CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

A Branch-and-Check Approach to Solve on Onshore Wind Turbine Maintenance Scheduling Problem

Aurélien Froger
Michel Gendreau
Jorge E. Mendoza
Eric Pinson
Louis-Martin Rousseau

November 2016

CIRRELT-2016-62

Bureaux de Montréal :
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

A Branch-and-Check Approach to Solve on Onshore Wind Turbine Maintenance Scheduling Problem

Aurélien Froger[1,2,*], Michel Gendreau[1], Jorge E. Mendoza[3], Éric Pinson[2], Louis-Martin Rousseau[1]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) and Department of Mathematics and Industrial Engineering, Polytechnique Montréal, P.O. Box 6079, Station Centre-Ville, Montréal, Canada H3C 3A7

[2] L'UNAM, Université Catholique de l'Ouest, LARIS EA 7315, Angers,France

[3] Université François-Rabelais de Tours, CNRS, LI EA 6300, OC ERL CNRS 6305, Tours, France

Abstract. In this report we deal with a maintenance scheduling problem rising in the onshore wind power industry. We address the problem on a short-term horizon considering a multi-skilled workforce. The objective is to schedule the maintenance operations in order to maximize the energy production while taking into account wind predictions, multiple task execution modes, and daily restrictions on the routes of the technicians. We first introduce two integer linear programming formulations of the problem. Then, building on top of one of our models, we propose a branch-and-check (B&C) approach that exploits both generic Benders cuts and cuts specially crafted for our problem. We report computational experiments on a 160-instance testbed proposed on an earlier article. For 80% of the instances, our exact approach finds an optimal solution in short execution times. For the remaining instances where the 3-hour time limit is reached, our B&C delivers solutions with average gaps of 1.7% with respect to upper bounds. The results suggest that our method significantly outperforms not only commercial solvers running our integer linear programming models but also an existing metaheuristic for the problem.

Keywords. Scheduling, maintenance, Benders decomposition, branch-and-check, cuts.

Acknowledgements. This work was supported by Angers Loire Métropole through its research grant program; and by the Natural Sciences and Engineering Research Council of Canada (NSERC) through a grant that enabled the collaboration with the Canadian company WPred, which we would like to thank for their expertise. This support is gratefully acknowledged.

---

* Corresponding author: Aurélien.Froger@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2016

Introduction

As the energy sector is facing major challenges to produce low-carbon power or carbon-free electricity, the share of renewable energies has significantly increased in recent years. Boosted by climate change mitigation and adaptation efforts (e.g., tax incentives, Paris climate change agreement) and the constantly-decreasing cost of turbines, wind energy is currently the world's fastest-growing source of electricity (63 GW of new wind power capacity in 2015), accounting nowadays for around 3.3% of the world electricity production[1].

Although wind turbines availability factor tops 95%[2], their capacity factor[3] is usually around 30-40% as a result of the intermittency of the wind and of design decisions (for a fixed wind speed, the larger are the blades the more electricity the turbine can produce). The impact of operational decisions on this value is also non negligible. As the wind industry is steadily growing, reliability and profitability of wind farms naturally becomes one of the priorities of the sector. In this context, developing optimization techniques to efficiently schedule wind turbine maintenance operations is essential to prevent unnecessary downtimes and excessive operational costs.

Maintenance planning and scheduling has been widely studied in the electricity industry. Most, or all, the contributions of Operations Research target thermal power plants and for the purpose of brevity, we refer the reader to Froger et al. (2016a) for a comprehensive review. The solution methods are inapplicable to the wind power industry where maintenance decisions are very specific. The consequence of shutting down an equipment on the power production depends on an uncontrollable factor, the weather, and this latter has a direct impact, due to safety concerns, on the possible concrete realization of the maintenance operations. Concerning the wind energy sector, studies related to maintenance optimization are primarily focused on reliability-centered maintenance (RCM). RCM is a methodology incorporating reactive, preventive and condition-based maintenance decisions. These decisions, essentially taken under financial considerations, are mostly based on monitoring and on the use of failure prediction models. Studies usually focused on a single turbine or a single wind farm and answer questions such as why failures happens, what should be done when it happens, how to predict or prevent each failure in order to determine the most effective maintenance approach. We refer the reader to Ding et al. (2013) for a survey since this topic is beyond the scope of this work. Using the results of these studies as valuable inputs to define the preventive maintenance operations that need to be perform in the short term and considering a fine-grained resource management result in building detailed maintenance plans that can be used on a daily or weekly basis. It also provides more accurate estimates of turbine downtimes and loss of production. Indeed, producing a maintenance plan in which no operations generate a loss of production (e.g., is scheduled during time periods where the wind speed is below 3.5 m.s^{-1}, which is too low to produce electricity) can almost never be achieved in practice, since human resources are a major bottleneck.

To our knowledge, only few studies have addressed this problem. Kovács et al. (2011) considered fine-grained resource management while scheduling maintenance operations on wind turbines on a one-day horizon. These authors aimed to minimize lost production due to maintenance and failures. They solved an integer linear programming (ILP) formulation of the problem with a commercial solver. With regard to offshore wind farms, Irawan et al. (2016) optimized a maintenance routing and scheduling problem minimizing labor, travel and penalty costs. They proposed a solution method based on Dantzig-Wolfe decomposition in which all the feasible routes for each vessel are generated a priori.

In this paper, we consider the wind turbine maintenance scheduling problem introduced in (Froger et al. 2016b). The problem – focusing on onshore wind farms – is to provide a maintenance plan on a short term-horizon that maximizes the wind electricity production while taking into account a fined-grained resource management involving task assignments to a multi-skilled workforce. To tackle this problem, Froger et al. (2016b) introduced several models based on both ILP and constraint programming (CP). They found that

[1]The Global Wind Energy Council - Global wind report annual market update 2015 - http://www.gwec.net/wp-content/uploads/vip/GWEC-Global-Wind-2015-Report_Avril-2016_22_04.pdf, last accessed: 2016-09-15

[2]the percentage of the time that the wind turbine is available to provide energy to the grid (mostly related to the downtime due to unexpected breakdowns and maintenance)

[3]the ratio of the net electricity generated, for the time considered, to the energy that could have been generated at continuous full-power operation during the same time period

the direct use of these models is unsuitable as the computational time grows prohibitively with the problem size; therefore they designed a CP-based large neighborhood search approach (CPLNS) as a solution method.

Another way to address this complex combinatorial problem may come from decomposition techniques that allow to decouple a large scale problem into several problems that are easier to solve. Based on this idea, the primary contribution of this paper is to propose an efficient exact approach for solving this wind turbine maintenance scheduling problem. The problem is decomposed into a task scheduling problem and a technician-to-task assignment sub-problem, and solved using a branch-and-check approach. More specifically, while solving the task scheduling problem, we discard, by means of cuts all along the branch-and-bound tree, maintenance plans that cannot be performed by the technicians. In addition to the generic Benders cuts, we introduce problem-specific cuts and demonstrate they are key to speed up the convergence of the approach.

The remainder of this paper is organized as follows. In Section 1 we describe the problem. In Section 2 we introduce new ILP formulations of the problem. In Section 3 we present a branch-and-check approach as an exact solution method. In Section 4 we report and discuss computational experiments on the 160-instance testbed proposed by (Froger et al. 2016b). Finally in Section 5 we present our conclusions and we outline research perspectives.

