance is primarily affected by the number of activities and the length of the planning horizon. The performances of the DT formulations are negatively affected by the length of the planning horizon because the numbers of variables and constraints depend on the number of time points considered. In contrast, the performances of the CT formulations are negatively affected by the number of activities because the number of sequencing variables increases exponentially with the number of activities. Typically, DT-based formulations are the most competitive and yield the best LP relaxations. However,

no formulation consistently dominates the others, as different formulations perform better for different problem settings.

In this study, we adapt different RCPSP formulations such that they can be applied to the ACP. From the DT formulations, we select the RCPSP formulations of Pritsker et al. (1969) and Kopanos et al. (2014). The basic DT formulation of Pritsker et al. (1969) still performs very well compared to newer formulations (cf., e.g., Koné et al. 2011). Kopanos et al. (2014) show that their two DT formulations outperform other DT formulations presented in the literature. Their DT formulations differ with regard to the modeling of the precedence constraints. For the ACP, these two formulations are identical because there are no precedence constraints. From the CT formulations, we adapt the formulations of Artigues et al. (2003) and Kopanos et al. (2014). The CT formulation of Artigues et al. (2003) performs well compared to other CT formulations if there are specific problem characteristics such as long activity durations (cf., e.g., Koné et al. 2011). Kopanos et al. (2014) show that their two CT formulations outperform other CT formulations presented in the literature; we adapted their best-performing CT formulation.

3.2 Comparative studies of MIP formulations

In addition to the aforementioned comparative studies of the RCPSP, the performances of alternative MIP formulations have also been compared for various other planning problems. In the following, we provide an overview of such comparative studies for extensions of the RCPSP and for some real-life problems.

Some extensions of the RCPSP for which alternative MIP formulations have been compared are as follows. In Koné et al. (2013), the performances of alternative DT and CT formulations are compared for an extension of the RCPSP with so-called storage resources. Storage resources are consumed and produced at the project activities' start times and completion times, respectively. As in Koné et al. (2011), the authors conclude that no MIP formulation consistently yields the best results. A comparative performance analysis of alternative DT formulations for the RCPSP with flexible resource profiles is provided in Naber and Kolisch (2014). With flexible resource profiles, the resource utilization of an activity is not constant but rather can be adjusted from period to period. The results of the comparative study in Naber and Kolisch (2014) indicate that an MIP formulation based on Bianco and Caramia (2013) dominates all other DT formulations. In the study of Zapata et al. (2008), alternative DT and CT formulations for the MRCPSP with multiple projects are compared. The authors conclude that the best MIP formulation depends on the specific characteristics of each problem instance.

Comparative analyses have also been conducted for MIP formulations in real-life applications. Stefansson et al. (2011) develop DT and CT formulations for a large-scale production scheduling problem originating from a pharmaceutical producer. In this problem, customers order specific products, which need to be produced in a four-stage production process such that the requested quantity and delivery date of the order are met. The results obtained for eight test instances indicate that the CT formulation obtains better solutions within shorter computational time than the DT formulation. Furthermore, in Chen et al. (2012), a comparative analysis of different

mixed-integer nonlinear programming formulations for the scheduling of crude-oil refinement operations is presented. The planning problem includes several processing steps, from unloading marine vessels to producing various crude-oil based products. In a recent study, Ambrosino et al. (2015) evaluated the performance of two alternative MIP formulations for the multi-port master bay plan problem. This problem involves the placement of containers on a containership such that the overall berthing costs of the ship's multi-port journey are minimized.

4 MIP formulations for the ACP

In this section, we present our five MIP formulations for the ACP. The notation of the MIP formulations is provided in Tables 3 and 4. In Sect. 4.1, we present the CT formulation that uses the assessor-assignment decisions to model the resource constraints (CT-A). In Sect. 4.2, we derive the CT formulation with resource-flow variables (CT-F). In Sect. 4.3, we present the CT formulation with overlapping variables (CT-O). In Sects. 4.4 and 4.5, we present the DT formulation with pulse variables (DT-P) and the DT formulation with on/off variables (DT-O), respectively.

4.1 Formulation CT-A

In this section, we present the continuous-time formulation that uses the assessor-assignment decisions to model the resource constraints (CT-A). In a preliminary version of this MIP formulation (cf. Grüter et al. 2014), each activity is split into

Table 3 Sets and parameters of the MIP formulations

C	Set of candidates
A	Set of assessors
P	Set of actors
N	Set of candidate-assessor pairs (c, a) with a no-go relationship
I	Set of activities $i = 1, \ldots, n$ (including lunch breaks)
I_c	Set of activities that require candidate $c \in C$
I^A, I^P	Set of activities that require assessors (I^A) and actors (I^P)
I^L	Set of lunch breaks
ES^L, LS^L	Earliest (ES^L) and latest (LS^L) start time for the lunch breaks
p_i	Duration of activity i
p_i^C	Preparation time of activity i for candidates
p_i^A, p_i^P	Evaluation time of activity i for assessors (p_i^A) and actors (p_i^P)
r_i^A, r_i^P	Number of assessors (r_i^A) and actors (r_i^P) required by activity i
M	Sufficiently large number
T	Upper bound on the duration of the assessment center

Table 4 Variables of the MIP formulations

D	AC duration
S_i	Start time of activity i for the candidate
X_{it}	$\begin{cases} = 1, & \text{if activity } i \text{ starts at time point } t; \\ = 0, & \text{otherwise} \end{cases}$
Y_{ij}^C	$\begin{cases} = 1, & \text{if activity } i \text{ is performed before } j > i \text{ by a candidate;} \\ = 0, & \text{otherwise} \end{cases}$
Y_{ij}^A	$\begin{cases} = 1, & \text{if activity } i \text{ is performed before } j \neq i \text{ by the assessors;} \\ = 0, & \text{otherwise} \end{cases}$
Y_{ij}^P	$\begin{cases} = 1, & \text{if activity } i \text{ is performed before } j \neq i \text{ by the actors;} \\ = 0, & \text{otherwise} \end{cases}$
Z_{ia}^A	$\begin{cases} = 1, & \text{if assessor } a \text{ is assigned to activity } i; \\ = 0, & \text{otherwise} \end{cases}$
Z_{ip}^P	$\begin{cases} = 1, & \text{if actor } p \text{ is assigned to activity } i; \\ = 0, & \text{otherwise} \end{cases}$
V_{ca}	$\begin{cases} = 1, & \text{if assessor } a \text{ is assigned to candidate } c \text{ at least once;} \\ = 0, & \text{otherwise} \end{cases}$
F_{ij}^C	$\begin{cases} = 1, & \text{if a candidate is sent from activity } i \text{ to } j; \\ = 0, & \text{otherwise} \end{cases}$
F_{ij}^A	Number of assessors sent from activity i to j
F_{ij}^P	Number of actors sent from activity i to j
$\widehat{Y}_{ij}$	$\begin{cases} = 1, & \text{if activity } i \text{ starts before or at the same time as } j \text{ for assessors;} \\ = 0, & \text{otherwise} \end{cases}$
O_{ji}^A	$\begin{cases} = 1, & \text{if activity } j \text{ finishes after the start of activity } i \text{ for assessors;} \\ = 0 \text{ or } 1, & \text{otherwise} \end{cases}$
O_{ji}^P	$\begin{cases} = 1, & \text{if activity } j \text{ finishes after the start of activity } i \text{ for actors;} \\ = 0 \text{ or } 1, & \text{otherwise} \end{cases}$
W_{it}	$\begin{cases} = 1, & \text{if } i \text{ is processed at time } t \text{ by the candidates;} \\ = 0, & \text{otherwise} \end{cases}$

several sub-activities to model the preparation, the execution, and the evaluation times. However, this results in an unnecessary large number of variables and constraints. In the following, we model the ACP without splitting the activities.

We distinguish between three types of resources: candidates, assessors, and actors. Each candidate is modeled as a renewable resource with capacity 1. The set of all assessors (actors) is modeled as one renewable resource with a capacity that equals the number of assessors (actors). Due to the capacity of 1, the resource constraints for the candidates are modeled using binary sequencing variables, i.e., $Y_{ij}^C = 1$ ($Y_{ij}^C = 0$) if activity i (j) is completed some time before the start of activity j (i) by the corresponding candidate. For the assessors and actors, the resource constraints are modeled using binary sequencing variables (Y_{ij}^A and Y_{ij}^P), and binary assignment variables (Z_{ia}^A and Z_{ip}^P). For the assessors, the sequencing variable Y_{ij}^A is equal to 1 if activity i is completed some time before the start of activity j. Otherwise, Y_{ij}^A is 0, i.e., activities i and j are processed simultaneously or j finishes

some time before i begins. Because the ACP does not include precedence relationships, there are no prescribed values for the sequencing variables. The assignment variable Z^A_{ia} is equal to 1 if assessor a is assigned to activity i; otherwise $Z^A_{ia} = 0$. For the actors, the sequencing and assignment variables (Y^P_{ij} and Z^P_{ip}) are interpreted in the same way. Finally, variable V_{ca} is used to model the assessor-assignment rule, i.e., $V_{ca} = 1$ if assessor a is assigned to candidate c at least once.

