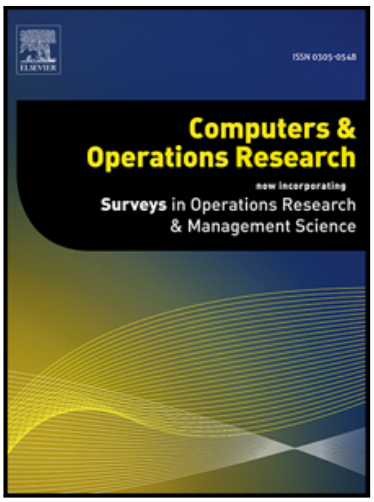
Accepted Manuscript



A heuristic procedure to solve the project staffing problem with discrete time/resource trade-offs and personnel scheduling constraints

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

PII: S0305-0548(18)30249-1
DOI: https://doi.org/10.1016/j.cor.2018.09.008
Reference: CAOR 4556

To appear in: *Computers and Operations Research*

Received date: 16 August 2017
Revised date: 4 June 2018
Accepted date: 17 September 2018

Please cite this article as: M. Van Den Eeckhout, B. Maenhout, M. Vanhoucke, A heuristic procedure to solve the project staffing problem with discrete time/resource trade-offs and personnel scheduling constraints, *Computers and Operations Research* (2018), doi: https://doi.org/10.1016/j.cor.2018.09.008

Highlights

- Project staffing with discrete time/resource trade-offs and calendar constraints
- An iterated local search procedure is proposed
- Different problem decomposition techniques are applied

A heuristic procedure to solve the project staffing problem with discrete time/resource trade-offs and personnel scheduling constraints

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

[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

When scheduling projects under resource constraints, assumptions are typically made with respect to the resource availability and activities are planned each with its own duration and resource requirements. In resource scheduling, important assumptions are made with respect to the staffing requirements. Both problems are typically solved in a sequential manner leading to a suboptimal outcome. We integrate these two interrelated scheduling problems to determine the optimal personnel budget that minimises the overall cost. Integrating these problems increases the scheduling flexibility, which improves the overall performance. In addition, we consider some resource demand flexibility in this research as an activity can be performed in multiple modes. In this paper, we present an iterated local search procedure for the integrated multi-mode project scheduling and personnel staffing problem. Detailed computational experiments are presented to evaluate different decomposition heuristics and comparison is made with alternative optimisation techniques.

Keywords: Multi-mode project scheduling; Personnel staffing; Heuristic optimisation

1 Introduction

Planning and scheduling is widely acknowledged as an important area in project management. All organisations and industries deal with projects where a set of activities or tasks need to be scheduled. These activities are typically carried out by a limited number of resources. Personnel resources are one of the most important resources in project planning and labour costs account for 30-50% of the total project costs (Adrian, 1987; Larson and Gray, 2011). Hence, it is essential to determine the personnel budget to carry out a particular project in an accurate manner. Since the supplied number of personnel resources are dependent upon the imposed calendar constraints, we devise a

staffing plan by constructing a baseline personnel roster. It is obvious that the scheduling of activities is a driver for the staffing requirements and/or that activities need to be scheduled in line with the personnel staffing and scheduling policies. However, in most cases project scheduling and resource scheduling are typically undertaken sequentially. In project scheduling, assumptions are typically made with respect to resource availability. In personnel scheduling, the resource requirements are known based on the project schedule. In an integrated problem, activities are scheduled and simultaneously a staffing plan is constructed. The integration of both problems may yield benefits in both directions. On the one hand, the inclusion of resource scheduling would provide additional flexibility for the project manager to schedule the project activities (Larson and Gray, 2011). On the other hand, practitioners frequently use demand management to steer their demand for service and to improve the resource utilisation (Easton and Rossin, 1997).

In this paper, we integrate project and personnel scheduling in order to introduce some flexibility in the scheduling process and to overcome the difficulties of composing isolated schedules. The problem primarily determines the minimum required number of personnel members to carry out a project. Furthermore, we consider some additional flexibility in the resource demand as we assume that an activity can be performed in multiple modes. Each activity mode is characterised by a certain duration and resource demand and the efficient modes of a particular activity ascertain a trade-off between the duration and resource demand. We consider only one type of renewable resources in the project scheduling problem, i.e. personnel or man units, and consider thus the discrete resource/time trade-off problem in project scheduling. The personnel staffing plan makes a further distinction of this single resource type to determine the ideal staffing composition in terms of regular personnel and temporary personnel to meet the staffing requirements. These personnel resources differ in cost and scheduling flexibility, which is relevant in staffing applications. Hence, we simultaneously determine the activity start times, the operation mode for each activity and the best resource mix in terms of cost given the project deadline.
Note that a suitable realistic background for the proposed problem is for example the software development sector. In the related scheduling problem, personnel members are assigned to tasks and the duration of the task is dependent on the number of assigned employees. The personnel members are not continuously available since they may not be fully dedicated to the project and no other resources require scheduling (Xiao et al., 2013). Moreover, the scope of the presented research can be broadened to other types of resources with calendar constraints, for which personnel is exemplary. Calendar constraints may also be relevant e.g. for machines such that the project schedule can account for machine maintenance and the scheduling of the required downtime is incorporated.

We propose a heuristic solution procedure to solve this integrated project scheduling and personnel staffing problem. We present different dedicated global improvement heuristics and explore different activity-based and personnel-based decomposition techniques. The

activity-based decomposition reduces the complexity of the project scheduling problem by selecting only a subset of activities in a (limited) time horizon. The personnel-based decomposition reduces the complexity of the personnel staffing problem by fixing some assignments in the master problem or the subproblem of the column generation procedure. These techniques are used in an iterated local search that considers both randomisation and solution quality to perform a perturbation move. Extensive computational experiments are conducted to compose a well-performing algorithm and to demonstrate the contribution of each component of the algorithm.

The further outline of the paper is as follows. In Section 2, we give an overview of the relevant literature on project scheduling, personnel staffing and the integrated problem. In Section 3, we provide a description and a mathematical formulation of the problem under study. In Section 4, the proposed heuristic algorithm is discussed and we present various alternative local optimisation strategies. The test design and computational experiments are discussed in Section 5. In Section 6, conclusions are drawn and directions for future research are provided.

2 Literature review

Project scheduling and personnel staffing are widely studied in the literature. In this section, we discuss the relevant literature on the project scheduling problem (Section 2.1) and the personnel staffing problem (Section 2.2). Section 2.3 gives an overview on the integration of both optimisation problems.

2.1 The project scheduling problem

In this research, we study the scheduling of a project where the execution of the project activities requires resources. Hartmann and Briskorn (2010) give an overview on resource-constrained project scheduling and discuss the variants and extensions of the resource-constrained project scheduling problem (RCPSP). This problem tries to minimise the project makespan subject to precedence relations between activities and under the assumption that the project is scheduled under a (constant) renewable resource availability. One of the most studied extensions of the basic RCPSP is the multi-mode resource constrained project scheduling problem (MRCPSP) where multiple execution modes for each activity are defined. Every mode is characterised by a specific duration and resource requirement. The MRCPSP considers typically renewable resources, which are limited per time unit (e.g. manpower, machines), and non-renewable resources, which are limited for the entire project (e.g. budget). The discrete time/resource trade-off problem (DTRTP) is closely related, but considers no non-renewable resources and only one renewable resource (Ranjbar et al., 2009). In the literature, several exact, heuristics and meta-heuristics are proposed to solve the MRCPSP and DTRTP. An overview can

be found in resp. Weglarz et al. (2011) and Ranjbar et al. (2009). Since the exact procedures are unable to solve large-sized realistic projects with multiple modes in a reasonable computation time, different single-pass heuristic and meta-heuristic procedures are presented. De Reyck et al. (1998) present several heuristic procedures that are based on the decomposition of the problem into a mode assignment phase and a resource-constrained project scheduling phase with fixed mode assignments. Lova et al. (2006) compare different heuristics based on single-pass and multi-pass priority rules to sort the activities and mode selection rules to assign an execution mode to each activity. Van Peteghem and Vanhoucke (2014) provide an excellent literature overview of developed meta-heuristic solution procedures for the MRCPSP. Different types of *meta-heuristic strategies* are proposed and mainly the evolutionary frameworks of genetic algorithm (Ranjbar and Kianfar, 2007; Lova et al., 2009; Van Peteghem and Vanhoucke, 2010; Coelho and Vanhoucke, 2011) and scatter search (Ranjbar et al., 2009; Van Peteghem and Vanhoucke, 2011) are visited.

Both MRCPSP and DTRTP consider makespan minimisation as objective similar to the basic RCPSP. There are, however, different other project scheduling problems where a cost minimisation objective is used and the context may better represent capacity planning and personnel resources. The resource availability cost problem (RACP) minimises the total cost of the constant renewable resources required to complete the project by a pre-specified project deadline (Möhring, 1984). The time-constrained project scheduling problem (TCPSP) considers a fixed resource availability and determines the need for overtime and temporary resources in order to complete a project within the deadline (Deckro and Herbert, 1989). The resource renting problem (RRP) considers fixed resource availability costs and variable renting costs such that the cost of hiring a single resource unit depends on the number of time periods (Nübel, 2001). The features of resource scheduling have been included by Kreter et al. (2016), who study the related RCPSP with general temporal and calendar constraints, implying that some resources are unavailable on certain days and, consequently, the execution of some activities needs to be delayed. The resource leveling problem (RLP) tries to level the resource usage by minimising the changes in this resource usage (Neumann and Zimmermann, 2000). Other min-cost project scheduling problems do not focus on resource costs. Maniezzo and Mingozzi (1999) minimise (irregular) activity start time costs. Achuthan and Hardjawidjaja (2001) consider two types of project costs, where the first type are earliness and tardiness costs related to milestone events. The crashing costs are the second type of costs which are incurred when the duration of an activity is decreased. All these project scheduling problems belong to the class of problems with a nonregular objective function, which are, in contrast to the RCPSP, DTRTP and MRCPSP, not a nondecreasing function of the activity completion times.

2.2 The personnel staffing problem

The underlying resource budgeting problem is a personnel staffing problem, which is studied in the academic literature in different guises and formulations (Ernst et al., 2004; Van den Bergh et al., 2013). The staffing decision is based upon the construction of a days off schedule for the personnel resources given the fixed resource requirements per time unit in order to minimise the workforce size or cost (Venkataraman and Brusco, 1996). Different models are discussed for the acyclic days off scheduling problem with a fluctuating but fixed daily demand in order to minimise labour costs. Beaumont (1997), Billionnet (1999) and Alfares (2003) use (mixed-)integer programming to solve a days off scheduling problem where the number of days off and the resulting work pattern may vary from week to week for an individual employee. These papers consider only a limited set of predefined days-off roster lines for individual employees. However, when the complexity increases, dedicated algorithms are required to solve workforce scheduling models. Caprara et al. (2003) present a number of mathematical models for different complex staff scheduling problems. The objective is to minimise the number of employees required to perform all daily assignments in the horizon. Several authors have solved personnel scheduling problems using column generation and closely connected branch-and-price approaches (Gamache et al., 1999; Mehrotra et al., 2000; Caprara et al., 2003; Beliën and Demeulemeester, 2006; Maenhout and Vanhoucke, 2010).

2.3 The integrated project scheduling and personnel staffing problem

In the project scheduling literature, different related papers do not construct a personnel roster that incorporates resource calendar constraints. However, these studies only assign personnel resources to tasks or compose a personnel budget for one or multiple projects. In this context, Tiwari et al. (2009) study a multi-project environment where activities can be assigned to a multi-skilled workforce according to different execution modes. Fernandez-Viagas and Framinan (2014) propose a greedy randomised adaptive search procedure to minimise the project duration for an integrated project scheduling and staff assignment problem with controllable processing times, i.e. the duration of the activities are dependent upon the number of employees assigned. Drezet and Billaut (2008) incorporate a labour constraint as they limit the number of assigned activities for individual employees. They minimise the lateness of a project scheduling problem and solve a staff assignment problem assuming a fixed days off manpower schedule. Kolisch and Heimerl (2012) propose a meta-heuristic algorithm for the staffing of multiple projects and consider internal and external resources with multiple skills. Their solution procedure first decides on the activity start times and then solves a staffing subproblem to come up with an efficient staffing plan.