1. Problem statement

The aim of the problem is to schedule a set $\mathcal{I}$ of maintenance tasks during a discrete and finite planning horizon $\mathcal{T}$ while maximizing the revenue from the electricity production of a set $\mathcal{W}$ of wind turbines.

The wind turbines are geographically spread across a set of locations $\mathcal{L}$ (consisting primarily of wind farms and possibly home depots). We denote $l_w \in \mathcal{L}$ the location of wind turbine $w \in \mathcal{W}$ and l_i the location where task $i \in \mathcal{I}$ has to be performed.

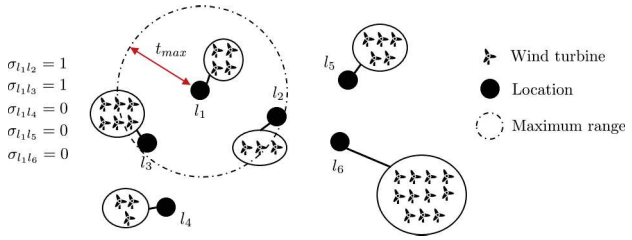
The time horizon is a totally ordered set partitioned into $|\mathcal{T}|$ time periods of identical length. $\mathcal{T}$ spans over several days from a set $\mathcal{D}$. We denote $\mathcal{T}_d$ the time periods that belongs to day $d \in \mathcal{D}$. Moreover, since the execution of a task can impact the production during non-working hours, we introduce a special time period (hereafter referred to as a *rest time period*) between two consecutive days to represent, for example, a night or a weekend. Maintenance tasks are non-preemptive, but, obviously, they are interrupted during rest time periods if they overlap different consecutive days (e.g., a technician can start a task at the end of one day and complete it at the beginning of the next day).

Although we do not include rest time periods in $\mathcal{T}$, we count in the objective function the loss of production generated by tasks overlapping these specific time periods. In more detail, we define the impact of the tasks on the availability of the turbines with two parameters. First, binary parameter b_{wi} takes the value 1 if and only if task $i \in \mathcal{I}$ shuts down turbine $w \in \mathcal{W}$ when technicians are effectively working on the task. Second, binary parameter $\widetilde{b}_{wi}$ takes the value 1 if and only if task i additionally shuts down turbine w during the rest time periods it overlaps. It must be noted that parameters b_{wi} and $\widetilde{b}_{wi}$ are equal to 0 if turbine w is not located at the location where the task i has to be performed (i.e., if $l_i \neq l_w$). Even if it is rather rare in practice, it is noteworthy that a maintenance task can shut down more than one turbine in a wind farm.

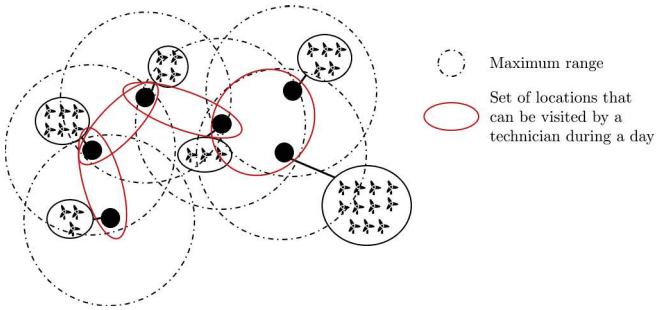
To execute the maintenance tasks, we have a finite set $\mathcal{R}$ of technicians. Each technician masters one or multiple skills from a set $\mathcal{S}$. We express technician skills by a binary vector λ_r over $\mathcal{S}$ such that $\lambda_{rs} = 1$ if and only if technician r masters skill $s \in \mathcal{S}$. We consider that a technician cannot perform more than one task during a given time period. Each task $i \in \mathcal{I}$ requires technicians with a specific skill $s_i \in \mathcal{S}$. For convenience, we define as $\mathcal{R}_i$ the set of technicians that can perform task i (i.e., $\mathcal{R}_i = \{r \in \mathcal{R} \mid \lambda_{rs_i} = 1\}$.

To avoid expensive travel times and save valuable time, we constraint technicians to work during a single day on tasks at *compatible locations*. Compatible locations are simply those that can be reached from each other in travel times that are negligible with respect to the duration of a time period in $\mathcal{T}$. Let us assume that t_{max} is the maximum travel time between two locations that we can consider "negligible" with respect to the duration of a time period. The top of Figure 1 then shows the locations that are compatible with l_1 (i.e., l_2 and l_3). To model these *daily location-based incompatibilities*, we introduce binary parameter $\sigma_{ll'}$

taking the value of 1 if and only if locations l and l' are compatible (naturally $\sigma_{ll'} = \sigma_{l'l}$). The bottom of Figure 1 shows the 4 sets of compatible locations in our example. During a single day, one should observe that a technician can only execute tasks at l_1 and l_2 or l_3 but not both. It is worth mentioning that wind turbine maintenance tasks usually span along hours (if not days), and therefore technicians tend to travel between very few locations during a single working day.



(a) Compatible locations with l_1



(b) Sets of compatible locations

Figure 1: Illustration of the daily location-based incompatibilities

Each technician $r \in \mathcal{R}$ has also an individual availability schedule expressed by a binary vector π_r, with $\pi_r^t = 1$ if and only if r is available during time period $t \in \mathcal{T}$. The availability schedule of every technician is related to training times, personal holiday times, and assignments to tasks (not part of the optimization) that have been already started or that are performed along with external companies. When the technician r is not available during a time period t, his or her location is fixed to $l_r^t \in \mathcal{L}$. For technician personal holidays and training sessions, this parameter is set to a dummy location l^* such that $\forall l \in \mathcal{L}, \sigma_{l^*l} = 1$. Notice that we assume that all the technicians work the same shift, which is a common practice in this industry.

Multiple execution modes are available for each task. For each execution mode of a task, there are an associated task duration and a number of required technicians. It is noteworthy that switching modes after starting the execution of a task is forbidden. Moreover, an important feature of the problem is that a technician assigned to a task has to work on it from the beginning to the end, even if the task overlaps one or multiple rest time periods.

Tasks can only be executed during some specific time periods. These take into account spare parts availability, safety work conditions (e.g., a technician cannot perform certain tasks on a turbine when the wind is too strong), and external restrictions imposed by the operator and/or the owner of wind farms. We also impose non-overlapping constraints on each subset of tasks that belongs to the set $ov(\mathcal{I})$.

The objective of the problem is to determine a schedule that maximizes the revenue generated by the electricity production of the wind farms while meeting the constraints previously described. We denote as g_w^t the value of the revenue generated by wind turbine w if it can produce electricity during time period $t \in \mathcal{T}$. Similarly, we denote as $\widetilde{g}_w^d$ the revenue generated by wind turbine w if it can produce electricity during the rest time period following day $d \in \mathcal{D}$. These revenue are estimated according to the forecasted wind speed.