The objective is to minimize the AC duration D.

$$\text{Min } D$$

The duration corresponds to the latest completion time of an activity that is defined by constraints (1).

$$D \geq S_i + p_i \quad (i \in I) \tag{1}$$

Constraints (2)–(5) determine the resource-feasible start times of the activities. Constraints (2) are binding if candidate c completes activity i before the start of activity j. Otherwise, constraints (3) are binding. Because candidate c is not required during the evaluation time, activity j can start at most p^A_i time units before the completion of activity i (cf. Fig. 3).

$$S_j \geq S_i - M + (p_i - p^A_i + M)Y^C_{ij} \quad (c \in C, i,j \in I_c : i<j) \tag{2}$$

$$S_i \geq S_j - M + (p_j - p^A_j + M)(1 - Y^C_{ij}) \quad (c \in C, i,j \in I_c : i<j) \tag{3}$$

Constraints (4) and (5) enforce a sequence of activities for the assessors and actors, respectively. In the case that activity i is executed before activity j by the assessors, constraints (4) are binding. Because the assessors are not required during the preparation time, activity j can start at most p^C_j time units before the completion of activity i (cf. Fig. 4). Similarly, constraints (5) are binding if activity i is executed before activity j by the actors. For the actors, activity i is completed after $p_i - p^A_i + p^P_i$ time units. Activity j can start at most p^C_j time units before that completion time (cf. Fig. 5).

$$S_j \geq S_i - M + (p_i - p^C_j + M)Y^A_{ij} \quad (i,j \in I^A : i \neq j) \tag{4}$$

$$S_j \geq S_i - M + (p_i - p^A_i + p^P_i - p^C_j + M)Y^P_{ij} \quad (i,j \in I^P : i \neq j) \tag{5}$$

Constraints (6) ensure that the lunch breaks are scheduled within the prescribed time window.

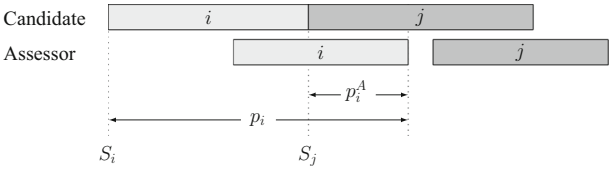


Fig. 3 Minimum time lag between start times of activities i and j for candidates

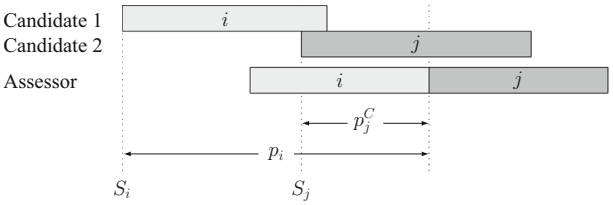


Fig. 4 Minimum time lag between start times of activities i and j for assessors

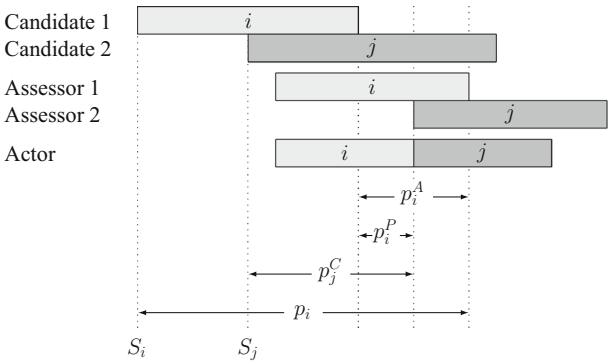


Fig. 5 Minimum time lag between start times of activities i and j for actors

$$ES^L \leq S_i \leq LS^L \quad (i \in I^L) \tag{6}$$

Constraints (7) and (8) imply that the required numbers of assessors and actors are assigned to each activity.

$$\sum_{a \in A} Z_{ia}^A = r_i^A \quad (i \in I^A) \tag{7}$$

$$\sum_{p \in P} Z_{ip}^P = r_i^P \quad (i \in I^P) \tag{8}$$

Constraints (9) and (10) link the assignment variables to the sequencing variables. If the same assessor a or the same actor p is assigned to two activities i and j, then a sequence between these two activities is enforced.

$$Y_{ij}^A + Y_{ji}^A \geq Z_{ia}^A + Z_{ja}^A - 1 \quad (i,j \in I^A, a \in A : i<j) \tag{9}$$

$$Y_{ij}^P + Y_{ji}^P \geq Z_{ip}^P + Z_{jp}^P - 1 \quad (i,j \in I^P, p \in P : i<j) \tag{10}$$

Constraints (11) and (12) ensure that either activity i precedes activity j, j precedes i, or i and j are processed in parallel.

$$Y_{ij}^A + Y_{ji}^A \leq 1 \quad (i,j \in I^A : i<j) \tag{11}$$

$$Y_{ij}^{P} + Y_{ji}^{P} \leq 1 \quad (i,j \in I^{P} : i<j) \tag{12}$$

Constraints (13) enforce that the number of assessors assigned to each candidate lies within the bounds imposed by the assessor-assignment rule.

$$\left\lfloor \frac{|A|}{2} \right\rfloor \leq \sum_{a \in A} V_{ca} \leq \left\lceil \frac{|A|}{2} \right\rceil + 1 \quad (c \in C) \tag{13}$$

Constraints (14) determine whether an assessor a has been assigned to a candidate c at least once. V_{ca} must be equal to 1 if assessor a is assigned to at least one activity that requires candidate c. If assessor a is never assigned to an activity that requires candidate c, then V_{ca} must be equal to 0.

$$\sum_{i \in I_c \setminus I^{L}} \frac{Z_{ia}^{A}}{|I_c \setminus I^{L}|} \leq V_{ca} \leq \sum_{i \in I_c \setminus I^{L}} Z_{ia}^{A} \quad (c \in C, a \in A) \tag{14}$$

Finally, constraints (15) model the no-go relationships.

$$V_{ca} = 0 \quad ((c,a) \in N) \tag{15}$$

In sum, formulation (CT-A) reads as follows:

$$\text{(CT-A)} \quad \begin{cases} \text{Min} & D & \\ \text{s.t.} & (1)-(15) & \\ & S_i \geq 0 & (i \in I) \\ & Y_{ij}^{C} \in \{0,1\} & (c \in C, i,j \in I_c : i<j) \\ & Y_{ij}^{A} \in \{0,1\} & (i,j \in I^{A} : i \neq j) \\ & Y_{ij}^{P} \in \{0,1\} & (i,j \in I^{P} : i \neq j) \\ & V_{ca} \in \{0,1\} & (c \in C, a \in A) \\ & Z_{ia}^{A} \in \{0,1\} & (i \in I^{A}, a \in A) \\ & Z_{ip}^{P} \in \{0,1\} & (i \in I^{P}, p \in P) \end{cases}$$

4.2 Formulation CT-F

In this section, we present the continuous-time formulation with resource-flow variables (CT-F), which is based on the RCPSP formulation of Artigues et al. (2003). This MIP formulation was first proposed in Zimmermann and Trautmann (2014). The following explanations closely follow that study.

To model the resource flows, formulation CT-F requires the dummy activities 0 and $n+1$; both have a duration of zero, and $r_0^{A} = r_{n+1}^{A} = |A|$ ($r_0^{P} = r_{n+1}^{P} = |P|$) is equal to the total number of available assessors (actors). Variable F_{ij}^{C} (F_{ij}^{A}, F_{ij}^{P}) denotes the quantity of candidates (assessors, actors) sent from activity i (upon completion) to activity j (at the beginning). This resource flow prevents the corresponding activities from being executed simultaneously. For the assessors

(actors), the sequencing variable Y_{ij}^{A} (Y_{ij}^{P}) is equal to 1 if some assessors (actors) are sent from activity i to activity j. Because each activity requires exactly one candidate, any flow of candidates between two activities will be either 0 or 1. Since the resource-flow variable F_{ij}^{C} is defined as binary, this variable is used simultaneously as a resource-flow and as a sequencing variable. As a sequencing variable, F_{ij}^{C} equals 1 if and only if activity j is executed after activity i.