There are only a few papers where a baseline personnel days off schedule is composed and time-related labour constraints are considered. Alfares and Bailey (1997)

and Alfares et al. (1999) minimise the project duration and personnel staffing costs via integer programming optimization using a commercial solver. The constructed personnel roster encompasses a cyclic days off schedule for a homogeneous workforce (Alfares and Bailey, 1997) and a multi-skilled heterogeneous workforce (Alfares et al., 1999). Maenhout and Vanhoucke (2016) construct a non-cyclic baseline roster and consider overtime and temporary resources in addition to regular personnel resources. They present an iterative branch-and-price procedure and compare their approach with a sequential approach, which first composes a project schedule and then constructs a staffing plan. Maenhout and Vanhoucke (2017) evaluate different (non-)cyclic scheduling policies and assess the impact of more flexible resource types. However, different from the problem under study, these studies considers only a single operation mode per activity and no decision is made on the duration and resource assignment of a single activity.

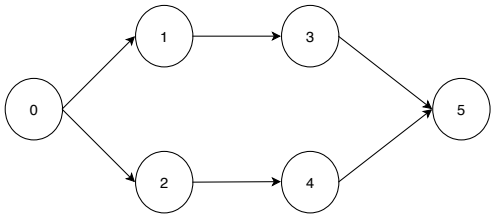
3 Problem definition and formulation

The problem under study comprises a strategic problem that simultaneously decides on the project schedule and the personnel budget. The project schedule determines the required number of resources per time unit to carry out the scheduled activities. Based on the staffing demand, the personnel staffing problem determines the personnel budget, which gives insight in the required personnel size and mix. Since personnel calendar constraints are imposed, a staffing plan is more accurately devised by constructing a feasible baseline personnel roster. By considering the project scheduling decision in the staffing decision, we incorporate demand flexibility such that we are able to determine the best staff composition in terms of personnel cost to carry out a single project. Moreover, we introduce some additional demand flexibility as the different activities may be performed according to several execution modes, each with their own duration and resource requirement. As a result, we have to decide on the timing and operation modes of the activities and the staffing plan. In multi-mode project planning, makespan minimisation is the most common objective, whereas minimising the resource cost is the common objective in personnel staffing. When both problems are integrated there is a trade-off between these two objectives, i.e. a shorter makespan will require a higher number of resources and vice versa. In this research, we defined a deadline for the project and focus on the minimisation of the personnel costs.

In the following sections, we describe the integrated multi-mode project scheduling and personnel staffing problem in detail. In Section 3.1, we define the activity and project scheduling characteristics and constraints. Section 3.2 describes the characteristics of the personnel staffing problem. Section 3.3 provides a mathematical formulation of the problem.

3.1 Activity and project scheduling characteristics

The activity and project scheduling characteristics can be defined as follows. The project is represented as an activity-on-the-node network $G = (N, A)$, where the set of nodes N represent the activities and the set of pairs A are the direct precedence relations between a pair of activities with a finish-start relationship with time lag 0. A project network and corresponding graph are illustrated in Figure 1. The set N (index i) contains $n + 2$ activities numbered from 0 to $n + 1$ where activities 0 and $n + 1$ represent the dummy start node and dummy end node respectively. The activities are to be scheduled without pre-emption. The availability of the renewable personnel resources for each time unit t of the planning horizon T is determined by the staffing plan via the construction of a baseline personnel roster.



	Mode 1		Mode 2		Mode 3	
Act i	d_{i1}	r_{i1}	d_{i2}	r_{i2}	d_{i3}	r_{i3}
0	**0**	**0**				
1	2	4	3	2	**4**	**1**
2	**3**	**4**	4	3	5	2
3	1	3	**2**	**2**	3	1
4	**1**	**5**	2	3	4	2
5	**0**	**0**				

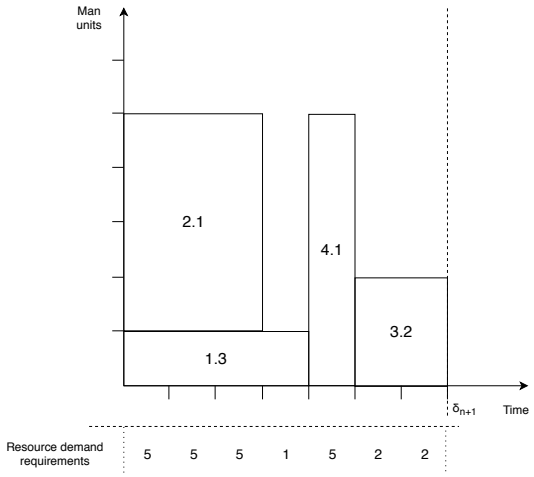


Figure 1: Example project network with a feasible project plan

Each activity $i \in N$ is executed according to a particular mode m_i, which is selected out of a set of modes M_i. Each mode has a deterministic duration d_{im} and requires r_{im} man units per time unit. An activity consumes only renewable resources. The dummy start and dummy end activity have zero duration and zero resource usage. A project schedule h is defined by a vector of activity start times $S^h = (s_1^h, \ldots, s_n^h)$ and a mode vector $P^h = (p_1^h, \ldots, p_n^h)$. We assume that there is a prescribed project deadline δ_{n+1}. In order to determine a project schedule, the start times of the activities v_{it} for all activities $i \in N$ and all time periods $t \in T$ should be determined together with the selection of an operation mode y_{im} for all activities $i \in N$. Let v_{it} be a binary variable that is equal to 1 if activity i starts at the beginning of time period t, 0 otherwise. Based on critical path analysis, the set of eligible start times is determined by the earliest (EST_i) and latest

(LST_i) start time of activity i, leading to $s_i^h = \sum_{t \in [EST_i, LST_i]} t\, v_{it}$ for project schedule h. The variable y_{im} is a binary variable that is equal to 1 if operation mode m is selected for activity i, 0 otherwise, and $p_i^h = \sum_{m \in M_i} m\, y_{im}$ for project schedule h.
A project schedule is feasible if it is non-preemptive and if the deadline and precedence constraints are satisfied. In Figure 1, a feasible project schedule is displayed based on the given project network and activity information. The graphical illustration denotes the activities based upon their activity number i and the executed mode m (notation $i.m$). For example, activity 1.3 implies that activity 1 is executed according to mode 3.

3.2 Personnel staffing characteristics

The integrated multi-mode project scheduling and personnel staffing problem under study considers only one type of renewable resources, i.e. personnel or man units. The execution of the project activities leads to a demand for personnel resources, which can be satisfied by regular and temporary personnel resources.

Regular personnel
The scheduling of the regular workers implies a manpower days-off scheduling problem (Alfares and Bailey, 1997; Alfares, 2001). All personnel members are anonymous and identical as they possess all skills to carry out the activities. The individual personnel schedules are determined by the imposed time-related constraints, i.e.

(i) The minimum number of working assignments w^{min} per unit time period l

(ii) The maximum number of working assignments w^{max} per unit time period l

(iii) The minimum number n^{min} of consecutive working assignments

(iv) The maximum number n^{max} of consecutive working assignments

(v) The minimum number f^{min} of consecutive days off

(vi) The maximum number f^{max} of consecutive days off

Constraints (i) and (ii) are counter constraints that are evaluated over a unit time period $l \in L$. The set of time units of time period l is denoted as $T_l \subset T$. E.g. a project with a makespan of 14 days may consist out of 2 unit time periods with a length of 7 days each, i.e. $T_1 = \{1, \ldots, 7\}$ and $T_2 = \{8, \ldots, 14\}$. Constraints (iii) to (vi) are sequence constraints, which stipulate the assignments to workers over successive days.
Depending on the rigidity of these regulations, personnel can be scheduled according to a cyclical or a non-cyclical manner. In *cyclic scheduling*, individual personnel members are scheduled according to a pre-defined work pattern. Their schedule is repeated over time in a cyclical manner. *Non-cyclic scheduling* is a more flexible way of scheduling that starts from an empty roster to create an ad hoc roster based upon the time-related constraints, which is performed in this research.

The decision variable x_j denotes the total number of regular workers assigned to personnel schedule $j \in J$. J represents the set of all feasible schedules, whereas $\bar{J} \subseteq J$ is a set of considered patterns. A personnel schedule is represented by a binary vector (a_{j1}, a_{j2}, ..., $a_{j|T|}$) with a length equal to the planning horizon $|T|$ where the parameter a_{jt} indicates a working day ($a_{jt} = 1$) or a day off ($a_{jt} = 0$) for day t.

Temporary personnel
In contrast to the regular workers, temporary personnel can be hired for a single day to satisfy the service demand. These temporary resources are external personnel members hired from e.g. subcontractors, agency bureaus. The decision variable o_t indicates the number of temporary personnel time units hired for time $t \in T$.

Figure 2 shows an example of a baseline personnel roster able to cover the personnel demand per time unit stemming from the project schedule in Figure 1. The regular workers are scheduled over a planning horizon of 7 days ($|T| = 7$). Individual workers perform exactly 5 working days ($w^{min} = w^{max} = 5$) over a unit time period of 7 days ($|T_l| = 7$), with the restriction that the individual workers should be assigned to blocks of at least two consecutive working days ($n^{min} = 2$). No restrictions are set on the consecutive days off or on the maximum consecutive workdays. The baseline roster reveals the required personnel budget since in total 5 regular workers and 2 temporary workers are planned to cover the resource demand. Note that due to the time-related constraints, the regular workers are not constantly available. The five regular workers are assigned to a working day on day 1, 2, 3 and 5 to cover the respective resource demand requirements. Hence, due to the minimum consecutive working assignments, day 4 or day 6 should also be a working day for these workers. The maximum number of working assignments for each unit time period imposes that all workers are assigned to a day off on day 7. Hence, the time-related constraints lead to a mismatch between the resource demand and supplied staff. On day 4, two regular workers have an idle duty whereas two temporary workers are scheduled on day 7.

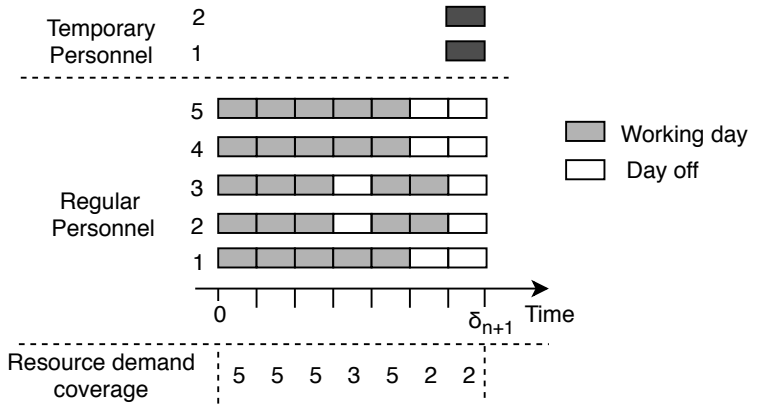


Figure 2: The personnel baseline roster based on the project schedule in Figure 1

3.3 Mathematical problem formulation

In the following, we provide a mathematical problem formulation for the integrated multi-mode project scheduling and personnel staffing problem.