One particularity of this problem is the possibility to postpone the scheduling of some tasks until the

next planning horizon. When task i is postponed, we apply to the objective a penalty of $o_i \geqslant 0$. The value of this penalty is fixed according to the relative degree of priority of the tasks. This priority depends on reliability consideration (the more a maintenance operation is delayed, the higher is the probability of failure) and contract commitments. If a task is postponed, it obviously does not impact the production of any wind turbines, and thus the value of the revenue. Therefore, if a task needs to be scheduled during the time horizon, this penalty has to be fixed in connection to the revenue in order to ensure that the postponement of this task is non-profitable. This penalty includes an estimation of the loss of revenue induced by the schedule of the corresponding task, to which may be added outsourcing costs (the decision maker then being responsible for the choice of outsourcing a task rather than postponing it). If the penalties are high enough, postponing a task is just triggered to overcome a possible lack of technicians. In short, the objective function to be maximized in the problem always corresponds to the difference between the revenue and the postponing penalties.

It is rather direct to note that the wind turbine maintenance scheduling problem (WTMSP) previously described includes various central scheduling problems as particular cases. Before analyzing more formally its complexity, we define the decision problem WTMSP^{dec} associated with WTMSP. In this problem, a parameter $G \in \mathbb{R}$ is given as a lower bound on the value of the objective function computed as the difference between the revenue and the penalties incurred due to postponed tasks. The problem WTMSP^{dec} is to decide whether there exists a schedule of the tasks such that the objective value is greater than or equal to G. By polynomially reducing the cumulative scheduling problem[4] – known to be NP-complete in the strong sense (Baptiste et al. 1999) – to WTMSP^{dec}, we easily prove that WTMSP is strongly NP-hard. In some particular cases, notice that solving WTMSP is trivial. For instance, if the postponement penalties are all equal to 0, the solution consisting in delaying all the tasks is always optimal. A rather similar observation is that each task having a postponement penalty equal to 0 can be set to be delayed, without affecting the value of the optimal solution, prior to the optimization.

2. Integer linear programming formulations

In this section, we propose an ILP model that will serve as a baseline for the exact approach. This formulation is broadly inspired by the model introduced in (Froger et al. 2016b). We then propose an alternative formulation of the problem to have a more relevant basis for comparison.

2.1. Baseline formulation

The baseline formulation relies on the prior generation for each task of all the possible combinations – according to the time periods during which it can be executed– of a starting time period and an execution mode. Such a combination is called a *plan*. Since the number of time periods is assumed to be short and there are only a few execution modes, the total number of plans is not so large. We denote $\mathcal{P}$ as the set of plans, i_p as the task involved in plan $p \in \mathcal{P}$, and conversely $\mathcal{P}_i$ as the set of all plans involving task i (i.e., $\mathcal{P}_i = \{p \in \mathcal{P} | i_p = i\}$). For each task i, we also create a plan $p_i^0 \in \mathcal{P}_i$ representing the postponement of the task. For a plan p, execution time periods of i_p are expressed by a binary vector a_p over $\mathcal{T}$ such that $a_p^t = 1$ if and only if i_p is executed during time period $t \in \mathcal{T}$. We also denote S_p and C_p the starting and completion time periods of plan p (i.e., $S_p = \min_{t \in \mathcal{T}} a_p^t t$ and $C_p = \max_{t \in \mathcal{T}} a_p^t t$). Similarly, we introduce another binary vector $\widetilde{a}_p$ over $\mathcal{D}$ such that $\widetilde{a}_p^d = 1$ if and only if i_p overlaps the rest time period following day $d \in \mathcal{D}$. We also denote $\mathcal{D}_p$ as the set of days overlapped by the plan p. For convenience and with a slight abuse of notation, we introduce parameters s_p, l_p, b_{wp}, and $\widetilde{b}_{wp}$ equal to s_{i_p}, l_{i_p}, b_{wi_p}, and $\widetilde{b}_{wi_p}$ respectively. Moreover, we denote $\mathcal{R}_p$ the set of technicians that can be assigned to plan p. More specifically, $\mathcal{R}_p$ contains all technicians $r \in \mathcal{R}_{i_p}$ such that for every time period t we have $\pi_r^t \geqslant a_p^t$ and for every day $d \in \mathcal{D}_p$ and every time period $t \in \mathcal{T}_d$, we have $\pi_r^t = 1$ or both $\pi_r^t = 0$ and $\sigma_{l_p l_r^t} = 1$. We also define q_p as the number

[4] A instance of the cumulative scheduling problem consists of a single resource with a given capacity C and a set $\mathcal{J}$ of n jobs where each job $j \in \mathcal{J}$ has a release date r_j, a deadline d_j, a processing time p_j and a capacity resource requirement a_j. The problem is to decide whether there exists a schedule of all the jobs satisfying the timing and the resource capacity constraints.

of technicians required to execute plan p. Finally, parameter o_p is the penalty incurred if plan p is selected (note that $\forall i \in \mathcal{I}, \forall p \in \mathcal{P}_i \backslash \{p_i^0\},\ o_p = 0$ and $o_{p_i^0} = o_i$).

The integer linear formulation relies on several types of decision variables. Binary variable x_p is equal to 1 if and only if plan $p \in \mathcal{P}$ is selected. Regarding the availability of the wind turbines, binary variable f_w^t is equal to 1 if and only if wind turbine $w \in \mathcal{W}$ can produce energy during time period $t \in \mathcal{T}$, and binary variable $\widetilde{f}_w^d$ is equal to 1 if and only if turbine w can produce energy during the rest time period following day $d \in \mathcal{D}$.

The fundamental difference between our model and that of Froger et al. (2016b) is that our model does not assign technicians to plans. Clearly, technicians can be indifferently assigned to plans sharing the same starting and completion time periods, the same location and the same skill. These previous four parameters define what we call a *pattern*. Directly assigning technicians to patterns results in a smallest number of assignment variables. We denote $\mathcal{H}$ and h_p respectively the set of all patterns and the pattern linked to plan $p \in \mathcal{P}$. We also define as $\mathcal{P}_h$ the set of plans sharing the same parameters than pattern $h \in \mathcal{H}$. For convenience, for a pattern h, we introduce parameters s_h, l_h and $\mathcal{R}_h$ that respectively define: its required skill, its location, and the set of technicians that can be assigned to it. Conversely, we define $\mathcal{H}_l = \{h \in \mathcal{H} \mid l_h = l\}$ the set of patterns associated with location $l \in \mathcal{L}$. For a pattern h, we express its active time periods by a binary vector a_h over $\mathcal{T}$ such that $a_h^t = 1$ if and only if h is active during time period $t \in \mathcal{T}$ (S_h and C_h are used to represent the starting and completion time periods of the pattern). Subsequently, for each pattern $h \in \mathcal{H}$ and each technician $r \in \mathcal{R}_h$, we introduce binary variable y_{rh} that is equal to 1 if and only if technician r is assigned to pattern h. Lastly, we define binary variable v_{rl}^t that is equal to 1 if and only if technician r is at location l during time period t.