The following constraints have to be considered. Constraints (16) determine resource-feasible start times of the activities for the candidates. The feasible start times of the activities for the assessors and actors are determined as in formulation CT-A. Constraints (16) are binding if a candidate is sent from activity i to activity j ($F_{ij}^{C} = 1$).

$$S_j \geq S_i - M + (p_i - p_i^{A} + M)F_{ij}^{C} \quad (c \in C; i,j \in I_c : i \neq j) \tag{16}$$

Constraints (17)–(22) are the resource-flow conservation constraints. Constraints (17) ensure that each activity i sends 1 unit of resource $c \in C$ to either an activity $j \neq i$ or the dummy activity $n+1$ (if activity i is the last activity performed by candidate c). Constraints (18) ensure that each activity j receives 1 unit of resource $c \in C$ from either an activity $i \neq j$ or the dummy activity 0 (if activity j is the first activity performed by candidate c).

$$\sum_{j \in I_c \cup \{n+1\}: j \neq i} F_{ij}^{C} = 1 \quad (c \in C; i \in I_c \cup \{0\}) \tag{17}$$

$$\sum_{i \in I_c \cup \{0\}: i \neq j} F_{ij}^{C} = 1 \quad (c \in C; j \in I_c \cup \{n+1\}) \tag{18}$$

Constraints (19)–(22) conserve the resource flow of assessors and actors, respectively. The number of assessors r_i^{A} (actors r_i^{P}) required by activity i must be sent to and received from other activities that require the same resource.

$$\sum_{j \in I^{A} \cup \{n+1\}: j \neq i} F_{ij}^{A} = r_i^{A} \quad (i \in I^{A} \cup \{0\}) \tag{19}$$

$$\sum_{j \in I^{P} \cup \{n+1\}: j \neq i} F_{ij}^{P} = r_i^{P} \quad (i \in I^{P} \cup \{0\}) \tag{20}$$

$$\sum_{i \in I^{A} \cup \{0\}: i \neq j} F_{ij}^{A} = r_j^{A} \quad (j \in I^{A} \cup \{n+1\}) \tag{21}$$

$$\sum_{i \in I^{P} \cup \{0\}: i \neq j} F_{ij}^{P} = r_j^{P} \quad (j \in I^{P} \cup \{n+1\}) \tag{22}$$

Constraints (23) and (24) link the resource-flow variables to the sequencing variables for assessors and actors, respectively.

$$F_{ij}^{A} \leq \min(r_i^A, r_j^A) Y_{ij}^{A} \quad (i,j \in I^A : i \neq j) \tag{23}$$

$$F_{ij}^{P} \leq \min(r_i^P, r_j^P) Y_{ij}^{P} \quad (i,j \in I^P : i \neq j) \tag{24}$$

The sequencing variables Y_{ij}^{A} and Y_{ij}^{P} are only used to link the flow variables F_{ij}^{A} and F_{ij}^{P} to the start times of the activities. The flow variables F_{ij}^{A} and F_{ij}^{P} can be greater than 1. For this reason, they cannot be used as sequencing variables.

Constraints (1), which determine the AC duration D, and the sequencing constraints for the assessors (4) and actors (5), and constraints (6), which specify the time window for the lunch breaks, are also included. The same applies to the assessor-assignment constraints (7), (9), and (11)–(15).
In sum, formulation (CT-F) reads as follows:

$$\text{(CT-F)} \quad \begin{cases} \text{Min} & D & \\ \text{s.t.} & (16)-(24) & \\ & (1),(4)-(7),(9) & \\ & (11)-(15) & \\ & S_i \geq 0 & (i \in I) \\ & F_{ij}^{C} \in \{0,1\} & (c \in C; i,j \in I_c \cup \{0, n+1\} : i \neq j) \\ & F_{ij}^{A} \geq 0 & (i,j \in I^A \cup \{0, n+1\} : i \neq j) \\ & F_{ij}^{P} \geq 0 & (i,j \in I^P \cup \{0, n+1\} : i \neq j) \\ & Y_{ij}^{A} \in \{0,1\} & (i,j \in I^A : i \neq j) \\ & Y_{ij}^{P} \in \{0,1\} & (i,j \in I^P : i \neq j) \\ & V_{ca} \in \{0,1\} & (c \in C, a \in A) \\ & Z_{ia}^{A} \in \{0,1\} & (i \in I^A, a \in A) \end{cases}$$

4.3 Formulation CT-O

In this section, we present the continuous-time formulation with overlapping variables (CT-O), which is based on the RCPSP formulation of Kopanos et al. (2014).

For activities that cannot be processed in parallel (i.e., two activities which require the same candidate), we use the sequencing variables Y_{ij}^{C}. For activities that can be processed in parallel, the resource constraints are modeled with the following binary variables.

- For the assessors and the actors, we introduce the sequencing variables $\widehat{Y}_{ij}$. Specifically, $\widehat{Y}_{ij} = 1$ if activity i starts before or at the same time as activity j for the assessors. These sequencing variables are not defined separately for assessors and actors, because the activities start at the same time for them.

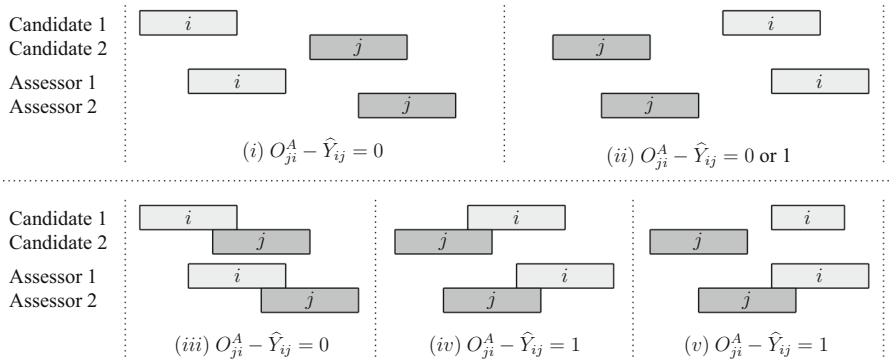


Fig. 6 Five possible cases (*i*)–(*v*) that illustrate the values of the sequencing and overlapping variables

- For the assessors, we introduce the overlapping variables O^A_{ji}. Specifically, $O^A_{ji} = 1$ if activity j finishes after the start of activity i for the assessors. If activity j finishes before or at the same time as activity i starts, then O^A_{ji} is equal to 0 or 1. The overlapping variables for the actors O^P_{ji} are defined in the same way.

To illustrate how these variables jointly determine whether two activities $i, j \in I^A$ are processed in parallel by the assessors, several possible cases are depicted in Fig. 6. For case (*ii*), the variable O^A_{ji} can be equal to zero or one, but for cases (*iv*) and (*v*), the variable must be equal to one.

Constraints (25) determine the resource-feasible start times of the activities for the candidates. Constraints (26) ensure that either activity i precedes activity j, or j precedes i. In contrast to constraints (2) and (3), the sequencing variables Y^C_{ij} are used for any pair of activities involving the same candidate.

$$S_i + p_i - p^A_i \leq S_j + MY^C_{ji} \quad (c \in C, i, j \in I_c : i \neq j) \tag{25}$$

$$Y^C_{ij} + Y^C_{ji} = 1 \quad (c \in C, i, j \in I_c : i > j) \tag{26}$$

Constraints (27)–(29) determine the resource-feasible start times of the activities which can be processed in parallel. Thereby, parameter λ is used to exclude some symmetric solutions, i.e., for two activities $i > j$ which start at the same time, it is specified that $\widehat{Y}_{ji} = 1$ and $\widehat{Y}_{ij} = 0$. As proposed in Kopanos et al. (2014), we set $\lambda = 0.1$.

$$S_j + p^C_j \leq S_i + p^C_i + M\widehat{Y}_{ij} \quad (i, j \in I^A : i > j) \tag{27}$$

$$S_i + p^C_i + \lambda \leq S_j + p^C_j + (M + \lambda)\widehat{Y}_{ji} \quad (i, j \in I^A : i > j) \tag{28}$$

$$\widehat{Y}_{ij} + \widehat{Y}_{ji} = 1 \quad (i, j \in I^A : i > j) \tag{29}$$

Constraints (30) and (31) link the overlapping variables to the start times of the activities.

$$(S_j + p_j) - (S_i + p_i^C) \leq MO_{ji}^A \quad (i,j \in I^A : i \neq j) \tag{30}$$

$$(S_j + p_j - p_j^A + p_j^P) - (S_i + p_i^C) \leq MO_{ji}^P \quad (i,j \in I^P : i \neq j) \tag{31}$$