$$\text{Minimise} \quad Z = \sum_{j \in \bar{J}} c_j \; x_j + \sum_{t \in T} c^{TP} \; o_t \tag{1}$$

$$\text{subject to} \quad \sum_{j \in \bar{J}} a_{jt} \; x_j + o_t - \sum_{i \in N} \sum_{m \in M_i} \sum_{t^*=t-d_i+1}^{t} r_{im} \; y_{im} \; v_{it^*} \geq 0 \qquad \forall t \in T \tag{2}$$

$$\sum_{t \in [EST_i, LST_i]} v_{it} = 1 \qquad \forall i \in N \tag{3}$$

$$\sum_{m \in M_i} y_{im} = 1 \qquad \forall i \in N \tag{4}$$

$$\sum_{t \in [EST_j, LST_j]} t \; v_{jt} - \sum_{t \in [EST_i, LST_i]} t \; v_{it} - \sum_{m \in M_i} d_{im} y_{im} \geq 0 \qquad \forall (i,j) \in A \tag{5}$$

$$\sum_{t \in [EST_{n+1}, LST_{n+1}]} t \; v_{n+1,t} \leq \delta_{n+1} \tag{6}$$

$$\begin{aligned} & x_j \geq 0 \text{ and integer} && \forall j \in \bar{J} \\ & o_t \geq 0 \text{ and integer} && \forall t \in T \\ & y_{im} \text{ binary} && \forall i \in N, \forall m \in M \\ & v_{it} \text{ binary} && \forall i \in N, \forall t \in T \end{aligned} \tag{7}$$

As the problem under study is a strategic budgeting problem, the objective [1] minimises the total personnel cost required to carry out a project. We consider different resource types. The *regular personnel* is rented for the complete planning horizon of the project. The cost c_j for assigning a regular worker to a schedule is determined based upon a daily cost c^{RG} and the project deadline δ_{n+1}, i.e. $c_j = c^{RG} \times \delta_{n+1}$. The use of *temporary personnel time units* is penalised with a variable cost c^{TP} per time period.
Constraint [2] embodies the staffing requirements for each time unit. This constraint links the project scheduling and personnel staffing problem as the staffing requirements per time unit are dependent on the start times of the activities and the selected activity mode. The term $\sum_{i \in N} \sum_{m \in M_i} \sum_{t^*=t-d_i+1}^{t} r_{im} \; y_{im} \; v_{it^*}$ calculates the amount of man units required at time t of the set of activities that are in progress. At each point in time $t \in T$, we have to decide on the resource mix (regular and temporary resources) scheduled such that all activities are carried out according to the project schedule.
Constraint [3] states that each activity can be started only once. Constraint [4] makes sure that each activity is performed in exactly one mode. Constraint [5] represents the direct precedence constraints with time lag 0, which stipulates that an activity cannot start before all its predecessors are finished. Constraint [6] stipulates that the dummy end activity $n+1$ should finish before δ_{n+1}, which means that the project should be finished before the project deadline. The non-negativity and integrality conditions on the

project and personnel schedule decision variables are stated in eqs. [7]. The personnel constraints are implicitly stated as these are incorporated in the definition of a feasible schedule $j \in \bar{J}$. In order to find the optimal solution, we need to consider the set of all feasible schedules, i.e. $\bar{J} = J$. However, since it is computationally not possible to incorporate all feasible schedules, we consider only a restricted set of generated schedules. A pattern j is called feasible if the following constraints hold:

$$\sum_{t \in T_l} a_{jt} \geq w^{min} \qquad \forall l \in L \qquad (8)$$

$$\sum_{t \in T_l} a_{jt} \leq w^{max} \qquad \forall l \in L \qquad (9)$$

$$\sum_{t}^{t+n^{min}-1} a_{jt} - n^{min} a_{jt}(1 - a_{j,t-1}) \geq 0 \qquad \forall t \in T \qquad (10)$$

$$\sum_{t}^{t+n^{max}} a_{jt} \leq n^{max} \qquad \forall t \in T \qquad (11)$$

$$\sum_{t}^{t+f^{min}-1} (1 - a_{jt}) - f^{min} a_{j,t-1}(1 - a_{jt}) \geq 0 \qquad \forall t \in T \qquad (12)$$

$$\sum_{t}^{t+f^{max}} (1 - a_{jt}) \leq f^{max} \qquad \forall t \in T \qquad (13)$$

$$a_{jt} \text{ binary} \qquad \forall t \in T \qquad (14)$$

Constraints [8] and [9] indicate the minimum and maximum number of working assignments for each unit time period. Constraints [10] and [11] stipulate the minimum and maximum number of consecutive working assignments. Constraints [12] and [13] impose the minimum and maximum number of consecutive days off. Equation [14] embodies the binary conditions.

The problem under study embodies different operating principles. Firstly, a trade-off is comprehended in the decision on the number of regular and temporary workers. Regular workers have a lower daily cost, but need to be paid the entire planning horizon. In contrast, temporary workers are relatively more expensive, but can be hired for a single day and embody thus a higher scheduling flexibility. Secondly, another trade-off arises on the activity planning level, where a relation exists between the duration and resource demand of an activity. More precisely, a higher duration will lead to a lower resource demand, and vice versa. A set of (non-dominated) execution possibilities or modes are present for each activity. Furthermore, on the project level, the activities should be scheduled given a fixed project deadline. This implies that trade-offs exist between the selected durations of different activities. Crashing the activity duration of one activity may allow the increase of the duration of another activity to reduce its resource demand, which would otherwise be impossible as a result of the defined precedence relationships between activities. Thirdly, time-related constraints, i.e. sequence and counter constraints, are imposed on the schedule of a regular worker. As a result of scheduling

regular workers according these constraints, the timing of activities should be matched with the scheduled personnel resource capacity. An inappropriate timing of activities and the associated resource demand may unnecessarily increase the personnel budget, i.e. the required number of regular workers and/or temporary workers.

These operating principles are illustrated based on the example introduced in Figure 1 and Figure 2. Figure 3 shows an alternative project schedule on the left and the associated staffing plan on the right. The new project schedule arises due to a different assignment of the project scheduling variables. More specifically, activity 2 is scheduled according to mode 2 instead of mode 1 ($y_{2,2} = 1, y_{2,1} = 0$), leading to a longer duration and a lower resource demand for this activity and thus other resource demand requirements. This exemplifies the trade-off introduced by the different mode assignments. This mode change leads to an improved personnel plan, where 4 regular and 5 temporary workers are needed to cover the demand. Setting $c^{RG} = 2$ and $c^{TW} = 4$, a personnel budget of 76 ($= c^{RG} \times \delta_{n+1} \times \sum_{j \in \bar{J}} x_j + c^{TP} \times \sum_{t \in T} o_t = 2$ x 7 x 4 + 4 x 5) is obtained. This is a cost decrease of 2 compared to the personnel budget of the personnel plan in Figure 2 needing 5 regular workers and 2 temporary workers, requiring a budget of 78 (= 2 x 7 x 5 + 4 x 2). This highlights the resource trade-off, since one regular worker is replaced by three temporary workers. The alternative project schedule of Figure 3 provides a better match with the staffing plan. The staffing plan in Figure 2 included two idle working duties on day 4 as a result of the time-related constraints imposed on the schedules of individual workers. In the staffing plan proposed in Figure 3, one of these idle duties is efficiently filled in as worker 4 is assigned to activity 2. In the new staffing plan, the line-of-work of worker 5 has been replaced by temporary workers. Two temporary workers are required to satisfy the additional resource demand resulting from the mode change of activity 2.

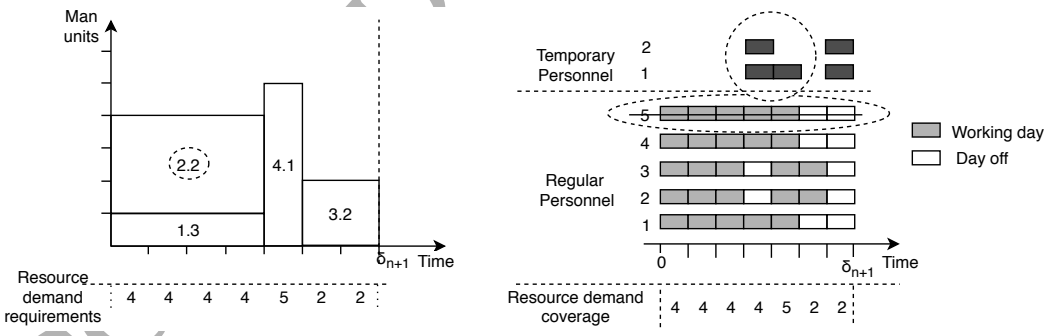


Figure 3: Example of a mode change leading to a lower personnel budget

Furthermore, considering the original project plan displayed in Figure 1, we see that the timing of activity 4 can be changed. Activity 4 starts in the proposed schedule at start time 4, although it could start immediately after activity 2 is finished, i.e. at start time 3. However, this will lead to extra temporary workers resulting in a worse objective function value. In general, changes in the project scheduling variables (v_{it} and/or y_{im})

impact the personnel scheduling variables (x_j and/or o_t) leading to a different personnel budget.

4 Procedure

In this section, we describe a heuristic solution procedure to solve the integrated multi-mode project scheduling and personnel staffing problem. Section 4.1 discusses the schedule representation and schedule generation of the proposed meta-heuristic. We propose an iterated local search (ILS) that is based on Lourenço et al. (2010). The literature review of Lourenço et al. (2010) gives a detailed description of iterated local search and acknowledges the beneficial performance for different applications. The pseudocode is displayed in Algorithm 1 and consists of the following steps. First, we generate a set of initial solutions randomly or via a constructive heuristic that is based on column generation (Section 4.2). An initial start solution is selected from this set based on the fitness value. The selected schedule is subject to a local improvement method (Section 4.3). The improvement heuristics are based on problem decomposition and optimise a series of smaller problems. We consider different local search heuristics based on activity-based decomposition and on personnel-based decomposition. After these two initial steps, an ILS iteration, which consists of three steps, is repeated until a termination condition is met. The first step of an iteration performs a perturbation based on the current base schedule to diversify the search and to escape from a local optimum (Section 4.4). Thereafter the local search procedure is again invoked to improve the (new) solution point. The last step of an iteration is the acceptance criterion, which determines if the base solution undergoing the perturbation step is changed. Since we aim to find good solutions in a small timeframe, we focus strongly on intensification and thus the base solution for perturbation is only changed when the (new) solution point has a better solution quality.

Algorithm 1 Iterated Local Search (Lourenço et al., 2010)

```
1: s_0 = Generate Initial Solution()
2: s* = Local Search (s_0)
3: repeat
4:     s' = Perturbation (s*, history)
5:     s'* = Local Search (s')
6:     s* = Acceptance Criterion (s*, s'*, history)
7: until termination condition met
```

In the following sections, we discuss each component into detail. For some components, we present alternative optimisation strategies to carefully design a well-performing algorithm. Different design choices have to be made thriving on the characteristics of the problem under study, finding the right balance between diversification and intensification and taking the trade-off in computational effort and solution quality into account.

4.1 Schedule representation and generation

In the proposed iterated local search, the different components and heuristic operators are applied on a schedule, which is represented by a schedule representation scheme. In order to represent a *project schedule* h, we make use of two vector notations, i.e. a vector of activity start times S^h and a mode vector P^h. For example, the project in Figure 1 is represented by the start time vector (0,0,5,4) and the mode vector (3,1,2,1). The *baseline personnel roster* is presented in worker-day view, which provides the individual lines-of-work, composed of days on and days off, for each hired worker. An example of the representation of a resource schedule is given in Figure 2.

A schedule is generated via integer programming using the mathematical formulation (1)-(7). In this way, a project schedule and the associated personnel staffing plan are determined via a branch-and-bound procedure based on a considered set of personnel patterns $\bar{J}$. Note that, unless otherwise stated, promising personnel patterns are added every time before solving the corresponding IP model by invoking a column generation step based on the LP formulation of the problem, i.e. equations (1)-(6), relaxing the integrality constraints (7). In this column generation process, an additional cut is added as a result of the required linearisation of the non-linear equation (2). An equivalent linear formulation together with the additional cut is given in appendix A. This schedule generation procedure is invoked at different stages in the procedure but sometimes in a different form with other conditions and constraints, which will become clear in the following sections.

4.2 Generate initial solution

In a meta-heuristic procedure, the initial starting solution is of great importance for the overall performance, since the computational time is limited (Lourenço et al. (2010)). For that purpose, we generate a set of feasible solutions G (index g), called a population, and select the best solution based on the fitness value as initial solution.
We propose five different methods to create the initial population of solution elements, which focus on the generation of a project representation scheme. A project schedule is constructed by designating the activity start times and the modes one-by-one. We first generate the set of project schedules and afterwards the associated personnel staffing plans are constructed. Note that the last three methods use information from the integrated problem setting. The details of these methods are indicated in the pseudocode in Algorithm 2 and are explained below.

