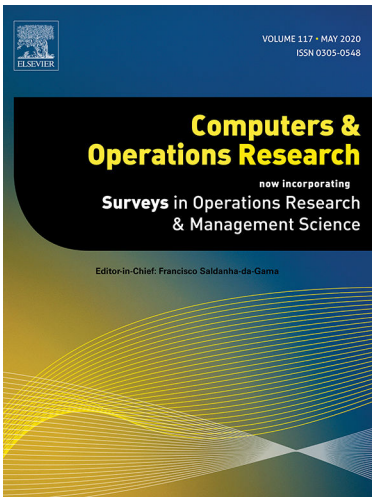
Journal Pre-proofs



A column generation-based diving heuristic to solve the multi-project personnel staffing problem with calendar constraints and resource sharing

M. Van Den Eeckhout, M. Vanhoucke, B. Maenhout

PII: S0305-0548(20)30280-X
DOI: https://doi.org/10.1016/j.cor.2020.105163
Reference: CAOR 105163

To appear in: *Computers and Operations Research*

Received Date: 2 November 2019
Revised Date: 7 September 2020
Accepted Date: 20 November 2020

Please cite this article as: M.V. Den Eeckhout, M. Vanhoucke, B. Maenhout, A column generation-based diving heuristic to solve the multi-project personnel staffing problem with calendar constraints and resource sharing, *Computers and Operations Research* (2020), doi: https://doi.org/10.1016/j.cor.2020.105163

A column generation-based diving heuristic to solve the multi-project personnel staffing problem with calendar constraints and resource sharing

M. Van Den Eeckhout[1], M. Vanhoucke[1,2,3], and B. Maenhout[1]

[1]Faculty of Economics and Business Administration, Ghent University, Tweekerkenstraat 2, 9000 Gent (Belgium), Mick.vanDenEeckhout@ugent.be,Broos.Maenhout@ugent.be,Mario.Vanhoucke@ugent.be
[2]Technology and Operations Management Area, Vlerick Business School, Belgium
[3]UCL School of Management, University College London, UK

Abstract

Project managers are often responsible for the management of multiple projects and associated personnel budgeting decisions. In order to determine the workforce size and mix accurately, we integrate the multi-project scheduling problem and personnel staffing problem. We construct a baseline personnel roster that takes the personnel scheduling constraints into account and model the workload stemming from the project activities as an endogenous variable in the model. In order to decrease the overall staffing budget, we consider the sharing of resources, i.e. personnel resources can be transferred between projects. In the problem under study, we consider unproductive resource transfer times and different restrictions related to the sharing of resources imposed on the schedule of individual workers, which is often neglected in the literature. We propose a multi-stage solution procedure that exploits the optimal linear programming solution, obtained via column generation, to find a high-quality integer solution based on a diving heuristic. In order to improve the computational performance, different speed-up mechanisms are included relying on a state space reduction. Detailed computational experiments are presented to evaluate and benchmark different alternative optimisation strategies. Managerial insights are provided which give insight in the benefits of resource sharing.

Keywords: Multi-project scheduling; Personnel staffing; Column generation; Diving heuristic; Resource sharing

1 Introduction

Personnel resources are one of the most prominent resources in project management to carry out the different activities from a project plan. Larson and Gray (2011) indicated that 30 to 50% of the total project costs are related to personnel resources. Therefore, determining the optimal personnel size and mix and the resulting staffing budget is an important decision for the project manager. As the working duties of personnel members are restricted by different calendar constraints, impacting the daily resource capacity, a baseline personnel roster is

constructed to determine a staffing plan in a more accurate manner (Ernst et al., 2004). The staffing plan can be further improved by integrating project scheduling and personnel staffing, which creates extra scheduling flexibility to lower the staffing costs (Maenhout and Vanhoucke, 2017). When the demand is modelled as an endogenous variable in the personnel staffing problem, the activity start times and the corresponding workload can be guided by the resource availabilities stemming from the baseline personnel roster and/or the resource availability can be changed in function of the activity start times to improve the resource utilisation.

In this paper, we study a multi-project staffing problem for which decisions should be made related to the number of personnel resources and the associated budget allocated to the different projects. Project managers, who typically deal with multiple projects simultaneously (Browning and Yassine, 2010), aim to minimise the personnel budget required to cover the staffing requirements resulting from the scheduling of the activities related to the different projects. To that purpose, we integrate the multi-project scheduling problem and the personnel staffing problem and introduce additional personnel scheduling flexibility by allowing the sharing of resources between projects to decrease the required personnel budget. We consider two resource types, i.e. regular workers, for which a personnel schedule is constructed, and temporary workers, which can be hired on a daily basis and are indispensable to obtain a cost-efficient budget (Wild and Schneewei, 1993; Maenhout and Vanhoucke, 2017). The regular personnel resources can be transferred from one project to another project to handle resource shortages. Since we compose an actual (baseline) personnel schedule to determine the staffing plan, we are able to incorporate unproductive resource transfer times and different time-related restrictions related to the sharing of resources, which is often neglected in the literature. The problem under study is relevant in multiple domains, e.g. accountants performing audits for different customers or consultants/software developers working on different projects for multiple clients (Kuo et al., 2014).

We present a column generation-based diving heuristic to obtain high-quality solutions for the problem under study. The solution methodology is composed out of multiple stages. In the first stage, we calculate the optimal linear programming (LP) solution via column generation. The formulation of the pricing problem, which is related to the scheduling of an individual worker, demonstrates a large complexity containing both the calendar and resource sharing constraints. To improve the computational performance, different speed-up techniques relying on a state space reduction are included. In the second stage, a depth-first diving heuristic is applied to find an integer staffing plan based on the optimal LP solution. Different variable fixing strategies are presented which iteratively fix a subset of the fractional activity start time variables and two variants are proposed to perform a branch-and-bound based on a restricted number of personnel schedules to find a high-quality integer staffing plan. In the third stage, the resource transfers in the solution resulting from the diving heuristic are fixed and the required personnel budget is determined for each project separately taking the transferred resource capacity into account. In the computational experiments, we validate

the design choices of our algorithm making the comparison between different variants. In addition, we benchmark the procedure with other solution procedures commonly applied in the literature. Managerial insights are provided into the benefits of resource sharing, especially related to the unproductive transfer times, the timing of transfers and the number of projects between which resources can be shared.

The remainder of this paper is organised as follows. Section 2 discusses the relevant literature concerning personnel staffing and project scheduling with resource sharing. The problem under study is explained in detail in Section 3. The solution methodology is presented in Section 4. Section 5 analyses the algorithm performance and provides managerial insights concerning the sharing of resources. Finally, conclusions are provided in Section 6.

2 Literature review

In this section, we review the literature related to the problem under study. In Section 2.1, we discuss personnel staffing and scheduling studies involving different departments or working locations. In Section 2.2, we discuss the project scheduling literature that considers the assignment of resources to multiple projects. The literature concerning integrated project and personnel staffing is reviewed in Section 2.3.

2.1 Personnel scheduling with resource transfers

Personnel staffing and scheduling is a well-studied domain in the operations research literature (cf the literature reviews of Ernst et al. (2004) and Van den Bergh et al. (2013)). Staffing problems involve long-term budgeting decisions related to the workforce size and the personnel mix. Scheduling problems consider the construction of a duty roster for a medium-term period given the strategic budgeting decisions by assigning specific workers to working days or tasks. In the remainder of this section, we review the relevant studies that consider the sharing of resources between different entities (e.g. departments, locations) and employee cross-training.

Personnel scheduling across multiple departments/locations

Mabert and Raedels (1977) and Bechtold (1988) are amongst the first authors that considered the sharing of resources for a multi-department personnel planning problem. In their problem definition, no restrictions are imposed on these transfers. In later studies, different authors incorporated resource transfers into their model via transfer costs (Dodin and Elimam, 1997; Bard and Wan, 2006; Batta et al., 2007; Nearchou et al., 2018; Attia et al., 2019; Dahmen et al., 2019). Dodin and Elimam (1997) recognise the relevance of travel times and costs but assume that travel times occur during non-working hours and only incorporate travel costs in their model formulation. Bard and Wan (2006) associate a resource transfer with a drop in productivity due to travel time and reorientation, which is represented by a cost

when moving between two work stations. In their study, they assume a fixed workforce size and minimise the total cost of resource transfers. The trade-off between staffing costs and transfer costs is studied in Batta et al. (2007). An important application of resource transfers is the staffing and scheduling of nursing personnel in a hospital. Worthington and Guy (1988) and Easton et al. (1992) observe that installing float nurses, which are shared between different wards, leads to significant cost improvements. Other studies (e.g. Franz et al. (1989)) study the resource transfers of nurses between individual care units in a hospital in order to satisfy the staffing requirements while minimising the costs related to the resource transfers. Nearchou et al. (2018) studied a personnel staffing problem considering a multi-site context where a set of sites should be staffed for a certain number of periods in the planning horizon. Each time workers are transferred to a site, the site is opened and both a cost and time penalty are incurred. The objective is to minimise the staffing costs and the transfer costs. Furthermore, the concept of transfer costs is also frequently used to model the resource transfers in operational allocation problems where short-term decisions are made in order to balance the resource capacity and the staff demand in response to operational variability (Dahmen et al., 2019).

In the literature, different restrictions have been imposed on the sharing of resources. Gendron (2005), Wright and Mahar (2013) and Nearchou et al. (2018) impose a limit on the number of resource transfers between departments/locations in the planning horizon. Kuo et al. (2014) investigate the impact of the allowed frequency of transfers between locations for the assignment of workers to tasks given a predefined duty roster. Bard and Wan (2008) restricted the resource transfer flexibility by prohibiting some resource transfers from one base workstation to another workstation. Bard and Wan (2008) and Dahmen et al. (2019) impose a minimum number of periods that an employee should be assigned to his home department. Maenhout and Vanhoucke (2013), Nearchou et al. (2018) and Dahmen et al. (2019) impose a minimum consecutive number of periods the workers can be assigned to a specific entity. Maenhout and Vanhoucke (2013) show that when this parameter increases, the staffing cost slightly increases.

Different methodologies have been proposed for the different types of multi-department or multi-location personnel planning problems. Note that most papers in the literature do not consider the demand as an endogenous variable in the model or do not construct an actual personnel roster but rather concern assignment problems (Batta et al., 2007; Nearchou et al., 2018). The relevant methodologies devised in the literature for multi-department staffing and/or scheduling problems are typically multi-stage heuristic methodologies. In a first stage, different studies relax some problem characteristics (e.g. idle times (Bard and Wan, 2006), resource transfer flexibility constraints (Bard and Wan, 2008)) or reduce the solution space (e.g. by solving the problem for a single entity (Attia et al., 2019), a cluster of multiple entities (Al-Yakoob and Sherali, 2007), a subset of employees (Al-Yakoob and Sherali, 2007) and/or a subset of tasks (Belien and Demeulemeester, 2006)). In later stages, improvements are accomplished by increasing the problem complexity and/or the solution space (Belien

and Demeulemeester, 2006; Al-Yakoob and Sherali, 2007; Attia et al., 2019). In some studies, these stages are organised in a hierarchical manner in the sense that decisions taken in earlier stages are imposed on later stages (Bard and Wan, 2008; Dahmen et al., 2019). Solution methodologies are applied such as branch-and-bound methods with a restricted subset of columns (Batta et al., 2007), variable-fixing heuristics (Batta et al., 2007; Al-Yakoob and Sherali, 2008; Maenhout and Vanhoucke, 2013), branch-and-price (Maenhout and Vanhoucke, 2013) and meta-heuristics (Bard and Wan, 2006; Nearchou et al., 2018). Most of these methods rely upon the decomposition of the problem and mathematical programming with application of Dantzig-Wolfe decomposition to define personnel column variables representing feasible lines-of-work for employees.

Cross-training

A large body of literature studied the concept of employee cross-training, which is defined as the training of employees enabling workers to carry out different roles or skills. In this domain, very similar concepts are encountered related to the multi-project staffing problem under study. Cross-training has mostly been studied in the context of a multi-skilling problem. For an overview of multi-skilled workforce scheduling problems, we refer to De Bruecker et al. (2015). Different studies analyse the benefits of cross-training by varying different factors such as the number of departments to which workers can be allocated, the level of cross-training and the demand variability (e.g. Campbell (1999); Brusco (2008); Taskiran and Zhang (2017)). These studies find that when including a small degree of cross-training, most benefits of an entirely cross-trained workforce can be captured. Only some of these studies (e.g. Kher and Malhotra (1994)) explicitly take transfer times or the loss of productive time into account as a result of the required orientation time at the new department. In their analysis, they show that cross-training for at least two departments increases the performance, even when large transfer times are imposed. Eitzen et al. (2004) assume that each employee has a core skill and a supplementary skill and impose the constraint that each employee should work at least one shift related to the supplementary skill each fortnight in order to preserve their skill level. The authors compare different solution methods exploiting the Dantzig-Wolfe decomposition that assign workers to a complete line-of-work, i.e. a column expansion method, the reduced column subset method and a branch-and-price method. Wirojanagud et al. (2007) and Fowler et al. (2008) decide on the amount of hiring, firing and cross-training in order to minimise cost in a job-shop environment with several machine groups. Wirojanagud et al. (2007) propose a decomposition approach with solution space reduction by assigning employees to a subgroup of machines with the assumption that cross-training workers for a subgroup of machines is likely to be almost as effective as cross-training workers among all machines in a factory. Fowler et al. (2008) propose two heuristic rounding mechanisms based on the optimal LP solution.

2.2 Multi-project scheduling with resource transfers

Scheduling activities of a single project given a set of precedence and resource constraints is known as the resource constrained project scheduling problem (RCPSP). Hartmann and Briskorn (2010) review the different variants and extensions within this domain. The resource constrained multi-project scheduling problem (RCMPSP) involves the scheduling of multiple projects simultaneously. For an exhaustive overview of the characteristics studied for the RCMPSP, we refer to Browning and Yassine (2010). In the following, we focus on the different aspects related to resource sharing in a (multi-)project environment.