Constraints (32) and (33) ensure that all activities that are executed in parallel do not require more than the available number of assessors and actors, respectively. Thereby, the term $O_{ji}^A - \widehat{Y}_{ij} = 1$ if activity j starts before activity i and if both activities overlap for the assessors. The same applies to the actors.

$$r_i^A + \sum_{j \in I^A : j \neq i} r_j^A (O_{ji}^A - \widehat{Y}_{ij}) \leq |A| \quad (i \in I^A) \tag{32}$$

$$r_i^P + \sum_{j \in I^P : j \neq i} r_j^P (O_{ji}^P - \widehat{Y}_{ij}) \leq |P| \quad (i \in I^P) \tag{33}$$

Constraints (34) and (35) ensure that the terms $O_{ji}^A - \widehat{Y}_{ij}$ and $O_{ji}^P - \widehat{Y}_{ij}$ are greater than or equal to zero.

$$\widehat{Y}_{ij} \leq O_{ji}^A \quad (i,j \in I^A : i \neq j) \tag{34}$$

$$\widehat{Y}_{ij} \leq O_{ji}^P \quad (i,j \in I^P : i \neq j) \tag{35}$$

Constraints (36) link the sequencing and overlapping variables to the assignment variables. If the same assessor a is assigned to two activities i and j, then both activities cannot overlap for the assessors.

$$(O_{ji}^A - \widehat{Y}_{ij}) + Z_{ia}^A + Z_{ja}^A \leq 2 \quad (a \in A, i,j \in I^A : i \neq j) \tag{36}$$

Constraints (1), which determine the AC duration D, and constraints (6), which specify the time window for the lunch breaks, are also included. The same applies to the assessor-assignment constraints (13)–(15).

In sum, formulation (CT-O) reads as follows:

$$\text{(CT-O)} \quad \begin{cases} \text{Min} & D & \\ \text{s.t.} & (25)-(36) & \\ & (1),(6),(7),(13)-(15) & \\ & S_i \geq 0 & (i \in I) \\ & Y_{ij}^C \in \{0,1\} & (c \in C, i,j \in I_c : i \neq j) \\ & \widehat{Y}_{ij} \in \{0,1\} & (i,j \in I^A : i \neq j) \\ & O_{ji}^A \in \{0,1\} & (i,j \in I^A : i \neq j) \\ & O_{ji}^P \in \{0,1\} & (i,j \in I^P : i \neq j) \\ & V_{ca} \in \{0,1\} & (c \in C, a \in A) \\ & Z_{ia}^A \in \{0,1\} & (i \in I^A, a \in A) \end{cases}$$

4.4 Formulation DT-P

In this section, we present the discrete-time formulation with pulse variables (DT-P), which is based on the RCPSP formulation of Pritsker et al. (1969). This formulation involves the discretization of the planning horizon into uniform time intervals. The endpoints of a time interval are denoted by the time points t and $t+1$, respectively ($t = 0, \ldots, T-1$). Binary pulse variables X_{it} state if activity i starts at time t. For each time point t, resource constraints are formulated that ensure that the resource capacities are not violated. We extend the resource constraints of the RCPSP formulation such that the preparation and evaluation times of the AC activities are considered.

For the ACP, the following constraints have to be taken into consideration. The AC duration corresponds to the latest completion time of an activity, which is defined by constraints (37).

$$D \geq \sum_{t=0}^{T-p_i} (t + p_i) X_{it} \quad (i \in I) \tag{37}$$

Constraints (38) and (39) ensure that each activity starts once. Furthermore, constraints (39) state that the lunch breaks are scheduled within the prescribed time window.

$$\sum_{t=0}^{T-p_i} X_{it} = 1 \quad (i \in I \backslash I^L) \tag{38}$$

$$\sum_{t=ES^L}^{LS^L} X_{it} = 1 \quad (i \in I^L) \tag{39}$$

Constraints (40) to (42) ensure that the resource capacities are not violated. Constraints (40) ensure that each candidate performs at most one activity at the same time t. Candidate c performs activity i at time t if the activity started between time $t - (p_i - p_i^A) + 1$ and t. Constraints (41) and (42) ensure that all activities that are scheduled in parallel do not require more than the maximum available numbers of assessors and actors, respectively. An assessor performs activity i at time t if the activity started between time $t - p_i + 1$ and $t - p_i^C$. An actor performs activity i at time t if the activity started between time $t - (p_i - p_i^A + p_i^P) + 1$ and $t - p_i^C$.

$$\sum_{i \in I_c} \sum_{\tau=\max(0, t-p_i+p_i^A+1)}^{t} X_{i\tau} \leq 1 \quad (c \in C, t = 0, \ldots, T) \tag{40}$$

$$\sum_{i \in I^A} \sum_{\tau=\max(0, t-p_i+1)}^{t-p_i^C} r_i^A X_{i\tau} \leq |A| \quad (t = 0, \ldots, T) \tag{41}$$

$$\sum_{i \in I^P} \sum_{\tau=\max(0,t-p_i+p_i^A-p_i^P+1)}^{t-p_i^C} r_i^P X_{i\tau} \leq |P| \quad (t = 0, \ldots, T) \tag{42}$$

Additionally, the assessor-assignment constraints (7), (9), (11), and (13)–(15) are also included. Constraints (43) link the variables X_{it} to the sequencing variables Y_{ij}^A.

$$\sum_{t=0}^{T-p_j} tX_{jt} \geq \sum_{t=0}^{T-p_i} tX_{it} - M + (p_i - p_j^C + M) Y_{ij}^A \quad (i,j \in I^A : i \neq j) \tag{43}$$

In sum, formulation (DT-P) reads as follows:

$$\text{(DT-P)} \quad \begin{cases} \text{Min} & D & \\ \text{s.t.} & (37)-(43) & \\ & (7), (9), (11), (13)-(15) & \\ & X_{it} \in \{0,1\} & (i \in I, t = 0, \ldots, T) \\ & Y_{ij}^A \in \{0,1\} & (i,j \in I^A : i \neq j) \\ & V_{ca} \in \{0,1\} & (c \in C, a \in A) \\ & Z_{ia}^A \in \{0,1\} & (i \in I^A, a \in A) \end{cases}$$

4.5 Formulation DT-O

In this section, we present the discrete-time formulation with on/off variables (DT-O), which is based on the RCPSP formulation of Kopanos et al. (2014). For the RCPSP, Kopanos et al. (2014) extend the formulation of Pritsker et al. (1969) with binary on/off variables W_{it}, which specify if activity i is in progress at time t. With these variables, the resource constraints can be modeled in a different manner than in Pritsker et al. (1969).

For the ACP, we extend the formulation DT-P (cf. Sect. 4.4) with binary on/off variables. Due to the preparation and the evaluation time, these on/off variables must be defined individually for candidates, assessors, and actors. However, this results in a large number of additional variables, which has a negative impact on the performance. For this reason, we only define the on/off variables for the candidates, and take the resource constraints of DT-P for the assessors and the actors. Hence, the resource constraints (40) for the candidates are replaced by constraints (44)–(46).

Constraints (44) ensure that each candidate performs at most one activity at a time.

$$\sum_{i \in I_c : t \leq T - p_i^A - 1} W_{it} \leq 1 \quad (c \in C, t = 0, \ldots, T) \tag{44}$$

Constraints (45) link the pulse variables X_{it} to the on/off variables W_{it}.

$$W_{it} = \sum_{\tau=\max(0,t-p_i+p_i^A+1)}^{t} X_{i\tau} \quad (i \in I, t = 0, \ldots, T - p_i^A - 1) \tag{45}$$

Constraints (46) are valid equalities that tighten the formulation.

$$\sum_{t=0}^{T-p_i^A-1} W_{it} = p_i - p_i^A \quad (i \in I) \tag{46}$$

In sum, formulation (DT-O) reads as follows:

$$\text{(DT-O)} \quad \begin{cases} \text{Min} & D & \\ \text{s.t.} & (44)-(46) & \\ & (37)-(39), (41)-(43) & \\ & (7), (9), (11), (13)-(15) & \\ & X_{it} \in \{0,1\} & (i \in I, t = 0, \ldots, T) \\ & W_{it} \in \{0,1\} & (i \in I, t = 0, \ldots, T - p_i^A - 1) \\ & Y_{ij}^A \in \{0,1\} & (i,j \in I^A : i \neq j) \\ & V_{ca} \in \{0,1\} & (c \in C, a \in A) \\ & Z_{ia}^A \in \{0,1\} & (i \in I^A, a \in A) \end{cases}$$

5 Lower bounds

In this section, we derive some lower bounds for the AC duration. In Sect. 5.1, we present four lower bounds based on the assessors' workload. In Sect. 5.2, we present two lower bounds based on the candidates' workload.

