Journal Pre-proof

The bilevel optimisation of a multi-agent project scheduling and staffing problem

P. Milička, P. Šůcha, M. Vanhoucke, B. Maenhout

PII: S0377-2217(21)00254-X
DOI: https://doi.org/10.1016/j.ejor.2021.03.028
Reference: EOR 17109

To appear in: *European Journal of Operational Research*

Received date: 11 June 2020
Accepted date: 13 March 2021

Please cite this article as: P. Milička, P. Šůcha, M. Vanhoucke, B. Maenhout, The bilevel optimisation of a multi-agent project scheduling and staffing problem, *European Journal of Operational Research* (2021), doi: https://doi.org/10.1016/j.ejor.2021.03.028

Highlights

- Multi-agent resource constrained project staffing problem
- Hierarchical interaction between team leader and project manager
- Mathematical programming-based approach with lazy constraints
- Different speed-up techniques to accelerate the solution method
- Classical multi-objective models fail to model this bilevel problem

The bilevel optimisation of a multi-agent project scheduling and staffing problem

P. Milička[1], P. Šůcha[1], M. Vanhoucke[2,3,4], and B. Maenhout[*,2]

[1]Czech Technical University in Prague, Czech Institute of Informatics, Robotics, and Cybernetics, Prague, The Czech Republic, pavel.h.milicka@gmail.com, suchap@cvut.cz

[2]Faculty of Economics and Business Administration, Ghent University, Tweekerkenstraat 2, 9000 Gent (Belgium), Broos.Maenhout@ugent.be, Mario.Vanhoucke@ugent.be

[3]Technology and Operations Management Area, Vlerick Business School, Belgium

[4]UCL School of Management, University College London, UK

Abstract

In this paper, we study a multi-agent project staffing problem involving a single project, which has to be scheduled under resource constraints. We consider a functional organisational structure where a team leader and a project manager are together responsible for the operational execution of a project. The team leader, which has the formal authority over the resources, is standing at the top of the hierarchy and determines the number and mix of (additional) employees and tries to level the workload over the planning period in order to avoid idle resource times. The project manager is responsible for the scheduling of the project activities and his/her objective is to minimise the project duration. The interaction between both agents in the decision-making process is, on the one hand, hierarchical, i.e. the team leader imposes his/her decision on the project manager. On the other hand, the decision taken by the team leader should comply to the objective of the project manager such that the staffing plan and project schedule is agreed by both parties. We propose a bilevel optimisation model that embeds a nested inner optimisation problem, i.e. the project manager decision problem, as a constraint in the outer optimisation problem, i.e. the decision problem of the team leader. The algorithm is a mathematical programming method thriving on the generation of additional lazy constraints via feasibility callbacks so that the team leader problem has to respect the requirements formulated in the project manager problem. In the computational experiments, we compare this solution approach to alternative classical optimisation approaches and we validate the design choices related to the proposed speed-up mechanisms and parameter settings.

Keywords: Project scheduling, Personnel staffing, Multi-agent, Bilevel Optimisation

*Corresponding author

1 Introduction

The need for adequate project planning and scheduling is widely recognised across various areas from construction industry to software development. The cornerstone of a successful delivery is balancing the project management triangle involving the dimensions of time, cost, and quality or scope (Dobson, 2004). As a result, in the literature, a project schedule has been evaluated using different performance measures (Brucker et al., 1999; Demeulemeester and Herroelen, 2002), e.g. completion time, resource cost, levelling of the resource demand, financial objectives. Different performance measures have been formulated related to human resource management since in many sectors projects are very labour-intensive, and personnel resources are recognised as one of the most important assets. In this perspective, different problem definitions and formulations have been proposed for the project staffing problem, which devises the required (personnel) resource capacity to deliver the project at a minimum cost possibly incorporating different other project scheduling objectives. The classical approach to approximate the decision-making process is to formulate the problem either as a non-preemptive optimisation problem with a single or multiple weighted objectives or to employ a preemptive or hierarchical approach solving multiple separated stages with different objectives, for which the decisions of higher hierarchical stages constrain the decisions of lower stages to come to a final schedule.

In this paper, we study a multi-agent project staffing problem that has been inspired by real-world practice, which is relevant in many industries (e.g. consultancy sector, construction sector, etc.). Every project requires a team for managing the project, decision-making related to the scheduling of the activities, and executing the project activities. A team may have various structures to suit the specifics of the company and/or the project (Bobera, 2008). We consider the interaction within a project-based team following a programmatic-based or functional organisational structure (Baligh, 2006), which sets up a hierarchical team structure and consists of three ranks, i.e. a team leader, a project manager, and a heterogeneous set of employees characterised by different categorical skill classes. The *team leader* is standing at the top of the team hierarchy and has the formal authority over the resources, determining the number and mix of (additional) employees of each skill class that should be hired. In addition, the team leader supervises the workload, ensuring the levelling of the personnel resource requirements as best as possible in order to avoid idle resource times. In this way, the objective of the team leader is to minimise the (additional) number of resources to staff a project properly and to level the resource requirements, minimising the sum of absolute deviations in resource usage, which is dependent on the constructed project schedule. The *project manager*, who is at a lower layer of the hierarchy, addresses the project scheduling. The project manager is responsible for determining the start and end times of the project activities, and his/her objective is to minimise the project duration. The project activities are executed by employees of different skill classes, representing the lowest level in the team hierarchy.

In the decision-making process, the interaction between both agents is hierarchical in the

sense that the realised outcome of any decision taken by the team leader to optimise the resource mix and the levelling of the workload is affected by the response of the lower-level project manager, who seeks to optimise the project duration.

Classical optimisation approaches fail to model the interaction between different agents correctly (Sinha et al., 2018). In order to overcome the disadvantages of those approaches, we propose a bilevel solution framework such that the team leader and the project manager come to an agreement on the resulting project schedule. In this way, the bilevel optimisation algorithm enables to model the decision behaviour leading to a game-theoretic equilibrium. The bilevel approach models the project manager decision as a nested inner optimisation problem, i.e. the *follower problem*, that is included as a constraint in the outer optimisation problem related to the decision of the team leader, i.e. the *leader problem*. In contrast to other approaches, the bilevel optimisation approach has the advantage to capture the interaction between both decision-makers, which are located at different hierarchical levels, each controlling one set of decision variables with different and possibly conflicting objectives. The higher-level decision-maker optimises its objective independently but may be affected by the reaction of the lower-level decision-maker, which executes its policies after the decision on the higher level. Bilevel optimisation enables to incorporate the feedback or response of the follower in the leader problem such that both parties come to a consensus. In this way, the bilevel approach allows for the integration of the project staffing decision of the team leader and the project scheduling decision, which is performed by the project manager, operating on the lower hierarchical level, such that an accurate staffing budget can be estimated to execute the project, meeting the desires of the project manager.
The proposed algorithm thrives on mathematical programming and conducts a search in the inducible region, i.e. the set of feasible points of the bilevel program, in order to come up with an optimal solution. A feasible solution satisfies the optimality conditions for the lower decision level and all upper and lower-level constraints. Throughout this search, additional constraints are added via feasibility callbacks so that the leader problem has to respect the requirements formulated in the follower problem. In the computational experiments, we compare this solution approach to alternative classical optimisation approaches and show that the latter approaches are inadequate to model the interaction between the two agents. In addition, we validate the impact of the proposed speed-up mechanisms and different parameter settings on the computational performance. Moreover, in a theoretical analysis we prove why a classical single-stage multi-objective approach and a multi-stage approach cannot guarantee to find an optimal solution.

The outline of this paper is as follows. Section 2 discusses the relevant literature. Section 3 explains the multi-agent project staffing problem in detail and provides a mathematical formulation of the resulting bilevel program. Section 4 presents the proposed solution procedure. Alternative single-stage and multi-stage solution approaches are discussed in Section 5. In addition, we show why single-stage multi-objective and multi-stage approaches cannot be used as exact approaches to solve the studied problem. In Section 6, we provide

a benchmark between the proposed method and the alternative optimisation methods and validate the performance of the different components and parameter settings related to the solution methodology. The conclusions and directions for future research are provided in Section 7.

2 Literature review

In this section, we first discuss the literature related to project scheduling and staffing problems in Section 2.1. Second, we review the relevant literature related to bilevel optimisation in Section 2.2.

2.1 Project scheduling and staffing

Project scheduling and staffing have been studied in the literature in various guises and formulations. Hartmann and Briskorn (2010) provide an overview of resource-constrained project scheduling and discuss different problem variants for which the project activities are characterised by a certain duration and require (personnel) resources. The common objective for different resource-constrained problems (e.g. the basic resource-constrained project scheduling problem (RCPSP), the multi-mode resource-constrained project scheduling problem (MRCPSP), and the discrete time/resource trade-off problem (DTRTP)) is to minimise the project duration, which is a regular measure of performance (French, 1982). A regular measure of performance is a non-decreasing function of the activity completion times. Other time-based objectives are to minimise the lateness or the tardiness. In these types of problems, the scheduling of the activities is subject to precedence constraints between activities and the limited resource availability. Each one of these problems considers renewable resources that are present each time period, for which personnel resources are exemplary.

There are, however, different other project scheduling problems where a resource cost minimisation objective is used, and the context may better represent the capacity planning of personnel resources. All the mentioned project scheduling problems belong to the class of problems with a nonregular objective function. The *resource availability cost problem* (RACP) minimises the total cost of the constant (unlimited) renewable resources required to complete the project by a pre-specified project deadline. This problem was introduced by Möhring (1984) as the resource investment problem. Exact solution methods have been devised by Demeulemeester (1995) and Rodrigues and Yamashita (2010). These methods propose an iterative procedure based on a subproblem solver that searches for the minimum project duration given a particular resource availability and gradually increases the number of resources until the project duration is smaller than or equal to the deadline. Rodrigues and Yamashita (2010) improved the procedure of Demeulemeester (1995) by calculating tighter

bounds and adding cuts to reduce the search space based on infeasible RACP-solutions. Kreter et al. (2018) discuss different mixed-integer linear and constraint programming formulations to solve the RACP with general constraints and calendar constraints. The problems are solved via mathematical programming and lazy constraint generation. Lazy constraints are able to remove integer-feasible solutions, in contrast to cutting planes that cut off non-integer solutions of the LP relaxation of the original problem. The *time-constrained project scheduling problem* (TCPSP) assumes a fixed resource availability and determines the need for flexible resources such as overtime and temporary resources in order to complete a project within the deadline (Deckro and Herbert, 1989). The related studies of Bassett (2000) and Wu and Sun (2006) present mixed-integer non-linear problem formulations for multi-project scheduling and staff allocation. The objective aims to minimise the outsourcing costs related to the temporal resources. The work of Heimerl and Kolisch (2010) presents a mixed-integer programming (MIP) model for the simultaneous scheduling and staffing of multiple projects. They incorporate internal and external resources, and the project activities have to start within certain time windows. The *resource renting problem* (RRP) considers fixed renting costs or resource availability costs and variable renting costs, where the cost of using a single resource unit is dependent on the time period the resource is hired (Nübel, 2001). The *resource levelling problem* (RLP) tries to level the resource usage by adjusting the activity start times to minimise the fluctuations in the patterns of resource utilisations over time, while maintaining compliance with a prescribed project completion time (Neumann and Zimmermann, 2000). No resource limiting constraints are imposed. Rieck et al. (2012) discuss different mixed-integer linear model formulations and propose various acceleration mechanisms such as domain-reducing preprocessing techniques and cutting planes to speed up the mathematical optimisation of the RLP. An overview of different exact methods for the RLP is given in Rieck and Zimmermann (2015).