Bassett (2000) aims to minimise outsourcing costs by maximising the resource utilisation of the employees by assigning them to different projects. Certa et al. (2009) study the allocation of resources to multiple projects and impose a limit on the number of projects an employee can be assigned to. Heimerl and Kolisch (2010) allocate multi-skilled internal and external human resources to staff and schedule different IT-projects. They investigate the impact of centralised planning with shared resources versus decentralised planning with dedicated resources. Felberbauer et al. (2019) extend the deterministic model of Heimerl and Kolisch (2010) by considering a stochastic demand and propose a matheuristic that decomposes the problem into a project scheduling subproblem and a staffing subproblem with fixed resource availabilities. Krüger and Scholl (2009) develop a heuristic procedure for considering resource transfers between activities and/or projects in single- and multi-project environments and introduce the resource constrained multi-project scheduling problem with transfer times (RCMPSPTT). The authors argue that transfers take time when resources are physically moved between locations or when a resource needs reorientation with respect to a new project content and make the linkage with set-up times in production and project scheduling (see Mika et al. (2006) and Allahverdi (2015) for an exhaustive overview). In a follow-up study, Krüger and Scholl (2010) include set-up costs leading to the RCMPSPTTC and propose a framework that classify the resource transfers and the roles of resources. Different MIP formulations are presented to model different resource transfer settings. Walter and Zimmermann (2016) formulate an MIP model that minimises the average team size allocated to different projects, assuming that the already scheduled projects have to be conducted on top of their regular departmental work. Switching between their departmental work and project work is time consuming since the workers should reacquaint themselves with the project status. Pinha et al. (2016) study the combinatorial multi-mode resource constrained multi-project scheduling problem (MMRCMPSP) and define multiple modes (e.g. hiring of temporary personnel) via the formulation of a combinatorial optimisation problem in order to better allocate resources between different projects. Besikci et al. (2015), on the other hand, study the MMRCMPSP, minimising the total weighted tardiness to complete the different projects with a dedicated resource policy prohibiting that resources are shared due to project and/or resource characteristics. The latter references do not consider transfer times. Transfer times are taken into account in the context of a single-project scheduling problem where tasks of a single project are performed in different locations (Laurent et al.,

2017; Kadri and Boctor, 2018).

2.3 Project staffing with calendar constraints

There are only a few studies that compose the project staffing plan accurately by simultaneously constructing a project schedule and a personnel roster taking calendar constraints into account. In these studies, only a single project has been scheduled and hence, resources are dedicated. All methods apply the Dantzig-Wolfe decomposition using personnel pattern variables to assign regular workers to individual days-off schedules in the (restricted) master problem. Alfares and Bailey (1997) and Alfares et al. (1999) study an integrated project task and cyclic manpower scheduling problem respectively with a homogenous and heterogeneous workforce. These authors propose a two-step procedure where the first phase focuses on the scheduling of the project activities and the second phase embodies the project staffing problem with fixed resource demand. Maenhout and Vanhoucke (2016) study the project staffing problem for which duty lines are constructed according to an acyclic days-off scheduling problem and additional personnel scheduling flexibility is introduced via overtime and temporary personnel. The restricted master problem considers the activity start time variables explicitly. A dedicated branch-and-price procedure is proposed that solves a column generation procedure in each node of the search tree. A pricing problem is solved once for the homogeneous set of workers. For the same problem, Maenhout and Vanhoucke (2017) investigate the impact of different kinds of personnel flexibility, i.e. the type of resources and time-related constraints included, on the staffing budget. Van Den Eeckhout et al. (2019) study the project staffing problem with discrete time/resource trade-offs, i.e. the activity duration depends on the number of assigned employees leading to multiple alternative execution modes. A matheuristic is presented that applies different activity-based and personnel-based types of decomposition in order to improve the project and personnel schedule locally in an iterated local search framework. For the same problem, Van Den Eeckhout et al. (2020) present an exact procedure based on an alternative formulation modelling the project scheduling constraints implicitly in the decision variables of the master problem via the Dantzig-Wolfe decomposition. As a result, two types of column variables are defined, i.e. personnel pattern variables and workload pattern variables, and the column generation algorithm needs to iterate between the restricted master problem and two pricing problems. Based upon the optimal LP solution, the procedure decomposes the search space by defining multiple tactical subproblems with a stipulated workforce size. Afterwards, the procedure searches for the optimal integer solution by solving the different tactical subproblems using branch-and-price. In order to speed up the procedure, different cuts are developed and added to the workload pricing problem.

3 Problem definition and formulation

We study a strategic budgeting problem that integrates the multi-project scheduling problem into the personnel staffing problem. In order to accurately determine the size and mix of the required resources, a staffing plan is composed by solving a manpower days-off manpower scheduling problem to construct a baseline personnel roster that is subject to calendar or time-related scheduling constraints and the staff coverage requirements. The employees are assigned to days on and days off and should carry out the scheduled activities of multiple projects. By integrating multi-project planning in the staffing problem, the resource demand is an endogenous variable and delivers flexibility related to the timing of the activities and the corresponding workload. The multi-project scheduling problem specifies the start times of the activities, each with their own duration and resource requirement, and, hence, determines the number of personnel resources required per time unit. The goal of the personnel staffing problem is to determine the minimum budget for regular and temporary workers based on the demand for staff stemming from the multi-project scheduling problem. We introduce additional resource flexibility as workers can be shared between projects. The flexibility of transferring resource capacity between projects is defined according to different resource sharing constraints. These constraints determine the number of projects between which resources are shared, the timing of resource transfers and the unproductive transfer times. The input characteristics of both scheduling problems are discussed in Section 3.1. Section 3.2 provides a mathematical problem formulation of the problem under study. Figure 1 illustrates the problem context and is explained in further sections below.

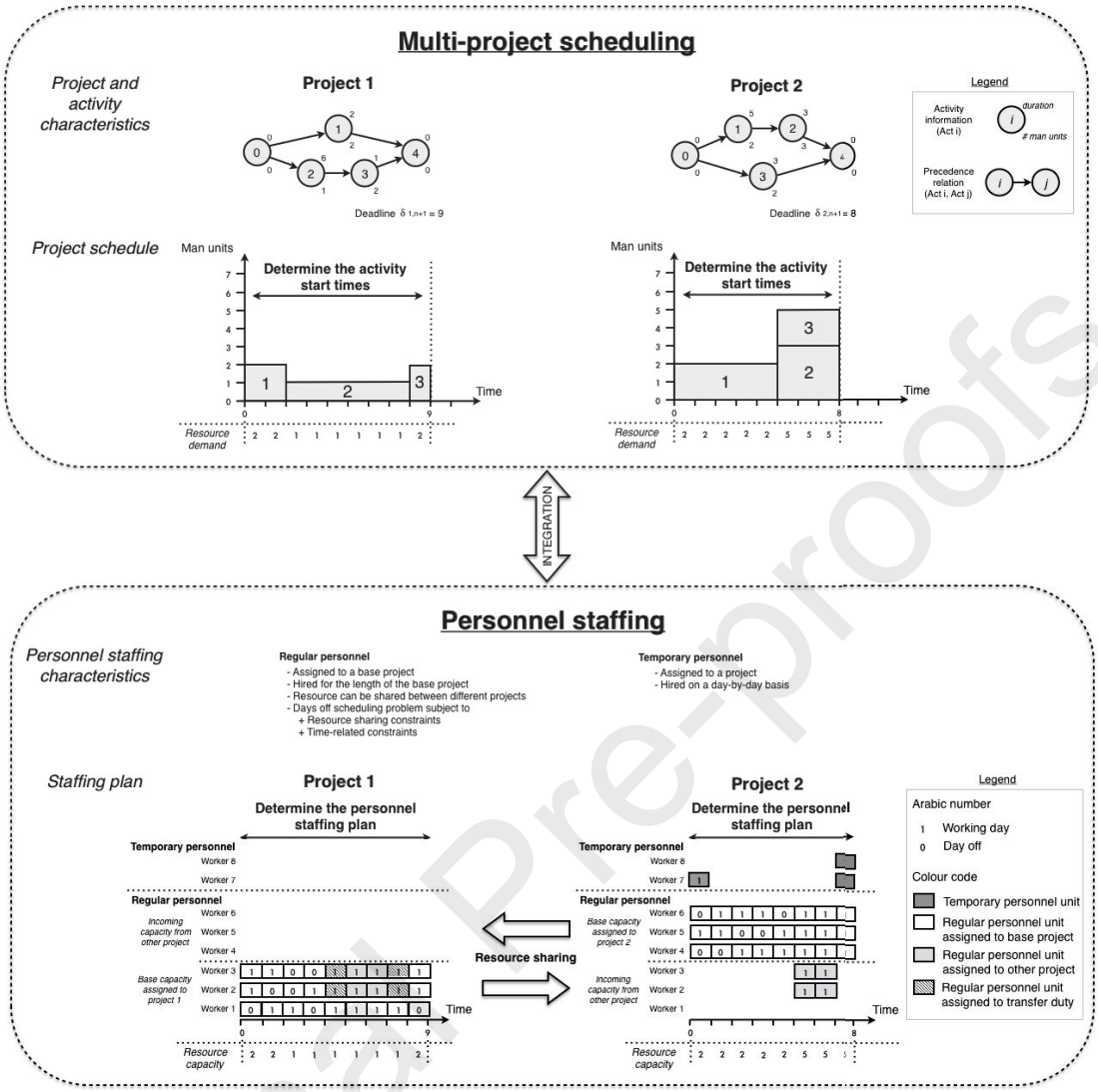


Figure 1: Illustration of the multi-project staffing problem with resource sharing

3.1 Problem context and input characteristics

The (multi-)project and activity characteristics

The demand planning problem embodies a multi-project scheduling problem that considers a set of projects P (index p). Each project p can be characterised by an activity-on-the-node network $G_p = (N_p, A_p)$. The set of nodes N_p encompasses the project activities and the arcs A_p between certain pairs of nodes represent the direct finish-start precedence relationship between a pair of activities with time lag 0. This implies that an activity can only start when all its predecessors are finished. Each project consists of $|N_p|$ activities (index i), numbered from 0 to $n+1$. Each activity $i \in N_p$ ($\forall p \in P$) has a single execution mode characterised by a particular duration d_{pi} and resource demand r_{pi}. The dummy start activity 0 and dummy end activity $n+1$ have a zero duration and zero resource demand. For each project

p, a deadline $\delta_{p,n+1}$ is imposed on the completion of the dummy end activity, which defines the planning period T_p (index t). Each activity should be executed without pre-emption. The scheduling of the activities is guided by the resource availability stemming from the personnel roster. Figure 1 displays the project and activity characteristics and a feasible activity schedule for two projects together with the calculated resource requirements. Note that the two projects have a different deadline.

The personnel staffing characteristics

A staffing plan is constructed for regular and temporary workers by composing a baseline personnel roster to accurately determine the resource availability. A decision should be made related to the required capacity and mix of regular and temporary workers to carry out the workload stemming from the multiple projects on each day of the planning horizon. The regular workers are assigned to a certain base project $p' \in P$ and individual personnel schedules or lines-of-work with a planning horizon $T_{p'}$. The length of the planning horizon is equal to the deadline of the base project p', i.e. $|T_{p'}| = \delta_{p',n+1}$. These individual lines-of-work embody a sequence of days on and days off. In addition, every workday embodies a working duty, i.e. a regular worker staffs either an activity stemming from project $p \in P$ or a transfer duty when the worker is switching between projects. The assignment of regular employees to working days is guided by a set of time-related or calendar constraints, i.e. counter and sequence constraints. Therefore, the planning period of a project $T_{p'}$ is divided into different unit time periods $l \in L_{p'}$ with length $|T_l|$. In each such unit time period, an employee should work a minimum (w^{min}) and maximum (w^{max}) amount of days. Furthermore, restrictions are imposed on the minimum (n^{min}) and maximum (n^{max}) consecutive workdays and on the minimum (f^{min}) and maximum (f^{max}) consecutive days off. These constraints restrict the availability of regular workers, which are hired for the entire planning period at a cost $c^{RG} \times |T_{p'}|$ with c^{RG} equal to the daily hiring cost of a regular worker. The temporary workers can be hired for a single day at a cost c^{TP} and are assigned to a particular project p.

The personnel resources can be shared between different projects to cover the peak demand in the schedule of a particular project or to handle the resource shortage in the staffing plan of the project. We assume a homogeneous workforce, i.e. each worker can work on all projects without productivity loss. Whenever a regular worker switches between projects, a resource transfer takes place. Based on the literature, the following restrictions related to the sharing of resources, limiting the resource transfer flexibility, are taken into account, i.e.

- Each regular worker can work at most upon ω projects in the planning horizon.
- An employee should be at least α^{min} consecutive days allocated to a certain project before he can be transferred to another project.
- An employee should perform at least one working duty at his base project in a consecutive period of α^{max} days.
- A resource transfer time of β days is incurred when a resource transfer takes place between

projects. These days represent the time workers need to familiarise with the new project and are therefore counted as working duties. As a result of this resource transfer time, workers cannot be assigned to project duties but are allocated to a transfer duty during β consecutive days.

- A regular worker should be assigned on the first and last day of the planning horizon to his base project. Similarly, the first day after a resource transfer, the workers should effectively staff a particular activity of the newly assigned project. This ensures that an employee works primarily duties related to his base project, unless there is a capacity shortage to staff activities of another project.

Figure 1 displays a staffing plan for each project, i.e. a baseline personnel roster for regular workers and the number of temporary workers in line with the project workload of the associated project. For project 1, three regular workers and no temporary workers are hired. For project 2, three regular workers and three temporary workers are hired. Note that some personnel resources, originally assigned to project 1, are shared with project 2 and a resource transfer takes place on day 5 and 6 and associated transfer duties are scheduled on day 4 and 7. In this way, a smaller number of budgeted resources is required to cover the resource demand stemming from the multiple project schedules.

3.2 Problem formulation

Different authors that studied the project staffing problem with calendar constraints (Maenhout and Vanhoucke, 2016; Van Den Eeckhout et al., 2019) have proposed a formulation based on the Dantzig-Wolfe decomposition embodying a master problem with a large number of column variables (Section 3.2.1) and a subproblem defining the feasibility of these columns (Section 3.2.2) (Vanderbeck, 2000). In the following, we explain the decision variables of the models, provide a mathematical formulation and discuss the relationship between both models.