5.1 Lower bounds based on the assessors' workload

In this section, we present four different lower bounds ($LB_1, \ldots, LB_4$) that are based on the assessors' workload. In contrast to lower bounds LB_1 and LB_2, lower bounds LB_3 and LB_4 consider the no-go relationships.

Lower bound LB_1 corresponds to the average workload of the assessors increased by the shortest preparation time of an activity. This preparation time is included because the assessors are never required before that time. The lower bound LB_1 reads as follows.

$$LB_1 = \left\lceil \sum_{i \in I^A} \frac{r_i^A (p_i - p_i^C)}{|A|} \right\rceil + \min_{i \in I^A} p_i^C$$

Lower bound LB_2 is obtained by considering only the activities that require two assessors. The total workload of these activities must be completed by an even

number of assessors. Hence, if the number of assessors $|A|$ is odd, then the following lower bound LB_2 is valid.

$$LB_2 = \left\lceil \sum_{i \in I^A : r_i^A = 2} \frac{2(p_i - p_i^C)}{|A| - 1} \right\rceil + \min_{i \in I^A} p_i^C$$

Lower bound LB_3 takes the no-go relationships of each assessor into consideration. The workload of all activities to which assessor a cannot be assigned due to no-go relationships is evenly distributed among the remaining $|A| - 1$ assessors and increased by the shortest preparation time. For each assessor $a \in A$, this corresponds to a lower bound.

$$LB_3 = \max_{a \in A} \left\lceil \sum_{c \in C : (c,a) \in N} \sum_{i \in I_c} \frac{r_i^A (p_i - p_i^C)}{|A| - 1} \right\rceil + \min_{i \in I^A} p_i^C$$

Lower bound LB_4 combines the underlying ideas of LB_2 and LB_3. We only consider activities that require two assessors and for which the corresponding candidates have a no-go relationship with assessor a. For these activities, an even number of assessors is required at any time. However, if the number of assessors is even and assessor a cannot be assigned to these activities due to the no-go relationships, it follows that one assessor $a^* \neq a$ is not needed. Hence, the workload of all activities that require two assessors and to which assessor a cannot be assigned is evenly distributed among the remaining $|A| - 2$ assessors. Again, the shortest preparation time of an activity is added to increase the lower bound. Hence, if the number of assessors $|A|$ is even, then lower bound LB_4 is valid.

$$LB_4 = \max_{a \in A} \left\lceil \sum_{c \in C : (c,a) \in N} \sum_{i \in I_c : r_i^A = 2} \frac{2(p_i - p_i^C)}{|A| - 2} \right\rceil + \min_{i \in I^A} p_i^C$$

5.2 Lower bounds based on the candidates' workload

In this section, we present two lower bounds for the AC duration based on the candidates' workload. The first lower bound (LB_5) is valid in general, and the second lower bound (LB_6) is only valid under certain conditions. Because each candidate must perform the same tasks, we do not need to differentiate between different candidates. Hence, in the following, we consider the tasks to be executed by each candidate and the lunch break rather than activities for individual candidates. The set of tasks and the lunch break are denoted by Q and l, respectively. It should be noted that the lunch break is not included in Q. Let p_q, p_q^C, and p_q^A be the duration, the preparation time, and the assessors' evaluation time of task $q \in Q$, respectively. The duration of the lunch break is p_l, and its preparation time (p_l^C) and evaluation time (p_l^A) are zero.

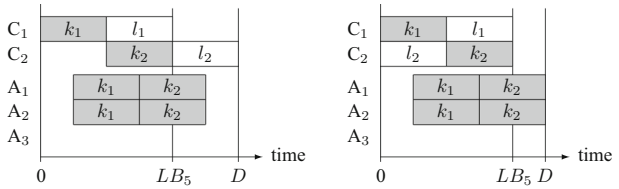


Fig. 7 Schedules of an example with (*left*) and without (*right*) waiting time for the candidates

Because the tasks and the lunch break must be performed sequentially, lower bound LB_5 is valid.

$$LB_5 = \sum_{q \in Q \cup \{l\}} \left(p_q - p_q^A \right)$$

The term $\min_{q \in Q \cup \{l\}} p_q^A$ could be added to LB_5 because the AC cannot end before all tasks and the lunch break are completed. However, the evaluation time of the lunch break is always equal to zero and, thus, this term is always zero. The lunch break cannot be excluded from this term, because each candidate can have the lunch break at the end if the latest possible start time is not violated.

To motivate lower bound LB_6, we first consider an illustrative example with two candidates and three assessors. Each candidate has to perform a task (activities k_1 and k_2) that requires two assessors and a lunch break (activities l_1 and l_2); activities k_1 and k_2 cannot be scheduled in parallel due to the limited number of assessors. Figure 7 depicts two feasible schedules for this example. In the schedule on the left, both candidates have the lunch break at the end. Due to the limited number of assessors, candidate C_2 has a waiting time. In this case, the AC duration D corresponds to the lower bound LB_5 plus the waiting time. In the schedule on the right, candidate C_2 performs the lunch break first. In this case, the AC duration D correspond to the lower bound LB_5 plus the evaluation time of the task.

In this example, either a candidate has a waiting time, or the last activity of a candidate does not correspond to the lunch break. With this in mind, we propose lower bound LB_6, which is valid under certain conditions. According to our industry partner, these conditions are fulfilled by a considerable number of real-life instances.

Theorem 1 *Let r be a task with the shortest evaluation time. If* (*i*) $\lfloor |A|/2 \rfloor < |C|$ *and* (*ii*) *all tasks except task r require two or more assessors, then the following lower bound is* valid.

$$LB_6 = \delta_0 + \min(\delta_1, \delta_2)$$

whereas

$$\delta_0 = \sum_{q \in Q \cup \{l\}} \left(p_q - p_q^A \right)$$

$$\delta_1 = \min_{q \in Q \setminus \{r\}} p_q^A$$

$$\delta_2 = \min_{q \in Q \setminus \{r\}} (p_q - p_q^C - p_q^A) + \min_{q \in Q \setminus \{r\}} p_q^A - \max(p_l, p_r - p_r^A)$$

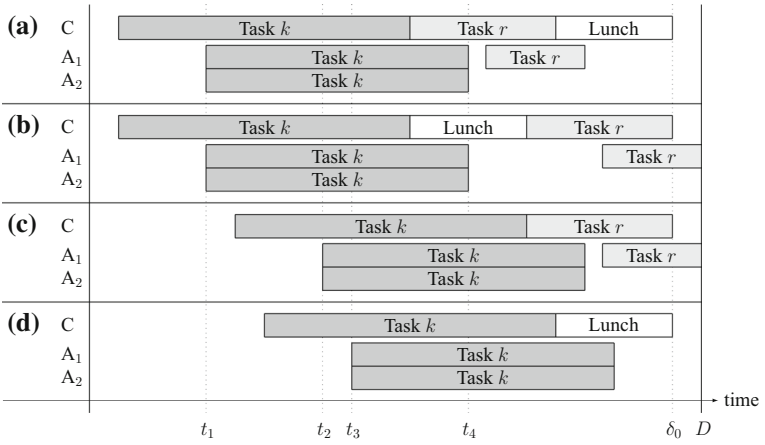


Fig. 8 All possible orders of the last activities and corresponding assessor requirements

Proof If the conditions (*i*) and (*ii*) hold for a given problem instance, any feasible solution belongs either to case 1 or to case 2.

- Case 1: The last activity of at least one candidate does not correspond to a lunch break or an activity of task *r*. It results that after the candidate completes this last activity, the assessors have an evaluation time of at least δ_1. Hence, $\delta_0 + \delta_1$ is a lower bound if the solution belongs to case 1.
- Case 2: The last activity of each candidate either corresponds to a lunch break or an activity of task *r*. We show that in this case, at least one candidate has a waiting time of at least δ_2 because condition *(i)* implies that not all candidates can perform an activity that requires two assessors at the same time. δ_2 corresponds to the length of the minimum time interval during which the required number of assessors exceeds the number of available assessors.
  Let *k* denote an arbitrary task that requires two assessors. To determine δ_2, we first consider the four possibilities for ordering the last activities such that the lunch break or task *r* are performed at the end by each candidate (cf. Fig. 8).

(a) The lunch break is performed at the end and preceded by task *r*. Task *r* is preceded by task *k*.
(b) Task *r* is performed at the end and preceded by the lunch break. The lunch break is preceded by task *k*.
(c) Task *r* is performed at the end and preceded by task *k*. The lunch break ends some time before task *k*.
(d) The lunch break is performed at the end and preceded by task *k*. Task *r* ends some time before task *k*.