Alfares and Bailey (1997) and Maenhout and Vanhoucke (2016) amongst others solve the project staffing problem in a more detailed manner by providing both a project schedule and an employee roster subject to different time-related resource scheduling constraints (e.g. the maximum length of consecutive working days, the minimum number of working days per time period). To that purpose, they propose to integrate the scheduling of project activities and personnel resources. The objective is to minimise the staffing budget consisting of regular, temporary, and overtime workers, as well as to minimise the activity start time costs. Note that all these problems are concerned with a single-stage optimisation problem that considers multiple weighted objectives simultaneously.

2.2 Bilevel programming

The investigation of bilevel programming models is strongly motivated by (real-world) applications. A bilevel program is an optimisation problem where the feasible set is partly determined through a solution set mapping of a second parametric optimisation problem (Dempe, 2002). Bilevel programming is applied when a single-level optimisation cannot be

utilised, and the problem can be interpreted as a hierarchical game of two decision makers (Colson et al., 2007). The leader makes his/her decision first and communicates it to the follower, which reacts to the decision of the leader optimising his/her own objective. The leader has information about the follower's objectives and, therefore, (s)he can anticipate their responses and adjust its strategy accordingly (Kalashnikov et al., 2016). As a result, the leader's optimisation problem is a nested problem, for which the lower-level optimisation problem represents the best response of the follower to the leader's decision. The two levels of the model are hierarchically organised with each level representing one entity (agent), and the upper level communicates the variables it controls to the lower level. The leader's objective function can contain variables from both levels, whereas the follower optimises only its own variables with the leader's variables influencing the constraints.

Depending on the domain of the variables (*continuous* versus *integer*) in the leader and follower problem, different types of bilevel programming problems have been formulated and different solution methodologies have been proposed in the literature. The algorithms developed for solving the bilevel problems are usually dedicated and make use of specific features of the problem under study. A solution approach is to transform a bilevel program into a related single-stage optimisation problem (Kalashnikov et al., 2016). In that perspective, several authors have proposed to efficiently solve a bilevel problem with continuous variables, linear constraints and a linear objective in both the leader and follower problem by converting the problem using linear programming duality or the Karush-Kuhn-Tucker conditions (e.g. Leyffer et al., 2006). A review of the methods for solving continuous bilevel problems is given in Colson et al. (2007) and Sinha et al. (2018). Introducing integrality conditions on the variables changes the shape of the inducible region and may lead to a disconnected search space. Moore and Bard (1990) proposed a branch-and-bound method for the mixed-integer linear bilevel problem. They discuss that for bilevel programming problems it is difficult to fathom nodes such that the algorithm's nested structure is not scalable beyond few integer variables. Several researchers included cuts to the bilevel program in order to accelerate the search for an optimal solution (Bard and Moore, 1992; Dempe, 2001). Other researchers have used an evolutionary heuristic approach to solve discrete bilevel programming problems (e.g. Hecheng and Yuping, 2008).

The literature related to bilevel project scheduling problems is limited. Gang et al. (2013) discussed a multi-project resource allocation problem organised according to a bilevel framework. On the upper level, the company manager aims to allocate the company's resources to multiple projects to achieve the lowest cost, which includes resource costs and a tardiness penalty. On the lower level, each project manager attempts to schedule their resource-constrained project minimising the project duration. To search for an optimal solution, a hybrid algorithm composed of an adaptive particle swarm optimisation, a genetic algorithm, and a fuzzy random simulation algorithm is proposed. Xu et al. (2015) study a bilevel project scheduling problem, which considers the interests of the project owner and the contractor. The project owner has the objective to maximise his/her profit and to minimise the project duration, while the contractor's objective is to minimise the resource costs. The problem is

solved by a fuzzy random simulation-based particle swarm optimisation technique. A multi-mode RCPSP with uncertainty is formalised as a bilevel multi-objective problem and solved by a swarm optimisation algorithm in Gan and Xu (2015). Feng et al. (2018) discussed the integration of a resource-constrained project scheduling problem and a resource supply chain problem in a bilevel framework. They proposed a genetic algorithm with double strings to search for an optimal equilibrium solution between the two players.

3 Problem definition and mathematical formulation

The problem under study comprises a strategic project staffing problem to determine the project schedule and the number of (additional) resources to carry out the project activities. Two agents are involved who have to agree on a proper project schedule. In the problem under study, we consider the roles of the project manager and the human resource manager or the team leader, which are hierarchically organised following a functional organisational structure (Kerzner, 2003). The team leader is concerned with the management of the personnel resources. (S)he should select the appropriate team (size and mix) to execute the project and is responsible for the day-to-day team performance. The team leader provides the necessary resources to the project manager to properly execute the project schedule. The project manager will seek to obtain the best possible alternative for project implementation, and his/her main responsibility is the successful completion of the project. In this way, (s)he is charged with the definition, planning, and tracking of the project. The context of the problem under study is illustrated via an example problem in Figure 1. The figure displays a feasible project schedule based on the given project, activity and resource characteristics. In this example, eight activities and only a single resource type are considered.
This paper focuses on the interplay between the hierarchically higher-ranked team leader and the lower-ranked project manager to decide on the scheduling of the project activities under resource constraints. The problem captures a real-world scenario, for which the individual parties have the following objectives

- The objective of the *team leader* (see Leader problem in Figure 1) is to minimise the costs for hiring additional workers on top of the already available workforce and to balance the workload as best as possible over the planning horizon. In this way, the utilisation of the personnel resources is maximised, and resource idle times are minimised (Kastor and Sirakoulis, 2009). The solution yielded by the leader problem in Figure 1 hires two additional resources and counts 10 unit changes in resource usage over the planning horizon.

- The objective of the *project manager* (see Follower problem in Figure 1) is to minimise the project duration in order to carry out a larger number of projects in the long term and to increase profit (Icmeli-Tukel and Rom, 1997). The project duration is his/her only interest and an agreement with the team leader can only be made when there is no

possibility of composing a project schedule with a shorter duration given the available workforce. The project schedule yielded by the team leader in Figure 1 satisfies the objective of the follower as the project duration cannot be reduced given the number of additional resources, which is determined via the leader problem.

In the following, we describe in detail the project and activity characteristics and the bilevel framework that surrounds this optimisation problem.

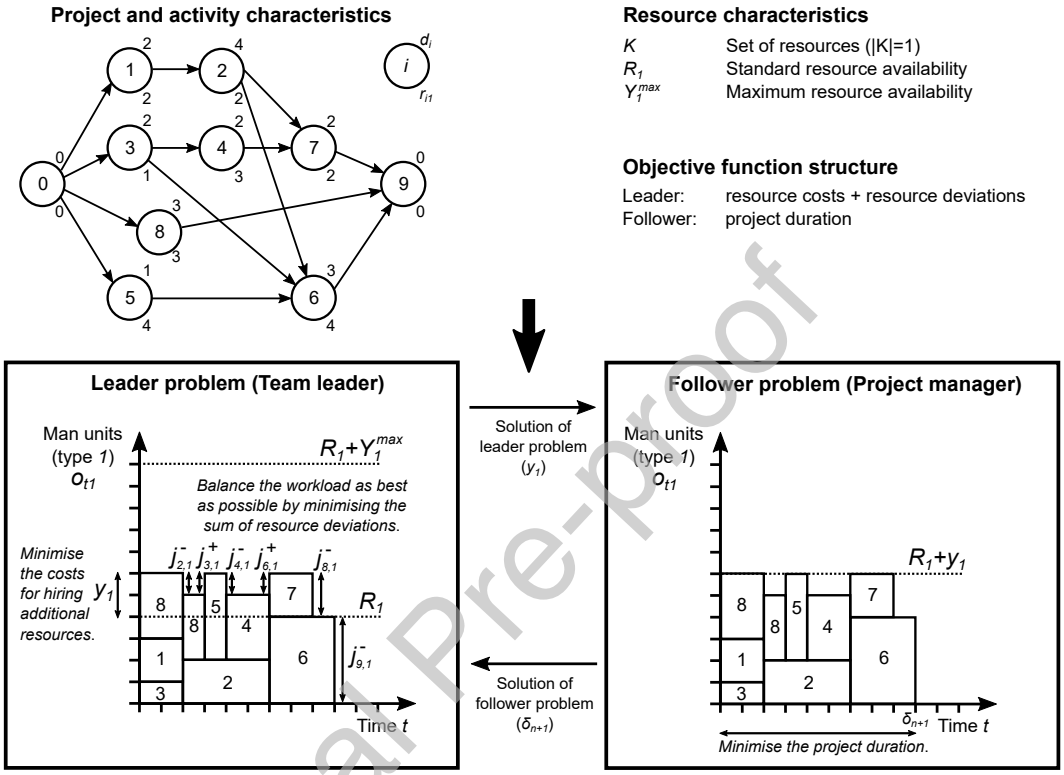


Figure 1: Problem context of the bilevel project scheduling problem

3.1 Problem definition and notation

In Section 3.1.1, we provide the notation of the involved sets and parameters to formulate the problem under study. Section 3.1.2 describes in a detailed manner the interaction between the two agents and the decision variables of the underlying bilevel optimisation problem.

3.1.1 Project scheduling and staffing characteristics

The relevant project scheduling and staffing characteristics can be stated as follows. In a project network in activity-on-the-node (AoN) format, we assume an acyclic graph $G = (N, A)$. The activities are represented by a set of nodes N and the precedence relations between activities by set A. The set N contains $n + 2$ activities, numbered from 0 to $n + 1$ with activity 0 and activity $n + 1$ representing the dummy start and dummy end activity.

The precedence relations are finish-start relations with time lag 0, i.e. an activity can be started as soon as all its predecessors are finished. These activities are to be scheduled without pre-emption requiring a set K of renewable personnel resource types. The standard resource availability, denoted as R_k, is defined for each resource type $k \in K$. Parameter R_k is constant for each time unit $t \in T$. The set $T = \{0, ..., t^{max}\}$ defines all feasible time indices ordered ascending from zero to the planning horizon t^{max}. The maximum number of resources that can be additionally hired is restricted by Y_k^{max} for each resource type k.
Each activity $i \in N$ has a deterministic duration d_i and requires r_{ik} man units of type k per time unit. The dummy activities 0 and $n+1$ have zero duration and zero resource usage. The other activities have a non-zero duration. An example of a project network and corresponding graph is illustrated in Figure 1.