3.2.1 The master problem

The master problem integrates the personnel staffing problem and the multi-project scheduling problem by composing the staffing plan with minimum cost and selecting adequate schedules for the different projects. In the formulation, we consider the start time variables for each project activity explicitly and personnel staffing variables to schedule regular workers and temporary workers over the planning horizon. The personnel staffing variables assign regular workers to feasible individual lines-of-work, which consider the personnel scheduling constraints implicitly, and are labelled as column variables. Note that the considered resource-constrained project scheduling problem, which arises when considering fixed resource availabilities e.g. based on a known staffing plan, has been shown to be NP-hard (Garey and Johnson, 1979; Blazewicz and Rinnooy Kan, 1983). When considering the re-

source demand as fixed e.g. stemming from a known project schedule, the problem embodies a non-cyclic personnel staffing problem. Findings by Garey and Johnson (1979) and Brucker et al. (2011) indicated that the non-cyclic staffing problem to minimise the number of employees to perform only one task is already NP-hard. Brunner and Kohler (2013) proved that the personnel staffing problem under study without resource sharing constraints and constraints related to the minimum number of working days is NP-complete.

Decision variables

The start times of the project activities are guided by the project scheduling variables v_{pit} which take the value 1 if activity i of project p starts at time t, 0 otherwise. The variables $x_{jp'}$ determine the number of workers assigned to pattern j with base project p', whereas the variables o_{pt} determine the number of temporary workers hired to work on project p on day t. A personnel line-of-work or pattern j, related to base project p', consists of a sequence of days-on and days-off assignments, allocating the worker on each day to a particular project or to a transfer duty, represented by the parameters $a_{jp'pt}$ and $b_{jp't}$. The parameters $a_{jp'pt}$ are equal to 1, if day t embodies a working day on project p, 0 otherwise. The parameters $b_{jp't}$ are equal to 1 if day t is a transfer duty, 0 otherwise. The set $J_{p'}$ denotes the set of patterns that postulate project p' as the base project. The total set of patterns J is equal to $\cup_{p' \in P} J_{p'}$. The cost of a personnel pattern j ($c_{jp'}$) is equal to the length of this planning period of project p' ($|T_{p'}|$) multiplied by the daily cost to hire regular workers (c^{RG}), i.e. $c_{jp'} = |T_{p'}| \times c^{RG}$.

Mathematical problem formulation

$$\text{Minimise} \quad \sum_{p' \in P} \sum_{j \in J_{p'}} c_{jp'}\, x_{jp'} + \sum_{p \in P} \sum_{t \in T_p} c^{TP}\, o_{pt} \tag{1}$$

$$\text{subject to} \quad \sum_{p' \in P} \sum_{j \in J_p} a_{jp'pt}\, x_{jp'} + o_{pt} - \sum_{i \in N_p} \sum_{t^* = max(1, t - d_{pi} + 1)}^{t} r_{pi}\, v_{pit^*} \geq 0 \quad \forall p \in P, \forall t \in T_p \tag{2}$$

$$\sum_{t \in T_p} v_{pit} = 1 \quad \forall p \in P, \forall i \in N \tag{3}$$

$$\sum_{t \in T_p} t\, v_{pjt} - \sum_{t \in T_p} t\, v_{pit} - d_{pi} \geq 0 \quad \forall p \in P, \forall (i,j) \in A_p \tag{4}$$

$$\sum_{t \in T_p} t\, v_{p,n+1,t} \leq \delta_{p,n+1} \quad \forall p \in P \tag{5}$$

$$x_{jp'} \geq 0 \text{ and integer} \quad \forall p' \in P, \forall j \in J_{p'}$$

$$o_{pt} \geq 0 \text{ and integer} \quad \forall p \in P, \forall t \in T_p$$

$$v_{pit} \text{ binary} \quad \forall p \in P, \forall i \in N, \forall t \in T_p \tag{6}$$

The objective function (1) minimises the staffing budget for all projects consisting of the cost of regular and temporary workers. Constraints (2) link the multi-project scheduling problem and the personnel staffing problem, enforcing that for each project and each day a sufficient number of personnel resources are available to cover the demand of the activities in progress. Constraints (3) assign exactly one start time to each activity. Constraints (4)

represent the precedence constraints, indicating that a successor activity cannot start before its predecessors are finished. The project deadlines are enforced by constraints (5). The domains of the different variables are stated in constraints (6).

3.2.2 The subproblem

The subproblem defines the feasibility of an individual line-of-work for a regular personnel member, impacting the set of patterns J and the characteristics of the personnel patterns considered in the master problem. The feasibility of a personnel pattern is related to (i) the sequence of days on and days off assignments and (ii) the allocation of the employee to the different projects and the timing of possible resource transfers. The subproblem is further defined by the base project to which the regular personnel members are assigned as the scheduling horizon is dependent on the deadline of the considered base project. The feasibility of a personnel pattern j when project p' is set as the base project is controlled by the following decision variables and constraints:

Decision variables
The duty assignments composing such a line-of-work are determined by the decision variables u_{pt} and e_{pt}. The variables u_{pt} (e_{pt}) are equal to 1 if day t is a working day (day off) assigned to project p, 0 otherwise. The variable y_p is equal to 1 if the worker has been assigned to project p on some day in the planning horizon. The variables q_t take the values 1 if day t is a transfer duty, 0 otherwise. The variables k_t are equal to 1 if a resource transfer starts at day t, 0 otherwise. Note that the decision variables u_{pt} and q_t define the values of the parameters $a_{jp'pt}$ and $b_{jp't}$ for personnel pattern j as considered in the master problem formulation (cf Section 3.2.1).

Personnel scheduling constraints
Both time-related constraints and restrictions related to the sharing of resources are included in this subproblem, i.e.

- **Time-related constraints -** Constraints (7) state that on each day, the employee should be either assigned to a working day or a day off and a particular project, or the worker is assigned to a transfer duty. Constraints (8) and (9) limit the minimum and maximum working assignments in a unit time period. Note that the transfer duty assignments are also considered as a working assignment. Constraint (10) and (11) restrict the minimum and maximum number of consecutive working assignments. More specifically, constraints (10) state that if day t is a working day and day $t-1$ is a day off, then the next n^{min} days (including day t) should be working days. Constraints (12)-(13) are the constraints related to the minimum and maximum consecutive number of days off.

$$\sum_{p\in P}(u_{pt}+e_{pt})+q_t=1 \qquad \forall t\in T_{p'} \tag{7}$$

$$\sum_{t\in T_l}(\sum_{p\in P}u_{pt}+q_t)\geq w^{min} \qquad \forall l\in L_{p'} \tag{8}$$

$$\sum_{t\in T_l}(\sum_{p\in P}u_{pt}+q_t)\leq w^{max} \qquad \forall l\in L_{p'} \tag{9}$$

$$\sum_{t}^{t+n^{min}-1}(\sum_{p\in P}u_{pt}+q_t)-n^{min}\,(\sum_{p\in P}u_{pt}+q_t)\,(1-\sum_{p\in P}u_{p,t-1}-q_{t-1})\geq 0 \qquad \forall t\in\{2,...,|T_{p'}|-(n^{min}-1)\} \tag{10}$$

$$\sum_{t}^{t+n^{max}}(\sum_{p\in P}u_{pt}+q_t)\leq n^{max} \qquad \forall t\in\{1,...,|T_{p'}|-n^{max}\} \tag{11}$$

$$\sum_{p\in P}\sum_{t}^{t+f^{min}-1}e_{pt}-f^{min}\sum_{p\in P}e_{p,t-1}\,(1-\sum_{p\in P}e_{pt})\geq 0 \qquad \forall t\in\{2,...,|T_{p'}|-(f^{min}-1)\} \tag{12}$$

$$\sum_{t}^{t+f^{max}}\sum_{p\in P}e_{pt}\leq f^{max} \qquad \forall t\in\{1,...,|T_{p'}|-f^{max}\} \tag{13}$$

- **Resource sharing constraints -** Constraints (14) enforce that the variables y_p are equal to 1 if the worker has been assigned to project p in the planning horizon. Constraint (15) states the employee can work at most upon ω projects during the planning horizon. Constraints (16) stipulate the minimum number of consecutive days a worker is assigned to a project before he can be transferred to another project. Constraints (17) state that in a timespan of α^{max} days, the employee should work at least one day on the base project p'. Constraints (18) and (19) model the resource transfers using the variable q_t and k_t. Constraint (18) models that a resource transfer is initiated ($k_t=1$) with the first transfer duty, i.e. an employee has no transfer duty at time $t-1$ ($q_{t-1}=0$), but has a transfer duty on day t ($q_t=1$). Constraints (19) state that the resource transfer requires β consecutive days. Note that the time-related constraints (7)-(13) view transfer duties as working duties such that a resource transfer cannot be interrupted by a day off. Constraints (20) and (21) state that the employee should be assigned to the base project p' on the first and last day of the planning period, respectively.

  The previous constraints are however not sufficient to correctly model the resource transfers when the transfer time $\beta>0$. For that purpose, we prohibit the consecutive assignment of a worker to different projects via constraint (22). This constraint stipulates that a worker cannot switch between projects on consecutive days. The worker can only be assigned to project p on day t, if the worker has not been assigned to a different project on day $t-1$. Constraints (23) state that the first day after a resource transfer should be a working day. An employee, allocated to a particular project, can

only be given a day off on day t when day $t-1$ was either a day off or a working duty on that project. Note that constraints (18), (19), (22) and (23) are only included when the transfer time $\beta > 0$.

$$\sum_{t \in T_p} u_{pt} \leq M\, y_p \qquad \forall p \in P \tag{14}$$

$$\sum_{p \in P} y_p \leq \omega \tag{15}$$

$$\sum_{t}^{t+\alpha^{min}-1} (u_{pt} + e_{pt}) - \alpha^{min}\, u_{pt}\, (1 - u_{p,t-1}) \geq 0 \qquad \forall t \in \{2, ..., |T_{p'}| - (\alpha^{min} - 1)\}, \quad \forall p \in P \tag{16}$$

$$\sum_{t}^{t+\alpha^{max}} u_{p't} \geq 1 \qquad \forall t \in \{1, ..., |T_{p'}| - \alpha^{max}\} \tag{17}$$

$$q_t - q_{t-1} \leq k_t \qquad \forall t \in \{2, ..., |T_{p'}|\} \tag{18}$$

$$\sum_{t}^{t+\beta-1} q_t \geq \beta\, k_t \qquad \forall t \in \{1, ..., |T_{p'}| - (\beta - 1)\} \tag{19}$$

$$u_{p'0} + e_{p'0} = 1 \tag{20}$$

$$u_{p'|T_{p'}|} + e_{p'|T_{p'}|} = 1 \tag{21}$$

$$\sum_{p^* \in P \setminus \{p\}} u_{p^*,t-1} + \sum_{p^* \in P \setminus \{p\}} e_{p^*,t-1} + u_{pt} + e_{pt} \leq 1 \qquad \forall t \in \{2, ..., |T_{p'}|\}, \quad \forall p \in P \tag{22}$$

$$e_{pt} \leq e_{p,t-1} + u_{p,t-1} \qquad \forall t \in \{2, ..., |T_{p'}|\}, \quad \forall p \in P \tag{23}$$

- **Domain constraints -** Constraints (24) state the domains of the different decision variables.

$$\begin{aligned} &u_{pt}, e_{pt} \text{ binary} && \forall p \in P, \quad \forall t \in T_{p'} \\ &y_p \text{ binary} && \forall p \in P \\ &k_t, q_t \text{ binary} && \forall t \in T_{p'} \end{aligned} \tag{24}$$

4 Solution procedure

In this section, we propose a Column Generation-based Diving Heuristic to solve the multi-project staffing problem under study with resource sharing and calendar constraints. In the literature, different methodologies have been proposed for related problems using column generation. However, the definition of a singular pricing problem that considers all project assignments during the construction of an individual line-of-work leads to a slow convergence of the column generation algorithm and is computationally inefficient. In order to accelerate the search for a high-quality solution, we propose a multi-stage solution methodology that includes different types of state space reduction. The different steps of the procedure are displayed in Algorithm 1 and are discussed in detail below.

In the *first* stage, the optimal LP solution in the root node of the search tree is obtained using a column generation algorithm since it is inefficient to generate all possible personnel

Algorithm 1 Column Generation-based Diving Heuristic

Find the optimal LP solution in the root node via column generation (Section 4.1 and 4.2)
Decompose multi-project staffing problem into multiple single-project staffing problems with dedicated resources
for each base project **do**
    Determine optimal personnel budget
    Retain generated personnel schedules
Calculate initial upper bound value by summing individual personnel budgets
Solve column generation algorithm in the root node of the search tree

Diving heuristic (Section 4.3)
if Optimal LP solution is fractional **then**
    do
        Fix activity start time variables with value 1
        Fix (set of) fractional activity start time variable(s) to 1
        Solve column generation algorithm
        Increase depth of the search tree
    while Fractional activity start time variables exist
    Apply restricted branch-and-bound to construct an integer multi-project staffing plan

Improvement step (Section 4.4)
    Fix resource transfers and determine incoming and outgoing resource capacity
    Decompose multi-project staffing problem into multiple single-project staffing problems
    for each base project **do**
        Adapt staffing constraints (eq. (2))
        Determine optimal personnel budget
    Calculate final upper bound value by summing individual personnel budgets

schedules. In Section 4.1, we present the column generation algorithm and the associated restricted master problem and pricing problem. Column generation relaxes the integrality conditions and works only with a subset of all possible variables. Additional variables are included via a pricing step based on the dual prices returned by the restricted master problem. The proposed column generation is improved by using different dedicated speed-up techniques, discussed in Section 4.2. In this way, we start the column generation procedure via an initialisation procedure that considers the multi-project staffing problem with dedicated resources, i.e. the project staffing problem is solved for the different projects independently. This efficient initialisation procedure devises an upper bound for the procedure and finds a set of columns to initialise the column generation procedure. In order to tackle the increased complexity resulting from the multi-project staffing problem with resource sharing, different speed-up mechanisms are implemented in the pricing procedure. The most prominent one is the state space reduction mechanism that reduces the number of projects a personnel member can be assigned to in a line-of-work in order to decrease the complexity of the pricing problem. In this way, different strategies are devised to determine the order and characteristics of the different pricing problems that are visited, to steer the convergence of the column generation algorithm towards the optimal LP solution. If the optimal LP solution is integer, we have found an optimal solution for the original integer problem. If not, we need additional steps to find a high-quality integer solution.