The baseline formulation is then defined as the following integer linear program denoted as $[P_1]$.

$$[P_1] \quad \max \sum_{w \in \mathcal{W}} \left(\sum_{t \in \mathcal{T}} g_w^t f_w^t + \sum_{d \in \mathcal{D}} \widetilde{g}_w^d \widetilde{f}_w^d \right) - \sum_{p \in \mathcal{P}} o_p x_p \tag{1}$$

subject to:

$$\sum_{p \in \mathcal{P}_i} x_p = 1 \quad \forall i \in \mathcal{I}, \tag{2}$$

$$\sum_{i \in \mathcal{B}} \sum_{p \in \mathcal{P}_i} a_p^t x_p \leqslant 1 \quad \forall \mathcal{B} \in ov\left(\mathcal{I}\right), \forall t \in \mathcal{T}, \tag{3}$$

$$f_w^t + \sum_{p \in \mathcal{P}_i} b_{wp} a_p^t x_p \leqslant 1 \quad \forall w \in \mathcal{W}, \forall i \in \mathcal{I}, \forall t \in \mathcal{T}, \tag{4}$$

$$\widetilde{f}_w^d + \sum_{p \in \mathcal{P}_i} \widetilde{b}_{wp} \widetilde{a}_p^d x_p \leqslant 1 \quad \forall w \in \mathcal{W}, \forall i \in \mathcal{I}, \forall d \in \mathcal{D}, \tag{5}$$

$$\sum_{i \in \mathcal{I} | s_i \in \bar{\mathcal{S}}} \sum_{p \in \mathcal{P}_i} a_p^t q_p x_p \leqslant |R_{\bar{\mathcal{S}}}^t| \quad \forall t \in \mathcal{T}, \forall \bar{\mathcal{S}} \subset \mathcal{S}, \tag{6}$$

$$\sum_{r \in \mathcal{R}_h} y_{rh} = \sum_{p \in P_h} q_p x_p \quad \forall h \in \mathcal{H}, \tag{7}$$

$$\sum_{h \in \mathcal{H}_l} a_h^t y_{rh} \leqslant \pi_r^t v_{rl}^t \quad \forall r \in \mathcal{R}, \forall l \in \mathcal{L}, \forall t \in \mathcal{T}, \tag{8}$$

$$\sum_{l \in \mathcal{L}} v_{rl}^t = 1 \quad \forall r \in \mathcal{R}, \forall t \in \mathcal{T}, \tag{9}$$

$$v_{rl_r^t}^t = 1 \quad \forall r \in \mathcal{R}, \forall t \in \mathcal{T} s.t. \pi_r^t = 0, \tag{10}$$

$$v_{rl}^t + \sum_{l' \in \mathcal{L} | \sigma_{ll'} = 0} v_{rl'}^{t'} \leqslant 1 \quad \forall r \in \mathcal{R}, \forall d \in \mathcal{D}, \forall (t, t') \in \mathcal{T}_d \times \mathcal{T}_d, t \neq t', \forall l \in \mathcal{L}, \tag{11}$$

$$x_p \in \{0, 1\} \quad \forall p \in \mathcal{P}, \tag{12}$$

$$f_w^t \in \{0, 1\} \quad \forall w \in \mathcal{W}, \forall t \in \mathcal{T}, \tag{13}$$

$$\widetilde{f}_w^d \in \{0,1\} \quad \forall w \in \mathcal{W}, \forall d \in \mathcal{D}, \tag{14}$$

$$y_{rh} \in \{0,1\} \ \forall h \in \mathcal{H}, \forall r \in \mathcal{R}_h \tag{15}$$

$$v_{rl}^t \in \{0,1\} \quad \forall r \in \mathcal{R}, \forall l \in \mathcal{L}, \forall t \in \mathcal{T}. \tag{16}$$

The objective in (1) is defined as the difference between the revenue generated by the wind turbines and the penalties related to the postponement of some tasks. Constraints (2) ensure that at least one plan involving each task is selected (i.e., each task is either executed or postponed). Constraints (3) are the non-overlapping constraints. Constraints (4) and (5) compute the impact of the tasks on the availability of the turbines to produce electricity. Constraints (7) ensure that the technician requirements are fulfilled. Constraints (8) couple the technicians location to the tasks they perform. Constraints (9) prevent technicians to perform multiple tasks during the same time period. When technicians are not available, constraints (10) ensure compliance with their known locations. Constraints (11) define the incompatibilities between the locations a technician can visit within a day. Finally, constraints (12)-(16) state the binary nature of the decision variables.

To strengthen the formulation $[P_1]$, we also add the cumulative scheduling constraints (6), even if they are redundant since they can be deduced from constraints (7) and (8). To build these constraints, we introduce for each time period $t \in \mathcal{T}$ the bipartite graph $\mathcal{G}^t = ((\mathcal{S}, \mathcal{R}^t), \mathcal{U}^t)$ in which, with a slight abuse of notation, vertices from $\mathcal{S}$ represent skills, vertices from $\mathcal{R}^t$ indicate technicians available during time period t (i.e., $\mathcal{R}^t = \{r \in \mathcal{R} | \pi_r^t = 1\}$), and edges from $\mathcal{U}^t$ are defined as follows: $\forall s \in \mathcal{S}, \forall r \in \mathcal{R}^t \ \ (s,r) \in \mathcal{U}^t$ if and only if $\lambda_{rs} = 1$. Applying a generalization of König-Hall theorem, the constraints (6) thus correspond to necessary and sufficient condition of the existence of a maximum cardinality b-matching from $\mathcal{S}$ to $\mathcal{R}^t$ where function b is defined for every vertex s that belongs to $\mathcal{S}$ by $b(s) = \sum_{i \in \mathcal{I} | s_i = s} \sum_{p \in \mathcal{P}_i} a_p^t q_p x_p$, and by $b(r) = 1$ for every vertex r that belongs to $\mathcal{R}^t$. To express these constraints, we denote $\mathcal{R}_{\bar{\mathcal{S}}}^t = \{r \in \mathcal{R} | \exists s \in \bar{\mathcal{S}}, \lambda_{rs} = 1 \wedge \pi_r^t = 1)\}$ the set of technicians available during time period t and mastering at least one skill in subset $\bar{\mathcal{S}} \subset \mathcal{S}$. Last but not least, the number of these constraints are exponential $\left(2^{|\mathcal{S}|} - 1\right) \times |\mathcal{T}|)$. In our experiments, however, the number of these constraints tends to be small; we therefore add all these constraints to our model.

2.2. Alternative formulation

A potential improvement of the previous model concerns the space-time tracking of the technicians. Observing that the number of constraints (11) is usually very large, we attempt to develop an alternative technician management strategy. This new strategy is based on finding all the maximal cliques (cliques that cannot be enlarged) in a graph where each vertex represents a location, and there exists an edge between two vertices if the underlying locations l and l' can be visited during the same day by the same technician (i.e., we have $\sigma_{ll'} = 1$). Figure 2 illustrates the computation of these maximal cliques. For the purpose of finding all the maximal cliques, we use the algorithm introduced by Bron and Kerbosch (1973).