In Fig. 8, the time point t_4 in (a) and (b) corresponds to the earliest possible finish time of task *k* for the assessors. The time points t_1, t_2, and t_3 correspond

to the possible start times of task k for the assessors if no candidate has a waiting time. The values of these time points are as follows.

$$t_1 = \delta_0 - p_l - (p_r - p_r^A) - (p_k - p_k^C - p_k^A)$$
$$t_2 = \delta_0 - (p_r - p_r^A) - (p_k - p_k^C - p_k^A)$$
$$t_3 = \delta_0 - p_l - (p_k - p_k^C - p_k^A)$$
$$t_4 = \delta_0 - p_l - (p_r - p_r^A) + p_k^A$$

Overall, the latest possible start time of task k for the assessors corresponds to

$$\max(t_1, t_2, t_3) = \delta_0 - \min(p_l, p_r - p_r^A) - (p_k - p_k^C - p_k^A).$$

If $t_4 > \max(t_1, t_2, t_3)$ and no candidate has a waiting time, then there is a time interval with a minimum length of $t_4 - \max(t_1, t_2, t_3)$ during which every candidate performs a task that requires two assessors. Because $\lfloor |A|/2 \rfloor < |C|$, the required number of assessors exceeds the available number of assessors in this interval. To resolve this conflict, at least one task k must be delayed, which leads to a minimum waiting time for at least one candidate of $t_4 - \max(t_1, t_2, t_3)$.

To derive a lower bound for the AC duration, we determine the smallest possible value of t_4 and the largest possible value of $\max(t_1, t_2, t_3)$ as follows.

$$t_4 \geq \delta_0 - p_l - (p_r - p_r^A) + \min_{q \in Q \setminus \{r\}} p_q^A$$
$$\max(t_1, t_2, t_3) \leq \delta_0 - \min(p_l, p_r - p_r^A) - \min_{q \in Q \setminus \{r\}} (p_q - p_q^C - p_q^A)$$

Hence, the minimum waiting time corresponds to

$$t_4 - \max(t_1, t_2, t_3) \geq \min_{q \in Q \setminus \{r\}} (p_q - p_q^C - p_q^A) + \min_{q \in Q \setminus \{r\}} p_q^A - \max(p_l, p_r - p_r^A)$$
$$= \delta_2.$$

Thereby, we used $\alpha + \beta - \min(\alpha, \beta) = \max(\alpha, \beta)$, where α, β are two arbitrary numbers. Hence, $\delta_0 + \delta_2$ is a lower bound if the solution belongs to case 2.

Overall, $LB_6 = \delta_0 + \min(\delta_1, \delta_2)$ is a lower bound for the AC duration if conditions (*i*) and (*ii*) hold. □

In the performance analysis, we use the maximum of these problem-specific lower bounds. If for an instance the necessary conditions for any of the lower bounds are not fulfilled, we set their respective value to 0.

$$LB^+ = \max(LB_1, LB_2, \ldots, LB_6)$$

6 Comparative analysis

We implemented the MIP formulations presented in Sect. 4 in AMPL, and we used the Gurobi Optimizer 6.0.5 as solver. All calculations were performed on an HP workstation with an Intel Xeon 2.67 GHz CPU and 24 GB RAM. The computational

experiment was performed using four real-life instances and 240 test instances derived from real-life data. We limited the CPU time of the solver to 3600 s for the real-life instances and to 600 s for the test instances. We used Gurobi with its default settings. Additionally, we applied Gurobi with the parameter MIPFocus set to 1. The parameter MIPFocus determines the MIP solution strategy of the solver. When this parameter is set to 1, Gurobi focuses on quickly generating good feasible solutions rather than increasing the lower bound. The default setting is 0, which aims to balance between finding good feasible solutions and proving optimality. For the DT formulations, the upper bound of the AC duration was set to $T = 200$ for all instances; this value is prescribed by the human resource provider.

In Sect. 6.1, we describe the instances that we used in our computational study. In Sect. 6.2, we discuss our computational results for the real-life instances. In Sect. 6.3, we provide the results for the test instances. In Sect. 6.4, we compare our problem-specific lower bounds.

6.1 Instances

The number of candidates $|C|$, assessors $|A|$, actors $|P|$, tasks $|E|$ and activities $|I|$ of the four real-life instances are listed in Table 5. The last column indicates whether at least one no-go relationship exists. We denote the real-life instances with RL1, …, RL4.

To test the different MIP formulations, we additionally devised a test set with 240 test instances based on real-life data. For the RCPSP, the well-known test instances of Kolisch and Sprecher (1997) were generated by systematically varying the complexity factors resource strength (*RS*), resource factor (*RF*), and network complexity (*NC*). These factors are only partially applicable to generate the ACP instances. The factor *NC* corresponds to the average number of precedence relationships per activity. Because there are no precedence relationships among the activities of the AC, we do not require such a factor. The factors *RF* and *RS* correspond to the average portion of the resources used by an activity and the scarcity of the resources, respectively. The factor *RF* can be interpreted as the average number of assessors required by an activity. To ensure that the instances are as close to reality as possible, we selected real-life tasks with given requirements for assessors and actors. Hence, we do not require a factor such as *RF*. The factor *RS* can be interpreted as the scarcity of the assessors. We use a similar factor to determine the number of available assessors. In total, we generated the 240 test instances by varying five complexity factors. Thereby, the employed experimental levels of each complexity factor were based on real-life data provided by the human resource management service provider. The complexity factors are as follows.

Table 5 Real-life instances

Instance	$\|C\|$	$\|A\|$	$\|P\|$	$\|E\|$	$\|I\|$	No-go relationships
RL1	7	10	2	5	42	No
RL2	11	11	3	5	66	No
RL3	9	11	3	5	54	Yes
RL4	6	9	3	5	36	No

The complexity factors n^C and n^E correspond to the number of candidates and tasks, respectively, and determine the number of activities of an instance. The tasks were randomly selected from a set of 15 real-life tasks. The experimental levels $n^C \in \{4, 5, \ldots, 10, 11\}$ and $n^E \in \{4, 5\}$ were used.

The complexity factor a^S corresponds to the average number of assignments per assessor. This factor is used to determine the number of assessors n^A of an instance. The number of assessors is equal to the nearest integer to $\sum_{i \in I^A} r_i^A / a^S$; thus, the numerator corresponds to the total number of assessor assignments. The experimental levels $a^S \in \{6.0, 8.5, 10.4\}$ correspond to the observed real-life minimum, average, and maximum.

The complexity factor a^N corresponds to the proportion of assessors who have one or more no-go relationships (no-go assessors). The number of no-go assessors is given by the nearest integer to $a^N n^A$. The no-go assessors were randomly selected from the set of all assessors. The experimental levels $a^N \in \{\frac{1}{6}, \frac{1}{3}\}$ were used.

The complexity factor a^R corresponds to the average number of no-go relationships per no-go assessor. The number of no-go relationships is equal to the product of a^R and the number of no-go assessors. The no-go relationships were randomly assigned to pairs of candidates and no-go assessors such that (1) each no-go assessor has at least one no-go relationship and (2) at least $\lfloor |A|/2 \rfloor$ different assessors can be assigned to each candidate. The experimental levels $a^R \in \{2, 3\}$ were used.

Because the actors are paid for each role play in which they actually perform, they are not considered to be a critical resource. Hence, the number of actors was set to 3 for all instances, which corresponds to the observed real-life maximum.

For each combination of complexity factor levels, an instance was generated; this leads to $8 \cdot 2 \cdot 3 \cdot 2 \cdot 2 = 192$ test instances. Additionally, $8 \cdot 2 \cdot 3 = 48$ test instances without no-go relationships (i.e., $a^N = a^R = 0$) were generated.

6.2 Computational results: real-life instances

For the real-life instances RL1, …, RL4, the results obtained by the solver using the MIP formulations CT-A, CT-F, CT-O, DT-P, and DT-O with MIPFocus set to 0 are reported in Table 6. We compare the objective function values (D) with the lower bounds obtained by the solver (LB) and the maximum value over all problem-

Table 6 Results for real-life instances with MIPFocus set to 0

Instance	CT-A		CT-F		CT-O		DT-P		DT-O		LB^+
	D	LB	D	LB	D	LB	D	LB	D	LB	
RL1	89	67	90	37	**88**	74	128	81	95	71	82
RL2	136	59	158	36	**132**	49	149	103	173	72	110
RL3	**106**	62	121	36	107	49	125	80	118	63	90
RL4	83	70	86	36	**82**	74	87	81	86	80	82

The best objective function values obtained are highlighted in boldface

Table 7 Results for real-life instances with MIPFocus set to 1

Instance	CT-A		CT-F		CT-O		DT-P		DT-O		LB^+
	D	*LB*	*D*	*LB*	*D*	*LB*	*D*	*LB*	*D*	*LB*	
RL1	**86**	49	**86**	36	88	49	98	76	88	70	82
RL2	**124**	49	128	36	129	54	159	70	150	69	110
RL3	102	49	**100**	36	108	49	118	59	114	63	90
RL4	**82**	56	84	36	84	55	**82**	82	**82**	76	82

The best objective function values obtained are highlighted in boldface

specific lower bounds (LB^+). The best objective function values obtained are highlighted in boldface. Using the default solver settings, the solver obtains on average the best objective function values with CT-O and the highest lower bounds with DT-P. For all real-life instances, these lower bounds are smaller than or equal to the problem-specific lower bound. The problem-specific lower bound of instance RL4 corresponds to the objective function value obtained with CT-O, i.e., this solution is optimal.