In the *second* stage, a greedy diving heuristic is applied to convert the fractional solution into a feasible integer solution, which is detailed in Section 4.3. The diving heuristic repetitively fixes one or several fractional assignments in a depth-first search without backtracking. The variable fixing is related to the project scheduling variables and we explore different diving strategies, which determine the number of variables that are fixed in each node. Whenever

variables are fixed, the depth of the branching tree is increased and a column generation procedure is invoked in each newly visited node in the tree. Once all activity start time variables have been fixed, a restricted branch-and-bound is applied given the set of personnel schedules that are generated in the process, to construct an integer multi-project staffing plan.

In the *third* stage, the incumbent solution is improved by optimising the staffing plans of the different projects separately given the resource transfers stemming from the solution found by the diving heuristic in the second step (Section 4.4). These resource transfers lead to additional or reduced resource capacity from regular workers who are assigned to a different project than their base project. In other words, we determine the in- and outgoing resource capacity transfers and optimise the staffing plan for each project independently given this transferred resource capacity.

4.1 Column generation

The column generation algorithm iterates between the so-called restricted master problem (RMP) and the pricing problem (PP). The restricted master problem is similar to the defined master problem (eqs. (1)-(6)) but takes only a subset of the column variables $\bar{J}_{p'} \subset J_{p'}$ into account. In order to find the optimal LP solution, the integrality conditions of the (restricted) master problem (eq. (6)) are relaxed. Note that as a result of the presence of temporary workers in the problem definition, a feasible solution to the RMP always exists, even if $\bar{J} = \emptyset$.

Based on the dual variables of the restricted master problem, the pricing problem searches for feasible columns with negative reduced cost in order to decrease the objective function value of the restricted master problem. The formulation of the personnel pricing problem is dependent on the considered base project p' and may be solved multiple times in different guises according to the selected characteristics following the embedded state space reduction approach (cf Section 4.2.2). The personnel pricing problem embodies an acyclic days-off scheduling problem that comprehends the personnel scheduling constraints as postulated in the subproblem definition (eqs. (7)-(24)). The objective function (eq. (25)) of the pricing problem minimises the reduced cost (RC), which is determined by the cost of the personnel pattern $c_{jp'}$ and the dual prices π_{pt} related to constraints (2) resulting from the restricted master problem. This pricing problem is a weighted resource-constrained shortest path problem with capacity constraints, which has been shown to be NP-hard (Garey and Johnson, 1979; Dumitrescu and Boland, 2003). In the literature, different approaches have been proposed for solving this problem, amongst them are branch-and-bound approaches, which is applied in this study, implemented via a commercial IP solver.

$$\text{Minimise} \qquad RC = c_{jp'} - \sum_{p \in P} \sum_{t \in T_p} \pi_{pt}\, u_{pt} \tag{25}$$

Whenever a new column has been added to the restricted master problem, a new iteration is performed and the restricted master problem is resolved. The procedure is terminated when no more columns with negative reduced cost can be found (Barnhart et al., 1998).

4.2 Speed-ups for the column generation algorithm

In order to decrease the computational time of the column generation algorithm, different speed-up mechanisms are implemented in the algorithm. In Section 4.2.1, the solution methodology is discussed to construct an initial heuristic solution. Section 4.2.2 presents the state reduction approach defining the sequence and characteristics of the pricing problems to be solved. Section 4.2.3 discusses other included speed-up mechanisms.

4.2.1 Initialisation of the column generation algorithm

In order to initialise the column generation algorithm, we first construct an initial upper bound assuming that all resources are dedicated to a specific project since the sharing of resources between different projects with explicit resource transfers times complicates the problem under study significantly. This initial heuristic applies a heuristic state space decomposition of the multi-project staffing problem setting ω equal to 1. The problem is decomposed into multiple single-project staffing problems determining the personnel budget for each project separately. Summing the personnel costs for the different projects leads to the total required personnel budget, which is set as an initial upper bound. This upper bound can be used to analyse the benefits of resource transfer flexibility, which will be discussed in the computational experiments. In order to find an initial solution, we apply the branch-and-price procedure of Van Den Eeckhout et al. (2020). Preliminary experiments showed that this solution procedure leads to better results to obtain the personnel budget for individual projects, compared to other solution procedures in the literature. Note that the different sets of personnel patterns that are generated to find the initial incumbent with resources dedicated to a specific project, are used to initialise the column generation algorithm for finding the optimal LP solution of the multi-project staffing problem with shared resources.

4.2.2 State space reduction to solve the pricing problem

In the proposed problem formulation, the sharing of the resources between projects is modelled implicitly in the personnel pattern variables of the restricted master problem and is explicitly modelled in the personnel pricing problem. As a result, this pricing problem contains a large number of variables and constraints, which impacts the problem complexity and associated solution time. In order to decrease the computational time to solve the personnel pricing problem, a state space reduction (SSR) approach is applied in which iteratively,

a subset of the assignments in the pricing problem are not eligible and are fixed to zero. The construction of the state space reduction algorithm is such that the column generation procedure is guaranteed to find the optimal LP solution. We start from a smaller number of eligible assignments and whenever we are not able to find a column with negative reduced cost, the state space is adapted by gradually increasing its size or changing the relevant set of variables until the complete solution space is explored. This is motivated by two factors. First, the computational time increases if the possible number of assignments and constraints increase. Second, previous research showed that by including a limited degree of flexibility, the largest cost improvements can already be ensured.

The state space reduction approach is guided by the parameters SSR^{strat}, $Project^{strat}$ and $Priority^{strat}$. These parameters steer the computational effort and the convergence of the column generation algorithm towards the optimal LP solution by defining the sequence of the type of pricing problems that are solved in a successive manner. The parameter SSR^{strat} determines the set of different state spaces S to be visited for a certain base project. The personnel pricing problem should be solved for each base project, which would lead to a total of $|S| \times |P|$ pricing problems. However, not all these subproblems should be solved in each column generation iteration. Once a personnel pattern with negative reduced cost is found, the restricted master problem can be revisited, leading to more accurate dual variables to solve the next pricing problem, which may again feature a lower number of variables and constraints. The parameter $Project^{strat}$ defines the project ordering according to which the related personnel pricing problems are visited. The parameter $Priority^{strat}$ indicates if priority should be given to solving all pricing problems successively related to a certain state space or related to a certain base project. Note that when the column generation algorithm reaches its final iterations, the impact of these parameters will decrease since more/all pricing problems should be solved to obtain a personnel pattern with negative reduced cost.

Definition and ordering of the visited state spaces (SSR^{strat})

The state space reduction strategy (SSR^{strat}) stipulates the amount $|S|$, the order and the characteristics of the state spaces that are potentially visited in the pricing step. The state space of a pricing problem is defined via the maximum number of projects ω and the set of eligible projects $P^e \subseteq P$ to which the regular workers can be assigned. These parameters determine the relevant decision variables and constraints in a pricing problem and the associated computational complexity of this pricing problem (cf Appendix A). The smaller the number of projects considered, the larger the number of variables in the pricing problem that are set to 0. As a result, constraint (26) is added to the personnel pricing problem (7)-(25) and the set P^e will replace the set P in constraints (7)-(24). When $\omega = 1$, constraint (27) can be added stating that no resource transfers are allowed.

$$\sum_{p \in P: p \notin P^e} \sum_{t \in T_{p'}} u_{pt} + \sum_{p \in P: p \notin P^e} \sum_{t \in T_{p'}} e_{pt} + \sum_{p \in P: p \notin P^e} y_p = 0 \tag{26}$$

$$\sum_{t \in T_{p'}} k_t + \sum_{t \in T_{p'}} q_t = 0 \tag{27}$$

Every state space reduction strategy starts from a state space without the sharing of resources between projects ($\omega = 1$, $|P^e| = 1$). When no eligible personnel pattern is found with a negative reduced cost, the state space is gradually increased and a different state space will be visited by increasing the maximum number of projects ω and/or adapting the set of eligible projects $P^e \subseteq P$. The pricing problem with the largest state space ($\omega = |P|$, $|P^e| = |P|$) is solved last in the pricing step. We consider the following strategies to define the different state spaces:

- $\mathbf{SSR^{strat} = 0}$: A single state space is constructed where an employee can be assigned to every project ($\omega = |P|$, $|P^e| = |P|$). This strategy resembles the base setting where no state space reduction is applied and only the original personnel pricing problem (7)-(25) is solved for each base project in the pricing step.
- $\mathbf{SSR^{strat} = 1}$: This strategy considers two state spaces, i.e. a state space without resource sharing ($\omega = 1$, $|P^e| = 1$) and a state space with complete resource sharing where the personnel resources are potentially shared between all projects ($\omega = |P|$, $|P^e| = |P|$).
- $\mathbf{SSR^{strat} = 2}$: In this strategy, we consider three state spaces, i.e. (i) without resource sharing ($\omega = 1$, $|P^e| = 1$); (ii) with a limited degree of resource transfer flexibility ($\omega = 2$, $|P^e| = |P|$), i.e. the employee is allowed to work on one additional project apart from his base project, without specifying the additional project; (iii) with complete resource sharing ($\omega = |P|$, $|P^e| = |P|$).
- $\mathbf{SSR^{strat} = 3}$: This strategy considers $|P|$ different state spaces, defined by the maximum number of projects to which an employee can be assigned. In this way, the parameter ω is gradually increased in the interval $[1, |P|]$. All projects are eligible for selection ($|P^e| = |P|$).
- $\mathbf{SSR^{strat} = 4}$: This strategy visits the same state spaces as defined by $SSR^{strat} = 2$. However, when $\omega = 2$, the additional project to which the regular worker can be assigned during the planning horizon apart from the base project is specified via the set P^e ($\omega = 2$, $|P^e| = \omega$).
- $\mathbf{SSR^{strat} = 5}$: This strategy visits the same state spaces as defined by $SSR^{strat} = 3$. However, when $\omega \in [2, |P| - 1]$, the additional project(s) to which the regular worker can be assigned during the planning horizon apart from the base project is specified via the set P^e ($|P^e| = \omega$).

To illustrate the defined state space strategies, Table 1 displays the characteristics of all pricing problems for the strategies $SSR^{strat} = 0$, 2 and 4 given $|P| = 4$. The second column displays the value of the parameter ω for each state space. The other columns represent the projects included in the set P^e, where the number before the semicolon indicates the base

project. For example '1: 2' indicates the state space where project 1 is the base project and employees can also be assigned to project 2. When no SSR is applied, $|P|$ pricing problems exist and only the order of base projects should be determined to visit the corresponding pricing problems. When SSR is applied, the number of pricing problems increase (e.g. 20 pricing problems for $SSR^{strat} = 4$).

	ω	$\mathbf{P^e}$			
$SSR^{strat} = 0$	4	1: 234	2: 134	3: 124	4: 123
$SSR^{strat} = 2$	1	1:	2:	3:	4:
	2	1: 234	2: 134	3: 124	4: 123
	4	1: 234	2: 134	3: 124	4: 123
$SSR^{strat} = 4$	1	1:	2:	3:	4:
	2	1: 2	2: 1	3: 1	4: 1
	2	1: 3	2: 3	3: 2	4: 2
	2	1: 4	2: 4	3: 4	4: 3
	4	1: 234	2: 134	3: 124	4: 123

Table 1: Illustration of state space characteristics for $SSR^{strat} = 0$, 2 and 4 with $|P| = 4$

Note that the state spaces are always visited with an increasing value for the parameter ω. However, for $SSR^{strat} = 4$ and $SSR^{strat} = 5$, the additional eligible project(s) should be specified as well leading to different state spaces with the same value for ω. Preliminary experiments showed that applying a random order with respect to the state space definition of P^e performed best.

Ordering personnel pricing problems according to the (base) project ($Project^{strat}$)
The parameter $Project^{strat}$ defines the order of base projects according to which the related pricing problems are visited. We distinguish the following categories:

- **Random**: The order of projects is determined in a random manner.
- **Deterministic**: The order of projects is based on instance information related to the different projects. More specifically, we consider the project deadline and work content in increasing/decreasing order, and the logical order of projects based upon the project numbering.
- **Dynamic**: The order of projects is based on information that is updated during the column generation algorithm. More specifically, the projects are ordered based on information of the last pricing problem related to a specific project, i.e. the required CPU time or the reduced cost (RC), or based on the information related to the solution of the restricted master problem, i.e. the number of workers assigned to the project.

Note that to fully explore the impact of the random and dynamic strategies, the base project ordering is re-determined each time the pricing step is invoked to find a suitable pattern

related to any base project.

Ordering personnel pricing problems according to the given priority ($Priority^{strat}$)
The parameter $Priority^{strat}$ defines the interplay between the visited state space and the base project ordering. We define the following priority-based strategies:

- **State space → Project ($Priority^{strat} = 0$)**: This strategy gives priority to the visited state space and solves first the pricing problems related to each base project for a given state space, before adapting the state space. The pseudocode for this strategy is displayed in Algorithm 2. Assuming the logical order is applied to visit the base projects, the pricing problems in Table 1 are visited in the order from left to right (given a particular SSR^{strat}).

Algorithm 2 Column generation algorithm for $Priority^{strat} = 0$

```
List all possible state spaces (SSR^strat)
do
    Solve restricted master problem
    Determine order for visiting state spaces
    for Each state space do
        Determine order of projects (Project^strat)
        for Each base project do
            Solve corresponding personnel pricing problem
            if Reduced cost < 0 then
                Break
        if Reduced cost < 0 then
            Break
while Reduced cost < 0
```

- **Project → State space ($Priority^{strat} = 1$)**: This strategy gives the priority to a certain base project and visits first the pricing problems related to all state spaces for a particular base project before considering another project as the base project. The pseudocode for this strategy is displayed in Algorithm 3. Assuming the logical order is applied to visit the base projects, the pricing problems in Table 1 are visited in the order from top to bottom (given a particular SSR^{strat}).

Algorithm 3 Column generation algorithm for $Priority^{strat} = 1$

```
List all possible state spaces (SSR^strat)
do
    Solve restricted master problem
    Determine order of projects (Project^strat)
    for Each base project do
        Determine order for visiting state spaces
        for Each state space do
            Solve corresponding personnel pricing problem
            if Reduced cost < then
                Break
        if Reduced cost < 0 then
            Break
while Reduced cost < 0
```

4.2.3 Other speed-up mechanisms

In the literature, different speed-ups are proposed to decrease the computational time to find the optimal LP solution in a column generation framework (Desaulniers et al., 2002). We implemented the following mechanisms, i.e.