Generation of project schedules

- **Init method 1.1**: The method generates all project representation schemes in a uniform random manner. We first create a mode list by selecting a uniform random mode for each activity. If the generated mode list is feasible with respect to the deadline constraint, we calculate the start time intervals of the activities based on

the mode list and determine the start times of the activities in a uniform random order. For each activity, a start time is selected according to a uniform probability distribution. The feasibility with respect to the precedence constraints and the deadline constraint is ensured since the start time intervals are updated after each start time assignment. The project schedule is evaluated by calculating the total work content of all activities, meaning that only temporary workers are used in the staffing plan. This method assumes that the workload of a project schedule is an important driver for the personnel budget.

- **Init method 1.2**: The method generates all project representation schemes in a uniform random manner in a similar manner as 'Init method 1.1'. The project schedule is evaluated by composing a staffing plan for both the regular and temporary workers and thus the trade-off between both types of workers is taken into account in order to select better project schedules. As a result, a column generation step is invoked based upon the fixed staffing requirements to carry out the project schedule.

- **Init method 2.1**: The method generates all project representation schemes in a random manner based on the problem formulation (eqs. (1)-(7)) relaxing the integrality conditions (eqs. (7)), i.e. the linear programming relaxation (LPR) that provides a lower bound on the optimal solution. The optimal LPR solution is obtained via column generation and the fractional values for the decision variables are used to bias the selection probabilities of the mode and start time selection. In particular, if for example $y_{i1} = 0.8$ and $y_{i2} = 0.2$, activity i has an 80% probability to be scheduled in mode 1 and 20% in mode 2. In this way, we may reduce the probability or even exclude the generation of some inferior project schedules with inappropriate activity start times or mode assignments. We thus explore information of the LPR solution to lead us to high-quality integer solutions. This method is conform to the research of Feltl and Raidl (2004), who used a linear programming relaxation (LPR) of their problem and a randomised rounding procedure to create an initial population for the generalised assignment problem. Once the project schedule is fixed, column generation is used as discussed for the 'Init method 1.2' to determine the staffing plan.

- **Init method 2.2**: The method is an extension of 'Init method 2.1' and adds an additional column generation when all modes are fixed by including equation (16) in the model. At this stage, the LPR relaxation provides more accurate information on the fractional values of the start time variables and the selection probabilities are adjusted correspondingly. Since the duration for each activity is known, the timing of activities is clearer, possibly increasing the quality of the generated project schedules.

- **Init method 2.3**: The method builds further on 'Init method 2.2' and adds a mutation step to the mode selection process. When the selection probabilities are biased corresponding to the LPR solution, it may be that the same mode

assignment m for activity i will be made for each population element (e.g. when $y_{im} = 1$). In order to avoid this strong bias, a mode mutation can occur with probability p_{mut} such that we select another mode according to a uniform random distribution. This mutation will lead to more diverse solutions in the population.

Algorithm 2 Generate Initial Solution

(Init method 2.1, 2.2, 2.3) Column generation
$g = 0$
while $g < G$ **do**
  for each activity **do**
    Select mode according to probability distribution
    (Init method 2.3) Mode Mutation
  end for
  Calculate deadline feasibility
  if feasible mode list **then**
    Update start time Intervals (based on mode list)
    (Init method 2.2, 2.3) Column generation (eq. (16) included)
    for each activity **do**
      Choose start time according to probability distribution
      Update start time intervals all activities
    end for
    Add solution to population ($g \leftarrow g + 1$)
  end if
end while
if *(Init method 1.2, 2.1, 2.2, 2.3)* **then**
  for $g \in G$ **do**
    Column generation (eqs. (15) and (16) included)
  end for
end if
for $g \in G$ **do**
  Determine staffing plan and fitness via branch-and-bound (eqs. (15) and (16) included)
end for
Sort population
Select best solution

Generation of personnel staffing plan

In order to obtain the resource staffing plan, we solve the integer problem formulation (1)-(7) to optimality using a branch-and-bound given a set of patterns $\bar{J}$, which are generated via column generation. Since the schedule representation of project schedule g is known, we include additional constraints (15) and (16), which respectively fix the activity start times $S^g = (s_1^g, \ldots, s_n^g)$ and the activity modes $P^g = (p_1^g, \ldots, p_n^g)$. The objective function value of the resulting resource schedule indicates the fitness of the solution element.

$$v_{i,s_i^g} = 1 \qquad \forall i \in N \qquad (15)$$

$$y_{i,p_i^g} = 1 \qquad \forall i \in N \qquad (16)$$

For reasons of computational efficiency, we delete the personnel patterns that are not included in one of the generated resource staffing plans for the different solutions $g \in G$. This results in a smaller set $\bar{J}$, which is input to the local search step.

4.3 Local Search

The proposed local search methods apply problem decomposition and optimise a series of smaller problems. For each such problem, a suitable mathematical integer program formulation is solved via a commercial solver. Decomposition is based on the principle of fixing certain decision variables. The number of fixed variables determines the complexity of the resulting model. A small neighbourhood with a large number of fixed variables has a small improvement potential. A large neighbourhood overcomes this risk but requires more computational effort. The decomposition and resulting neighbourhood size should be balanced in order to avoid being trapped in a local optimum.
The problem under study includes both project scheduling variables and personnel scheduling variables, which allows performing activity-based decomposition (Section 4.3.1) and personnel-based decomposition (Section 4.3.2). Similar to the initialisation method, promising personnel patterns are added by invoking a column generation step before solving the corresponding IP model. However, the local search determines the project schedule and the staffing plan simultaneously via global improvement heuristics. As a result, the personnel patterns added in the local search will have a higher quality in contrast to the initialisation method, which composes a staffing plan based upon a fixed project schedule.

4.3.1 Activity-based decomposition

Activity-based decomposition is a common type of decomposition in the project scheduling literature (e.g. Palpant et al. (2004)). Using a certain selection rule and based upon a predefined number of activities to select, a set of activities N' ($N' \subseteq N$) is determined and these activities can be scheduled in a different manner. For the problem under study, this implies that these activities can be assigned to another mode and/or another start time. All the other activities $i \in N \setminus N'$ are fixed according to their schedule representation.
Given a project schedule h with an activity start time vector $S^h = (s_1^h, \ldots, s_n^h)$ and a mode vector $P^h = (p_1^h, \ldots, p_n^h)$, the local search optimises the integer problem formulation (1)-(7) to optimality given the set of patterns $\bar{J}$ under consideration and includes the following additional constraints

$$v_{i,s_i^h} = 1 \qquad \forall i \in N \setminus N' \tag{17}$$

$$y_{i,p_i^h} = 1 \qquad \forall i \in N \setminus N' \tag{18}$$

Equations (17) and (18) fix resp. the start times and modes of the set of activities $N \setminus N'$. The selection of the set of activities N' is determined by both the features of the activities and the considered time horizon, which are both explained below.

Activity feature-based selection
In the literature, different selection rules exist based upon the characteristics of the activities, which can be classified into activity-based, network-based and schedule-based rules (Maenhout and Vanhoucke (2016)). The activity features have an impact on the scheduling flexibility of activities. For the problem under study, activities with a high (low) duration, a high (low) resource demand, a high (low) workload and a small (large) amount of slack are harder (easier) to schedule. Therefore, focusing on re-scheduling activities with a low scheduling flexibility may be beneficial, or vice versa. For example, when activities with a higher duration are selected, their mode assignment can be adapted to a shorter duration allowing other activities to increase their duration given the deadline constraint. However, to avoid focusing on a certain set of activities, a biased selection percentage was calculated instead of using a deterministic cutoff value. This means that for example if the selection rule is based on the highest duration, an activity with a duration of 8 has a four-times higher selection probability compared to an activity with a duration of 2. When using a cutoff value of for example 4, activities with a lower duration could never be selected.

Horizon-based selection
The horizon-based selection determines the time period out of which scheduled activities are eligible for selection. We define four different strategies for determining this time period, taking the degree of intensification and different problem-specific characteristics into account, which are illustrated in Figure 4, as follows

- **Entire planning horizon**: The standard strategy for which all activities over the entire planning horizon are available for selection.
- **Unit time period** l: This strategy selects a horizon with a fixed length of $|T_l|$ time periods, i.e. equal to the pattern length over which an individual personnel schedule is evaluated. The start of the selected horizon may be any time period in the planning horizon, such that in total $|T| - |T_l| + 1$ time periods are eligible. Moreover, the selection probabilities for these time periods may be uniform random or biased based on the number of scheduled idle or temporary resources. The higher these costly resources in a particular period, the higher the selection probability of that period. Hence, although activity-based decomposition focuses on the project schedule, this strategy takes the information from the staffing plan into account.
- **Chronological selection**: A time period of one day is selected and is expanded in a chronological manner with consecutive time periods until $|N'|$ activities are selected.

This strategy leads to a more focused selection since in the previous strategies not all activities may be selected in the defined interval. Moreover, this strategy takes information from the project network and the staffing plan into account. When the resource availability is high and/or the project network has a high degree of parallelism, a large number of activities may be scheduled in parallel and therefore the defined interval will be small (and vice versa).

- **Block strategy**: The block strategy of Palpant et al. (2004) selects one (random) activity and all its contiguous and parallel neighbours in the project schedule. The width of the considered horizon is thus defined by the start times of the parallel and contiguous neighbours. This strategy takes the precedence relations of the project network into account on top of the project schedule and the staffing plan. The contiguous neighbours may be predecessor or successors of the selected activity.

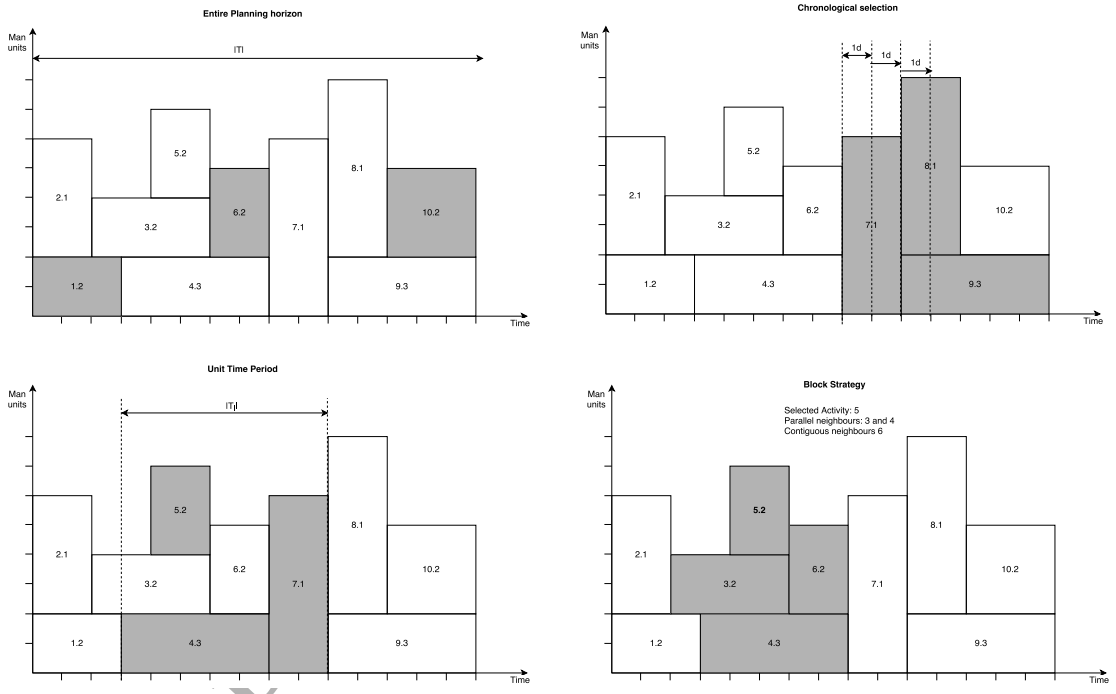


Figure 4: The four types of horizon selection

Note that the first two strategies select a fixed length of days, whereas for the other two strategies the selected time horizon is dependent on the project schedule. The above strategies differ from a pure time-based strategy, which selects all activities in the corresponding time interval. In the computational results, we show that imposing a maximum on the number of selected activities is beneficial.