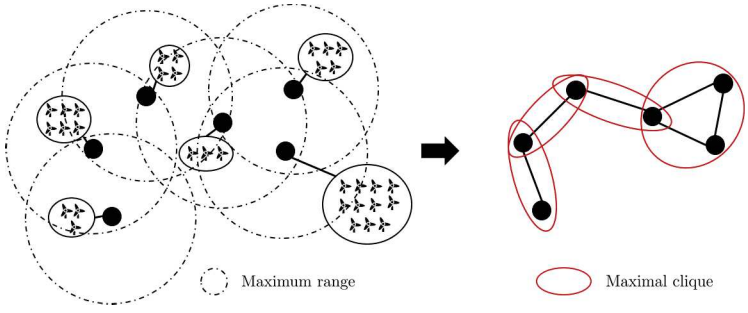


Figure 2: Example of how maximal cliques are computed

Denoting $\mathcal{K}$ the set of all maximal cliques in the previous graph, we define $\mathcal{K}_r^d$ as the set of cliques to which technician r can be assigned during day d. More specifically, $\mathcal{K}_r^d$ is equal to $\{k \in \mathcal{K} \mid \forall t \in \mathcal{T}_d, \pi_r^t = 1 \vee (\pi_r^t = 0 \wedge l_r^t \in k)\}$. We then introduce binary variable u_{rk}^d that takes value 1 if and only if during day

$d \in \mathcal{D}$ technician $r \in \mathcal{R}$ can only performed tasks at locations included in clique $k \in \mathcal{K}_r^d$. Denoting d_t as the day to which time period t belongs, we can track the location of the technicians with the following constraints:

$$\sum_{k \in \mathcal{K}_r^d} u_{rk}^d = 1 \quad \forall r \in \mathcal{R}, \forall d \in \mathcal{D}, \tag{17}$$

$$\sum_{h \in \mathcal{H} | r \in \mathcal{R}_h} a_h^t y_{rh} \leqslant \pi_r^t \quad \forall r \in \mathcal{R}, \forall t \in \mathcal{T}, \tag{18}$$

$$\sum_{h \in \mathcal{H}_l} a_h^t y_{rh} \leqslant \sum_{k \in \mathcal{K}_r^{d_t} | l \in k} u_{rk}^{d_t} \quad \forall r \in \mathcal{R}, \forall l \in \mathcal{L}, \forall t \in \mathcal{T}, \tag{19}$$

$$u_{rk}^d \in \{0,1\} \quad \forall r \in \mathcal{R}, \forall d \in \mathcal{D}, \forall k \in k \in \mathcal{K}_r^d \tag{20}$$

Constraints (17) state that a technician is assigned to only one clique on each day. This ensures the compliance with the location-based incompatibilities. Constraints (18) prevent a technician to be assigned to multiple tasks during a given time period and constraints (19) couple assignment and space-time tracking variables.

For the sake of clarity, we define as $[P_2]$ the model resulting from replacing constraints (8), (9), (10), (11), and (16) by constraints (17), (18), (19), and (20) in the baseline formulation $[P_1]$.

3. An exact solution approach

3.1. Problem decomposition

The exact approach presented in this section takes advantage of the intrinsic decomposition of the problem into a task scheduling problem and a technician-to-task assignment sub-problem. The task scheduling problem consists in selecting a plan for each task in order to maximize the revenue from the wind electricity production. In this problem, technician considerations have been partially dropped. If we assume a fixed selection of plans, the technician-to-task assignment sub-problem (hereafter occasionally referred to simply as the sub-problem) checks if the technicians requests can be satisfied while meeting the daily location-based incompatibilities and coping with individual resource availability time periods. The aim of our approach is thus to design a coordination procedure between these two problems. Note that an optimal solution to the scheduling problem leading to a feasible technician-to-task assignment sub-problem is optimal for the whole problem.

First, let us introduce the scheduling problem. An initial formulation $[ShP_1]$ of this problem reads:

$$[ShP_1] \quad \max \sum_{w \in \mathcal{W}} \left(\sum_{t \in \mathcal{T}} g_w^t f_w^t + \sum_{d \in \mathcal{D}} \widetilde{g}_w^d \widetilde{f}_w^d \right) - \sum_{p \in \mathcal{P}} o_p x_p \tag{21}$$

subject to:

(2), (3), (4), (5), (6), (12), (13), (14)

Let us now assume a fixed selection $(\bar{x}_p)_{p \in \mathcal{P}}$ of plans (hereafter referred to simply as $\bar{x}$) solution to $[ShP_1]$. An ILP formulation $[SP_2(\bar{x})]$ of the technician-to-task assignment sub-problem reads:

$$[SP_2(\bar{x})] \quad \min \sum_{h \in \mathcal{H}} \theta_h \tag{22}$$

subject to:

$$\sum_{r \in \mathcal{R}_h} y_{rh} + \theta_h = \sum_{p \in P_h} q_p \bar{x}_p \quad \forall h \in \mathcal{H}, \tag{23}$$

$$\sum_{\substack{h \in H \\ s.t.\ r \in \mathcal{R}_h}} y_{rh} \leqslant 1 \quad \forall r \in \mathcal{R}, \forall H \in \mathcal{C}^{max}(G), \tag{24}$$

$$\theta_h \geqslant 0 \quad \forall h \in \mathcal{H}, \tag{25}$$

$$y_{rh} \in \{0,1\} \ \forall h \in \mathcal{H}, \forall r \in \mathcal{R}_h \tag{26}$$

We introduce slack variables $(\theta_h)_{h\in\mathcal{H}}$ for the technicians requirement constraints (23). Notice that the unavailability time periods of each technician are respected by definition of the set $\mathcal{R}_h$. The clique constraints (24) ensure that the technician-task assignments comply with the daily location-based incompatibilities. More specifically, set $\mathcal{C}^{max}(G)$ contains all the maximal cliques in a graph G where each vertex represents a pattern in $\mathcal{H}$, and there exists an edge between two vertices if the underlying patterns h and h' cannot be visited by the same technician. More precisely, this edge exists if the following clause holds:

$$(S_{h'} \leqslant C_h \ \wedge \ S_h \leqslant C_{h'}) \ \vee \ \big((\exists d \in \mathcal{D}, \mathcal{T}_d \cap \{S_h, ..., C_h\} \neq \varnothing \wedge \mathcal{T}_d \cap \{S_{h'}, ..., C_{h'}\} \neq \varnothing) \wedge \sigma_{l_h l_{h'}} = 0\big) \tag{27}$$

One can visualize graph G as an extended version of an interval graph. To define the sub-problem for a given solution $\bar{x}$ (and therefore to compute all the maximal cliques), we only need to consider the patterns of set $\mathcal{H}(\bar{x}) = \{h \in \mathcal{H} \mid \sum_{p\in P_h} q_p \bar{x}_p > 0\}$. This allows to significantly reduce the number of variables in $[SP_2(\bar{x})]$ and the number of clique constraints (24) since we consider a sub-graph of G.

It is worth noting that the technician-to-task assignment sub-problem is NP-complete. This can be proved by its equivalence to a L-coloring problem. Nonetheless, under certain assumptions, the sub-problem becomes solvable in polynomial time. For the sake of clarity, we do not present any details here. We refer the reader to Appendix A for a thorough discussion on the complexity of the sub-problem.

Notice also that the cumulative scheduling constraints (6) in $[ShP_1]$ help to speed-up the convergence of a coordination procedure between the two problems. Indeed, solving the task scheduling problem without any information regarding the availability of the technicians would result in selection of plans that would unlikely lead to a feasible technician-to-task assignment sub-problem.

It is fairly easy to observe that $[SP_2(\bar{x})]$ always admits a feasible solution thanks to the slack variables $(\theta_h)_{h\in\mathcal{H}}$. However, we can conclude that the technician-to-task assignment sub-problem is feasible only if the value of the optimal solution is equal to zero.