3.1.2 Multi-agent project staffing problem in a bilevel optimisation framework

The *leader problem* involves the decisions, objectives, and constraints of the team leader. The team leader tries to minimise the integer number of additional resources hired of resource type k, which is denoted by the decision variable y_k. This amount is constrained by the maximum number of additional resources Y_k^{max}. In addition, the leader aims to minimise the sum of absolute deviations in resource usage during the execution of the project in order to optimise the resource levelling. The goal of resource levelling is to minimise the resource fluctuations, i.e. the adjustments in resource demand designated by the staffing requirements, which are calculated using the auxiliary variables o_{tk}, j_{tk}^+ and j_{tk}^-. The integer variable o_{tk} represents the staffing requirements in time period t for resource type k. In the literature, there exists several different objective function formulations to level the resource usage over the duration of the project (Damci and Polat, 2014). In this paper, we minimise the absolute deviation between the staffing requirements of consecutive time periods, indicated by the positive deviation j_{tk}^+ and the negative deviation j_{tk}^- (Senouci and Eldin, 2004; El-Rayes and Jun, 2009). These resource deviations are the direct consequence of the project schedule that is determined by the project manager in the follower problem. In this way, the objective of the team leader considers two non-regular measures of performance with weights α and β determining the importance of resource levelling and the additional staffing costs, respectively.
In the follower problem, a project schedule is constructed by assigning a start time to each activity by using the binary variables v_{it}, which are equal to 1 if activity i starts at time period t and are zero otherwise. The objective of the follower is to minimise the project duration δ_{n+1} designated by the start time of the dummy end activity $n+1$, i.e. $\delta_{n+1} = \sum_{t \in T} t \cdot v_{(n+1)t}$, representing a regular measure of performance. The staffing requirements o_{tk} result from the summation of the workload for resource k of all active activities at time t, i.e. $o_{tk} = \sum_{i \in N} \sum_{t^*=t-d_i+1}^{t} r_{ik}\ v_{it^*}$. A feasible solution to the follower problem respects the precedence constraints and the resource availability constraints. These latter constraints link the resource capacity decision of the leader and the project scheduling decision of the

follower. In the formulation of this bilevel programming problem, the variables y_k, j_{tk}^{+} and j_{tk}^{-} represent the upper-level decision variables. All other variables v_{it}, o_{tk} and δ_{n+1} represent the lower-level decision variables and are related to the construction of the project schedule, which is the responsibility of the project manager. The upper-level decision variables j_{tk}^{+} and j_{tk}^{-} are a function of the lower-level decision variables o_{tk}.

The bilevel optimisation problem under study implicitly uses the optimistic assumption (Kalashnikov et al., 2016), i.e. when the follower problem possesses alternative project schedules with an optimal project duration, then the project schedule will be selected that minimises the sum of absolute deviations in resource usage. In other words, out of all feasible solutions for a fixed number of resources hired (y_k, $\forall k \in K$) and associated minimal project duration (δ_{n+1}), the schedule will be chosen that is the most favourable for the leader. The team leader has the authority to decide upon both the additional number of resources hired and the project schedule, as long as the project schedule is executed according to the minimum project duration given the amount of additional resources hired.

3.2 Mathematical problem formulation

In this section, we present the bilevel problem (BL) formulation for the proposed multi-agent project staffing problem as follows

$$\text{Min } F^{BL} \tag{1}$$

s.t. $F^{BL} = \alpha \sum_{t \in T} \sum_{k \in K} (j_{tk}^{+} + j_{tk}^{-}) + \beta \sum_{k \in K} y_k$ (2)

$o_{tk} + j_{tk}^{+} - j_{tk}^{-} = o_{(t+1)k}$ $\forall k \in K, \forall t \in T$ (3)

$y_k \leq Y_k^{max}$ $\forall k \in K$ (4)

$j_{tk}^{+}, j_{tk}^{-} \geq 0$ $\forall t \in T, \forall k \in K$

$y_k \geq 0$ and integer $\forall k \in K$ (5)

$(v_{it}, o_{tk}, \delta_{n+1}) \in argmin\{f^{BL} = \delta_{n+1} | (7) - (13)\}$ (6)

s.t.

$\delta_{n+1} = \sum_{t \in T} t \; v_{(n+1)t}$ (7)

$\sum_{t \in T} v_{it} = 1$ $\forall i \in N$ (8)

$\sum_{t^*=t}^{LST_i} v_{it^*} + \sum_{t^*=EST_j}^{min(LST_j, t+d_i-1)} v_{jt^*} \leq 1$ $\forall (i,j) \in A, \forall t \in \{EST_i, LST_i\}$ (9)

$o_{tk} = \sum_{i \in N} \sum_{t^*=t-d_i+1}^{t} r_{ik} \; v_{it^*}$ $\forall k \in K, \forall t \in T$ (10)

$o_{tk} \leq R_k + y_k$ $\forall k \in K, \forall t \in T$ (11)

$v_{it} \in \{0, 1\}$ $\forall i \in N, \forall t \in T$ (12)

$o_{tk} \geq 0$ $\forall t \in T, \forall k \in K$

$\delta_{n+1} \geq 0$ (13)

The objective of the mathematical model (1) together with constraint (2) represent the objective function of the team leader denoted as F^{BL}. This entity aims to minimise the additional number of resources y_k of each resource type k and the absolute deviations in the resource demand j_{tk}^{+} and j_{tk}^{-} resulting from the project schedule, which is a function of the staffing requirements o_{tk} determined in the follower problem. This resource levelling and staffing objectives compose a multi-objective function using the weights α and β, respectively. The absolute resource deviation variables j_{tk}^{+} and j_{tk}^{-} are determined in equations (3), which contain both upper-level and lower-level decision variables but are upper-level constraints since they do not impact the feasibility of a solution to the follower problem. Constraints (4) and (5) define the domain of the upper-level decision variables. Constraints (4) impose an upper bound on the additional resource capacity that can be hired for each resource type k. The integrality conditions imposed on the variables in the leader problem are stated via constraints (5).

The bilevel problem formulation further comprehends the lower-level problem of the follower in the constraints of the leader problem. Equation (6) indicates that a feasible solution for the leader problem should be an optimal solution for the follower problem given its objective f^{BL}, i.e. the minimisation of the project duration δ_{n+1}. This objective is formulated as a function of the activity start time of the dummy end activity $n+1$ via equation (7). The follower's feasible region is a set of decisions satisfying the follower's constraints (8)-(13). Constraints (8) make sure that exactly one start time is assigned to all project activities. Constraints (9) represent the direct finish-start precedence constraints with time lag 0 between certain

pairs of activities. Constraints related to activity i are only listed for the range of t defined by the earliest start time (EST_i) and the latest start time (LST_i). The activity and network information can be used to define explicitly the time window during which an activity can be started. The earliest start time EST_i of activity i is determined by only considering the precedence requirements and the duration of the activities without the resource requirements. The latest start time LST_i is the latest time activity i can start given the end of the planning horizon t^{max}. The staffing requirements o_{tk} are calculated via constraints (10). Constraints (11) state that the required number of staff resulting from the activity scheduling should be lower than the number of available resources for each type k. These available resources stem from the standard resource availability R_k and the additional number of resources hired by the team leader. Constraints (11) include both upper and lower-level decision variables, whereas constraints (7), (8), (9), and (10) only embody lower-level decision variables. The domain of the variables related to the follower problem is stated in constraints (12) and (13). A solution is called bilevel feasible if it satisfies all constraints (3)-(13). The feasible region that contains all bilevel feasible solutions is called the inducible region. These solution points embody an optimal follower decision, i.e. a minimum project duration, in response to the leader decision. A bilevel optimal solution is bilevel feasible and optimises the leader's objective F^{BL} (eq. (1)). In conclusion, the problem of the team leader is to find the best schedule according to the combination of performance measures involving resource levelling and resource cost minimisation, and at the same time to construct a project schedule with the shortest possible duration with the available resources. The optimal solution value of the upper and lower-level problem is indicated as $\hat{F}^{BL}$ and $\hat{f}^{BL}$, respectively.

Note that the parameters α and β have a large influence on the problem structure. If α is set to a very low value compared to β, the problem basically collapses into a variation of the RACP, i.e. finding a shortest possible feasible project schedule, where additional personnel is hired only when the current pool of workers is insufficient to create a feasible schedule. On the other hand, when α is set to a very large value compared to β, the problem changes into a variant of the RLP.

4 Bilevel solution methodology

The main obstacle for formulating the bilevel project scheduling problem as a single MIP is that the follower problem cannot be transformed to a linear program. Therefore, it is not possible to use strong duality of linear programming to formulate the objective of the follower, as it is used, for example, in Kis and Kovacs (2013). As a result, the construction of the inducible region is computationally a hard task and a mathematical programming approach is employed that searches through the bilevel search space and continuously adds so-called lazy constraints restricting the region such that the leader has to respect the requirements of the follower. Hence, in order to solve the problem under study to optimality, we decompose the problem into a leader problem and a follower problem. Both these problems contain integer

variables and are modelled as a mixed-integer programming problem. The formulations and solution procedure for the leader problem and the follower problem are discussed in detail in Section 4.1 and Section 4.2, respectively.

The proposed solution method is a depth-first branch-and-bound search method that operates on the leader problem, which relies on a binary representation of the integer number of resources hired. In this way, the leader is the supremal unit and has control over the decision variables y_k. This implies that the procedure thrives upon the linear programming relaxation (LPR) of the leader problem relaxing the integrality constraints (5) and (12). The individual steps of the proposed algorithm are displayed in Algorithm 1.
The procedure is initiated by creating an upper bound based on a resource levelling heuristic for different project durations (Section 4.4). These upper bounds are used to initialise the incumbent value accelerating the search for an optimal solution of the leader problem and to generate lazy constraints, which tighten the solution space of the branch-and-bound search. After these upper bound computations, an optimal LPR solution of the leader problem is calculated in the root node of the branch-and-bound tree. If this solution is fractional or bilevel infeasible, additional steps are required to drive this fractional solution to integrality and to a bilevel feasible solution. In an optimal LPR solution of any node, two types of (binary) variables may be fractional, i.e. the personnel staffing variables and/or the activity start time variables. Branching is performed on the variable with the largest fractional value without any priority between these two types of decision variables. A binary branching rule is employed creating two disjoint subproblems setting the selected branching variable relevant in the leader problem either to 0 or 1.

When an integer solution is encountered in any node in the branch-and-bound tree, the follower problem is solved to check if the yielded integer solution is bilevel feasible. Based upon the outcome of the follower problem, lazy constraints are formulated connecting the required number of additional resources (leader decision) with the minimum project duration (follower response) in order to incorporate information related to the bilevel feasibility of integer solutions and to tighten the search space (Section 4.3). These lazy constraints are added to the formulation of the leader problem in order to cut off integer solution points for which the project duration is not minimal given the number of resources hired and, hence, are not bilevel feasible. If the integer solution for the leader problem is bilevel feasible, the objective value of the solution is compared with the incumbent and the incumbent is updated if the new solution has a lower value for F^{BL}. If the integer solution is not bilevel feasible, the linear relaxation of the leader problem is solved again with the generated lazy constraints included, which cuts away the bilevel infeasible solution leading to a different solution for the leader problem.
Nodes are only fathomed when either the retrieved solution is bilevel feasible or when the lower bound at a particular node is larger than or equal to the incumbent value. In these cases, backtracking is applied and the search is continued with the most recent node or the node with the best bound that has not been fathomed. When all nodes are fully explored, the

procedure is stopped. In order to accelerate the procedure, we have implemented different speed-up mechanisms, which are detailed in the relevant sections below.