4.3.2 Personnel-based decomposition

In personnel-based decomposition, we focus on the personnel scheduling variables and reduce the problem space to solve smaller subproblems of the resource scheduling problem. For this local search type, we make a distinction between two types of decomposition, i.e. worker and (personnel) pattern decomposition. In worker decomposition, we fix the individual schedules assigned to a specific number of workers. In pattern decomposition, we fix certain days on and days off in the pattern generation such that only a limited time horizon is set variable in an exhaustive pattern generation step and the additional considered set of patterns is restricted. Personnel-based decomposition is always combined with activity-based decomposition such that only a subset of activities is selected, otherwise the resulting decomposition model becomes intractable.

Worker decomposition
This type of decomposition focuses on the required personnel members and ensures the assignment of certain patterns. The general idea is that the workload stemming from a project plan may be satisfied by different personnel rosters and therefore we re-use information of the staffing plan by fixing the individual schedules of a number of workers. We can discern a set of patterns $\bar{J}'$ for which a minimum assigned number of personnel members is fixed, which is denoted as x_j^{min}. The local search optimises the integer problem formulation (1)-(7) to optimality given the set of patterns $\bar{J}$ under consideration and includes the following additional constraint, i.e.

$$x_j \geq x_j^{min} \qquad \forall j \in \bar{J}' \tag{19}$$

Furthermore, equations (17) and (18) are added to fix resp. the start times and modes of a set of activities as a result of the activity-based decomposition. We define different strategies to fix certain patterns, which are illustrated in Figure 5, as follows

- **Random**: This standard strategy fixes a percentage p_{fix1} of the assigned lines-of-work in accordance with the current staffing plan. The individual schedules are selected in a uniform random manner. If $p_{fix1} = 0\%$, no lines-of-work are fixed. If $p_{fix1} = 100\%$, the individual schedules of all hired staff members in the current staffing plan are fixed.
- **Constant base coverage**: This strategy ensures a constant supply of workers over the time horizon by fixing certain personnel patterns of the current staffing plan. This strategy is motivated by the observation that although different project schedules generate a different daily workload, a part of this workload has to be covered for all possible daily workload profiles. In order to prevent generating an entire staffing plan for all staff members each time, we fix some staff schedules such that only a staffing plan has to be computed to cover the workload on top of this base coverage. The constant supply of workers is determined as a percentage p_{fix2} of the workforce size that results from the current staffing plan. The set of patterns $\bar{J}'$ and the values for

x_j^{min} are selected by an IP model (cf. Appendix B), based on the current staffing plan that delivers a coverage that is closest to the required coverage, which is not a trivial task due to the imposed time-related constraints. Hence, the model tries to minimise the overall deviation, which is modelled via the deviation variables d_t^+ and d_t^-, between the defined required base coverage and the coverage provided by the set of patterns $\bar{J}'$.

- **Variable base coverage**: This strategy ensures a variable supply of workers over the time horizon in line with the staffing requirements by fixing certain personnel patterns of the current staffing plan. This strategy assumes that the current project schedule is of high-quality and therefore the associated staffing requirements are used to define a base coverage as the percentage p_{fix3} of these staffing requirements. Hence, although this is a personnel-based decomposition strategy, information from the project scheduling problem is taken into account. The set of patterns $\bar{J}'$ and the values of x_j^{min} are selected in a similar manner as for the constant base coverage strategy (cf. Appendix B).

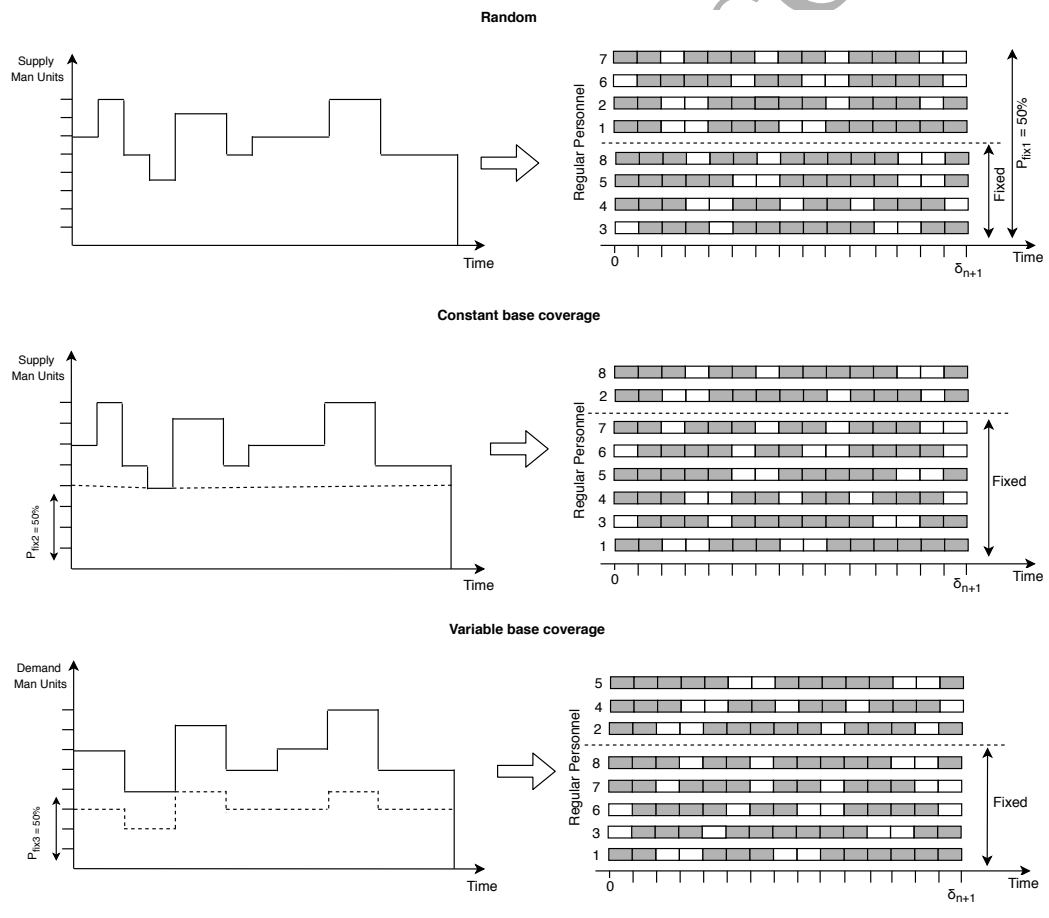


Figure 5: Different strategies for worker decomposition

Pattern decomposition

While worker decomposition focuses on the assigned schedules to workers and the corresponding x_j variables, pattern decomposition focuses on the set of patterns $\bar{J}$ that can be assigned to the workers and the underlying a_{jt} variables. The goal is to increase the set of considered personnel patterns $\bar{J}$ to obtain a high-quality staffing plan. To that purpose, we select a time horizon and consider for the current set of patterns the days on and days off assignments outside this horizon as fixed. We invoke a pattern generation step that enumerates, for the current set of patterns, all feasible patterns given the time-related constraints and the fixed assignments outside the time horizon. This decomposition strategy does not fix any decision variable of the integer problem formulation (1)-(7) but rather imposes a restriction on the set of all feasible patterns J, which allows the exhaustive enumeration and an increase in the number of considered patterns $\bar{J}$. The generated patterns are deleted after the local search in order to keep the total number of patterns in $\bar{J}$ relatively small for computational reasons. This decomposition is linked with activity-based decomposition that selects the activities over a (limited) time horizon and is illustrated in Figure 6.

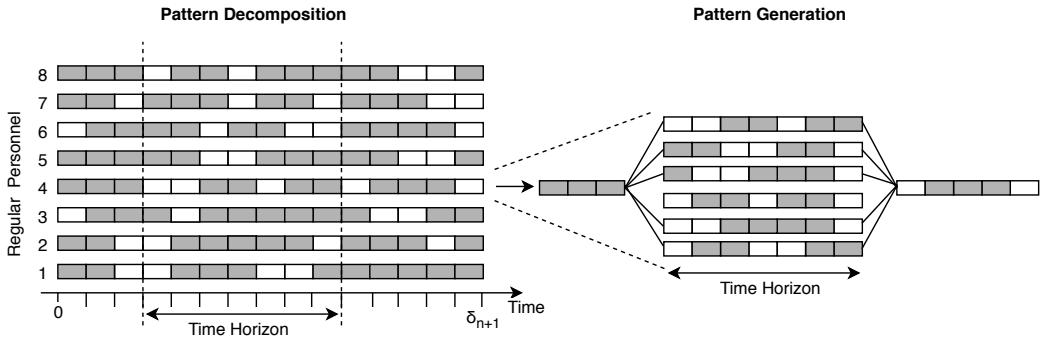


Figure 6: Illustration of pattern decomposition

4.4 Perturbation

Different strategies can be applied to escape from a local optima in an ILS framework. The most common perturbation is the random change of solution components, where the number of changes is called the perturbation strength (Lourenço et al., 2010). In this perspective, the most drastic strategy is to construct a completely random solution. Other perturbation strategies take the solution quality into consideration to perform a perturbation move. Congram et al. (2002) propose a backtrack strategy and go back to a previous visited local optimum. Lourenço et al. (2010) mention the optimisation of a sub-part of the problem as a good perturbation strategy, which is for example applied in the job shop scheduling problem (Lourenço, 1995). We propose a perturbation move that is based on both randomisation and solution quality. As long as we do not get trapped in a local optimum, we start a new iteration with the (improved) solution

of the previous iteration. We define a solution as a local optimum if this solution is the start solution for five iterations without improvement, which represents the history component in Algorithm 1. If a local optimum is reached, we perform a random move and we backtrack to the next best solution created in the initial population.

5 Computational experiments

This section provides computational insights into the proposed algorithm. In Section 5.1, we describe the test design for our computational experiments. Section 5.2 validates the performance of the different components and parameter settings of the algorithm. A benchmark with alternative solution procedures on different datasets is performed in Section 5.3. The algorithm was coded in C++ with the use of Gurobi 6.5. All tests were performed on an Intel Core E5-2680v3 processor with 2.5GHz.

5.1 Test design

In the following we describe the input characteristics to compose a project schedule and a personnel staffing plan and the applied stopping criteria.

5.1.1 Activity and project scheduling characteristics

We consider the network topology of the instances of the multi-mode PSPLIB benchmark dataset with 30 activities (Kolisch and Sprecher, 1996). For these instances, the network topologies are generated using ProGen (Kolisch et al., 1995) based on the network size, the minimum/maximum number of predecessors/successors of an activity and the network complexity indicator NC, i.e. the average number of non-redundant arcs per node. These instances can be further characterised by different network topology measures described by Vanhoucke et al. (2008). For example, the serial or parallel indicator (I_2) indicates how close the network is to a serial or parallel graph of activities ($I_2 = 0$ if all activities are in parallel, $I_2 = 1$ if all activities are in series). Due to the high computational requirements, a subset of 30 instances is selected in such a way that the range of values for the different network topology measures is representative for the entire dataset.

The multi-mode PSPLIB dataset considers two renewable and two non-renewable resources. Since we consider only one type of renewable resources, most of the modes of this dataset become inefficient meaning that a higher duration of an activity does not always lead to a lower (renewable) resource demand. Therefore, all the activity mode characteristics are deleted and newly generated via a random *mode generation* to create meaningful mode trade-offs. For each activity, we generated an initial duration and resource demand randomly in the interval [1,10]. Based upon this initial mode, two other

efficient modes are generated in this interval. However, the latter is not always possible, for example when the initial mode has a duration and resource demand equal to 10. This leads to an average of 2.91 modes per activity.