- **Cut-off value**: Since we are only interested in personnel patterns with a negative reduced cost, a cut-off value can be imposed on the objective function (25) and constraint (28) is included in the pricing problem. Jourdan et al. (2009) classify this technique also as a state space reduction approach. This speed-up mechanism reduces the number of eligible columns in the restricted master problem and the associated solution time.

$$c_{jp'} - \sum_{p \in P} \sum_{t \in T} \pi_{pt} \; u_{pt} < 0 \tag{28}$$

- **Adding multiple personnel patterns**: Instead of only adding a single personnel pattern with the most negative reduced cost, we add all feasible personnel patterns that are encountered during the search for the optimal solution.
- **Partial pricing**: Since the pricing problems for different base projects use the same dual information, the procedure tends to produce days-off schedules covering the same workload and projects. In order to accelerate the procedure, only a small group of subproblems is solved at each iteration rather than all subproblems for all projects. In the next iteration, more accurate dual variables will be obtained to generate better patterns for the other pricing problems. As the iterations progress, it becomes increasingly difficult to find a column with negative reduced cost. Therefore, more and more subproblems are solved until, ultimately, all subproblems need to be solved. When partial pricing is applied, the RMP is re-solved from the moment a single pattern with negative reduced cost is found for any project. When partial pricing is not applied, the RMP is re-solved when a single pattern with negative reduced cost is found after solving the pricing problem for every base project.

4.3 Diving heuristics

The optimal LP solution obtained by the column generation algorithm may be fractional and branch-and-price has been commonly applied as a methodology to convert a fractional solution into the optimal integer solution (Barnhart et al., 1998). However, due to the high computational time of the column generation algorithm, a branch-and-price procedure will be very time consuming. We propose a diving heuristic to derive high-quality integer solutions. We refer to Joncour et al. (2010) and Sadykov et al. (2019) for an overview of different diving techniques. The diving heuristic applies a depth-first search without backtracking until an integer solution is found. The heuristic repetitively fixes one or multiple (fractional) assignments to a positive integer value and uses column generation in each node of the

branching tree to include promising personnel patterns. In this way, the LP is re-optimised taking the diving constraints into account.

In each node of the branching tree, two type of variables may be fractional, namely the activity start time variables v_{pit} and the personnel pattern variables $x_{jp'}$. Based on preliminary experiments, diving is only performed on the activity start time variables. We do not fix variables related to the personnel staffing plan since diving on these variables requires significant adaptations to the pricing problem (Vanderbeck, 2000). Moreover, the preliminary experiments showed that diving on these variables requires a significant amount of time and dives, and may lead to an infeasible solution. As a result, a restricted branch-and-bound based on the set of generated personnel patterns is additionally required to drive the solution to integrality. In the following, we discuss the variable fixing performed by the diving heuristic and the branch-and-bound to devise integer solutions.

Variable fixing

The variables stemming from the optimal LP solution may be integer or fractional. We first fix all activity start time variables with a value equal to one. Related to the fixing of fractional activity start time variables, we devise the following strategies:

- **High. frac.**: This strategy fixes only the start time variable with the fractional value closest to 1.
- **Cut-off**: All fractional start time variables with a value greater than θ are fixed to 1. Note that to preserve feasibility, this cut-off should be greater than 0.5. When no fractional value satisfies this cut-off value, the start time variable with the largest fractional value is selected for diving.
- **% frac.**: This strategy determines the number of activities with fractional activity start times and assigns a start time to a certain percentage (θ) of these activities by fixing its highest fractional variable to 1. To preserve feasibility, the start time windows are adapted after fixing a certain activity start time and therefore, no fractional value may exist in the resulting start time window of an activity. In this case, a random start time in this time window is selected. To the best of our knowledge, this technique has not been applied in the literature.

Restricted branch-and-bound

Due to the partial diving algorithm, an integer solution is calculated by applying a branch-and-bound procedure on the restricted master problem formulation (eqs. (1)-(6)). This branch-and-bound only considers a restricted set of personnel patterns, which have been generated via column generation. The procedure is known as a restricted branch-and-bound procedure (Van Den Eeckhout et al., 2019), a reduced column subset method (Eitzen et al., 2004) or a restricted master heuristic (Joncour et al., 2010). We consider the following two strategies to find an integer solution via the restricted branch-and-bound, i.e.

- RBB^f: This strategy assumes that all activity start time variables are fixed by the diving algorithm and only the personnel staffing plan has to be constructed for the different

projects.

- *RBB*: This strategy relaxes the assignments fixed by the diving heuristic. The branch-and-bound procedure constructs a schedule for the multiple projects and corresponding personnel staffing plans simultaneously. The restricted branch-and-bound considers a large set of personnel patterns generated via the diving heuristic and the column generation algorithm.

4.4 Improvement step

In order to further improve the upper bound resulting from the diving heuristic, we apply an improvement procedure that improves the personnel staffing plan per project given the realised resource transfers in the solution resulting from the diving heuristic. This solution indicates the amount and timing of resource transfers. In this improvement step, we fix these resource transfers and try to improve the project schedule and associated personnel staffing plan per project in subsequent order. The resource transfers show (i) the transfer of resource capacity from personnel assigned to a different base project towards the project under consideration, i.e. incoming resource capacity, and (ii) the transfer of resource capacity from the workforce associated with a particular (base) project to the other projects, i.e. outgoing resource capacity. We define this resource capacity using the following parameters:

- **Incoming resource capacity**: The incoming resource capacity $\Delta^{+}_{p't}$ on a particular day t for project p' is the additional capacity (i.e. the sum of working duties) of workers assigned to a different base project $p \neq p'$ that is transferred to project p'. This additional transfer capacity can be calculated as follows, where the x_{jp} values are based on the upper bound solution of the diving heuristic:

$$\Delta^{+}_{p't} = \sum_{p \in P \setminus \{p'\}} \sum_{j \in \bar{J}_p} a_{jpp't} x_{jp} \qquad \forall t \in T_{p'} \tag{29}$$

- **Outgoing resource capacity**: The outgoing resource capacity ($\Delta^{-}_{p't}$) for project p' is the capacity (i.e. sum of working duties) of workers assigned to base project p' that is transferred to other projects $p \neq p'$. This transferred capacity can be calculated as follows:

$$\Delta^{-}_{p't} = \sum_{p \in P \setminus \{p'\}} \sum_{j \in \bar{J}_{p'}} a_{jp'pt} x_{jp'} + \sum_{j \in \bar{J}_{p'}} b_{jp't} x_{jp'} \qquad \forall t \in T_{p'} \tag{30}$$

Taking the parameter values for Δ^{+}_{pt} and Δ^{-}_{pt} into account, the multi-project problem can be decomposed per project into multiple single-project staffing problems by adapting the staffing constraints linking the project scheduling problem and the staffing problem (cf eq. (2)) as follows:

$$\sum_{j \in \bar{J}_p} a_{jpt}\, x_{jp} + o_{pt} - \sum_{i \in N} \sum_{t^* = max(1, t - d_{pi} + 1)}^{t} r_{pi}\, v_{pit^*} + \Delta_{pt}^{+} - \Delta_{pt}^{-} \geq 0 \qquad \forall p \in P, \forall t \in T \qquad (31)$$

In order to optimise the personnel staffing plan for a single project given the resource transfers, the decomposed branch-and-price procedure of Van Den Eeckhout et al. (2020) is applied for each individual project separately, given the adapted resource linking constraints (eq. (31)). The result will be an optimal personnel plan for each project given the resource transfers and may improve the solution resulting from the diving heuristic.

5 Computational experiments

In this section, we provide computational insights to calibrate the proposed procedure and analyse the impact of resource sharing in a multi-project environment. In Section 5.1, we describe the test design and parameter settings used in the experimental analysis. Section 5.2 demonstrates the performance of the devised strategies associated with the column generation algorithm and the diving heuristic to design a well-performing algorithm. In Section 5.3, we provide managerial insights into the impact of resource sharing and the definition of transfer flexibility. Because of the randomness included in the proposed state space reduction approach, we have carried out 10 independent runs for all experiments in order to present coherent and unbiased results. All computational tests are performed on a Intel Core E5-2660v3 processor with 4 GB RAM and 2.6GHz. The algorithm is coded in C++ using XCode (version 8.0) and relies on Gurobi 7.0 to solve mathematical problem formulations with a time limit imposed of 3600 seconds, unless otherwise stated.

5.1 Dataset

The problem instances used in the computational experiments comprehend project scheduling, personnel staffing and resource sharing information.

Project scheduling information

In the literature, different datasets have been developed for the RCMPSP (e.g. Krüger and Scholl (2009); Browning and Yassine (2010); Homberger (2012); Wauters et al. (2016)). These datasets, however, are not relevant to validate the proposed procedure since we deal with only a single renewable resource and consider both the scheduling of activities and resources, which gives rise to a higher complexity. As a result, we have generated multi-project instances in line with the literature. In our test design, we utilise the single-project instances of Vanhoucke et al. (2001), which have been constructed using the network generator Progen/Max, to generate a set of 50 multi-project instances by combining different single-project instances in a random manner. Each multi-project instance comprehends 4 single-project instances

consisting of 10 activities. From these instances, the network topology and the activity durations are retained. Since the instances of Vanhoucke et al. (2001) consider multiple renewable resources, we generate the resource demand of each activity randomly using a discrete uniform distribution on the interval [1,10], which is in line with the resource demand generation of Vanhoucke et al. (2001). For each project in the multi-project instances, a deadline is set equal to the critical path of the related single-project instance. Note that in an integrated project and personnel scheduling context, the deadline of a project is, apart from the number of activities, a determinant for the instance size and computational complexity since resources are scheduled over this planning horizon. The average project deadline of the generated instances is 26.5 days, which is in line with the planning period of 28 days commonly applied in personnel scheduling problems.

Personnel staffing parameters and time-related scheduling constraints
The parameters related to the personnel staffing problem are taken from related project staffing problems in the literature (Maenhout and Vanhoucke, 2016; Van Den Eeckhout et al., 2020). More specifically, we set the length of a unit time period to 7 ($|T_l|$), the minimum and maximum number of assignments equal to 5 (w^{min} and w^{max}), the minimum and maximum number of consecutive working days to respectively 2 (n^{min}) and 6 (n^{max}) and the minimum and maximum number of consecutive days off to respectively 1 (f^{min}) and 2 (f^{max}). The objective function weights, i.e. the daily cost for hiring regular and temporary workers, are set to respectively 2 (c^{RG}) and 4 (c^{TW}).

Resource sharing parameters
The parameters related to the resource sharing constraints are set in such a way that the transfer flexibility is varied. To that purpose, we consider different resource sharing scenarios that are displayed in Table 2, by setting specific values related to the number of eligible projects (ω), the minimum consecutive days workers should be assigned to another project (α^{min}), the maximum consecutive days a worker is at least once assigned to his base project p' (α^{max}) and the transfer time (β). Scenario 1 represents the resource sharing scenario with maximal flexibility. The other scenarios are characterised by a lower flexibility. Note that in all these scenarios constraints (20) and (21) are not considered. In Section 5.2, these scenarios are utilised, leading to $5 \times 50 = 250$ instances, to validate the performance and design choices of the algorithm. In Section 5.3, a wider range of settings is considered related to the parameter values defining the resource transfer flexibility to study the managerial impact of resource sharing on the staffing budget.

5.2 Algorithmic performance

In this section, we give insight in the computational performance of the different components and devised optimisation strategies. Section 5.2.1 investigates the impact of the different speed-up mechanisms on the computational performance of the column generation algorithm.

Scenario	Constraint parameter			
	ω	α^{min}	α^{max}	β
1	4	1	$\lvert T_{p'}\rvert + 1$	1
2	2	1	$\lvert T_{p'}\rvert + 1$	1
3	4	5	$\lvert T_{p'}\rvert + 1$	1
4	4	1	7	1
5	4	1	$\lvert T_{p'}\rvert + 1$	2

Table 2: Test design - Parameter settings for different resource sharing scenarios

The impact of the different diving strategies and parameter settings is discussed in Section 5.2.2. In Section 5.2.3, we show the contribution of the diving heuristic and the improvement procedure related to the progress of the incumbent solution value. The results in this section represent the average values over the different resource sharing scenarios defined in Table 2 in order to come up with general findings. An analysis of the results showed similar (relative) findings for the different resource sharing scenarios, such that average results can be presented without loss of information.

5.2.1 Computational analysis of the column generation algorithm

We evaluate the convergence and the related computational performance of the column generation algorithm to determine the optimal LP solution in the root node of the search tree as follows. First, the impact of the different state space reduction strategies (SSR^{strat}) and the priority of visiting the different state spaces ($Priority^{strat}$) is investigated. Second, the impact of the order for visiting the different base projects ($Project^{strat}$) is analysed. Third, the overall impact of the different speed-ups applied in the procedure is highlighted. Note that for the analysis in this section, a time limit is imposed of 10 hours.

Impact of strategies SSR^{strat} and $Priority^{strat}$

In this analysis, we focus on the state space reduction approach for solving the pricing problem presented in Section 4.2.2 and explore the impact of different parameter values for SSR^{strat} and $Priority^{strat}$. The relevant parameters which are not considered in this analysis are set to their best values. Table 3 shows the computational performance for different settings of SSR^{strat} and $Priority^{strat}$ solely related to the column generation algorithm in the root node. We report the average total required CPU time (CPU^{CG}), the average number of personnel patterns generated ($|\bar{J}|$), the average number of solved restricted master problems ($\#RMP$), the average number of solved personnel pricing problems ($\#PP$) visited and the average CPU time for solving a single problem (CPU^{RMP}_{avg}, CPU^{PP}_{avg}). The best performing setting for a certain $Priority^{strat}$ is indicated in bold.