Algorithm 1 The branch-and-bound solution methodology

```
Construct an initial upper bound (Section 4.4)
do
  Solve the linear programming relaxation (LPR) of the leader problem (Section 4.1)
  If (LPR solution of the leader problem ≥ Incumbent)
    ∨ (LPR of the leader problem is infeasible) then
    Backtrack
  Else
    If LPR solution of the leader problem is fractional then
      Branch on the variable with the largest fractional value
      Create subproblems and increase depth of the branching tree
    Else
      Solve the follower problem and check bilevel feasibility (Section 4.2)
      Generate lazy constraints (Section 4.3)
      If LP solution of the leader problem is bilevel feasible then
        If LP solution value of the leader problem < Incumbent then
          Incumbent = LP solution value of the leader problem
        Backtrack
While Branching tree is not fully explored
```

4.1 The leader problem

The leader problem is formulated as a MIP problem and follows equations (1)-(13), where the follower objective, i.e. the constraint (6), is omitted. In this problem, the assumption is made that the leader has the entire control over the decision variables. In addition, variable y_k is replaced by a set of auxiliary binary variables w_{kl} where $l \in \{0, ..., Y_k^{max}\}$. Variable w_{kl} equals 1 if the number of additional resources type k is l and 0 otherwise. These variables replace variable y_k by setting $y_k = \sum_{l=0}^{Y_k^{max}} l \; w_{kl}$, and thus they provide another form representing the number of additional resources. This binary representation allows to interconnect the leader problem with the follower problem via the lazy constraint mechanism (Section 4.3). The solution returned in a particular node of the branching tree is represented by the specific values for the involved decision variables (indicated by '^'), i.e. the start times of activities $\hat{v}_{it}^{LPR}$ ($i \in N$, $t \in T$), the amount of additional resources $\hat{y}_k^{LPR} = \sum_{l=0}^{Y_k^{max}} l \; \hat{w}_{kl}^{LPR}$ ($k \in K$), the resource deviations $\hat{j}_{tk}^{+,LPR}$ and $\hat{j}_{tk}^{-,LPR}$ ($t \in T$, $k \in K$), and a particular project duration $\hat{\delta}_{n+1}^{LPR}$. The upper index LPR denotes that the solution results from the linear relaxation of the leader problem.

4.2 The follower problem and bilevel feasibility

The follower problem is solved to check whether an integer solution found by the leader problem is bilevel feasible and there does not exist a solution with a shorter duration given

the number of additional resources $\hat{y}_k^{LPR}$ $(k \in K)$ returned by the leader problem. The objective of the follower is to minimise the project duration δ_{n+1} for the given integer number of additional resources $\hat{y}_k^{LPR}$, which is now considered as a constant input parameter. As a result, the follower problem can be stated as a simple MIP model as follows

$$\min f^{FP} = \delta_{n+1} \tag{14}$$

$$\text{s.t. } \delta_{n+1} = \sum_{t \in T} t \, v_{(n+1)t} \tag{15}$$

$$\sum_{t \in T} v_{it} = 1 \qquad \forall i \in N \tag{16}$$

$$\sum_{t^*=t}^{LST_i} v_{it^*} + \sum_{t^*=EST_j}^{min(LST_j, t+d_i-1)} v_{jt^*} \leq 1 \qquad \forall (i,j) \in A, \forall t \in \{EST_i, LST_i\} \tag{17}$$

$$o_{tk} = \sum_{i \in N} \sum_{t^*=t-d_i+1}^{t} r_{ik} \, v_{it^*} \qquad \forall k \in K, \forall t \in T \tag{18}$$

$$o_{tk} \leq R_k + \hat{y}_k^{LPR} \qquad \forall k \in K, \forall t \in T \tag{19}$$

$$v_{it} \in \{0,1\} \qquad \forall i \in N, \forall t \in T$$

$$o_{tk} \geq 0 \qquad \forall t \in T, \forall k \in K$$

$$\delta_{n+1} \geq 0 \tag{20}$$

The model uses the same schedule representation as the leader problem using the variables v_{it}, for which the binary conditions are imposed via equation (20). The model is equivalent to equations (7)-(13) (cf. Section 3.2). The only difference is that the number of additional resources $\hat{y}_k^{LPR}$ is an integer constant. The solution of the follower problem is denoted via the activity start times $\hat{v}_{it}^{FP}$ and the project duration $\hat{\delta}_{n+1}^{FP}$.
Whenever the optimal solution value of the follower problem $\hat{\delta}_{n+1}^{FP}$ is equal to the project duration $\hat{\delta}_{n+1}^{LPR}$ of the project schedule constructed by the leader problem, the solution point under consideration is called bilevel feasible since it is feasible for both parties. If $\hat{\delta}_{n+1}^{FP} < \hat{\delta}_{n+1}^{LPR}$, the solution of the leader problem is not feasible from the follower's point of view.

Initialisation of the follower problem ($UB - FP$)
The project schedule obtained via the leader problem can be passed along to initialise the procedure solving the follower problem. The project schedule of the leader, which is defined by the start times of the activities ($\hat{v}_{it}^{LPR}$), is used as the initial solution and the project duration $\hat{\delta}_{n+1}^{LPR}$ as initial upper bound.

4.3 Lazy constraint generation

In order to guide the leader problem towards a bilevel feasible solution, we incorporate information of the follower problem into the leader problem via a so-called lazy constraint mechanism. Whenever an integer feasible solution is found by the leader problem, the bilevel feasibility is checked via the follower problem. The follower problem returns the (minimum)

project duration $\hat{\delta}_{n+1}^{FP}$ given the number of (additional) resources hired $\hat{y}_k^{LPR}$ set by the leader problem. Based upon this insight, two types of lazy constraints can be defined for every encountered integer solution of the leader problem based upon the solution of the follower problem. The lazy constraints are added to the formulation of the leader problem in order to cut off integer solution points that are not feasible for both parties. The definition of lazy constraints is motivated by the fact that the leader problem specified in Section 4.1 is actually a relaxation of the true bilevel optimisation problem. At the start of the procedure, it is not possible to identify these constraints, which are identified via bilevel feasibility callbacks at certain points in the search. Adding a lazy constraint to the leader problem increases the ability of the leader problem in search for an optimal solution of the true bilevel optimisation problem.

Lazy constraint 1 ($LC1-LP$)
The lazy constraint connecting an integer solution of the leader problem and a corresponding optimal solution of the follower problem is modelled in inequalities (21).

$$\sum_{k \in K} w_{k,\hat{y}_k^{LPR}} - |K| + 1 \leq v_{(n+1)\hat{\delta}_{n+1}^{FP}}. \tag{21}$$

Constraint (21) states that if the number of (additional) resources hired y_k equals $\hat{y}_k^{LPR}$ for each $k \in K$ (such that $\sum_{k \in K} w_{k,\hat{y}_k^{LPR}} = |K|$), the project duration returned by the leader problem should be equal to $\hat{\delta}_{n+1}^{FP}$ (as the associated binary variable $v_{(n+1)\hat{\delta}_{n+1}^{FP}} \geq 1$). All solutions having a larger project duration for this number of additional resources $\hat{y}_k^{LPR}$ are not bilevel feasible and are cut off via this constraint. For any other combination of resources, the constraint is redundant.

Lazy constraint 2 ($LC2-LP$)
Lazy constraint 1 can be generalised in order to cut off a larger space of infeasible solutions. The generalisation is based on the fact that by increasing any number of additional resources y_k higher than $\hat{y}_k^{LPR}$, the follower's best response δ_{n+1} can only be smaller than or equal to $\hat{\delta}_{n+1}^{FP}$. Based upon this insight, lazy constraint 2 is modelled via inequalities (22).

$$\left(\sum_{k \in K} \left(\sum_{l \in \hat{y}_k^{LPR}, \ldots, Y_k^{max}} w_{kl} \right) \right) - |K| + 1 \leq \sum_{t=0}^{\hat{\delta}_{n+1}^{FP}} v_{(n+1)t}. \tag{22}$$

The right-hand side of constraint (22) embodies the sum of the project durations smaller than or equal to $\hat{\delta}_{n+1}^{FP}$. The left-hand side equals 1, if the number of additional resources y_k for all resource types are larger than or equal to $\hat{y}_k^{LPR}$. When this is the case, the project duration set via the leader problem is forced to be smaller than or equal to $\hat{\delta}_{n+1}^{FP}$. When the left-hand side is lower or equal to zero, the constraint is redundant.

Example
In order to illustrate the lazy constraints, we consider the example project that has been

defined in Figure 1, considering only a single resource type ($|K| = 1$). Assume that the Figure 2(a) provides an integer and feasible project schedule yielded via solving the leader problem in any iteration with $\hat{y}_1^{LPR} = 1$ (or $\hat{w}_{11}^{LPR} = 1$) and $\hat{\delta}_{n+1}^{LPR} = 14$. Solving the follower problem, for which the returned project schedule is provided in Figure 2(b), leads to the conclusion that the candidate integer project schedule resulting from the leader problem is not bilevel feasible as the follower solution provides a project schedule with a shorter duration $\hat{\delta}_{n+1}^{FP} = 12$. In order to cut off this integer solution and continue the search for the bilevel optimal solution, a lazy constraint (21) of type 1 can be defined collapsing to $w_{1,1} \leq v_{(n+1),12}$, which ensures that the leader will return a schedule with a project duration equal to 12 if only one additional resource is hired. A lazy constraint (22) of type 2 further strengthens this cut by defining the constraint $\sum_{l=1}^{Y_k^{max}} w_{kl} \leq \sum_{t=0}^{12} v_{(n+1)t}$, ensuring that any project schedule resulting from the leader problem with a number of additional resources larger than or equal to 1 has a project duration smaller than or equal to the follower response $\hat{\delta}_{n+1}^{FP} = 12$.

4.4 Initial upper bound for the leader problem ($UB - LP$)

The initialisation step aims to install an adequate upper bound on the leader's objective and to generate a set of lazy constraints in the beginning of the search for an optimal solution in order to reduce the size of the branch-and-bound tree. This is accomplished by generating multiple bilevel feasible project schedules in a heuristic manner inspired by the algorithm of Burgess and Killebrew (1962), which is a single-pass improvement procedure and constructs a single project schedule by sequentially adjusting the start times of slack activities one by one, in a predefined order stipulated by an activity priority list. Algorithm I in the Online Appendix A presents the individual steps of this initialisation step, which applies the project scheduling heuristic for a predefined set of project durations and priority lists, i.e.