The average number of efficient modes is further influenced by the *project deadline*. A shorter deadline impacts the latest finish time of an activity i (LFT_i), leading to a smaller interval $[EST_i\ ,\ LFT_i]$. All modes of activity i for which the duration d_{im} exceeds $LFT_i - EST_i$ can be pruned and therefore a shorter deadline will result in a lower number of modes. The deadline formula presented in Vanhoucke and Debels (2007) is applied where the parameter k determines how close the deadline δ_{n+1} is to the longest path LP or shortest path SP according to the formula $\delta_{n+1} = SP + k \times (LP - SP)$. The shortest path, respectively longest path, is calculated by scheduling all the activities in the shortest mode, respectively the longest mode. When $k = 0\%$, the average number of modes drops to 2.06. We set the parameter k to 50% to retain the number of efficient modes at 2.91.

5.1.2 Personnel characteristics

The *objective function weights* of the personnel scheduling problem are determined based upon the literature (Campbell, 2012; Maenhout and Vanhoucke, 2017). Accordingly, the cost of regular c^{RG} and temporary personnel c^{TP} are set at 2 and 4 respectively.

The parameter values of the *time-related constraints* are set as follows, i.e. a worker can be assigned to

- minimum 5 and maximum 5 working days per unit time period l with $|T_l| = 7$ days (w^{min} and w^{max}).
- minimum 2 and maximum 6 consecutive working days (n^{min} and n^{max}).
- minimum 1 and maximum 2 consecutive days off (f^{min} and f^{max}).

5.1.3 Stop criteria

As we want to obtain high-quality solutions within a small time interval, we employ a stop criterion based on time and on the number of iterations of the iterated local search. We evaluate the performance of the procedure after 100 iterations and after 300 seconds in total to make an appropriate trade-off between different decomposition strategies. A time limit of 4 seconds was set for each generated subproblem in the local search step.

5.2 Validation of algorithmic parameters

We compare different optimisation strategies and parameter settings within the framework of the proposed meta-heuristic. Section 5.2.1 evaluates the different strategies for

constructing an initial solution. Section 5.2.2 validates different activity-based decomposition strategies, whereas the focus of Section 5.2.3 is on personnel-based decomposition.

For our analysis, we report the results with the following symbols, i.e.

Sol^x_{ILS}	Average solution quality obtained by the proposed ILS procedure (Stop criterion x)
Sol_{LP}	Average LP solution quality obtained by column generation
$\%Dev^x_{LB}$	Percentage deviation from the optimal LP solution (Stop criterion x) (= (Sol^x_{ILS} - Sol_{LP})/Sol_{LP})
Cpu^x	Average required CPU time (in seconds) (Stop criterion x)
$\#Pat^x$	Average number of generated personnel patterns (Stop criterion x)
$\#Upd^x_{SQ}$	Average number of accepted new solutions based upon solution quality (Stop criterion x)
$\#Upd^x_{RAND}$	Average number of accepted new solutions based upon randomisation (Stop criterion x)

The presented results are based on the average of ten different runs in order to reduce the impact of randomness. The results are displayed for a specific stop criterion x, i.e. 100 iterations of the iterated local search (x = 100i) or 300 seconds (x = 300s).

5.2.1 Impact of the initialisation method

Table 1 displays the performance of the different initialisation methods (cf. Section 4.2) for different population sizes ranging from 1 to 100. We display the solution quality obtained by the initialisation step and by the proposed procedure after 100 iterations. The deviation from the LPR lower bound and the CPU time after the initialisation step are indicated by $\%Dev^{Init}_{LB}$ and Cpu^{Init} resp. Since a trade-off is established based on the incorporated complexity in the initialisation method, we also analyse the algorithm based on the time-based stop criterion of 300 seconds. The local search is based on activity-based decomposition with $|N'| = 6$ and the time period is selected according to the 'entire planning horizon' strategy. Activity features are not taken into account.

Initialisation stage - The results show that the method that evaluates the project schedules based on work content only delivers the worst initial solutions since only temporary workers are scheduled ('Init method 1.1'). When including the scheduling of regular workers in the objective function ('Init method 1.2'), the deviation from the LPR drops to 60.03%, which results from the lower daily cost of regular workers. For these methods, which determine the activity start times in a uniform random manner, only 33% of the generated project schedules are feasible. When taking information of the LPR into account to bias the selection probabilities, about 67% of the generated project schedules are feasible and better initial solutions are generated. The method 'Init method 2.2' outperforms the other methods at the expense of some computational effort. This shows that information of the LPR leads indeed to better initial solutions. In general, the additional column generation after fixing the mode assignments improves the deviation from the LPR lower bound compared to 'Init method 2.1'. This results from considering the more accurate information on the activity start times.

Initialisation method	Pop. size	Init. stage		Overall performance				
		$\%Dev_{LB}^{Init}$	Cpu^{Init}	$\%Dev_{LB}^{100i}$	Cpu^{100i}	$\#Pat^{100i}$	$\#Upd_{SQ}^{100i}$	$\%Dev_{LB}^{300s}$
Method 1.1	**1**	140.95	0.0	16.45	237	238	67	15.78
	10	113.20	0.0	15.30	197	246	67	14.10
	20	107.76	0.0	15.11	197	243	67	14.09
	50	101.14	0.1	14.77	194	244	67	13.79
	100	97.12	0.2	14.51	192	238	67	13.71
Method 1.2	**1**	98.80	0	16.00	195	232	68	15.07
	10	73.90	1	14.95	219	270	67	14.15
	20	69.19	2	14.85	234	296	66	14.21
	50	63.91	5	14.54	251	342	65	14.08
	100	60.03	11	14.42	278	405	64	14.28
Method 2.1	**1**	77.77	38	15.59	189	223	67	14.87
	10	58.83	37	14.65	201	245	65	14.07
	20	54.79	40	14.56	208	265	65	14.07
	50	50.16	43	14.54	227	312	64	14.20
	100	47.68	47	14.73	243	369	63	14.48
Method 2.2	**1**	68.86	38	15.42	208	233	65	15.00
	10	51.36	41	14.68	217	261	64	14.19
	20	47.60	43	14.24	231	283	63	13.99
	50	44.07	49	14.44	252	330	61	14.38
	100	41.66	59	14.40	268	389	60	14.36
Method 2.3	**1**	71.79	39	15.82	201	230	65	15.02
	10	53.58	40	14.49	219	266	64	14.07
	20	49.08	42	14.00	229	285	63	13.62
	50	44.55	48	14.15	252	339	61	14.06
	100	42.27	59	14.06	272	401	60	14.03

Table 1: Results for different initialisation methods

Overall performance - When evaluating the performance of the different initialisation methods after 100 iterations, it is clear that a larger population size leads to a better performance for 'Init method 1.1 and 1.2'. However, when the information of the LPR biases the selection probabilities, the best results are obtained when an initial population of 20 solutions is considered. Despite the observation that 'Init method 2.2' leads to the best initial solution, the results for the overall performance reveal that the inclusion of a mutation step ('Init method 2.3') improves the overall performance after 100 iterations. After 100 iterations, 'Init method 2.3' performs best with a deviation of 14.00%. Although focusing on information from the LPR results in a good initial solution, the mutation step is valuable to compensate for the loss in the diversity of the initial population. When considering the time-based stop criterion of 300 seconds, we observe that 'Init method 1.1' (population size = 100) and 'Init method 2.3' (population size = 20) have a comparable performance. Hence, the worse performance of the 'Init method 1.1' in the initial stage of the algorithm is compensated by the fact that more iterations of the iterated local search are carried out. This shows that promising schedules can be identified by evaluating only the work content of a project schedule. The results in the remainder of the paper are based upon 'Init method 2.3' with an initial population size of 20.

5.2.2 Local search: activity-based decomposition

The activity-based decomposition selects a set of activities N' based upon a predefined number of activities, an activity feature selection rule and the time period selection.

Number of selected activities - Figure 7 shows the performance of the algorithm ($\%Dev^x_{LB}$) as a function of the number of selected activities ($|N'|$). We consider the entire planning horizon but the results are similar for other time horizon selections (cf. Table 3). Figure 7 displays for both stop criteria a convex behaviour. When the number of selected activities is low, the improvement potential of the decomposed problem is small. When the number of selected activities is high, the decomposed problem becomes too complex and leads to a poor performance as a result of the time limit imposed on the effort to find the optimal staffing plan.

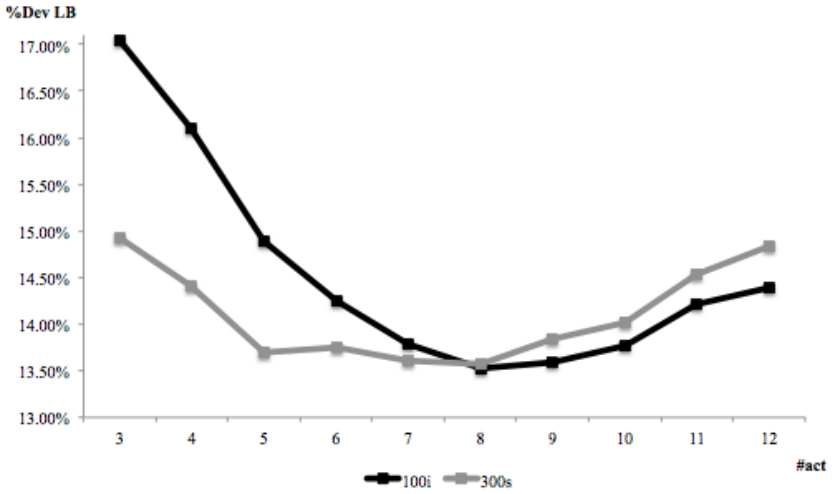


Figure 7: Impact of the number of selected activities on the algorithm performance

Activity feature selection rule - Table 2 shows the algorithm performance for different activity selection rules for a stop criterion of 100 iterations, considering the entire planning horizon as selection period. We consider the uniform random selection rule and different other rules that are biased by an activity characteristic, i.e. duration, resource demand, workload and the activity slack. The probability bias 'High' ('Low') indicates that a higher selection probability is given to activities with a higher (lower) characteristic value.

Probability bias	**High**		**Low**	
Activity feature	$\%Dev_{LB}^{100i}$	Cpu^{100i}	$\%Dev_{LB}^{100i}$	Cpu^{100i}
Duration	14.58	232	16.49	186
Demand	14.90	225	16.25	191
Workload	14.50	239	18.78	173
Slack	15.86	264	14.99	196
Uniform random	14.00	229	na	na

Table 2: Impact of the activity feature selection rule on the algorithm performance

The results reveal that a biased selection probability based upon activity features do not improve the algorithm performance, independent of the assigned probability bias. This means that taking the scheduling flexibility into account does not lead to improvements, indicating that it is more beneficial to select some activities with high and some activities with low scheduling flexibility.

Time period selection - Table 3 displays the results for the selected horizon from which activities are eligible for selection. Since the selected strategy may impact the interval and the number of selected activities, we display the average length of the determined interval $width_{avg}^{100i}$ and the average number of selected activities $\#act_{avg}^{100i}$. The results in the table lead to the following observations:

- When the *entire planning horizon* is considered, the selection of a subset of 8 activities yields the best results. Notice that the number of patterns, and thus the complexity of the decomposed problem, increases when more activities are selected and is larger compared to other horizon selection strategies. This is explained by the fact that the starting times and modes of activities may be changed on a larger number of days. As a result, a larger number of personnel patterns are generated in order to find the best possible match between the workload and the workforce capacity. Moreover, there is an inverse relation with the number of updates, which implies that more local optima are encountered when the number of activities is higher.
- When the *unit time period* strategy is applied and a horizon with length $|T_l|$ (i.e. 7 days) is selected, we observe that biasing the probabilities for the time period selection based upon the temporary or idle resources does not increase the performance. Both strategies are outperformed by the strategy that applies a uniform random probability to select a horizon of 7 days. This means that the presence of idle resources and temporary workers can not be avoided and therefore should not receive more than proportional attention. Each horizon should be given equal consideration to improve the solution quality. Moreover, the results reveal that a pure time-based selection strategy, which selects all activities in the interval ('All'), is not appropriate for the problem under study since the best performance is reached when maximum 8 activities are selected. Note that the average number of selected

activities is below the predefined number of activities, which indicates that most intervals have less than 8 activities.