$Priority^{strat}$	SSR^{strat}	CPU^{CG}	$\lvert\bar{J}\rvert$	$\#RMP$	$\#PP$	CPU^{RMP}_{avg}	CPU^{PP}_{avg}
-	**0**	**495**	**644**	**65**	**108**	**0.03**	**3.71**
	1	425	887	155	474	0.04	0.74
	2	340	886	158	486	0.04	0.57
0	**3**	371	888	158	491	0.04	0.63
	4	**121**	**1067**	**199**	**710**	**0.04**	**0.14**
	5	145	1068	199	726	0.04	0.17
	1	363	651	90	192	0.03	1.46
	2	335	642	92	218	0.03	1.27
1	**3**	427	643	92	241	0.03	1.48
	4	**126**	**733**	**115**	**263**	**0.03**	**0.39**
	5	167	736	115	307	0.03	0.46

Table 3: Results related to the different strategies for SSR^{strat} and $Priority^{strat}$ on the column generation algorithm in the root node

Table 3 clearly shows that state space reduction to steer the convergence of the column generation algorithm is beneficial compared to the benchmark approach that constructs only a single pricing problem per base project ($SSR^{strat} = 0$ and $Priority^{strat} =$ -), which requires 495 seconds to find the optimal LP solution. The strategies visiting different state spaces significantly decrease the required CPU time. The best strategy ($SSR^{strat} = 4$ and $Priority^{strat} = 0$) requires only 121 seconds. The results reveal that this is mainly thanks to the decrease in complexity of most pricing problems solved, i.e. the required CPU time CPU^{PP}_{avg} decreases from 3.71 to 0.14 seconds, which compensates for the larger number of pricing problems ($\#PP$) and column generation iterations ($\#RMP$).

When comparing the different SSR strategies (SSR^{strat}), we observe that those strategies specifying the set of eligible projects P^e, i.e. $SSR^{strat} = 4$ and 5, perform best in terms of CPU time (CPU^{CG}). Using only the parameter ω to limit the number of projects an employee can be assigned to, as done in $SSR^{strat} = 1$, 2 and 3, leads to only a slightly lower CPU time compared to $SSR^{strat} = 0$. When P^e has been specified, the computational effort decreases significantly because of the dramatic decrease in the number of relevant variables and constraints in the pricing problem. This is especially observed in the required average computational time for solving a pricing problem (CPU^{PP}_{avg}), which is significantly lower for $SSR^{strat} = 4$ and 5. The number of column generation iterations ($\#RMP$) and visited pricing problems ($\#PP$) are, in contrast, significantly higher. Note that applying state space reduction increases the number of generated personnel patterns ($\lvert\bar{J}\rvert$), but this has only a very limited impact on the solution time of the restricted master problem (CPU^{RMP}_{avg}). These conclusions are similar for the different priority strategies ($Priority^{strat}$).

The different values of the parameter $Priority^{strat}$ have a smaller impact on the performance, which implies that the construction of different state spaces (guided by SSR^{strat}) is more important than the order of visiting these state spaces (guided by $Priority^{strat}$). However, a pairwise comparison between the different SSR strategies (SSR^{strat}) reveals that visiting the pricing problems per state space ($Priority^{strat} = 0$) performs better than visiting all state spaces per base project ($Priority^{strat} = 1$) for those strategies that define a larger

number of state spaces ($SSR^{strat} = 3$, 4 or 5). This can be explained by the smaller average CPU time required to solve the pricing problems (CPU^{PP}_{avg}) for $Priority^{strat} = 0$, despite the larger number of pricing problems ($\#PP$) and restricted master problems ($\#RMP$) solved. When $SSR^{strat} = 4$, the performance between both priorities is similar, i.e. 121 seconds ($Priority^{strat} = 0$) versus 126 seconds ($Priority^{strat} = 1$).

Impact of strategies $Project^{strat}$

The impact of the order in which the different (base) projects are visited is investigated only for the best performing strategies resulting from previous analysis, i.e. SSR^{strat} is set to 4 and $Priority^{strat}$ is set to 0 or 1. Table 4 shows the results for different strategies $Project^{strat}$ to determine the project ordering (cf Section 4.2.2), i.e. the random order ('Random'), a deterministic order (logical - 'Logical', based on the deadline - '$\delta_{p,n+1}$', based on the work content of project p - 'wc_p') or a dynamic ordering (based on the number of employees assigned to the project p in the last RMP solution ($|W|^{RMP}_p$), based on the CPU time or on the reduced cost of the last pricing problem solved for project p - 'CPU^{PP}_p' and 'RC_p' respectively). Note that these orderings can be ranked from low to high ('$L \rightarrow H$') or from high to low ('$H \rightarrow L$'). We report the average total required CPU time (CPU^{CG}), the average number of solved restricted master problems ($\#RMP$) and the average number of solved personnel pricing problems ($\#PP$) visited. Note that for some settings the column generation algorithm did not converge to the optimal LP solution within the time limit of 10 hours. These instances are not taken into account when displaying the results, which is indicated by an asterisk (*).

		Priority^{strat} = 0 - SSR^{strat} = 4			**Priority^{strat} = 1 - SSR^{strat} = 4**		
Category	$Project^{strat}$	CPU^{CG}	$\#RMP$	$\#PP$	CPU^{CG}	$\#RMP$	$\#PP$
Random	Random	123	199	674	276	121	381
Determ.	Logical $L \rightarrow H$	162	236	831	2080*	235*	1222*
	$\delta_{p,n+1}$ $L \rightarrow H$	125	200	849	1788*	173*	1320*
	$\delta_{p,n+1}$ $H \rightarrow L$	190	282	842	3613*	377*	1617*
	wc_p $L \rightarrow H$	154	234	866	2028*	253*	1418*
	wc_p $H \rightarrow L$	156	243	832	2296*	268*	1318*
Dynamic	$\|W\|^{RMP}_p$ $L \rightarrow H$	174	249	808	2289*	306*	1453*
	$\|W\|^{RMP}_p$ $H \rightarrow L$	130	215	842	1464*	184*	1018*
	CPU^{PP}_p $L \rightarrow H$	**121**	**199**	**710**	**126**	**115**	**263**
	RC_p $L \rightarrow H$	131	215	590	279	144	336

Table 4: Results related to the different strategies for $Project^{order}$ on the column generation algorithm in the root node

Table 4 reveals that the CPU time (CPU^{CG}) is fairly similar for different project ordering strategies when pricing problems are visited per state space ($Priority^{strat} = 0$). This is in contrast to the performance when pricing problems are visited per base project ($Priority^{strat} = 1$), which is very much dependent on the project ordering, as a result from the definition of this latter priority strategy. In addition, for every project ordering, the former strategy ($Priority^{strat} = 0$) outperforms the latter ($Priority^{strat} = 1$). This stems from the fact that when $Priority^{strat} = 1$ also larger state spaces for a certain project need to be explored, before the pricing problem of another base project can be explored, which

may find a pattern with negative reduced cost in a smaller state space.
When analysing the results for '$Priority^{strat} = 0 - SSR^{strat} = 4$', we see that the dynamic measure '$CPU_p\ L \rightarrow H$' yields the best performance ($CPU^{CG} = 121$ seconds). Since the CPU time is a proxy for the problem complexity, we prefer to solve pricing problem with a smaller complexity first, in line with the state space reduction. The strategies 'Random' and '$\delta_{p,n+1}\ L \rightarrow H$' also perform very well. For '$Priority^{strat} = 1 - SSR^{strat} = 4$', we also observe that the strategy '$CPU_p\ L \rightarrow H$' performs best. The performance of most other metrics is unacceptable.
In the remainder of the computational experiments, we employ the best settings, i.e. $SSR^{strat} = 4$, $Priority^{strat} = 0$ and $Project^{strat} =$ '$CPU_p\ L \rightarrow H$', in the column generation algorithm.

Impact of all speed-up mechanisms
The contribution of all speed-up mechanisms (cf Section 4.2) is displayed in Table 5. The impact of the speed-up techniques is determined via the comparison of different variants of the proposed column generation algorithm. In this regard, we consider a variant without any speed-up mechanism ('No speed-up'), with all speed-up mechanisms included ('All speed-ups') and a number of variants that excludes one specific speed-up technique leaving the rest of the algorithm unchanged, i.e. without creating an upper bound solution to obtain a good initial set of personnel patterns ('w/o initialisation step'), without the state space reduction approach ('w/o SSR'), without partial pricing ('w/o partial pricing'), without adding multiple patterns per pricing problem ('w/o multiple patterns.') and without imposing the cut-off on the objective function of the pricing problem ('w/o cut-off').

Variant	CPU^{CG}	$\lvert\bar{J}\rvert$	$\#RMP$	$\#PP$	RMP^{cpu}_{avg}	PP^{cpu}_{avg}
No speed-ups	1143	212	59	235	0.02	4.14
All speed-ups	142	594	206	731	0.03	0.15
w/o initialisation	136	533	278	879	0.03	0.12
w/o SSR	543	394	66	109	0.03	4.01
w/o partial pricing	473	477	45	485	0.03	0.68
w/o multiple patterns	186	485	315	1078	0.03	0.14
w/o cut-off	121	1067	199	710	0.04	0.14

Table 5: Impact of speed-up mechanisms on the performance of the column generation algorithm

Table 5 reveals that incorporating the proposed speed-up mechanisms decreases the required computational effort (CPU^{CG}) from 1143 seconds ('No speed-ups'), which is used as a benchmark to evaluate the performance of the algorithm, to 142 seconds ('All speed-ups'). Dropping the partial pricing or the state space reduction mechanism has the largest impact on the CPU time, leading to an increase in CPU time to 473 seconds, respectively 543 seconds. Only including a single personnel pattern per pricing problem ('w/o multiple patterns') leads to an increase to 186 seconds.
When no cut-off is imposed on the objective function value of the pricing problem ('w/o cut-off'), CPU^{CG} decreases from 142 to 121 seconds. This is a consequence of the interplay with the speed-up 'Adding multiple patterns'. When a cut-off is imposed on the objective

function, a smaller number of feasible solutions will be encountered during the search and thus less patterns are added to $\bar{J}_p$, which is clear from the number of personnel patterns $|\bar{J}|$ (594 versus 1067 personnel patterns). This implies that also including patterns with a positive reduced cost helps to reduce the CPU time. Starting the column generation algorithm with an empty set of personnel patterns ('w/o initialisation') leads to a slightly lower CPU time for the column generation algorithm, decreasing the CPU time from 142 to 136 seconds. This means that an initial set of dedicated personnel patterns, constructed without taking (dual) information from other projects into account, does not lead to a faster convergence. However, note that this initialisation step ensures a reasonable upper bound solution with dedicated resources, which is be used as a benchmark to value the sharing of resources and is exploited as initial incumbent to generate an integer solution via the restricted branch-and-bound implemented in the diving heuristic.

5.2.2 Impact diving heuristic

In order to obtain insight in the performance of the diving heuristic (cf Section 4.3), we vary the strategies and parameter settings related to the variable fixing ('High. frac.', 'Cut-off' and '% frac.') and the restricted branch-and-bound (RBB^f and RBB). Table 6 displays the results of the different diving strategies for different threshold values θ. The first line provides the benchmark setting where no diving is performed and the RBB is directly solved based on the columns that have been generated in the root node of the search tree via column generation. We report the average optimality gap ($\%Gap = (UB - LB)/LB$) comparing the incumbent solution and the optimal LP solution, the average depth of the search tree (#Dives) and the average computation time of the diving algorithm (CPU^{DH}), the restricted branch-and-bound (CPU^{RBB}) and the entire proposed column generation-based diving heuristic procedure (CPU^{CGDH}).

Variable fixing strategy		**Restricted branch-and-bound strategy**									
		RBB^f					**RBB**				
Type	θ	%Gap	#Dives	CPU^{DH}	CPU^{RBB}	CPU^{CGDH}	%Gap	#Dives	CPU^{DH}	CPU^{RBB}	CPU^{CGDH}
-	**-**	-	-	-	-	-	2.46%	0	0	1656	1814
High. frac.	**-**	3.22%	11.4	688	478	1311	2.66%	11.4	688	1542	2364
Cut-off	**0.9**	3.24%	11.0	681	505	1330	2.63%	11.0	681	1573	2394
	0.8	3.24%	9.8	651	489	1285	2.61%	9.8	651	1599	2386
	0.7	3.15%	8.3	590	528	1270	2.50%	8.3	590	1652	2381
	0.6	3.10%	6.2	512	580	1235	2.46%	6.2	512	1714	2365
	0.5	3.08%	3.9	383	590	1130	2.28%	3.9	383	1884	2404
% frac.	**1**	3.61%	1.0	132	443	735	2.13%	1.0	132	2082	2354
	0.75	3.15%	2.1	244	596	996	**2.12%**	**2.1**	**244**	**2016**	**2401**
	0.5	2.98%	4.0	363	618	1133	2.18%	4.0	363	1940	2445
	0.25	3.05%	6.7	503	558	1217	2.36%	6.7	503	1767	2414
	0.1	3.24%	10.0	645	500	1286	2.60%	10.0	645	1578	2363

Table 6: Computational results for different strategies related to the diving heuristic

Table 6 reveals that the components associated with the diving heuristic, i.e. the search tree and especially the restricted branch-and-bound, to find a high-quality solution for the problem under study require significant computational effort (CPU^{DH} and CPU^{RBB}). The

benchmark procedure that applies a restricted branch-and-bound on the patterns found in the root node without diving, i.e. with variable fixing strategy '-', requires on average 1814 seconds finding a solution with an optimality gap of 2.46%.

When diving is applied, the different diving strategies and associated threshold value θ impact the solution quality (%Gap) and the depth of the search tree (#Dives). When only the highest fractional variable is fixed (High. frac.), on average 11.4 dives are required to obtain an integer multi-project schedule. This depth gradually decreases for the strategy 'Cut-off' when a lower threshold is imposed, from 11.0 to 3.9. For the strategy '% frac.', an opposite behaviour is observed, in line with the definition of this strategy, i.e the depth of the tree increases from 1.0 ($\theta = 1$) to 10.0 ($\theta = 0.1$). Note that there is no clear relationship between the depth of the tree and the solution quality, as it may have been expected that a larger depth in the search tree may lead to a larger set of patterns generated and consequently result in a higher solution quality of the staffing plan of the incumbent solution. For the strategy 'Cut-off', the best solution quality is obtained when θ is low ($\theta = 0.5$). For the strategy '% frac.', the best solution quality is obtained when θ is average to high ($\theta = 0.5$, 0.75 and 1). When comparing the different diving strategies, Table 6 shows that the strategy 'High frac.', which is commonly applied in the literature, is outperformed by the other two strategies. The strategy '% frac.' delivers the best results.