- The set of priority lists PL: Based upon preliminary computational experiments, we have selected (i) the earliest start time of the activities and (ii) the duration of the activities (d_i) as relevant priority lists.
- The set of possible project durations D: The lowest project duration ($\min(D)$) is computed by constructing the earliest start project schedule without taking the resource requirements into account, which corresponds to finding the shortest path between the dummy start and end node. The longest project duration ($\max(D)$) is calculated via the follower problem as the minimal project duration with the minimum additional resources hired ($y_k = \max(0, \max_i(r_{ik} - R_k))$, $\forall k \in K$), which is known as the resource-constrained shortest path problem (Blazewicz et al., 1983). A logarithmic step size is used to visit different project durations for reasons of scaling between different instances as the size of set D is determined by the activity characteristics.

The project scheduling heuristic creates a single project schedule given a particular project

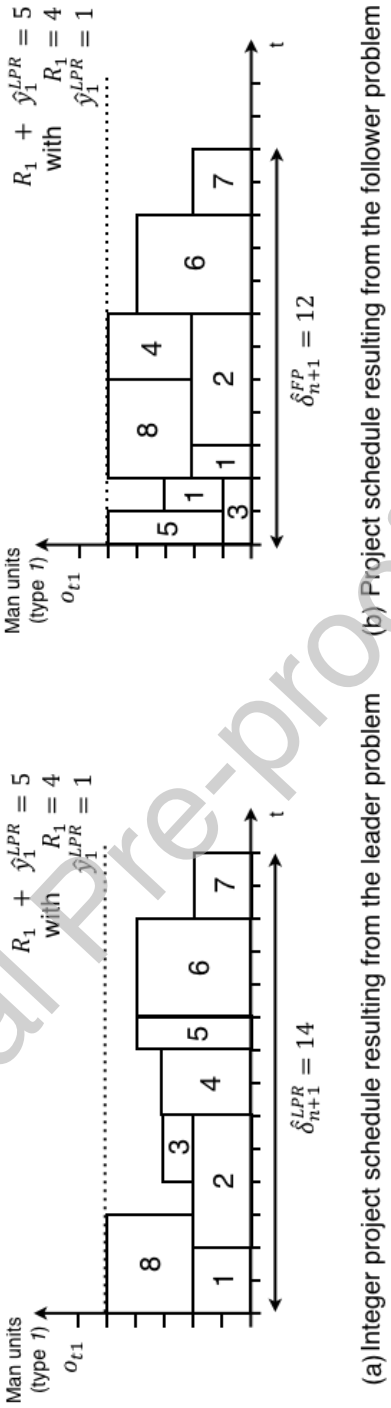


Figure 2: Illustration of lazy constraint generation

duration d and the activity priority list p. In a preprocessing step, the earliest and latest start time are calculated for each activity and the start times are set at their earliest start times. The activities are selected one by one according to the priority list and the start time of the considered activity is varied to identify the best starting time, which is the one that minimises

the performance measure. The latter is modelled as a preemptive objective function in the sense that a hierarchy is established between two objectives, i.e. the primary objective is to minimise the number of additional resources, whereas the secondary objective is to minimise the sum of absolute deviations in resource usage. The detailed functioning of this project scheduling heuristic is designated by Algorithm II in the Online Appendix A.

For a particular priority list and project duration, the single-pass heuristic returns a project schedule that is characterised by the start times for all activities ($\hat{ST} = \{\forall i \in N : \hat{ST}_i\}$ with $\hat{ST}_i = \sum_{t \in T} t \cdot \hat{v}_{it}$), the number of additional resources hired ($\hat{Y} = \{\forall k \in K : \hat{y}_k\}$) and the project duration ($\hat{\delta}_{n+1} = d$). Subsequently, we evaluate whether the returned project schedule is not dominated by any other schedule that has been already generated and stored in the archive set DS. The dominance is evaluated based on the project duration and the additional number of resources hired, which embody the characteristics based upon which relevant lazy constraints are defined. A schedule is dominated by another schedule if (i) it is completed in an equal or smaller project duration and (ii) it consumes an equal or smaller number of resources for each resource type. If the project schedule is dominated, we continue the search and construct another project schedule applying the next priority list and/or project duration. Otherwise, the devised project schedule is a suitable solution based upon which a bilevel feasible solution is constructed. To that purpose, we solve the follower problem and retain the resulting solution. The list DS is updated by adding the constructed bilevel feasible solution, designated as ($\tilde{ST}$, $\tilde{Y}$, $\tilde{\delta}_{n+1}$), and deleting those project schedules that are dominated by the newly found project schedule. In this way, only project schedules leading to relevant lazy constraints are retained.

The initialisation step is terminated whenever all relevant project durations and priority lists have been visited or when the time limit for the initial heuristic has been reached, which is installed to maintain the balance between the initialisation step and the branch-and-bound search given the overall time limit for the entire procedure. At that point, we evaluate every solution on the final list DS according to the leader's objective F^{BL} with appropriate weights α and β. The best upper bound found, which is bilevel feasible, will be set as the incumbent for the leader problem at the start of the branch-and-bound search. The solution quality of the incumbent solution resulting from the initialisation step is denoted as $\hat{F}^{UB}$. Note that the solution points added to the list DS allow the definition of lazy constraints (cf. Section 4.3) linking the heuristic solutions for the leader problem with the follower response to cut off a bilevel infeasible part of the solution space. These constraints are added a priori to the leader problem with the aim to speed up its convergence.

5 Alternative multi-stage and single-stage solution methodologies

In order to benchmark the proposed solution procedure, we devise different alternative solution methodologies. In Section 5.1, we approach the bilevel problem via multiple stages that are solved in a sequential manner. The stages follow the hierarchy of the leader and the follower in the problem under study and the decisions of the team leader constrain the decisions of the project manager. In Section 5.2, we formulate a multi-criteria optimisation problem to model the interaction between the two decision makers and solve the problem via a single-level methodology, which is common in the scheduling literature (Kis and Kovacs, 2002). The relation between multi-criteria optimisation and bilevel linear programming have been studied in Dempe (2002) as well as Kis and Kovacs (2002). The alternative single- and multi-stage approaches devised in this section are *standard* optimisation techniques trying to capture the bilevel optimisation problem. However, these approaches may not be fully equivalent leading to solutions that are not bilevel feasible, i.e. the notions of optimality in the different approaches are different.

5.1 Multi-stage sequential approach (*Seq*)

The multi-stage sequential approach solves the problem in an iterative and sequential manner. For that purpose, we calculate the set D (index d), which contains all the possible project durations (see Section 4.4). We iterate over the possible project durations and solve the leader problem for a particular project duration d by including constraint (23).

$$\delta_{n+1} = d \tag{23}$$

The leader problem is intrinsically a multi-objective optimisation problem, considering the resource levelling and resource cost objective via a weighted sum. However, in the project scheduling literature both objectives have not been tackled simultaneously in a single optimisation problem. In order to make use of the provided formulations in the literature, these objectives can also be tackled in two stages solving different known project scheduling problems as subproblems in a sequential order. These subproblems involve the RACP and the RLP, i.e.

- The RACP minimises the minimum number of (additional) employees needed to achieve the given project duration (Kreter et al., 2018). The objective function is to minimise $F^{Seq,RACP}$ (cf. equation (24)) subject to constraints (4), (5), (7)-(13), and (23).

$$F^{Seq,RACP} = \sum_{k \in K} y_k \tag{24}$$

- The RLP tries to shuffle the activity start times in order to minimise the sum of the absolute deviations in resource usage (Rieck et al., 2012). The objective function is to minimise $F^{Seq,RLP}$ (cf. equation (25)) subject to constraints (3)-(5), (7)-(13), and (23).

$$F^{Seq,RLP} = \sum_{t \in T} \sum_{k \in K} (j_{tk}^{+} + j_{tk}^{-}) \tag{25}$$

Depending upon the postulated hierarchy in solving these two subproblems, the outcome of the subproblem that is solved first will be imposed on the second problem leading to different eventual project schedules. As a result, the leader problem can be solved in three different manners for a given deadline d, i.e.

- RACP $\uplus$ RLP: A solution for the leader problem is obtained by optimising the objective function (1), which considers both objectives simultaneously give the respective weights α and β, subject to constraints (3)-(5), (7)-(13), and (23).

- RACP $\rightarrow$ RLP: A solution is obtained by first solving the RACP, leading to the optimal value $\hat{F}^{Seq,RACP}$ and the additional resource $\hat{y}_k$ ($\forall k \in K$). This solution of the RACP will be imposed on the solution of the RLP by adding constraints (26).

$$y_k = \hat{y}_k \qquad \forall k \in K \tag{26}$$

- RLP $\rightarrow$ RACP: A solution is obtained by first solving the RLP, leading to the optimal value $\hat{F}^{Seq,RLP}$. This solution of the RLP will be imposed on the solution of the RACP by adding constraint (27).

$$\sum_{t \in T} \sum_{k \in K} (j_{tk}^{+} + j_{tk}^{-}) = \hat{F}^{Seq,RLP} \tag{27}$$

The general flow of the algorithm is denoted in Algorithm 2. Note that the objective function of the leader problem (eq. (1)) is used to evaluate the project schedule given a particular project duration d. The project schedule with the smallest objective function value will be selected. The optimal solution value will be denoted as $\hat{F}^{Seq,RACP \uplus RLP}$, $\hat{F}^{Seq,RACP \rightarrow RLP}$, and $\hat{F}^{Seq,RLP \rightarrow RACP}$. In case the same objective function value for the leader problem has been obtained for different project durations, the solution with the smallest project duration will be selected in accordance with the follower's objective.

Algorithm 2 Multi-stage sequential approach

1: Calculate the set D of possible project durations (index d)
2: **For** each project duration $d \in D$ **do**
3: Solve leader problem (RACP $\uplus$ RLP, RACP $\rightarrow$ RLP, RLP $\rightarrow$ RACP)
4: Return the solution with the best objective function value for the leader problem (in case of ties: select the solution with the smallest d)

Proposition 1. *Any algorithm solving the bilevel problem introduced in Section 3 as a multi-stage sequential problem (RACP ⊎ RLP, RACP → RLP, RLP → RACP) cannot guarantee to find an optimal solution of the bilevel problem.*

Proof. If there were a multi-stage sequential algorithm able to solve the bilevel problem to optimum, it would solve any instance of the problem to the optimum. Therefore, the proof of the proposition is based on a simple contradiction showing that there are instances that such an algorithm cannot solve optimally.

Let us assume a simple instance of the bilevel problem having 6 activities and a single resource ($|K| = 1$) with standard resource availability $R_1 = 2$. The duration of all activities is 2, activity 2 requires 2 units of the resource, while the other activities consume only one unit. The instance defines eight precedence relations illustrated in Figure 3. Finally, the weight of resource levelling is $\alpha = 1$, and the weight of the staffing objective is $\beta = 1$.