Let $[SP_2^{LR}(\bar{x})]$ be the linear relaxation of formulation $[SP_2(\bar{x})]$. Constraints (26) of $[SP_2(\bar{x})]$ are substituted in $[SP_2^{LR}(\bar{x})]$ by the following constraints:

$$y_{rh} \leqslant 1 \quad \forall h \in \mathcal{H}, \forall r \in \mathcal{R}_h, \tag{28}$$

$$y_{rh} \geqslant 0 \quad \forall h \in \mathcal{H}, \forall r \in \mathcal{R}_h, \tag{29}$$

Because the constraint matrix of $[SP_2^{LR}(\bar{x})]$ is not totally unimodular, integrity constraints (26) on variables y_{rh} cannot be relaxed while ensuring they will be satisfied by any optimal solution to $[SP_2(\bar{x})]$.

When the cost of a solution to $[SP_2(\bar{x})]$ is strictly positive (i.e., the technician-to-task assignment sub-problem is infeasible), we therefore use a *combinatorial Benders cut* as introduced in (Codato and Fischetti 2006) to invalidate the current solution $\bar{x}$ in the scheduling problem $[ShP_1]$. Note that these cuts are similar to the cuts defined in the Logic-based Benders decomposition by Hooker and Ottosson (2003) and share some similarities with the integer optimality cuts introduced by Laporte and Louveaux (1993) for the integer L-shaped method. Using the binary variables $(x_p)_{p\in\mathcal{P}}$, we can define such a cut as follows:

$$\sum_{p\in\mathcal{P}|\bar{x}_p=0} x_p + \sum_{p\in\mathcal{P}|\bar{x}_p=1} (1 - x_p) \geqslant 1 \tag{30}$$

Clearly, this cut states that, in the next iteration, at least one of the variables related to the selection of plans in the scheduling problem $[ShP_1]$ must change its value with respect to $\bar{x}$. This cut is also known as a *no-good* cut. Observing that a solution contains always $|\mathcal{I}|$ non-zero variables x_p since exactly one plan has to be selected per task, we can then replace inequality (30) by the following cover inequality (denoting

$\mathcal{P}(\bar{x}) = \{p \in \mathcal{P} | \bar{x}_p = 1\}$):

$$\sum_{p \in \mathcal{P}(\bar{x})} x_p \leqslant |\mathcal{I}| - 1, \tag{31}$$

Hereafter, we refer to the combinatorial Benders cuts (31) as CB cuts.

Since the feasible region of our problem is bounded, the number of integer points satisfying all the constraints of $[ShP_1]$ is finite and, thus, the same holds for the number of combinatorial Benders' cuts. Let us denote $\bar{\mathcal{F}}$ the set of all solutions $\bar{x}$ to $[ShP_1]$ that lead to an infeasible technician-to-task assignment sub-problem. The whole maintenance scheduling problem can therefore be reformulated as the following master problem $[P]$:

$$\begin{aligned}
[P] \quad & \max \sum_{w \in \mathcal{W}} \left(\sum_{t \in \mathcal{T}} g_w^t f_w^t + \sum_{d \in \mathcal{D}} \widetilde{g}_w^d \widetilde{f}_w^d \right) - \sum_{p \in \mathcal{P}} o_p x_p \\
& \text{subject to:} \\
& (2), (3), (4), (5), (6), (12), (13), (14) \\
& \sum_{p \in \mathcal{P}(\bar{x})} x_p \leqslant |\mathcal{I}| - 1 \quad \forall \bar{x} \in \bar{\mathcal{F}} \qquad (31)
\end{aligned}$$

In the remainder of the document, we denote $[RMP]$ a restricted master problem of problem $[P]$ that contains none or only a small subset of constraints (31).

3.2. Cut generation procedure

For every solution $\bar{x}$ to the restricted master problem $[RMP]$, we need to check the feasibility of the technician-to-task assignment sub-problem. We can directly solve $[SP_2(\bar{x})]$ to optimality using a commercial solver. Nevertheless, this approach has two major drawbacks. First, since $[SP_2(\bar{x})]$ is a pure ILP model, solving the model may be too time-consuming. Second, if the cost of the solution is strictly positive, the resulting CB cut (31) may be too weak because it does not identify the causes of the infeasibility of the technician-to-task assignment sub-problem. Indeed, the infeasibility is likely to be caused by only a subset of the selected plans. To overcome these drawbacks and to build up stronger cuts, we propose 3 different cut generation strategies based on different approximations to the sub-problem.

3.2.1. Benders feasibility cuts

First, we can generate cuts based on solving the linear relaxation $[SP_2^{LR}(\bar{x})]$ of the formulation $[SP_2(\bar{x})]$. Since a solution $\bar{x}$ to $[RMP]$ is feasible for the whole problem only if the optimum of $[SP_2^{LR}(\bar{x})]$ is zero, $\bar{x}$ is feasible for the whole problem only if the optimum of the dual $[DSP_2^{LR}(\bar{x})]$ of $[SP_2^{LR}(\bar{x})]$ is less or equal to zero (duality theorem). Let us associate the dual variables ι_h, ϱ_r^H, and φ_{rh} to constraints (23),(24), and (28), respectively. The objective function of $[DSP_2^{LR}(\bar{x})]$, denoted as $\Theta_{\bar{x}}(\iota, \varrho, \varphi)$, reads:

$$\Theta_{\bar{x}}(\iota, \varrho, \varphi) = \sum_{h \in \mathcal{H}} \left(\sum_{p \in \mathcal{P}_h} q_p \bar{x}_p \iota_h + \sum_{r \in \mathcal{R}_h} \varphi_{rh} \right) + \sum_{r \in \mathcal{R}} \sum_{\substack{H \in \mathcal{C}(\mathcal{H}) \\ s.t.\ r \in \bigcup_{h \in H} \mathcal{R}_h}} \varrho_r^H \tag{32}$$

Let D be the polyhedron defined by the constraints of the dual problem $[DSP_2^{LR}(\bar{x})]$. Since $[SP_2^{LR}(\bar{x})]$ always admits a feasible solution, the dual problem $[DSP_2^{LR}(\bar{x})]$ is bounded and achieves its optimum on an extreme point of D. Denoting $\eta^1, \eta^2, ..., \eta^n$ (with $\eta^k = (\iota^k, \varrho^k, \varphi^k)$) the finite set of extreme points of D, by weak duality theorem, the following inequalities must hold to ensure the existence of a zero value solution to $[SP_2^{LR}(x)]$:

$$\Theta_x(\iota^k, \varrho^k, \varphi^k) \leqslant 0 \quad \forall k \in \{1, ..., n\} \tag{33}$$

Constraints (33) are the classical Benders feasibility cuts (hereafter referred to as BF cuts). Since a solution to $[SP_2^{LR}(\bar{x})]$ is not guaranteed to be a solution to $[SP_2(\bar{x})]$, a cut generation algorithm responsible for identifying violated constraints (33) will therefore not, in general, retrieve a feasible solution to $[P]$. Nevertheless, identifying violated BF cuts may help to generate less CB cuts. The advantages are that: i) BF cuts are faster to compute than CB cuts since we only need to solve a continuous linear model and ii) they may discard more solutions than just the current solution to the restricted master problem $[RMP]$. One persistent drawback of BF cuts is that they are generic, and therefore likely to be weak.