- When the activities are *chronologically selected*, the results reveal that selecting 7 activities leads to the best performance. The average length of the selected time interval is 7.9 days. Compared to the first two strategies, the activities are selected in a more condense manner. As a result of the associated increase in scheduling flexibility, the maximum number of selected activities is reduced to 7 to obtain the best results.
- For the *block strategy*, the results reveal that selecting 6 activities, i.e. 1 activity and 5 parallel or contiguous neighbours, yields the best results. Since both predecessors and successors are selected, the scheduling flexibility has increased and therefore the number of selected activities is the lowest of all strategies. Although less activities are selected as opposed to the chronological selection, the average width of the interval is 0.4 days larger. The selection of a contiguous activity before and after the random activity explains this larger interval. The higher scheduling flexibility and the larger selection interval lead to more complex subproblems and therefore the performance deteriorates compared to the previous two strategies. The selection of all contiguous and parallel neighbours thus leads to slightly worse results.

Overall, we observe that the strategies that focus on a narrower time interval outperform the standard strategy 'Entire planning horizon' allowing the selection of activities in the entire planning period. The best strategy is to select a horizon with a length of $|T_l|$ days in accordance with the defined unit time periods. The strategies 'Chronological selection' and 'Block strategy' lead both to worse results. This shows that it is difficult to select one day (activity) starting from which the chronological selection (block strategy) is initiated. For the remainder of the paper, the results are based upon the strategy where maximum 8 activities are randomly selected within the selected unit time horizon.

5.2.3 Local search: personnel-based decomposition

The personnel-based decomposition fixes a number of decision variables related to the composition of the staffing plan and is carried out on top of the activity-based decomposition. We can apply a worker or a pattern decomposition.

Worker decomposition - Table 4 displays the results for the worker decomposition. For each proposed strategy, we varied the corresponding percentage to fix a number of assignments between 10% and 30%. We display explicitly the average number of fixed decision variables $\#SA_{fix}^{100i}$.

Table 4 reveals that after 100 iterations the worker decomposition deteriorates the performance of the algorithm slightly due to an increase of the number of patterns generated in the column generation step. When certain individual patterns are already fixed, the

Horizon type	$\|N'\|$	Sol_{ILS}^{100i}	$\%Dev_{LB}^{100i}$	Cpu^{100i}	$\#Pat^{100i}$	$\#Upd_{SQ}^{100i}$	$\#act_{avg}^{100i}$	$width_{avg}^{100i}$
Entire horizon	**5**	1461.5	14.89	187	275	63	5	-
	6	1450.2	14.00	229	285	63	6	-
	7	1447.5	13.79	276	306	61	7	-
	8	**1444.1**	**13.53**	**317**	**322**	**59**	**8**	**-**
	9	1445.0	13.60	387	342	56	9	-
Unit Time Period *Uniform probability*	**5**	1442.2	13.41	216	259	60	4.8	7
	6	1442.0	13.35	252	259	59	5.5	7
	7	1440.0	13.20	272	258	57	5.9	7
	8	**1439.4**	**13.16**	**261**	**228**	**56**	**6.3**	**7**
	9	1439.8	13.18	288	255	56	6.4	7
	All	1440.9	13.27	288	253	55	6.5	7
Unit Time Period *Biased by TW*	**5**	1446.6	13.72	231	265	60	4.8	7
	6	**1439.5**	**13.16**	**269**	**263**	**59**	**5.6**	**7**
	7	1441.7	13.33	292	264	57	6.1	7
	8	1441.7	13.34	301	263	56	6.4	7
	9	1440.3	13.22	304	264	56	6.6	7
	All	1442.8	13.42	306	255	55	6.7	7
Unit Time Period *Biased by IR*	**5**	1447.5	13.79	216	256	59	4.8	7
	6	1444.3	13.54	254	254	58	5.5	7
	7	**1439.4**	**13.16**	**276**	**259**	**57**	**5.9**	**7**
	8	1443.1	13.45	285	257	56	6.2	7
	9	1441.9	13.35	290	257	55	6.4	7
	All	1441.9	13.35	293	255	55	6.5	7
Chronological	**5**	1450.0	13.99	215	253	60	5	4.8
	6	1442.2	13.37	278	262	57	6	6.4
	7	**1442.1**	**13.37**	**324**	**259**	**54**	**7**	**7.9**
	8	1447.8	13.81	358	266	52	8	9.5
	9	1457.5	14.57	383	255	48	9	11.1
Block	**5**	1453.1	14.23	203	254	61	4.7	7.8
	6	**1442.4**	**13.39**	**238**	**258**	**60**	**5.3**	**8.3**
	7	1445.6	13.54	258	256	58	5.8	8.6
	8	1444.8	13.58	268	257	57	6.1	8.8
	9	1443.6	13.48	272	256	56	6.3	8.9
	All	1444.8	13.58	276	252	55	6.6	9.0

Table 3: Impact of the horizon selection on the algorithm performance

higher number of patterns is required to construct a high-quality staffing plan. The decreased complexity of the personnel staffing problem does not compensate for the decrease of flexibility to schedule the project activities. When more than 10% of the assignments are fixed, the CPU time decreases as a result of fixing a higher number of variables.

For a stop criterion of 300 seconds, the results reveal the beneficial performance of the worker decomposition when 10% of the assignments are fixed. The 'Variable base coverage' strategy, which fixes the assignments in line with the staffing requirements, yields the best improvement. This shows that incorporating characteristics of the project schedule, i.e. the daily workload, in the staff scheduling problem improves the performance of the algorithm.

Strategy	%fix	Sol_{ILS}^{100i}	$\%Dev_{LB}^{100i}$	Cpu^{100i}	$\#Pat^{100i}$	$\#Upd_{SQ}^{100i}$	$\#SA_{fix}^{100i}$	$\%Dev_{LB}^{300s}$
Random	**10%**	1441.0	13.28	281	269	57	1.4	13.26
	20%	1441.5	13.32	277	289	57	2.9	13.29
	30%	1441.9	13.35	274	311	58	4.4	13.32
Constant base coverage	**10%**	1440.6	13.25	285	282	57	1.9	13.28
	20%	1445.3	13.62	270	315	58	3.9	13.44
	30%	1448.3	13.86	258	353	58	5.8	13.33
Variable base coverage	**10%**	1439.7	13.17	285	268	58	1.2	13.11
	20%	1442.0	13.36	283	279	57	2.8	13.41
	30%	1441.0	13.28	280	290	58	4.2	13.67
None	**0%**	1439.4	13.16	261	228	56	0.0	13.29

Table 4: Impact of the worker decomposition on the algorithm performance

Pattern decomposition - Table 5 displays the results for the pattern decomposition. We vary the width of the time horizon between 1 and 11 days to investigate the impact of the number of patterns considered. When the width is equal to 0, pattern decomposition is not applied. The time horizon is positioned symmetrically within the interval selected for activity-based decomposition, which is fixed at 7 days (cf. Section 5.2.2). E.g. a time horizon of 3 days is positioned on day 3, day 4 and day 5 of the selected interval for activity-based decomposition. An interval larger than 7 days may increase the flexibility of days-on assignments during these seven days.

Width	Sol_{ILS}^{100i}	$\%Dev_{LB}^{100i}$	Cpu^{100i}	$\#Pat^{100i}$	$\#Upd_{SQ}^{100i}$
0	1439.4	13.16	261	228	56
1	1438.3	13.06	274	350	58
3	1436.2	12.90	277	390	57
5	1435.9	12.88	283	448	57
7	1435.2	12.83	291	501	57
9	1438.1	13.05	305	548	56
11	1440.2	13.21	332	601	56

Table 5: Computational results of pattern-based decomposition

The results show that pattern decomposition improves the performance of the algorithm. The interval for pattern decomposition is best as large as the considered interval for activity-based decomposition, i.e. 7 days. A larger interval for pattern decomposition is not desirable since the staffing requirements mainly change within this interval. Exploiting pattern decomposition improves the deviation from the optimal LP solution to 12.83%. We further observe that the increase in computational time is not proportional to the significant increase in the number of patterns.

5.3 Comparison with other solution procedures

In this section, we benchmark the proposed algorithm with respect to other solution procedures on different datasets, i.e.

Solution Procedures

- *Branch-and-Price* - We adapt the branch-and-price procedure of Maenhout and Vanhoucke (2016) for the single mode problem to the problem under study. We include a branching strategy to branch on the mode assignments that have fractional values. We truncate the branch-and-price after 3,600 seconds.
- *Restricted Branch-and-Bound* - This procedure solves the integer problem formulation (1)-(7) using the standard commercial software solver Gurobi. Due to the acyclic nature of the personnel patterns, incorporating all feasible patterns J in a branch-and-bound procedure will be too complex and only a reduced set of patterns $\bar{J}$ is considered. The set of patterns is determined based on the upper bound calculations of Maenhout and Vanhoucke (2016). We truncated the branch-and-bound after 3,600 seconds.
- *Multi-start heuristic* - This algorithm applies the proposed local search in each iteration on a completely random solution and allows us to validate the added value of the ILS framework.
- *ILS* - This algorithm is proposed in Section 4 and relies on iterated local search and different decomposition techniques.

Datasets

For our computational experiment, we consider different datasets and generate 30 instances for each dataset according to the same principle outlined in Section 5.1. The datasets have the following characteristics, i.e.

- MPSPLIB with 10 activities (k = 0% i.e. deadline = critical path)
- MPSPLIB with 10 activities (k = 50%)
- MPSPLIB with 20 activities (k = 50%)
- MPSPLIB with 30 activities (k = 50%) (= test set)
- MPSPLIB with 30 activities (k = 50%) (= validation set)
- MMLIB (Van Peteghem and Vanhoucke, 2014) with 50 activities (k = 50%)

The comparison of our procedure with the different datasets is given in Table 6. Apart from the deviation from the LP lower bound (%Dev_{LB}), the CPU time (Cpu) and the considered number of patterns (#Pat), we report the percentage deviation from the best performing procedure for a particular dataset (%Dev_{Best}).

The set of instances with the lowest complexity (10 act, $k = 0\%$) can be solved to optimality via the branch-and-price procedure within 37 seconds on average. The proposed ILS is able to find near optimal solution for these small instances, since its performance deviates only 0.07% from the optimal IP solutions after 100 iterations. Only for this dataset, the multi-start procedure performs better than the ILS.

For the other datasets with a larger number of activities or a higher k parameter value, we observe that the proposed algorithm outperforms all procedures. The comparison with the multi-start heuristic reveals the need for a meta-heuristic framework in the search process. When the number of activities is larger than 20, the restricted branch-

	δ_{n+1}	Sol_{LP}	%Dev$_{\text{Best}}$	%Dev$_{\text{LB}}$	Cpu	#Pat
MPSPLIB, 10 act, k = 0%	15	618				
Branch-and-Price			0.00	5.32	37	50
Restricted Branch-and-Bound			0.66	6.04	0	18
Multi-Start			0.03	5.35	18	146
ILS			0.07	5.42	27	158
MPSPLIB, 10 act, k = 50%	25	450				
Branch-and-Price			0.46	24.30	3389	781
Restricted Branch-and-Bound			1.06	25.04	32	49
Multi-Start			2.55	26.89	45	245
ILS			0.00	23.74	48	211
MPSPLIB, 20 act, k = 50%	40	790				
Branch-and-Price			10.86	32.10	3610	1025
Restricted Branch-and-Bound			0.27	13.23	2535	134
Multi-Start			10.66	33.71	151	485
ILS			0.00	12.82	170	391
MPSPLIB, 30 act, k = 50% (test set)	48	1272				
Branch-and-Price			17.08	32.10	3633	892
Restricted Branch-and-Bound			0.36	13.23	3563	246
Multi-Start			18.51	33.71	256	530
ILS			0.00	12.83	291	501
MPSPLIB, 30 act, k = 50% (validation set)	53	1382				
Branch-and-Price			18.80	34.14	3636	885
Restricted Branch-and-Bound			0.10	12.91	3600	415
Multi-Start			18.14	33.39	291	583
ILS			0.00	12.79	302	513
MMLIB, 50 act, k = 50%	48	2150				
Branch-and-Price			16.38	32.59	3720	457
Restricted Branch-and-Bound			3.38	17.78	3473	198
Multi-Start			23.13	40.28	281	572
ILS			0.00	13.92	341	586

Table 6: Comparison with other solution techniques

and-bound performs much better than the branch-and-price procedure. The procedures based on mathematical programming, i.e. the branch-and-price and the restricted branch-and-bound, are characterised by far larger computational times.