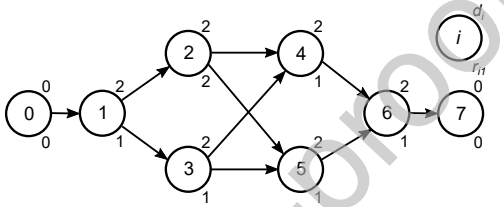
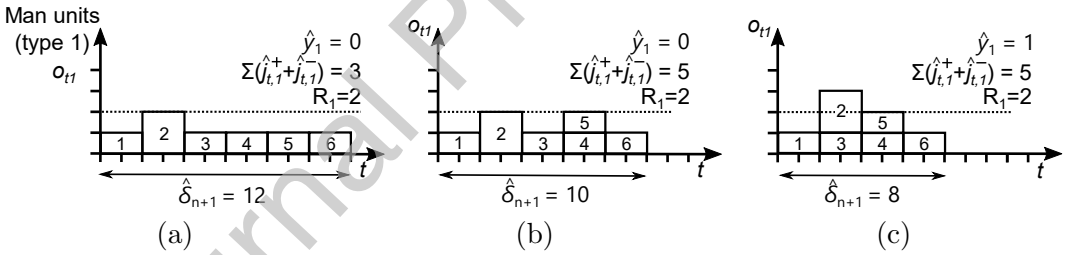


Figure 3: An example instance



(a) (b) (c)

Figure 4: Possible solutions of the example instance

The instance has only three possible solutions that are not dominated by any other solution w.r.t resource levelling $\sum_{t\in T}(\hat{j}^+_{t,1} + \hat{j}^-_{t,1})$ and staffing $\hat{y}_1$ objective values. The solutions are illustrated in Figure 4. Solution (a) has $\sum_{t\in T}(\hat{j}^+_{t,1} + \hat{j}^-_{t,1}) = 3$ (the resource fluctuation at the start of time zero is not counted) and $\hat{y}_1 = 0$, such that the value of leader's objective is $\hat{F}^{BL} = 3$. This solution is not bilevel feasible, since its project duration can be decreased from 12 to 10 as it is illustrated in case (b). Solution (b) is bilevel feasible having $\sum_{t\in T}(\hat{j}^+_{t,1} + \hat{j}^-_{t,1}) = 5$, $\hat{y}_1 = 0$, and $\hat{F}^{BL} = 5$. The last solution (c) is also bilevel feasible with $\sum_{t\in T}(\hat{j}^+_{t,1} + \hat{j}^-_{t,1}) = 5$, $\hat{y}_1 = 1$, and $\hat{F}^{BL} = 6$. Therefore, the optimal bilevel solution is case (b) having the minimal $\hat{F}^{BL}$.

For this problem instance, approach RACP ⊎ RLP will clearly find the solution (a) as the optimal one since it minimises the objective function (1). Nevertheless, this solution is not bilevel feasible. Approach RACP → RLP, minimising the number of additional resources,

will determine that $\hat{y}_1 = 0$. Its second stage obviously favourites solution (a) as it minimises $\sum_{t \in T}(j^+_{t,1} + j^-_{t,1})$. Therefore, this approach does not deliver a bilevel feasible solution as well. The last approach, i.e. RLP $\rightarrow$ RACP, will select solution (a) since it minimises the resource levelling objective, i.e. $\sum_{t \in T}(\hat{j}^+_{t,1} + \hat{j}^-_{t,1}) = 3$. Its second stage will confirm this solution since no additional resources are needed. Therefore, none of the three approaches will find the optimal bilevel solution. □

Please note that the multi-stage approaches are fully equivalent to the bilevel approach for a special class of the optimisation problem, i.e. when $\alpha = 0$. In this case, the decision of the follower cannot affect the objective of the leader. Therefore, all three multi-stage approaches will select the minimal number of additional resources for which Algorithm 2 takes the solution with the minimum δ_{n+1}. On the other hand, the same does not hold when $\alpha > 0$ and $\beta = 0$. The difference is that the follower's decision affects the sum of the absolute deviations in resource usage in the leader's objective. For the example instance of Figure 4, all three variants of the multi-stage sequential algorithm will find solution (a), which is not bilevel feasible.

5.2 Single-stage multi-objective approach (*Int*)

The single-stage multi-objective approach combines the objectives of the leader problem and of the follower problem into an integrated optimisation problem with a single weighted objective. The weights α, β, and γ are assigned to the objectives of resource levelling, the minimisation of the (additional) resource capacity cost, and project duration minimisation, respectively. The objective function is to minimise F^{Int} (cf. equation (28)) subject to constraints (3)-(5) and (7)-(13). The optimal solution value related to the leader objectives of the original bilevel problem is denoted as $\hat{F}^{Int}$ and the solution value for the project duration as $\hat{f}^{Int}$, designating the value for the follower objective.

$$\text{Min } F^{Int} = \alpha \sum_{t \in T} \sum_{k \in K} (j^+_{tk} + j^-_{tk}) + \beta \sum_{k \in K} y_k + \gamma \delta_{n+1} \tag{28}$$

A suitable value for γ is fixed beforehand reflecting the relative priorities between the three different objectives. Since γ is associated with the follower objective, the relative priority will be set low (i.e. $\gamma < \alpha$ and $\gamma < \beta$). The same value for γ is set for all instances. Note that this approach may be myopic as it is very difficult to assess the impact of the relative priorities reflected by the objective function coefficients on the bilevel feasibility of the eventual schedule outcome given a particular problem instance. However, a multi-objective approach iterating over all possible values of γ does also not guarantee to mirror the bilevel optimisation problem under study and to find a bilevel optimal solution (see Proposition 2), such that an approach with clear (relative) priorities between the leader and follower objectives is most unambiguous and assessed in this study.

Proposition 2. *Any algorithm solving the bilevel problem introduced in Section 3 as a multi-objective problem with objective (28) arbitrarily iterating over γ cannot guarantee to find an optimal solution of the bilevel problem.*

Proof. If there were a multi-objective algorithm iterating over γ able to solve the bilevel problem to optimum, it would solve any instance of the problem to the optimum. As in the previous proposition, the proof of the proposition is based on a simple contradiction showing there are instances that such an algorithm cannot solve optimally.

Let us assume the same instance of the bilevel problem as in Proposition 1. The instance has only three possible solutions as was illustrated in Figure 4, while (a) is not bilevel feasible and (b) and (c) are bilevel feasible; solution (b) is the optimal one. The objective values of the individual solutions (with $\alpha = \beta = 1$) in a multi-objective algorithm can be formulated as $\hat{F}_a^{Int} = 3 + 12\gamma$, $\hat{F}_b^{Int} = 5 + 10\gamma$, and $\hat{F}_c^{Int} = 6 + 8\gamma$. By a simple comparison of the objective functions we can identify for which γ solution (a) dominates solution (b). It is for $3 + 12\gamma < 5 + 10\gamma$, i.e. $\gamma < 1$. In the same way, we can identify when solution (c) dominates solution (b). For those we get $6+8\gamma < 5+10\gamma$, i.e. $\gamma > 0.5$. From these two comparisons one can see that even though solution (b) is the optimal bilevel solution it is always dominated by (a) or (c) in the multi-objective algorithm for any $\gamma \in [0, +\infty[$. □

As before, the multi-objective problem is equivalent to the bilevel approach on instances with $\alpha = 0$. In this case, it is sufficient to determine $\gamma << \beta$ such that lexicographic optimality is guaranteed. Such an objective directly corresponds to the bilevel approach.

6 Computational results

In this section, we provide computational insights in the performance of the proposed solution methodology. The characteristics of the test dataset are described in Section 6.1. In Section 6.2, we compare the proposed algorithm with alternative single-stage and multi-stage solution methodologies. Section 6.3 discusses the impact of the different components of the algorithm and validates the composition of the algorithm by justifying the design choices. The code is implemented in Python and the tests are carried out on two Intel(R) Xeon(R) E5-2690 v4 processors @ 2.60GHz and 256 GB of memory. The solver used to conduct the experiments is Gurobi 8.1.

6.1 Test design

In order to devise general findings related to the performance of the proposed solution methodology considering different activity and project characteristics, we consider a synthetic set of project scheduling problem instances from the literature. The problem in-

stances are constructed in an artificial manner, varying different settings and complexity indicators in a systematic and controlled way by means of a well-known network generator (Vanhoucke et al., 2016). Given the problem definition, research objective and the available datasets from literature, we selected the DC1 dataset. This dataset has been developed by Vanhoucke et al. (2001) using the network generator Progen/Max for the resource-constrained project scheduling problem with discounted cash flows (RCPSPDC). In our experiments, we only consider the instances with 10, 20, and 30 activities. The imposed time limit in all experiments is set at 1800 seconds per instance. On top of the experiments to endorse the performance of the algorithm, we discuss in the Online Appendix B the computational sensitivity related to different parameters relevant for the problem under study, i.e. the objective function coefficients and the standard resource availability, to validate the employed parameter settings and verify the stable performance of the algorithm.

Instance characteristics

The characteristics of the problem instances contained in this dataset are displayed in Table 1. The resource requirements of each activity are randomly generated in the interval [1,10] based on the resource parameters indicated in the table. In the experiments, we set the standard resource availability R_k equal to the original resource availability indicated in these instances multiplied by a factor 0.5. This factor has been determined such that a suitable trade-off can be realised between hiring additional resources and levelling the resource usage (cf. Online Appendix B.2. The maximum resource availability Y_k^{max} is equal to $\sum_{i \in N} r_{ik}$ for each resource type $k \in K$. The *duration* of each activity is randomly generated in the interval [1,10]. For additional information regarding the complexity indicators and settings, we refer to Vanhoucke et al. (2016).

Characteristics	**Parameter settings**
Network topology	
# activities (I_1 or $\|N\|$)	10, 20, 30
Coefficient of Network Complexity (CNC)	[0.7, 2.3]
Order Strength (OS)	[0.25, 0.824]
Serial or Parallel indicator (SP or I_2)	[0.222, 0.846]
Activity Distribution indicator (AD or I_3)	[0, 0.917]
I_4 - I_6	[0, 1]
Resource parameters	
# resources ($\|K\|$)	4
Resource Factor (RF)	0.25, 0.5, 0.75, 1
Resource Strength (RS)	[0, 1]
Resource Constrainedness (RC)	[0.261, 0.825]
Activity characteristics	
Resource requirements (r_{ik})	[1,10]
Duration (d_i)	[1,10]

Table 1: Dataset DC1 characteristics

Objective function coefficients

The leader objective is parametrised by setting the objective function weight α for the levelling of resources to 10 and the weight β of the resource cost objective to 100 based on preliminary computational experiments for finding a balanced trade-off between both leader

objectives and avoiding extreme outcomes (cf. Online Appendix B.1).

Entire dataset
Based upon the variation of the resource parameters and network topology characteristics, we consider a single instance from each category generated in the dataset. As a result, the test data under consideration involves 108 instances, i.e. 36 instances for a particular number of activities ($|N|$).