The efficiency of a coordination procedure between the two problems relies primarily on finding reduced subsets of plans causing the infeasibility of the technician-to-task assignment sub-problem. In the following subsections, we then describe two different problem-specific procedures to find these reduced subsets, and we show how we build up stronger problem-specific cuts.

First, we introduce the following notation to refer to the constraints of the sub-problem:

[C1] The technician requirements for each task have to be fulfilled by technicians mastering the desired skill.

[C2] A technician cannot perform more than one task during a given time period.

[C3] The technician assignments must not violate the daily location-based incompatibilities.

[C4] Each technician has an availability schedule which must be respected.

[C5] A technician assigned to a task has to work on it from the beginning to the end, even if the task overlaps some rest time periods.

Obviously, the sub-problem is feasible if and only if constraints [C1], [C2], [C3], [C4], and [C5] are all satisfied.

Second, to simplify the discussion, we introduce the concept of *jobs* that, hereafter, simply refer either to patterns or to technician unavailability time periods. We define a job j with the following notation $(l_j, S_j, C_j, s_j, \mathcal{R}_j, q_j)$ where l_j denotes the location where j is executed, S_j its starting time period, C_j its completion time period, $\mathcal{S}_j$ a set of skills such that a technician should master at least one of these skills in order to perform j, $\mathcal{R}_j$ the set of technicians who can perform j, and q_j the number of technicians required for executing job j. For every unavailability time period of a technician $r \in \mathcal{R}$ occurring at a time period t ($t \in \mathcal{T}$ such that $\rho_r^t = 1$), we build an artificial job defined by the vector $(l_r^t, t, t, \mathcal{S}_r, \{r\}, 1)$ where $\mathcal{S}_r = \{s \in \mathcal{S} | \zeta_{rs} = 1\}$ is the set of skills mastered by technician r. If a technician is unavailable during contiguous time periods and if he or she is assigned each time at the same location (with regard to l_r^t), we associate only one job for his or her unavailability time periods. We denote as $\mathcal{J}^{\mathcal{R}}$ the set of jobs associated with technician unavailability time periods. We also associate with each solution $\bar{x}$ to the restricted master problem $[RMP]$ a set $\mathcal{J}(\bar{x})$ of jobs. We build this set following two steps. First, we add to $\mathcal{J}(\bar{x})$ all the jobs of $\mathcal{J}^{\mathcal{R}}$. Then, for every pattern $h \in \mathcal{H}$ such that $q_h(\bar{x}) > 0$ (with $q_h(\bar{x}) = \sum_{p \in \mathcal{P}_h} q_p \bar{x}_p$), we create a job defined by the vector $(l_h, S_h, C_h, \{s_h\}, \mathcal{R}_h, q_h(\bar{x}))$. Hereafter, we denote as $\mathcal{J}^{\mathcal{H}}(\bar{x})$ the set of jobs associated with patterns. We also define the parameter h_j as the pattern associated with job $j \in \mathcal{J}^{\mathcal{H}}(\bar{x})$.

3.2.2. Maximum cardinality b-matching cuts

In this section, we aim to extend the idea used to generate constraints (6) by partially taking into account the constraints [C3] and [C5] while fully taking into account the constraints [C1], [C2], and [C4]. More precisely, when building the potential assignments of the technicians to the tasks, we consider the unavailability time periods of the technicians and the restriction of not switching technicians during the execution of a task. We then show that the sub-problem can be approximated solving a series of maximum cardinality b-matching problems (as many as the length of the time horizon).

First, for a fixed time period $t \in \mathcal{T}$ of the time horizon and for a given solution $\bar{x}$ to the restricted master problem $[RMP]$, we introduce an undirected graph $\breve{G}^t(\bar{x})$ composed of:

- a set of vertices $\breve{\mathcal{V}}^t$ where $\breve{\mathcal{V}}^t = \breve{\mathcal{V}}^t_{\mathcal{J}} \cup \breve{\mathcal{V}}^t_{\mathcal{R}}$

- $\breve{\mathcal{V}}^t_{\mathcal{J}}$: for each job $j \in \mathcal{J}(\bar{x})$ such that $S_j \leqslant t \leqslant C_j$, we add a vertex in $\breve{\mathcal{V}}^t_{\mathcal{J}}$. Parameter j_ν denotes the job associated with a vertex $\nu \in \breve{\mathcal{V}}^t_{\mathcal{J}}$. Conversely ν_j denotes the vertex associated with job j.
- $\breve{\mathcal{V}}^t_{\mathcal{R}}$: a vertex of $\breve{\mathcal{V}}^t_{\mathcal{R}}$ represents a technician $r \in \mathcal{R}$ during time period t. We denote r_v the technician associated with a vertex $\nu \in \breve{\mathcal{V}}^t_{\mathcal{R}}$.

- a set of edges $\breve{\mathcal{U}}^t$ defined such that $\forall \nu_1 \in \breve{\mathcal{V}}^t_{\mathcal{J}}, \forall \nu_2 \in \breve{\mathcal{V}}^t_{\mathcal{R}}$: $(\nu_1, \nu_2) \in \breve{\mathcal{U}}^t \Leftrightarrow r_{\nu_2} \in \mathcal{R}_{j_{\nu_1}}$

We now formally describe in Proposition 3.1 and Corollary 3.1 the link between the technician-to-task assignment sub-problem and a series of maximum cardinality b-matching problems defined in the graphs previously introduced.

Proposition 3.1. *Let $\bar{x}$ be a solution to the restricted master problem $[RMP]$ and assume that constraints [C3] and [C5] are relaxed. The technician-to-task assignment sub-problem for $\bar{x}$ is equivalent to a series of $|\mathcal{T}|$ maximum cardinality b-matching problems in graph $\breve{G}^t(\bar{x})$ where for each time period $t \in \mathcal{T}$ function b is defined by $b_\nu = q_{j_\nu}$ for every vertex $\nu \in \breve{\mathcal{V}}^t_{\mathcal{J}}$ and by $b_\nu = 1$ for every vertex $\nu \in \breve{\mathcal{V}}^t_{\mathcal{R}}$.*

Proof. Notice that since constraints [C3] and [C5] are relaxed, the sub-problem can be independently solved for each time period of the planning horizon.

Assuming that during each time period t we can find a maximum cardinality b-matching in each of the graphs $G^t(\bar{x})$ with a cardinality equal to $\sum_{j \in \mathcal{J}(\bar{x})} \mathbb{1}_{\{S_j \leqslant t \leqslant C_j\}} q_j$, we can immediately build a solution to the technician-to-task assignment sub-problem from the selected edges by making the underlying assignments.

Assume now that we know a feasible solution to the technician-to-task assignment sub-problem. For each time period t, we can build a b-matching in $\breve{G}^t(\bar{x})$ from the working schedule of each technician during this specific time period. If a technician $r \in \mathcal{R}$ is assigned to a pattern $h \in \mathcal{H}$ during time period t, we select the edge $(\nu_1, \nu_2) \in \mathcal{U}$ where $j_{\nu_1} = j_h$ (j_h denoting the job associated with the pattern h) and $r_{\nu_2} = r$. This construction ensures the building of a b-matching. Moreover, since all the requirements are fulfilled, this b-matching has the maximum possible cardinality. □

Corollary 3.1. *If we assume that constraints [C3] and [C5] are relaxed, the technician-to-task assignment sub-problem is feasible for a solution $\bar{x}$ to the restricted master problem $[RMP]$ if and only if the maximum cardinality b-matching in graph $\breve{G}^t(\bar{x})$ for each time period $t \in \mathcal{T}$ contains $\sum_{j \in \mathcal{J}(\bar{x})} \mathbb{1}_{\{S_j \leqslant t \leqslant C_j\}} q_j$ edges of $\breve{\mathcal{U}}^t$.*

Proof. This is a direct consequence of Proposition 3.1.