When analysing the performance of the two branch-and-bound strategies, we notice that the solution quality (%Gap) when fixing the activity start times according to the diving heuristic (RBB^f) is worse compared to the strategy that relaxes the start time assignments resulting from the diving heuristic (RBB). The required solution times (CPU^{CGDH}), however, are significantly lower. The best setting with RBB^f (i.e., % frac, $\theta = 0.5$) results in a optimality gap equal to 2.98% (%Gap) and a CPU time of 1133 seconds (CPU^{CGDH}). When RBB is applied, different variants of the algorithm with diving are outperformed by the algorithm without diving. Since additional personnel patterns are generated during the diving iterations, the restricted branch-and-bound with a larger number of patterns should lead to better results. However, due to the imposed time limit of 3600 seconds, we notice that this is not always the case. The time required for diving (CPU^{Diving}) leads to less time remaining for the restricted branch-and-bound (CPU^{RBB}) to search for high-quality solutions. Only when the depth of the search tree is limited (#Dives $\leq$ 6.7), a better solution quality is obtained with diving. The best performing strategy with RBB (i.e., % frac, $\theta = 0.75$) leads to an optimality gap of 2.12% (%Gap) and a CPU time of 2401 seconds (CPU^{CGDH}). This setting is used in further experiments.

In order to better estimate the quality of the incumbent solutions based on the optimality gap, we calculated the optimal LP solution value when no resource sharing is allowed ($\omega = 1$), which equals to 3447.8. The initial upper bound solution value, which provides the optimal integer resource budget without resource sharing, leads to a solution value equal to 3524.2. The resulting optimality gap equals 2.26% and, consequently, we can conclude that the proposed procedure with application of the diving heuristic is able to find high-quality solutions.

5.2.3 Contribution of the multistage solution methodology

Table 7 shows the average value of the incumbent solution found after each stage x (UB^x) and the average required computational time for each stage x (CPU^x). The improvement of the incumbent solution compared to the previous stage is displayed (UB^x_{impr}).

x	CPU^x	UB^x	UB^x_{impr}
Stage 1	121	3524.2	-
Initialisation CG	5	3524.2	-
Column generation	116	3524.2	-
Stage 2: Diving heuristic	2262	3364.2	-4.55%
Stage 3: Improvement step	18	3359.6	-0.13%
CGDH	2401	3359.6	-

Table 7: Computational results for the different stages

The results show that the diving heuristic, which constructs an integer staffing plan for different projects simultaneously with resource transfers, is the most expensive step in terms of CPU time ($CPU^x = 2262$ seconds). When the problem is decomposed and a staffing plan is derived for each project independently ('Initialisation step CG' and 'Improvement step'), the CPU time amounts to 5, respectively 18 seconds. This shows that resource sharing has a tremendous impact on the computational complexity. Related to the solution quality, we observe that the initial upper bound solution is improved with 4.55% by the diving heuristic, which is an indication of the added value of resource sharing. The improvement step further improves the incumbent solution value with 0.13%, showing that only minor improvements are made when calculating the optimal solution for individual projects with fixed timing for the resource transfers.

5.3 Managerial insights into the impact of resource transfer flexibility

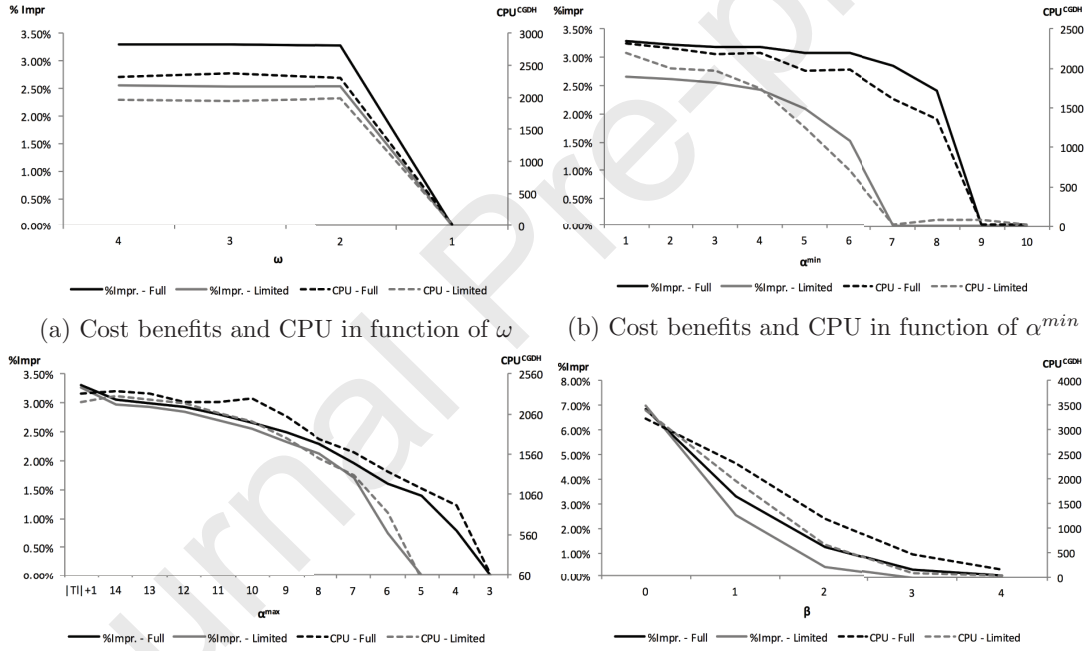
In this section, we evaluate the impact of transfer flexibility on the sharing of resources between projects and the resulting personnel staffing budget. In the problem under study, the resource transfer flexibility is defined via different parameters and constraints, i.e. (i) the maximum number of projects an employee can be assigned to (ω); (ii) the minimum number of consecutive days an employee should be stationed at a project (α^{min}); (iii) the period during which an employee should be assigned once to his base project (α^{max}); (iv) the duration of the resource transfer (β). The required assignment of an employee on the first and last day of the planning period to his base project (eqs. (20)-(21)) are included in these experiments. The impact of the parameters is analysed for three different scenarios, for which different base parameter values are considered, i.e.

- *Full flexibility ('Full')*: This scenario assumes maximum transfer flexibility for sharing of resources between different projects. The base parameter values are set as follows: $\omega = 4$,

$\alpha^{min} = 1$, $\alpha^{max} = |T_p| + 1$ and $\beta = 1$.

- *Limited flexibility ('Limited')*: This scenario assumes resources may be shared between different projects to a limited extent. The base parameter values are set as follows: $\omega = 2$, $\alpha^{min} = 3$, $\alpha^{max} = 10$ and $\beta = 1$.
- *No flexibility*: This scenario assumes that all projects are carried out by dedicated resources and that there is no sharing of resources between projects ($\omega = 1$). The other parameters and constraints are not relevant. This scenario serves as a benchmark to calculate the benefits of resource sharing given the defined resource transfer flexibility.

Figure 2 reveals the impact of varying the parameters related to resource transfer flexibility in an independent manner. The figures display the percentage improvement in staffing budget comparing the scenarios with resource sharing ('Full' or 'Limited') versus the scenario without resource sharing (No flexibility) ('%Impr') and the CPU time required by the solution procedure (CPU^{CGDH}). Note that the parameter values are arranged from more flexible towards more restrictive settings.



(a) Cost benefits and CPU in function of ω

(b) Cost benefits and CPU in function of α^{min}

(c) Cost benefits and CPU in function of α^{max}

(d) Cost benefits and CPU in function of β

Figure 2: Impact of constraint parameter values on the personnel budget and CPU time

Figure 2 shows a clear relation between the defined transfer flexibility, the percentage improvement ('%Impr') and the CPU time (CPU^{CGDH}). In general, allowing less resource flexibility leads to a lower added value of sharing resources and a lower required CPU time. Figure 2(a) confirms that when employees are allowed to work on one additional project on top of the base project ($\omega = 2$), the largest benefits in terms of cost improvement are obtained. Further increasing ω improves only marginally the staffing budget or even deteriorates the budget due to the larger complexity. Note that a larger value for ω does not

significantly impact the number and the average duration of the transfers realised in the incumbent solutions.

Figure 2(b) shows that increasing α^{min} leads initially to only a small decrease in the cost improvements, i.e. '%Impr' is larger than 3% as long as $\alpha^{min} \leq 6$ for the scenario 'Full flexibility'. When the scenario 'Limited flexibility' is considered, a dramatic decrease in the benefits of resource sharing ('%Impr') is observed when $\alpha^{min} > 4$. Note that when α^{min} increases, the number of resource transfers decrease, whereas the duration of the transfers is stable. Figure 2(c) shows that the cost benefits are gradually decreasing when α^{max} decreases. When $\alpha^{max} \leq 8$, the slope becomes steeper and the cost benefits rapidly reduce. Note that when α^{max} decreases, the duration of the resource transfers obviously decrease whereas the number of resource transfers is rather stable.

Figure 2(d) shows that the resource transfer time β is a main determinant for the cost benefits related to resource sharing. When the resource transfer time β is equal to 0, the cost improvements are significantly higher (approximately 6.8%). These benefits reduce rapidly when β increases. The cost benefits range between 2.5% and 3.5% when $\beta = 1$ and are smaller than 0.5% when $\beta \geq 3$. The large decrease of the cost improvements when β is increased from 0 to 1, is not explained by the number or the duration of the resource transfers, which are more or less stable for $\beta = 0$ and 1. In this way, the unproductive time associated with the resource transfers is the main responsible for the decrease in the cost improvements. Only when β is increased to 2, there is a large decrease in the number and the duration of the transfers. The latter resulting from the parameter value set for α^{max}.

In general, the results show that a similar trend is observed for both scenarios 'Full flexibility' and 'Limited flexibility', when varying a parameter to define resource transfer flexibility. Substantial cost improvements related to the project staffing budget are obtained as a result of resource sharing, although the obtained cost improvements are obviously higher for the scenario 'Full flexibility'. The largest impact stems from the resource transfer time parameter β which is rarely taken into account in previous research.

6 Conclusion

In this paper, we studied the multi-project staffing problem with calendar constraints and resource sharing. In order to accurately assess the staffing budget of the different projects, we simultaneously construct the project schedule of the different projects and a baseline personnel schedule giving insight in the minimum number of regular and temporary workers. The timing of the workload stemming from the project activities is an endogenous variable in the model. The problem definition includes resource flexibility via resource sharing subject to resource transfer constraints to decrease the staffing budget.

We proposed a multi-stage solution methodology to tackle the problem under study labelled as a Column Generation-based Diving Heuristic. First, we calculate the optimal LP solution methodology via column generation. Due to the high complexity of the pricing problem stem-

ming from the embedded time-related and resource sharing constraints, different speed-up mechanisms relying on a state space reduction are presented to increase the efficiency of the column generation algorithm. Second, a diving heuristic is applied to convert the fractional LP solution into an integer solution. A depth-first search is conducted by repetitively fixing a subset of the fractional activity start time variables. An integer staffing plan is obtained by conducting a restricted branch-and-bound using the patterns generated via column generation in the different nodes of the search tree. Third, an improvement step improves the incumbent solution by fixing the resource transfer capacity and determining the corresponding optimal personnel budgets for each project separately. In the computational experiments, we demonstrated that the proposed state reduction approach significantly outperforms the standard approach defining a singular pricing problem per base project. Furthermore, we showed that fixing a percentage of the fractional variables leads to high-quality solutions, outperforming existing approaches to obtain integer solutions based on the optimal LP solution.

Finally, we provided managerial insights concerning the impact of imposing restrictions on the resource transfer flexibility. The unproductive transfer times stemming from resource transfers have the largest impact on the staffing budget, followed by the constraint that defines the maximum time period during which an employee should be assigned at least once to his base project. Last, most cost improvements of resource sharing are obtained when allowing workers to work on one additional project, which is in accordance with the literature.

Acknowledgements The computational resources (Stevin Supercomputer Infrastructure) and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by Ghent University, FWO and the Flemish Government - department EWI.

References

Al-Yakoob, S. M. and Sherali, H. D. (2007). Mixed-integer programming models for an employee scheduling problem with multiple shifts and work locations. *Annals of Operations Research*, 155(1):119–142.

Al-Yakoob, S. M. and Sherali, H. D. (2008). A column generation approach for an employee scheduling problem with multiple shifts and work locations. *Journal of the Operational Research Society*, 59(1):34–43.

Alfares, H. K. and Bailey, J. E. (1997). Integrated project task and manpower scheduling. *IIE transactions*, 29(9):711–717.

Alfares, H. K., Bailey, J. E., and Lin, W. Y. (1999). Integrated project operations and personnel scheduling with multiple labour classes. *Production Planning & Control*, 10(6):570–578.

Allahverdi, A. (2015). The third comprehensive survey on scheduling problems with setup times/costs. *European Journal of Operational Research*, 246(2):345–378.

Attia, D., Bürgy, R., Desaulniers, G., and Soumis, F. (2019). A decomposition-based heuristic for large employee scheduling problems with inter-department transfers. *EURO Journal on Computational Optimization*, pages 1–33.

Bard, J. F. and Wan, L. (2006). The task assignment problem for unrestricted movement between workstation groups. *Journal of Scheduling*, 9(4):315–341.

Bard, J. F. and Wan, L. (2008). Workforce design with movement restrictions between workstation groups. *Manufacturing & Service Operations Management*, 10(1):24–42.

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W., and Vance, P. H. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3):316–329.

Bassett, M. (2000). Assigning projects to optimize the utilization of employees' time and expertise. *Computers & Chemical Engineering*, 24(2-7):1013–1021.

Batta, R., Berman, O., and Wang, Q. (2007). Balancing staffing and switching costs in a service center with flexible servers. *European Journal of Operational Research*, 177(2):924–938.

Bechtold, S. E. (1988). Implicit optimal and heuristic labor staffing in a multiobjective, multilocation environment. *Decision Sciences*, 19(2):353–372.

Belien, J. and Demeulemeester, E. (2006). Scheduling trainees at a hospital department using a branch-and-price approach. *European Journal of Operational Research*, 175(1):258–278.

Besikci, U., Bilge, U., and Ulusoy, G. (2015). Multi-mode resource constrained multi-project scheduling and resource portfolio problem. *European Journal of Operational Research*, 240:22–31.

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

Browning, T. R. and Yassine, A. A. (2010). A random generator of resource-constrained multi-project network problems. *Journal of Scheduling*, 13(2):143–161.

Brucker, P., Qu, R., and Burke, E.K. (2011). Personnel scheduling: Models and complexity. *European Journal of Operational Research*, 210:467–473.

Brunner, J.O. Bard, J. and Kohler, J. (2013). Bounded flexibility in days-on and days-off scheduling. *Naval Research Logistics*, 60(8):678–701.