6.2 General performance and benchmark with other solution approaches

In this section, we discuss the general performance of the algorithm and benchmark the proposed bilevel methodology with the multi-stage and single-stage solution methodologies given their formulation discussed in Section 5. The objective of the experiment is (i) to identify the potential of classical multi-stage approaches, the classical single-stage approach and the proposed algorithm to return (optimal) solutions that are bilevel feasible and (ii) to compare these approaches to each other with respect to the required computational effort. In this way, insight is provided in how classical approaches are able to return relevant solutions for the hierarchical optimisation problem, although their notion of optimality is different as these approaches relax the constraint involving the follower objective. To model the integrated single-stage approach, the weight γ minimising the project duration is fixed and set to 0.1. Table 2 displays the results related to the average performance of the different solution methods for the different tested problem sizes. The comparison is based on both the solution quality and the computational performance. The solution quality is measured via the percentage of optimal solutions that are bilevel feasible (%BL Feas), the percentage of instances solved to optimality (%Opt), the leader objective ($\hat{F}^x$), the number of additionally hired resources ($\sum_k \hat{y}_k$), the sum of the absolute deviations in resource usage ($\sum_{tk}(\hat{j}^+_{tk}+\hat{j}^-_{tk})$), and the follower objective ($\hat{f}^x$). The computational performance is reflected by the CPU time in seconds (CPU^x). The upper index 'x' refers to the type of solution method, i.e. the bilevel method (x = BL), the multi-stage method (x = Seq), and the single-stage method (x = Int).

Table 2 reveals that the classical approaches fail to guarantee to end up with a bilevel feasible solution for all visited instances and are, hence, not proper solution methods for solving the bilevel optimisation problem under study. The multi-stage method 'RACP $\rightarrow$ RLP' mimics best the rationale embedded in a bilevel optimisation problem as for a large percentage of the instances the yielded solutions are bilevel feasible. The other approaches perform worse as bilevel feasible solutions are only attained for approximately 30% to 40% of the instances in case $|N| = 30$. The multi-stage method 'RLP $\rightarrow$ RACP' delivers the worst results as only 72.2% of the instances with $|N| = 10$ are bilevel feasible. This low percentage is due to the fact that the number of resources hired are minimised only in the second stage.
The complexity associated with the notion of bilevel feasibility rises with an increasing number of activities. In this perspective, we observe that (i) not only the classical approaches

$\|N\|$	Solution approach (x)	%BL Feas	%Opt	CPU^x	$\hat{F}^x$	$\sum_k \hat{y}_k$	$\sum_{tk}(\hat{j}^+_{tk} + \hat{j}^-_{tk})$	$\hat{f}^x$
10	**Bilevel** (BL)	100%	100%	16.7	2308.3	13.4	96.4	45.2
	Multi-stage (Seq)							
	RACP ⊎ RLP	83.3%	94.4%	147.6	2351.9	13.9	96.3	44.8
	RACP → RLP	94.4%	97.2%	114.6	2315.8	13.6	96.0	44.8
	RLP → RACP	72.2%	94.4%	190.6	2781.4	18.9	89.3	38.8
	Single-stage (Int)	94.4%	100.0%	12.6	2304.2	13.5	95.4	44.9
20	**Bilevel** (BL)	100.0%	66.7%	776.5	3806.7	17.3	207.6	73.3
	Multi-stage (Seq)							
	RACP ⊎ RLP	55.6%	55.6%	855.3	3703.1	18.7	183.6	69.2
	RACP → RLP	83.3%	66.7%	679.9	3484.2	16.3	185.9	72.1
	RLP → RACP	44.4%	50.0%	975.5	4255.0	25.6	169.4	64.9
	Single-stage (Int)	72.2%	61.1%	748.2	3248.3	13.9	186.2	78.7
30	**Bilevel** (BL)	100.0%	38.9%	1541.6	6546.7	30.8	346.6	93.4
	Multi-stage (Seq)							
	RACP ⊎ RLP	36.1%	33.3%	1218.5	5656.4	27.6	289.5	94.4
	RACP → RLP	61.1%	44.4%	1078.6	5338.9	23.8	296.4	98.6
	RLP → RACP	30.6%	30.6%	1363.5	6296.9	35.4	275.3	90.4
	Single-stage (Int)	41.7%	36.1%	1182.7	5989.7	25.7	342.3	116.9

Table 2: Benchmark comparison between different solution approaches

deliver a smaller percentage of bilevel feasible solutions when the problem size increases, but also (ii) the computational effort (CPU) of the proposed bilevel solution approach increases more rapidly compared to the other approaches as a result of the more complex problem definition and the search for a bilevel optimal solution. The computational effort to solve the problem using the proposed bilevel optimisation method results in very small CPU times for instances with $|N| = 10$, outperforming the classical multi-stage approaches. When tackling larger instances ($|N| = 20$ or 30), the associated CPU time of the proposed bilevel optimisation method significantly increases while the percentage of instances solved to optimality decreases. Moreover, an analysis of the computational performance of all methods reveals that the encountered (rise in) complexity is also contingent on the multi-objective problem definition, the type of decomposition into different hierarchical stages and the specific nature of the tackled problem in the first stage. The incorporation of the resource levelling objective in the primary optimisation problem entangles the search for an optimal solution. Iterative multi-stage optimisation approaches, which calculate a project schedule for every possible duration, also denote large CPU times, especially the approaches 'RACP ⊎ RLP' and 'RLP → RACP'. The single-stage approach, which adds the minimisation of the project duration to the single objective, leads to a relatively high CPU time compared to classical multi-stage approaches as the model needs to be solved only once. However, compared to the bilevel approach, the CPU times are smaller as the single-stage method does not account for the bilevel feasibility such that a better bounding is possible.

When comparing the solutions yielded by the classical approaches and the bilevel optimisation method, we observe that the classical approaches that include the resource cost in the primary objective, i.e. all classical approaches except 'RLP → RACP', find an equal or better quality for the leader objective when instances are solved to optimality. This finding is not surprising since in the bilevel problem the follower objective is included as a constraint

in contrast to the other approaches, which do not guarantee to find an optimal solution that is bilevel feasible. The approach 'RLP → RACP' functions differently as first the resource demand is levelled as best as possible without taking the cost for additional resources into account, such that the number of additional resources hired is larger compared to the other approaches. Note that the results in the table are slightly biased since we were not able to find optimal solutions for all instances within the time limit because of the large(r) complexity or specific functioning of the (iterative) solution methodology.
The relative performance related to the follower objective is dependent upon the characteristics of the approach the bilevel optimisation approach is compared to. Corresponding to the larger number of resources hired, the project durations for the approach 'RLP → RACP' are the smallest. In contrast, the solutions retrieved by the other methods show a significantly smaller number of additional resources hired and a larger project duration, comparable to the project duration resulting from the bilevel solution approach. The notion of bilevel feasibility, however, gives the bilevel approach a particular advantage in the negotiation process relative to multi-stage hierarchical approaches. A candidate - bilevel feasible - solution is characterised by an optimal or minimum project duration, which prohibits that the follower can improve his/her solution and consequently negatively impact the outcome for the leader. A similar conclusion can be made when comparing to the single-stage approach where the project duration is very much determined by the objective weight assigned to the follower objective and which does not require an optimal project duration to identify a feasible solution.

6.3 Impact of algorithm improvement mechanisms

In order to determine the importance and the impact of the different improvement techniques introduced in the procedure, we compare different versions of the proposed algorithm finding (optimal) bilevel solutions. Each version leaves out one specific improvement technique keeping the rest of the algorithm unchanged to specify their contribution and to validate the made design choices. Hence, we exclude either the upper bound calculations to initialise the MIP search of the leader problem (*w/o UB-LP*) or the generation of the tighter lazy constraints of type 2 to cut off bilevel infeasible solutions and formulate constraints of type 1 instead (*w/o LC2-LP*). We also considered a version that excludes or includes all improvement techniques, i.e. (*w/o All*) and (*All*), respectively. Table 3 presents the impact of these settings on the computational performance of the proposed algorithm, averaging the results for the different problem sizes. We report in this table the number of lazy constraints generated of type 1 (cf. equation (21)) ($\#LC1$), the number of lazy constraints generated of type 2 (cf. equation (22)) ($\#LC2$), the number of nodes explored in the branch-and-bound tree ($\#nodes$), the CPU time spent to derive suitable upper bound solutions in the initialisation heuristic ($CPU^{UB,LP}$), the CPU time required to solve the leader problem ($CPU^{BL,LP}$), the CPU time required to solve the follower problem ($CPU^{BL,FP}$), the total required CPU time to solve the bilevel problem to optimality (CPU^{BL}), the percentage of instances solved to

optimality (%Opt), and the percentage of instances for which no feasible solution has been found (%Inf).

$\|N\|$	**Metric**	*w/o UB-LP*	*w/o LC2-LP*	*w/o All*	*All*
10	$\#LC1$	0.0	13.7	22.1	0.0
	$\#LC2$	1.7	0.0	0.0	1.3
	$\#nodes$	7598.3	8161.7	8012.8	7597.4
	$CPU^{UB,LP}$	0.5	7.3	0.5	7.3
	$CPU^{BL,LP}$	4.4	4.6	4.5	4.5
	$CPU^{BL,FP}$	4.0	9.2	11.6	4.9
	CPU^{BL}	9.0	21.1	16.6	16.7
	%Opt	100.0%	100.0%	100.0%	100.0%
	%Inf	0%	0%	0%	0%
20	$\#LC1$	0.0	244.8	299.6	0.0
	$\#LC2$	19.7	0.0	0.0	17.9
	$\#nodes$	198838.3	72677.3	96257.5	185663.1
	$CPU^{UB,LP}$	22.3	77.9	22.4	78.0
	$CPU^{BL,LP}$	432.4	242.8	244.3	437.8
	$CPU^{BL,FP}$	267.8	619.7	607.8	260.8
	CPU^{BL}	722.6	940.4	874.4	776.5
	%Opt	66.7%	58.3%	58.3%	66.7%
	%Inf	17%	0%	36%	0%
30	$\#LC1$	0.0	257.4	242.5	0.0
	$\#LC2$	23.7	0.0	0.0	28.5
	$\#nodes$	117527.2	45575.9	63664.5	112729.1
	$CPU^{UB,LP}$	87.2	361.5	87.3	362.3
	$CPU^{BL,LP}$	942.8	716.1	781.4	914.7
	$CPU^{BL,FP}$	229.7	511.4	447.3	264.6
	CPU^{BL}	1259.7	1589.0	1316.0	1541.6
	%Opt	36.1%	33.3%	33.3%	38.9%
	%Inf	50%	0%	64%	0%

Table 3: Impact of the speed-up mechanisms

Table 3 reveals the beneficial impact of the improvement techniques, which may be either related to the solution quality of the returned solutions or to the computational effort. When taking both these measures into account, the proposed procedure (*All*) leads to the best performance, able to find a feasible solution for all instances in an acceptable time span. When $|N| = 10$, all instances can be solved to optimality. Increasing the number of activities reduces the percentage of instances solved to optimality within a time limit of 1800s to 66.7% ($|N| = 20$) and 38.9% ($|N| = 30$) as the required computational effort increases significantly. The initialisation heuristic has foremost a significant impact on the solution quality, especially for the instances with 20 and 30 activities. Leaving out the initialisation heuristic (*w/o UB-LP*) has a negative impact on the number of instances for which a feasible solution is returned. Without the initialisation heuristic, the proposed procedure is not able to find a feasible solution for 17% of the instances ($|N| = 20$) and 50% of the instances ($|N| = 30$). In addition, the inclusion of the initialisation heuristic increases the number of instances solved to optimality ($|N| = 30$) as the construction of initial upper bound solutions enables the generation of some relevant lazy constraints reducing the size of the branching

tree. As additional CPU time is required to compute these initial upper bound solutions (see $CPU^{UB,LP}$), the total CPU time spent by the proposed procedure CPU^{BL} increases respectively from 9.0s to 16.7s ($|N| = 10$), from 722.6s to 776.5s ($|N| = 20$), and from 1259.7s to 1541.6s ($|N| = 30$). The inclusion of the lazy constraints of type 2 leads to a significant improvement in terms of computational effort. The results reveal that, when solely relying on lazy constraints of type 1 (*w/o LC2-LP*), the algorithm detects a large(r) number of (bilevel infeasible) integer solutions for which a callback in the procedure is invoked to solve the follower problem (see $\#LC1$), consuming significant computational time (see $CPU^{BL,FP}$). Including the strengthened lazy constraint of type 2 significantly reduces the number of (bilevel infeasible) integer solutions encountered (see $\#LC2$). As a result, when leaving out the lazy constraints of type 2, the CPU time (CPU^{BL}) increases significantly from 16.7s to 21.1s ($|N| = 10$), from 776.5s to 940.4s ($|N| = 20$), and from 1541.6s to 1589.0s ($|N| = 30$). Furthermore, the number of instances solved to optimality drop from 66.7% to 58.3% ($|N| = 20$) and from 38.9% to 33.3% ($|N| = 30$).