Table 7 lists the results obtained by the solver with MIPFocus set to 1. The best objective function values obtained are highlighted in boldface. Except for CT-O, the average AC duration is improved. However, on average, the lower bounds are worse. CT-A devises the best solutions for three instances, CT-F for two instances, and DT-P and DT-O for one instance. The smallest instance (RL4) is even solved to optimality using formulation DT-P. Both, CT-A and DT-O, also find a solution with an optimal objective function value, but they do not prove optimality within the prescribed CPU time.

6.3 Computational results: test instances

Based on the number of activities $|I|$, we divide the 240 test instances into small-sized (20–34 activities, 75 instances), medium-sized (35–49 activities, 90 instances), and large-sized (50–66 activities, 75 instances) instances. For these three ranges of $|I|$, the average number of variables and constraints for the different formulations are presented in Fig. 9. Regardless of the number of activities, DT-O has the highest number of variables. For small- and medium-sized instances, DT-O has also the highest number of constraints. However, with an increasing number of activities, the number of constraints increases less for the DT formulations than for

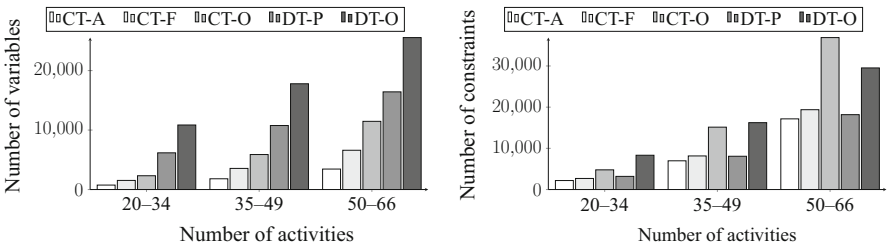


Fig. 9 Average number of variables (*left*) and constraints (*right*)

Table 8 Aggregated results for all 240 test instances

Formulation	MIP-focus	CT-A	CT-F	CT-O	DT-P	DT-O
Average gap^+ (in %)	0	**10.3**	15.1	12.5	27.7	19.4
	1	**9.3**	11.1	11.2	26.8	18.5
Average *gap* (in %)	0	44.7	59.8	50.4	37.5	**36.6**
	1	56.7	65.0	55.1	44.6	**37.8**
Number of feasible solutions	0	**240**	216	**240**	**240**	234
	1	**240**	235	**240**	**240**	238
Number of optimal solutions	0	**36**	29	32	22	19
	1	27	24	**30**	22	27
Number of best solutions	0	**170**	51	80	22	60
	1	**161**	69	81	27	57

The best results are highlighted in boldface

the CT formulations. For the large-sized instances, CT-O has the highest number of constraints.

Table 8 reports the average relative gaps between the obtained solutions and the problem-specific lower bound ($gap^+ = (D - LB^+)/D$), as well as the average relative gaps between the obtained solutions and the lower bounds obtained by the solver ($gap = (D - LB)/D$). To evaluate the quality of the solutions, we use gap^+. To evaluate the quality of the lower bounds provided by the solver, we use *gap*. The best results are highlighted in boldface.

Regardless of the solver settings employed, the best gap^+ is obtained with CT-A (10.3 % for MIPFocus set to 0 and 9.3 % for MIPFocus set to 1), and the worst gap^+ is obtained with DT-P. In contrast, the smallest *gap* is obtained with DT-O. Similarly to the results of Kopanos et al. (2014), better solutions are obtained with DT-O than with DT-P. We conclude that the CT formulations provide better solutions, and that the DT formulations provide better lower bounds. For all formulations, *gap* considerably exceeds gap^+. We deduce that the problem-specific lower bounds are considerably higher than the lower bounds obtained by the solver within the prescribed CPU time limit.

With CT-A, CT-O, and DT-P, feasible solutions are obtained for all 240 test instances within the prescribed CPU time limit. With CT-F and MIPFocus set to 0, feasible solutions are obtained only for 216 instances (i.e., 90 % of the instances). With MIPFocus set to 1, this number increases to 235 (i.e., 97.9 %); feasible solutions could not be obtained for five of the large-sized instances.

To determine the number of optimal solutions, we compare the objective function value obtained with the maximum value over all problem-specific lower bounds and the lower bound obtained by the solver. With 36 instances, CT-A obtains the highest number of optimal solutions.

The number of best solutions corresponds to the number of times that a formulation generates a best solution. With MIPFocus set to 0, CT-A provides a best solution for 170 instances. This means that the other formulations generate better solutions for 70 instances only.

Table 9 Average gap^+ for different instance characteristics

Instance characteristics		MIP-focus	Average gap^+				
			CT-A	CT-F	CT-O	DT-P	DT-O
$\|I\|$	20–34	0	**1.8**	3.1	2.6	10.7	11.0
		1	**2.4**	3.0	2.6	6.8	6.0
	35–49	0	**8.3**	14.9	11.4	28.0	12.7
		1	**7.8**	9.3	9.5	28.0	13.8
	50–66	0	**21.2**	31.8	23.6	44.4	36.5
		1	**18.1**	22.0	21.8	45.4	37.3
a^S	6	0	**10.1**	15.2	12.8	29.3	22.4
		1	**9.6**	11.3	11.9	23.4	20.3
	8	0	**12.1**	17.8	14.3	31.0	20.4
		1	**11.2**	12.9	12.5	32.2	19.4
	10.4	0	**8.8**	11.9	10.4	22.9	15.3
		1	**7.1**	9.0	9.1	24.8	15.8
a^N	0	0	**10.7**	16.9	12.5	25.9	18.8
		1	**9.2**	10.6	11.0	24.9	17.5
	0.17	0	**10.3**	14.9	12.8	27.4	19.4
		1	**9.4**	11.3	11.2	26.9	17.5
	0.33	0	**10.2**	14.5	12.2	29.0	19.6
		1	**9.3**	11.2	11.2	27.6	20.0
a^R	0	0	**10.7**	16.9	12.5	25.9	18.8
		1	**9.2**	10.6	11.0	24.9	17.5
	2	0	**10.3**	15.9	12.4	26.8	18.0
		1	**9.4**	11.3	10.8	27.0	17.8
	3	0	**10.2**	13.5	12.6	29.6	21.1
		1	**9.3**	11.2	11.7	27.6	19.8
f	11–13	0	**9.3**	13.0	12.2	25.6	10.6
		1	**8.6**	10.0	10.5	26.8	11.2
	13–15	0	**8.9**	13.5	10.7	24.4	18.8
		1	**7.3**	9.4	9.8	22.9	17.5
	15–17	0	**11.5**	17.0	13.5	30.3	23.2
		1	**10.7**	12.4	12.2	28.8	21.9

The best results are highlighted in boldface

With MIPFocus set to 1, the average solution quality for all formulations is improved. This is indicated by a reduction of gap^+. For CT-F, this reduction is quite considerable (from 15.1 to 11.1 %). This might indicate that the MIP solution strategy used by the solver exploits the resource-flow information in an efficient manner. However, the average *gap* is larger with MIPFocus set to 1 because this solver setting focuses less on improving the lower bounds but gives priority to the quick generation of good feasible solutions. Therefore, the number of feasible solutions is increased for CT-F. Surprisingly, for the CT formulations CT-A, CT-F and CT-O, the number of optimal solutions obtained is lower with MIPFocus set to 1.

Table 10 Comparison of problem-specific lower bounds

	LB_1	LB_2	LB_3	LB_4	LB_5	LB_6
Number of instances with best lower bound	93	90	0	8	32	22

Table 9 reports the average results for all instances with the same problem characteristics. The best results are highlighted in boldface. The overall results show that with MIPFocus set to 1 the best solutions are obtained. However, for CT-A and small-sized instances, the solver performs better with MIPFocus set to 0.