Brusco, M. J. (2008). An exact algorithm for a workforce allocation problem with application to an analysis of cross-training policies. *IIE Transactions*, 40(5):495–508.

Campbell, G. M. (1999). Cross-utilization of workers whose capabilities differ. *Management Science*, 45(5):722–732.

Certa, A., Enea, M., Galante, G., and Manuela La Fata, C. (2009). Multi-objective human resources allocation in r&d projects planning. *International Journal of Production Research*, 47(13):3503–3523.

Dahmen, S., Rekik, M., Soumis, F., and Desaulniers, G. (2019). A two-stage solution approach for personalized multi-department multi-day shift scheduling. *European Journal of Operational Research*.

De Bruecker, P., Van den Bergh, J., Beliën, J., and Demeulemeester, E. (2015). Workforce planning incorporating skills: State of the art. *European Journal of Operational Research*, 243(1):1–16.

Desaulniers, G., Desrosiers, J., and Solomon, M. M. (2002). Accelerating strategies in column generation methods for vehicle routing and crew scheduling problems. In *Essays and Surveys in Metaheuristics*, pages 309–324. Springer.

Dodin, B. and Elimam, A. (1997). Audit scheduling with overlapping activities and sequence-dependent setup costs. *European Journal of Operational Research*, 97(1):22–33.

Dumitrescu, I. and Boland, N. (2003). Improved preprocessing, labeling and scaling algorithms for the weight-constrained shortest path problem. *Networks*, 42:135–153.

Easton, F. F., Rossin, D. F., and Borders, W. S. (1992). Analysis of alternative scheduling policies for hospital nurses. *Production and Operations Management*, 1(2):159–174.

Eitzen, G., Panton, D., and Mills, G. (2004). Multi-skilled workforce optimisation. *Annals of Operations Research*, 127(1-4):359–372.

Ernst, A. T., Jiang, H., Krishnamoorthy, M., and Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, 153(1):3–27.

Felberbauer, T., Gutjahr, W. J., and Doerner, K. F. (2019). Stochastic project management: multiple projects with multi-skilled human resources. *Journal of Scheduling*, 22(3):271–288.

Fowler, J. W., Wirojanagud, P., and Gel, E. S. (2008). Heuristics for workforce planning with worker differences. *European Journal of Operational Research*, 190(3):724–740.

Franz, L. S., Baker, H. M., Leong, G. K., and Rakes, T. R. (1989). A mathematical model for scheduling and staffing multiclinic health regions. *European Journal of Operational Research*, 41(3):277–289.

Garey, M. and Johnson, D. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman.

Gendron, B. (2005). Scheduling employees in quebecs liquor stores with integer programming. *Interfaces*, 35(5):402–410.

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):1–14.

Heimerl, C. and Kolisch, R. (2010). Scheduling and staffing multiple projects with a multi-skilled workforce. *OR Spectrum*, 32(2):343–368.

Homberger, J. (2012). A (μ, λ)-coordination mechanism for agent-based multi-project scheduling. *OR Spectrum*, 34(1):107–132.

Joncour, C., Michel, S., Sadykov, R., Sverdlov, D., and Vanderbeck, F. (2010). Column generation based primal heuristics. *Electronic Notes in Discrete Mathematics*, 36:695–702.

Jourdan, L., Basseur, M., and Talbi, E.-G. (2009). Hybridizing exact methods and metaheuristics: A taxonomy. *European Journal of Operational Research*, 199(3):620–629.

Kadri, R. L. and Boctor, F. F. (2018). An efficient genetic algorithm to solve the resource-constrained project scheduling problem with transfer times: The single mode case. *European Journal of Operational Research*, 265(2):454–462.

Kher, H. and Malhotra, M. (1994). Acquiring and operationalizing worker flexibility in dual resource constrained job shops with worker transfer delays and learning losses. *Omega*, 22(5):521–533.

Krüger, D. and Scholl, A. (2009). A heuristic solution framework for the resource constrained (multi-) project scheduling problem with sequence-dependent transfer times. *European Journal of Operational Research*, 197(2):492–508.

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

Kuo, Y.-H., Leung, J. M., and Yano, C. A. (2014). Scheduling of multi-skilled staff across multiple locations. *Production and Operations Management*, 23(4):626–644.

Larson, E. and Gray, C. (2011). *Project management : the managerial process*. New York: McGraw-Hill Irwin, 5th edition.

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

Mabert, V. A. and Raedels, A. R. (1977). The detail scheduling of a part-time work force: A case study of teller staffing. *Decision Sciences*, 8(1):109–120.

Maenhout, B. and Vanhoucke, M. (2013). An integrated nurse staffing and scheduling analysis for longer-term nursing staff allocation problems. *Omega*, 41(2):485–499.

Maenhout, B. and Vanhoucke, M. (2016). An exact algorithm for an integrated project staffing problem with a homogeneous workforce. *Journal of Scheduling*, 19(2):107–133.

Maenhout, B. and Vanhoucke, M. (2017). A resource type analysis of the integrated project scheduling and personnel staffing problem. *Annals of Operations Research*, 252(2):407–433.

Mika, M., Waligóra, G., and Weglarz, J. (2006). Modelling setup times in project scheduling. In *Perspectives in Modern Project Scheduling*, pages 131–163. Springer.

Nearchou, A., Giannikos, I., and Lagodimos, A. (2018). Multisite and multishift personnel planning with set-up costs. *IMA Journal of Management Mathematics*.

Pinha, D., Ahluwalia, R., and Senna, P. (2016). The combinatorial multi-mode resource constrained multi-project scheduling. *International Journal of Supply and Operations Management*, 3(3):1391–1412.

Sadykov, R., Vanderbeck, F., Pessoa, A., Tahiri, I., and Uchoa, E. (2019). Primal heuristics for branch and price: The assets of diving methods. *INFORMS Journal on Computing*.

Taskiran, G. K. and Zhang, X. (2017). Mathematical models and solution approach for cross-training staff scheduling at call centers. *Computers & Operations Research*, 87:258–269.

Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., and De Boeck, L. (2013). Personnel scheduling: A literature review. *European Journal of Operational Research*, 226(3):367–385.

Van Den Eeckhout, M., Maenhout, B., and Vanhoucke, M. (2019). A heuristic procedure to solve the project staffing problem with discrete time/resource trade-offs and personnel scheduling constraints. *Computers & Operations Research*, 101:144–161.

Van Den Eeckhout, M., Vanhoucke, M., and Maenhout, B. (2020). A decomposed branch-and-price procedure for integrating demand planning in personnel staffing problems. *European Journal of Operational Research*, 280(3):845–859.

Vanderbeck, F. (2000). On dantzig-wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm. *Operations Research*, 48(1):111–128.

Vanhoucke, M., Demeulemeester, E., and Herroelen, W. (2001). On maximizing the net present value of a project under renewable resource constraints. *Management Science*, 47(8):1113–1121.

Walter, M. and Zimmermann, J. (2016). Minimizing average project team size given multi-skilled workers with heterogeneous skill levels. *Computers & Operations Research*, 70:163–179.

Wauters, T., Kinable, J., Smet, P., Vancroonenburg, W., Berghe, G. V., and Verstichel, J. (2016). The multi-mode resource-constrained multi-project scheduling problem. *Journal of Scheduling*, 19(3):271–283.

Wild, B. and Schneewei, C. (1993). Manpower capacity planning a hierarchical approach. *International Journal of Production Economics*, 30:95–106.

Wirojanagud, P., Gel, E. S., Fowler, J. W., and Cardy, R. (2007). Modelling inherent worker differences for workforce planning. *International Journal of Production Research*, 45(3):525–553.

Worthington, D. and Guy, M. (1988). Allocating nursing staff to hospital ward a case study. *European Journal of Operational Research*, 33(2):174–182.

Wright, P. D. and Mahar, S. (2013). Centralized nurse scheduling to simultaneously improve schedule cost and nurse satisfaction. *Omega*, 41(6):1042–1052.

A Impact of the state space reduction on the computational complexity

The computational time complexity of the column generation algorithm and the associated strategies to conduct a state space reduction are primarily assessed in a empirical manner via computational experimentation (cf Section 5.2.1). However, in order to give the reader an indication on the reduced complexity of the pricing step, we report some characteristics of a singular pricing problem (i.e. the number of decision variables, constraints and feasible patterns) and the number of pricing problems that needs to be solved to proof LP optimality (i.e. $|S| \times |P|$) for different parameter settings defined by the state space reduction strategies. The state space reduction is primarily guided by the parameter SSR^{strat} via the definition of the following two parametrised settings, i.e. (i) the maximum number of projects (ω) and the set of eligible projects (P^e). The ordering strategies $Project^{strat}$ and $Priority^{strat}$ do not impact the characteristics of a singular pricing problem nor the theoretical number of pricing problems to proof LP optimality but only the order in which the different subproblems are visited, impacting the number of column generation iterations for finding the optimal LP solution.

Table 8 displays the theoretical number of times a pricing problem needs to be solved to proof the LP optimality for each parameter setting of SSR^{strat} when the number of projects $|P|$ is equal to 4. When $SSR^{strat} = 0$, the state space is not reduced and four pricing problems are required to proof LP optimality. When $SSR^{strat} = 5$, the state space is reduced based on different settings for ω and P^e and in total 68 pricing problems are required to proof LP optimality.

	$\mathbf{SSR^{strat}}$					
	0	**1**	**2**	**3**	**4**	**5**
Maximum # projects (ω)	4	1 or 4	1, 2 or 4	1, 2, 3 or 4	1, 2 or 4	1, 2, 3 or 4
# eligible projects ($\lvert P^e\rvert$)	$\lvert P\rvert$	$\lvert P\rvert$	$\lvert P\rvert$	$\lvert P\rvert$	ω	ω
# state spaces ($\lvert S\rvert$) per base project						
$\omega = 1$	0	1	1	1	1	1
$\omega = 2$	0	0	1	1	3	3
$\omega = 3$	0	0	0	1	0	12
$\omega = 4$	1	1	1	1	1	1
Total	1	2	3	4	5	17
# pricing problems ($=\lvert S\rvert \times \lvert P\rvert$)	4	8	12	16	20	68

Table 8: The number of pricing problems required to proof LP optimality ($|P| = 4$)

Table 9 provides an overview how the different parameter settings for SSR^{strat} impact the number of decision variables and the number of feasible patterns. This table gives insight in the impact of different constraints and groups of constraints (time-related constraints, resource sharing constraints) and primarily in the parametrised settings ω and P^e. The table displays for each setting both the number of feasible patterns and the number of constraints between brackets. These characteristics are devised for problem instances with $|P| = 4$ and $|T_{p'}| = 21$.

	# eligible projects			
	$\lvert P^e\rvert = \mathbf{1}^{(a)}$	$\lvert P^e\rvert = \mathbf{2}$	$\lvert P^e\rvert = \mathbf{3}$	$\lvert P^e\rvert = \mathbf{4}$
# eligible decision variables	42	128	171	214
# feasible patterns				
- No constraints	$2^{(2\times21)}$ (0)	$2^{(5\times21)}$ (0)	$2^{(7\times21)}$ (0)	$2^{(9\times21)}$ (0)
- 1 assignment/day constr (7)	2^{21} (21)	5^{21} (21)	7^{21} (21)	2^{21} (21)
- Time-related constr (7)-(13)	1675 (100)	$\approx 2.5 \times 10^{12}$ (100)	$\approx 2.6 \times 10^{15}$ (100)	$\approx 3.9 \times 10^{17}$ (100)
+ Resource sharing constr (14)-(23)				
$\omega = 1^{(a)}$	1675 (100)	1675 (100)	1675 (100)	1675 (100)
$\omega = 2$	1675 (100)	$\approx 12 \times 10^6$ (266)	$\approx 13 \times 10^6$ (327)	$\approx 19 \times 10^6$ (388)
$\omega = 3$	1675 (100)	$\approx 12 \times 10^6$ (266)	$\approx 50 \times 10^6$ (327)	$\approx 66 \times 10^6$ (388)
$\omega = 4$	1675 (100)	$\approx 12 \times 10^6$ (266)	$\approx 50 \times 10^6$ (327)	$\approx 151 \times 10^6$ (388)

$^{(a)}$When $|P^e|$ or ω equals 1, a personnel member is only assigned to his base project subject to the time-related constraints. The decision variables and constraints related to resource sharing are redundant and are not counted.

Table 9: Impact of SSR^{strat} on the number of decision variables and the feasible number of patterns for project p ($|P| = 4$, $|T_{p'}| = 21$, constraint parameters as defined in Section 5.1)

The table reveals that the number of (binary) decision variables in the pricing problem is determined by both the time horizon $|T_{p'}|$ of project p' considered as base project and the number of projects $|P|$. However, as a result of the definition of the subset P^e, a subset of the decision variables is fixed to zero such that the solution space of the pricing problem is significantly reduced. Table 9 displays only the number of eligible variables, which is equal to $2 \times |T_{p'}| \times (1 + |P^e|) + |P^e|$. The number of relevant constraints is determined by $|T_{p'}|$, $|P^e|$ and the constraint input parameters as defined in Section 5.1. Hence, when $|P^e| = 4$, the number of binary decision variables is 214 and the number of constraints is 388. When $|P^e| = 2$, the number of eligible decision variables drops from 214 to 128 and the number of relevant constraints from 388 to 266, as listed between brackets. Note that when $\omega = 1$ or $|P^e| = 1$, only assignments are allowed to the base project such that the pricing problem

can be formulated without decision variables related to other projects or transfer duties and only time-related constraints included.

Further, the table gives insight in the number of feasible patterns contained in the pricing problems related to the parameter setting of the state reduction strategy SSR^{strat}. Both the time-related and resource sharing constraints have a significant impact on the number of feasible patterns, adding both to the complexity of the pricing problem. Further, we observe that the number of feasible patterns are largely reduced for pricing problems defined by a smaller state space. For the standard pricing problem that considers all projects ($\omega = |P^e| = 4$), the total number of feasible patterns amounts approximately 151×10^6. Reducing the maximum number of projects (e.g. $\omega = 2$) leads to only 19×10^6 feasible patterns. Specifying the set of eligible projects (e.g. $|P^e| = 2$) further diminishes the number of feasible patterns to approximately 12×10^6. In general, the definition of a restricted set of eligible projects P^e has the largest impact on the feasible solution space.