6 Conclusion

In this paper, we determine the personnel staffing budget in order to carry out a project. Since we assume different personnel calendar constraints, a baseline personnel roster is constructed to determine the staffing budget in an accurate way. To avoid a mismatch stemming from the isolated schedule composition, we integrate the personnel staffing problem and the project scheduling problem such that the demand for staff and the scheduling of the resources is determined simultaneously. In addition, we assume that the activities can be performed in multiple execution modes, reflecting a trade-off between

the number of personnel resources and the duration of activities. Due to the complexity of the resulting problem, we developed an iterated local search where the local search is based on decomposing the overall problem into smaller subproblems. Different dedicated decomposition techniques are presented, which can be classified in activity-based and personnel-based decomposition techniques.
In the computational experiments, we varied the different parameters of the initialisation method and decomposition techniques. Results showed that the decomposition should consider only a limited time horizon. A subset of activities should be selected within this time horizon and all possible work patterns should be considered in this timeframe to yield the best performance. Although the proposed decompositions strategies mainly focus on one problem, i.e. the project scheduling problem or the personnel staffing problem, considering information from the other scheduling problem improves the performance of the algorithm. This shows the added value of an integrated approach.

Future research should focus on a better management of the set of considered patterns, i.e. more promising patterns should be included while limiting the total amount of patterns. This could also lead to an increase in performance of the worker decomposition. Furthermore, in future research the problem definition should be expanded in connection with real-life problems and dedicated algorithms should be developed accordingly. In this perspective, we suggest to incorporate other types of flexibility for scheduling projects e.g. flexible network structures, activity pre-emption, etc. In the resource capacity plan, we may account for multi-skilled personnel resources or multiple types of resources. In the latter case, resources with calendar constraints (e.g. personnel or machines with scheduled maintenance) and resources with constant availability (e.g. equipment) can be considered.

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.

A Linearisation constraints

Before implementing the proposed model of section 3.3 and its adaptions in Gurobi, the non-linearity in the staffing constraint ($r_{im}y_{im}v_{it*}$) should be avoided. Therefore an AND-relation between y_{im} and v_{it} is incorporated by including the extra variables z_{imt} and the following constraints:

$$z_{imt} \leq y_{im} \qquad \forall i \in N, \forall t \in T, \forall m \in M \qquad (20)$$

$$z_{imt} \leq v_{it} \qquad \forall i \in N, \forall t \in T, \forall m \in M \qquad (21)$$

$$z_{imt} \geq y_{im} + v_{it} - 1 \qquad \forall i \in N, \forall t \in T, \forall m \in M \qquad (22)$$

It should be noted that when relaxing the linearisation constraints, the extra constraint $\sum_{t\in T}\sum_{m\in M} z_{imt} = 1 \quad \forall i \in N$ is needed to ensure that each activity is scheduled in a certain mode on a certain day and thus to ensure that the resource demands are taken into account in the staffing constraints. Otherwise, due to the AND-relation between y_{im} and v_{it}, z_{imt} can be zero when for example $y_{im} = v_{it} = 0.5$, which can results in a lower bound of zero.

The model formulation of the discrete time/resource trade-off problem only considers these z_{imt} variables with the additional constraint $\sum_{t\in T}\sum_{m\in M} z_{imt} = 1 \quad \forall i \in N$ (Ranjbar et al., 2009). However, using the y_{im} and v_{it} variables yielded a better linear programming relaxation.

B Mathematical formulation worker decomposition

Parameters

x_j	The number of regular workers assigned to pattern j in solution s'
a_{jt}	1, if personnel schedule j stipulates a working day for time period t, 0 otherwise
req_t	The staffing requirements at time t (see section 4.3.2)

Decision variables

x_j^{min}	The number of regular workers whose assignment to pattern j is fixed
d_t^+	Positive deviation variable for time t
d_t^-	Negative deviation variable for time t

$$
\begin{aligned}
\text{Minimise} \quad & Z = \sum_{t\in T} d_t^+ + \sum_{t\in T} d_t^- \\
\text{subject to} \quad & \sum_{j\in J} a_{jt}\, x_j^{min} + d_t^- = req_t + d_t^+ && \forall t \in T \\
& x_j^{min} \le x_j && \forall j \in J \\
& x_j^{min} \ge 0 \text{ and integer} && \forall j \in J \\
& d_t^+, d_t^- \ge 0 \text{ and integer} && \forall t \in T
\end{aligned}
\tag{23}
$$

References

Adrian, J. (1987). *Construction productivity improvement.* Elsevier, New York.

Alfares, H. (2003). Four-day workweek scheduling with two or three consecutive days off. *Journal ofMathematical Modelling and Algorithms*, 2:67–80.

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

Alfares, H. K. (2001). Efficient optimization of cyclic labor days-off scheduling. *OR-Spektrum*, 23(2):283–294.

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

Beaumont, N. (1997). Using mixed integer programming to design employee rosters. *Journal of the Operational Research Society*, 48:585–590.

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

Billionnet, A. (1999). Integer programming to schedule a hierarchical workforce with variable demands. *European Journal of Operational Research*, 114:105–114.

Campbell, G. M. (2012). On-call overtime for service workforce scheduling when demand is uncertain. *Decision Sciences*, 43(5):817–850.

Caprara, B., Monaci, M., and Toth, P. (2003). Models and algorithms for a staff scheduling problem. *Mathematical Programming*, 98:445–476.

Coelho, J. and Vanhoucke, M. (2011). Multi-mode resource-constrained project scheduling using rcpsp and sat solvers. *European Journal of Operational Research*, 213:73–82.

Congram, R. K., Potts, C. N., and van de Velde, S. L. (2002). An iterated dynasearch algorithm for the single-machine total weighted tardiness scheduling problem. *INFORMS Journal on Computing*, 14(1):52–67.

De Reyck, B., Demeulemeester, E., and Herroelen, W. (1998). Local search methods for the discrete time/resource trade-off problem in project networks. *Naval Research Logistics*, 45:553–578.

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

Drezet, L. and Billaut, J. (2008). A project scheduling problem with labour constraints and time-dependent activities requirements. *International Journal of Production Economics*, 112:217–225.

Easton, F. and Rossin, D. (1997). Overtime schedules for full-time service workers. *Omega*, 25(3):285 – 299.

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

Feltl, H. and Raidl, G. R. (2004). An improved hybrid genetic algorithm for the generalized assignment problem. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 990–995. ACM.

Fernandez-Viagas, V. and Framinan, J. M. (2014). Integrated project scheduling and staff assignment with controllable processing times. *The Scientific World Journal*, 2014.

Gamache, M., Soumis, F., Marquis, G., and Desrosiers, J. (1999). A column generation approach for large-scale aircrew rostering problems. *Operations Research*, 47:247–263.

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.

Kolisch, R. and Heimerl, C. (2012). An efficient metaheuristic for integrated scheduling and staffing it projects based on a generalized minimum cost flow network. *Naval Research Logistics*, 59:111–127.

Kolisch, R. and Sprecher, A. (1996). Psplib-a project scheduling problem library: Or software-orsep operations research software exchange program. *European journal of operational research*, 96(1):205–216.

Kolisch, R., Sprecher, A., and Drexl, A. (1995). Characterization and generation of a general class of resource-constrained project scheduling problems. *Management Science*, 41(10):1693–1703.

Kreter, S., Rieck, J., and Zimmermann, J. (2016). Models and solution procedures for the resource-constrained project scheduling problem with general temporal constraints and calendars. *European Journal of Operational Research*, 251(2):387–403.

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

Lourenço, H. R. (1995). Job-shop scheduling: Computational study of local search and large-step optimization methods. *European Journal of Operational Research*, 83(2):347–364.

Lourenço, H. R., Martin, O. C., and Stützle, T. (2010). Iterated local search: Framework and applications. In *Handbook of metaheuristics*, pages 363–397. Springer.

Lova, A., Tormos, P., and Barber, F. (2006). Multi-mode resource constrained project scheduling: scheduling schemes, priority rules and mode selection rules. *Inteligencia artificial*, 30(10):69–86.

Lova, A., Tormos, P., Cervantes, M., and Barber, F. (2009). An efficient hybrid genetic algorithm for scheduling projects with resource constraints and multiple execution modes. *International Journal of Production Economics*, 117(2):302–316.

Maenhout, B. and Vanhoucke, M. (2010). Branching strategies in a branch-and-price approach for a multiple objective nurse scheduling problem. *Journal of Scheduling*, 13:77–93.

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.

Mehrotra, A., Murphy, K., and Trick, M. (2000). Optimal shift scheduling: A branch-and-price approach. *Naval Research Logistics*, 47:185–200.

Möhring, R. (1984). Minimizing costs of resource requirements in project networks subject to a fixed completion time. *Operations Research*, January-February.

Neumann, K. and Zimmermann, J. (2000). Procedures for resource leveling and net present value problems in project scheduling with general temporal and resource constraints. *European Journal of Operational Research*, 127(2):425–443.

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

Palpant, M., Artigues, C., and Michelon, P. (2004). Lssper: Solving the resource-constrained project scheduling problem with large neighbourhood search. *Annals of Operations Research*, 131(1-4):237–257.

Ranjbar, M., De Reyck, B., and Kianfar, F. (2009). A hybrid scatter-search for the discrete time/resource trade-off problem in project scheduling. *European Journal of Operational Research*, 193:35–48.

Ranjbar, M. R. and Kianfar, F. (2007). Solving the discrete time/resource trade-off problem in project scheduling with genetic algorithms. *Applied Mathematics and Computation*, 191(2):451–456.

Tiwari, V., Patterson, J., and Mabert, V. (2009). Scheduling projects with heterogeneous resources to meet time and quality objectives. *European Journal of Operational Research*, 193:780–790.

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 Peteghem, V. and Vanhoucke, M. (2010). A genetic algorithm for the preemptive and non-preemptive multi-mode resource-constrained project scheduling problem. *European Journal of Operational Research*, 201(2):409–418.

Van Peteghem, V. and Vanhoucke, M. (2011). Using resource scarceness characteristics to solve the multi-mode resource-constrained project scheduling problem. *Journal of Heuristics*, 17(6):705–728.

Van Peteghem, V. and Vanhoucke, M. (2014). An experimental investigation of metaheuristics for the multi-mode resource-constrained project scheduling problem on new dataset instances. *European Journal of Operational Research*, 235(1):62–72.

Vanhoucke, M., Coelho, J., Debels, D., Maenhout, B., and Tavares, L. V. (2008). An evaluation of the adequacy of project network generators with systematically sampled networks. *European Journal of Operational Research*, 187(2):511–524.

Vanhoucke, M. and Debels, D. (2007). The discrete time/cost trade-off problem: extensions and heuristic procedures. *Journal of Scheduling*, 10(4-5):311–326.

Venkataraman, R. and Brusco, M. (1996). An integrated analysis of nurse staffing and scheduling policies. *OMEGA International Journal of Management Science*, 24:57–71.

Weglarz, J., Józefowska, J., Mika, M., and Waligóra, G. (2011). Project scheduling with finite or infinite number of activity processing modes–a survey. *European Journal of Operational Research*, 208(3):177–205.

Xiao, J., Ao, X.-T., and Tang, Y. (2013). Solving software project scheduling problems with ant colony optimization. *Computers & Operations Research*, 40(1):33–46.