The number of activities $|I|$ and the level of complexity factor a^S, which defines the number of available assessors, have a significant impact on both relative gaps. In contrast, the levels of complexity factors a^N and a^R, which define the no-go relationships, have no systematic impact on the relative gaps. Parameter f corresponds to the average duration of the activities. The performance of DT-O is affected most by the value of f. For instances with short activities ($11 \leq f \leq 13$), the performance of DT-O is almost as good as the performance of CT-A. However, for the instances with longer activities, the average gaps are much higher. Surprisingly, such an effect is not observed with DT-P.

According to the results obtained by Koné et al. (2011) for the RCPSP, DT formulations are better for instances with activities that have a short duration. Although the durations of the AC activities are quite short, we do not observe similar results for the ACP. Overall, the CT formulations provide the best solutions. A drawback of the DT formulations may be the large number of variables (cf. Fig. 9) which depend on the number of time points considered. In the RCPSP, the number of variables is reduced considerably with a simple preprocessing like the definition of earliest and latest start times for the activities. However, this preprocessing is based on precedence relationships, which do not exist in the ACP. Considering the CT formulations, CT-A performs best, and CT-O performs better than CT-F.

6.4 Computational results: problem-specific lower bounds

Table 10 compares the six problem-specific lower bounds presented in Sect. 5. The last row shows the number of instances for which the different lower bounds obtained the highest values. LB_1 and LB_2 each provide the highest lower bounds for more than 90 instances. However, lower bounds that consider no-go relationships (LB_3 and LB_4) only provide the highest values for a few instances. If the conditions for LB_6 hold, this lower bound provides the highest values for 22 instances.

7 Conclusions

Comparisons of alternative MIP formulations in the literature for project scheduling problems are primarily based on generic test instances. In this study, we analyzed the performance of two discrete-time and three continuous-time MIP formulations

in a real-life application of project scheduling. We considered the problem of planning assessment centers. For this problem, we developed new MIP formulations, and we provided problem-specific lower bounds. In contrast to the results generally obtained for the RCPSP, our comparative study indicates that the CT formulations outperform the DT formulations in terms of solution quality. However, using the DT formulations, the best MIP-based lower bounds are obtained.

The assessment center planning problem is an interesting and challenging optimization problem for future research. An important area is the development of heuristic solution procedures. Preliminary versions of an MIP-based heuristic and a list-scheduling heuristic are presented in Rihm and Trautmann (2016) and Zimmermann and Trautmann (2015). In the MIP-based heuristic, first, the activities are scheduled without assessor assignments; second, the assessors are assigned to the activities using the CT formulation with resource-flow variables presented in this study. In the list-scheduling heuristic, the activities are scheduled sequentially based on problem-specific priority rules. The MIP formulations and the problem-specific lower bounds presented in this paper can be used to analyze the performance of such heuristic approaches.

References

Alvarez-Valdes R, Tamarit J (1993) The project scheduling polyhedron: dimension, facets and lifting theorems. Eur J Oper Res 67(2):204–220

Ambrosino D, Paolucci M, Sciomachen A (2015) Experimental evaluation of mixed integer programming models for the multi-port master bay plan problem. Flex Serv Manuf J 27(2–3):263–284

Artigues C, Michelon P, Reusser S (2003) Insertion techniques for static and dynamic resource-constrained project scheduling. Eur J Oper Res 149(2):249–267

Artigues C, Koné O, Lopez P, Mongeau M (2015) Mixed-integer linear programming formulations. In: Schwindt C, Zimmermann J (eds) Handbook on project management and scheduling, vol 1. Springer, Cham, pp 17–41

Bianco L, Caramia M (2013) A new formulation for the project scheduling problem under limited resources. Flex Serv Manuf J 25(1–2):6–24

Bixby RE (2012) A brief history of linear and mixed-integer programming computation. Doc Math Extra ISMP:107–121

Chen X, Grossmann I, Zheng L (2012) A comparative study of continuous-time models for scheduling of crude oil operations in inland refineries. Comput Chem Eng 44:141–167

Christofides N, Alvarez-Valdés R, Tamarit JM (1987) Project scheduling with resource constraints: a branch and bound approach. Eur J Oper Res 29(3):262–273

Collins JM, Schmidt FL, Sanchez-Ku M, Thomas L, McDaniel M, Le H (2003) Can basic individual differences shed light on the construct meaning of assessment center evaluations? Int J Sel Assess 11(1):17–29

Grüter J, Trautmann N, Zimmermann A (2014) An MBLP model for scheduling assessment centers. In: Huisman D, Louwerse I, Wagelmans A (eds) Operations research proceedings 2013. Springer, Berlin, pp 161–167

Kaplan L (1988) Resource-constrained project scheduling with preemption of jobs. PhD thesis, University of Michigan

Klein R (2000) Scheduling of resource-constrained projects. Kluwer, Amsterdam

Koch T, Achterberg T, Andersen E, Bastert O, Berthold T, Bixby RE, Danna E, Gamrath G, Gleixner AM, Heinz S et al (2011) MIPLIB 2010. Math Progr Comput 3(2):103–163

Kolisch R, Sprecher A (1997) PSPLIB-a project scheduling problem library: OR software-ORSEP operations research software exchange program. Eur J Oper Res 96(1):205–216

Koné O, Artigues C, Lopez P, Mongeau M (2011) Event-based MILP models for resource-constrained project scheduling problems. Comput Oper Res 38(1):3–13

Koné O, Artigues C, Lopez P, Mongeau M (2013) Comparison of mixed integer linear programming models for the resource-constrained project scheduling problem with consumption and production of resources. Flex Serv Manuf J 25(1–2):25–47

Kopanos GM, Kyriakidis TS, Georgiadis MC (2014) New continuous-time and discrete-time mathematical formulations for resource-constrained project scheduling problems. Comput Chem Eng 68:96–106

Mingozzi A, Maniezzo V, Ricciardelli S, Bianco L (1998) An exact algorithm for the resource-constrained project scheduling problem based on a new mathematical formulation. Manag Sci 44(5):714–729

Naber A, Kolisch R (2014) MIP models for resource-constrained project scheduling with flexible resource profiles. Eur J Oper Res 239(2):335–348

Pritsker AAB, Waiters LJ, Wolfe PM (1969) Multiproject scheduling with limited resources: a zero-one programming approach. Manag Sci 16(1):93–108

Rihm T, Trautmann N (2016) A decomposition approach for an assessment center planning problem. In: Ruiz R, Alvarez-Valdes R (eds) Proceedings of the 15th international conference on project management and scheduling, Valencia, pp 206–209

Stefansson H, Sigmarsdottir S, Jensson P, Shah N (2011) Discrete and continuous time representations and mathematical models for large production scheduling problems: a case study from the pharmaceutical industry. Eur J Oper Res 215(2):383–392

Vanhoucke M, Coelho J, Debels D, Maenhout B, Tavares LV (2008) An evaluation of the adequacy of project network generators with systematically sampled networks. Eur J Oper Res 187(2):511–524

Vielma JP (2015) Mixed integer linear programming formulation techniques. SIAM Rev 57(1):3–57

Zapata JC, Hodge BM, Reklaitis GV (2008) The multimode resource constrained multiproject scheduling problem: alternative formulations. AIChE J 54(8):2101–2119

Zimmermann A, Trautmann N (2014) Scheduling of assessment centers: an application of resource-constrained project scheduling. In: Fliedner T, Kolisch R, Naber A (eds) Proceedings of the 14th international conference on project management and scheduling, Munich, pp 263–266

Zimmermann A, Trautmann N (2015) A list-scheduling approach for the planning of assessment centers. In: Hanzálek Z, Kendall G, McCollum B, Šůcha P (eds) Proceedings of the multidisciplinary international scheduling conference: theory and application, Prague, pp 541–554

Tom Rihm received his BSc and MSc degree in Mathematics (2009, 2012) and his MSc degree in Management, Business and Economics (2013) from the RWTH Aachen University (Germany). Currently he is a research and teaching assistant at the Department of Business Administration at the University of Bern (Switzerland). His research interests include combinatorial optimization, personnel scheduling, project management and project scheduling, production planning, and network-based optimization.

Norbert Trautmann received his BSc and MSc in Business Engineering/Management Science and his PhD in Business and Economics from the University of Karlsruhe (Germany) in 1994, 1997, and 2000, respectively. Currently he holds the Chair in Quantitative Methods in Business Administration at the University of Bern (Switzerland). His research interests are in combinatorial optimization, project management and project scheduling, production planning and control, and portfolio optimization.

Adrian Zimmermann received his BSc and MSc degree in Business Administration (2011, 2012) from the University of Bern (Switzerland). Currently he is a research and teaching assistant at the Department of Business Administration at the University of Bern. His research interests are combinatorial optimization, project management and project scheduling, and service operations management.