7 Conclusions

In this paper, we have studied a bilevel project staffing problem involving a single project. The required staffing budget results from a negotiation process between a team leader and a project manager and is determined in an accurate manner based upon the scheduling of activities under resource constraints. The contribution of this paper is threefold. First, we have introduced a new optimisation model representing the decision-making process in a hierarchically organised team related to the staffing and scheduling of a project under resource constraints in a functional organisation. The model considers two entities, i.e. a (team) leader who is in charge of hiring additional employees and tries to make the resource usage as levelled as possible, and a project manager (or follower) trying to minimise the project duration. A bilevel programming formulation is proposed to model the interaction between the distinct entities pursuing their individual goals. Second, we have proposed an exact solution approach thriving on mathematical programming that searches through the bilevel search space and continuously adds so-called lazy constraints restricting the region such that the ability of the leader problem to find an optimal solution of the bilevel optimisation problem increases. We added different speed-up mechanisms to accelerate the performance of the algorithm. Third, we demonstrate that classical single-stage and multi-stage optimisation approaches are inadequate to model and solve the hierarchically organised optimisation problem under study.

There are different ways to extend this study in future research. First, the problem definition can be extended by considering a variant of the resource-constrained project scheduling problem (e.g. multiple projects, different types of (non-)dedicated resources, multiple activity modes). In particular, the proposed bilevel framework is very suitable to model the negotiation process between a team leader and different project managers in a multi-project problem

environment with dedicated resources and independent, non-cooperative project managers. In such an environment, a team leader is standing at the top of the hierarchy and determines the number and mix of (additional) employees allocated to every project whereas the different project managers are responsible for the operational execution of different projects. Second, a solution for the bilevel project staffing problem may be devised in a heuristic manner, enabling to solve larger-sized problem instances. A promising pathway may be the use of an evolutionary population-based meta-heuristic method, which has been very popular in the research involving resource-constrained project scheduling problems, combined with an exact repair mechanism to convert a solution to a bilevel feasible solution. Another direction is to further study the relationship between bilevel optimisation and multi-criteria optimisation to explore the conditions under which the proposed bilevel optimisation problem can be solved as an equivalent - and possibly more scalable - multi-criteria approach.

Acknowledgements

This work was supported by the European Regional Development Fund under the project AI&Reasoning (Reg. No. CZ.02.1.01/0.0/0.0/15_003/0000466).

References

Alfares, H. and Bailey, J. (1997). Integrated project task and manpower scheduling. *IIE Transactions*, 29:711–717.

Baligh, H. (2006). *Organization Structures: Theory and Design, Analysis and Prescription*. Information and Organization Design Series. Springer US.

Bard, J. and Moore, J. (1992). An algorithm for the discrete bilevel programming problem. *Naval Research Logistics*, 39:419–435.

Bassett, M. (2000). Assigning project to optimize the utilization of employees' time and expertise. *Computers and Chemical Engineering*, 24:1013–1021.

Blazewicz, J., Lenstra, J., and Rinnooy Kan, A. (1983). Scheduling subject to resource constraints: classification and complexity. *Discrete Applied Mathematics*, 5:11–24.

Bobera, D. (2008). Project management organization. *Management information systems*, 3:3–9.

Brucker, P., Drexl, A., Mohring, R., Neumann, K., and Pesch, E. (1999). Resource-constrained project scheduling: Notation, classification, models, and methods. *European Journal of Operational Research*, 112:3–41.

Burgess, A. and Killebrew, J. (1962). Variation in activity level on a cyclical arrow diagram. *Journal of Industrial Engineering*, 13:76–83.

Colson, B., Marcotte, P., and Savard, G. (2007). An overview of bilevel optimization. *Annals of Operations Research*, 153:235–256.

Damci, A. and Polat, G. (2014). Impacts of different objective functions on resource levelling in construction projects: A case study. *Journal of Civil Engineering and Management*, 20:537–547.

Deckro, R. and Herbert, J. (1989). Resource constrained project crashing. *Omega International Journal of Management Science*, 17:69–79.

Demeulemeester, E. (1995). Minimizing resource availability costs in time-limited project networks. *Management Science*, 41:1590–1598.

Demeulemeester, E. and Herroelen, W. (2002). *Project scheduling: A research handbook*. International series in operations research & management science (49). Kluwer Academic Publishers, Boston, US.

Dempe, S. (2001). Discrete bilevel optimization problems. Technical report, Inst. für Wirtschaftsinformatik.

Dempe, S. (2002). *Foundations of Bilevel Programming*. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Dobson, M. (2004). *The Triple Constraints in Project Management*. Project Management Essential Library. Berrett-Koehler Publishers, Oakland, US.

El-Rayes, K. and Jun, D. (2009). Resource leveling in construction projects. *Journal of Construction Engineering and Management*, 115:302–316.

Feng, C., Ni, T., and Zhou, G. (2018). Bi-level optimization for an integrated project scheduling-resources supply system and its application. In Xu, J., Cooke, F., Gen, M., and Ahmed, S., editors, *Proceedings of the Twelfth International Conference on Management Science and Engineering Management (ICMSEM 2018). Lecture Notes on Multidisciplinary Industrial Engineering*, pages 119–134. Springer, New York, US.

French, S. (1982). *Sequencing and Scheduling: An introduction to the mathematics of the Job-shop*. John Wiley & Sons, New Jersey, US.

Gan, L. and Xu, J. (2015). Control risk for multimode resource-constrained project scheduling problems under hybrid uncertainty. *Journal of Management in Engineering*, 31:14–41.

Gang, J., Xu, J., and Y., X. (2013). Multiproject resources allocation model under fuzzy random environment and its application to industrial equipment installation engineering. *Journal of Applied Mathematics*, 2013:1–19.

Hartmann, S. and Briskorn, D. (2010). A survey of variants and extensions of the resource-constrained project scheduling problem. *European Journal of Operational Research*, 207:1–15.

Hecheng, L. and Yuping, W. (2008). Exponential distribution-based genetic algorithm for solving mixed-integer bilevel programming problems. *Journal of Systems Engineering and Electronics*, 19:1157–1164.

Heimerl, C. and Kolisch, R. (2010). Scheduling and staffing multiple projects with a multi-skilled workforce. *OR Spectrum*, 32:343–368.

Icmeli-Tukel, O. and Rom, W. (1997). Ensuring quality in resource constrained project scheduling. *European Journal of Operational Research*, 103:483–496.

Kalashnikov, V., Dempe, S., Perez-Valdes, G., Kalashnykova, N., and Camacho-Vallejo, J.-F. (2016). Bilevel programming and applications. *Mathematical problems in Engineering*, 2015:1–16.

Kastor, A. and Sirakoulis, K. (2009). The effectiveness of resource levelling tools for resource constraint project scheduling problem. *International Journal of Project Management*, 27:493–500.

Kerzner, H. (2003). *Project management: A systems approach to planning, scheduling, and controlling*. John Wiley & Sons, New York, US.

Kis, T. and Kovacs, A. (2002). On bilevel machine scheduling problems. *OR Spectrum*, 34:43–68.

Kis, T. and Kovacs, A. (2013). Exact solution approaches for bilevel lot-sizing. *European Journal of Operational Research*, 226:237–245.

Kreter, S., Schutt, A., Stuckey, P., and Zimmermann, J. (2018). Mixed-integer linear programming and constraint programming formulations for solving resource availability cost problems. *European Journal of Operational Research*, 266:472–486.

Leyffer, S., Lopez-Calva, G., and Nocedal, J. (2006). Interior methods for mathematical programs with complementarity constraints. *SIAM Journal on Optimisation*, 17:52–77.

Maenhout, B. and Vanhoucke, M. (2016). An exact algorithm for an integrated project staffing problem with a homogeneous workforce. *Journal of Scheduling*, 19:107–133.

Möhring, R. (1984). Minimizing costs of resource requirements in project networks subject to a fixed completion time. *Operations Research*, 32:89–120.

Moore, J. T. and Bard, J. F. (1990). The mixed integer linear bilevel programming problem. *Operations Research*, 32:911–921.

Neumann, K. and Zimmermann, J. (2000). Procedures for resource leveling and net present value problems in project scheduling with general temporal and resource constraints. *European Journal of Operational Research*, 127:425–443.

Nübel, H. (2001). The resource renting problem subject to temporal constraints. *OR Spektrum*, 23:359–381.

Rieck, J. and Zimmermann, J. (2015). Exact methods for resource levelling problems. In Schwindt, C. and Zimmermann, J., editors, *Handbooks on Project Management and Scheduling (Vol 1)*, pages 361–387. Springer International Publishing, Switzerland.

Rieck, J., Zimmermann, J., and Gather, T. (2012). Mixed-integer linear programming for resource levelling problems. *European Journal of Operational Research*, 221:27–37.

Rodrigues, S. and Yamashita, D. (2010). An exact algorithm for minimizing resource availability costs in project scheduling. *European Journal of Operational Research*, 206:562–568.

Senouci, A. and Eldin, N. (2004). Use of genetic algorithms in resource scheduling of construction projects. *Journal of Construction Engineering and Management*, 130:869–877.

Sinha, A., Malo, P., and Deb, K. (2018). A review on bilevel optimization: From classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, 22:276–295.

Vanhoucke, M., Coelho, J., and Batselier, J. (2016). An overview of project data for integrated project management and control. *Journal of Modern Project Management*, 3:6–21.

Vanhoucke, M., Demeulemeester, E., and Herroelen, W. (2001). On maximizing the net present value of a project under renewable resource constraints. *Management Science*, 47:1113–1121.

Wu, M. and Sun, S. (2006). A project scheduling and staff assignment model considering learning effect. *International journal of advanced manufacturing technology*, 28:1190–1195.

Xu, J., Ma, Y., and Xu, Z. (2015). A bilevel model for project scheduling in a fuzzy random environment. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45:1322–1335.