□

Let us assume a fixed time period t. Solving the maximum cardinality b-matching in $\breve{G}^t(\bar{x})$ from $\breve{\mathcal{V}}^t_{\mathcal{J}}$ to $\breve{\mathcal{V}}^t_{\mathcal{R}}$ is equivalent to solving a maximum flow problem in a slightly modified version of this graph. We use this equivalence to derive new cuts. We denote $\widehat{G}^t(\bar{x})$ this new directed graph and $\widehat{\mathcal{V}}^t$ and $\widehat{\mathcal{U}}^t$ the new sets of vertices and arcs. We build graph $\widehat{G}^t(\bar{x})$ as follows:

1. We define $\widehat{\mathcal{V}}^t$ as $\widehat{\mathcal{V}}^t = \breve{\mathcal{V}}^t \cup \{\overline{\omega}^t, \underline{\omega}^t\}$ where the two new vertices $\overline{\omega}^t$ and $\underline{\omega}^t$ represent the source and the sink vertices.
2. For every directed arc $(\nu_1, \nu_2) \in \widehat{\mathcal{U}}^t$, we denote as $\gamma^{max}_{\nu_1\nu_2}$ its maximal capacity and as $f_{\nu_1\nu_2}$ the number of units of flow on the arc. We formally define $\widehat{\mathcal{U}}^t$ as follows:
   - $\forall \nu_1 \in \breve{\mathcal{V}}^t_{\mathcal{J}}, \forall \nu_2 \in \breve{\mathcal{V}}^t_{\mathcal{R}}$: $(\nu_1, \nu_2) \in \widehat{\mathcal{U}}^t \Leftrightarrow r_{\nu_2} \in \mathcal{R}_{j_{\nu_1}}$ and $\gamma^{max}_{\nu_1\nu_2} = +\infty$
   - $\forall \nu \in \breve{\mathcal{V}}^t_{\mathcal{J}}$: $(\overline{\omega}^t, \nu) \in \widehat{\mathcal{U}}^t$ and $\gamma^{max}_{\overline{\omega}^t\nu} = q_{j_\nu}$
   - $\forall \nu \in \breve{\mathcal{V}}^t_{\mathcal{R}}$: $(\nu, \underline{\omega}^t) \in \widehat{\mathcal{U}}^t$ and $\gamma^{max}_{\nu\underline{\omega}^t} = 1$

Let us denote $f^*(t, \bar{x})$ the value of the maximum flow in $\widehat{G}^t(\bar{x})$. If $f^*(t, \bar{x}) < \sum_{\nu \in \breve{\mathcal{V}}^t_{\mathcal{J}}} q_{j_\nu}$ then the jobs of $\mathcal{J}(\bar{x})$ overlapping t cannot be fully scheduled during this time period. We therefore have to to discard the solution $\bar{x}$. We first compute the minimum flow cut in graph $\widehat{G}^t(\bar{x})$ (see Figure 3). The minimum flow cut

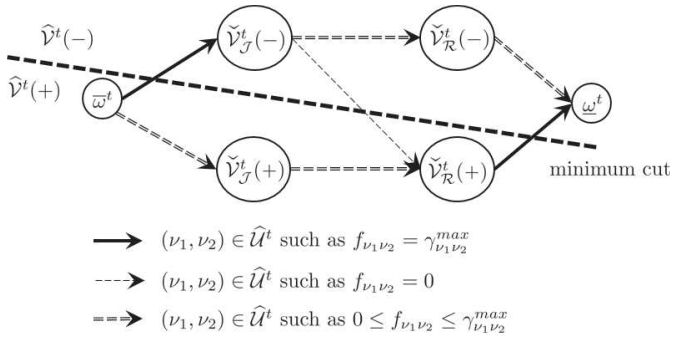


Figure 3: Minimum cut in graph $\widehat{G}^t(\bar{x})$

can be described by the sets $\widehat{\mathcal{V}}^t(+)$ and $\widehat{\mathcal{V}}^t(-)$ that are composed as follows: $\widehat{\mathcal{V}}^t(+) = \{\overline{\omega}^t\} \cup \breve{\mathcal{V}}^t_{\mathcal{J}}(+) \cup \breve{\mathcal{V}}^t_{\mathcal{R}}(+)$ and $\widehat{\mathcal{V}}^t(-) = \breve{\mathcal{V}}^t_{\mathcal{J}}(-) \cup \breve{\mathcal{V}}^t_{\mathcal{R}}(-) \cup \{\underline{\omega}^t\}$ with $\breve{\mathcal{V}}^t_{\mathcal{J}} = \breve{\mathcal{V}}^t_{\mathcal{J}}(-) \cup \breve{\mathcal{V}}^t_{\mathcal{J}}(+)$ and $\breve{\mathcal{V}}^t_{\mathcal{R}} = \breve{\mathcal{V}}^t_{\mathcal{R}}(-) \cup \breve{\mathcal{V}}^t_{\mathcal{R}}(+)$.

Applying the *max-flow/min-cut* theorem on graph $\widehat{G}^t(\bar{x})$, we can state that:

$$f^*(t, \bar{x}) = \sum_{\nu \in \breve{\mathcal{V}}^t_{\mathcal{J}}(-)} \gamma^{max}_{\overline{\omega}^t v} + \sum_{v \in \breve{\mathcal{V}}^t_{\mathcal{R}}(+)} \gamma^{max}_{v\underline{\omega}^t} \tag{34}$$

If we replace the capacity of each arc by its value, we obtain:

$$f^*(t, \bar{x}) = \sum_{\nu \in \breve{\mathcal{V}}^t_{\mathcal{J}}(-)} q_{j_\nu} + |\breve{\mathcal{V}}^t_{\mathcal{R}}(+)| \tag{35}$$

The valid minimum flow cut (that invalidates $\bar{x}$) reads:

$$\sum_{\nu \in \breve{\mathcal{V}}^t_{\mathcal{J}}(-)} q_{j_\nu} + |\breve{\mathcal{V}}^t_{\mathcal{R}}(+)| \geqslant \sum_{\nu \in \breve{\mathcal{V}}^t_{\mathcal{J}}} q_{j_\nu} \tag{36}$$

which we can reformulate as follows:

$$\sum_{\nu \in \breve{\mathcal{V}}^t_{\mathcal{J}}(+)} q_{j_\nu} \leqslant |\breve{\mathcal{V}}^t_{\mathcal{R}}(+)| \tag{37}$$

Note that inequality (37) leads to the following valid constraint (hereafter referred to as a maximum cardinality b-matching (MCbM) cut) that eliminates the solution $\bar{x}$ from the feasible region of the restricted master problem $[RMP]$:

$$\sum_{\substack{j \in \mathcal{J}^{\mathcal{H}}(\bar{x}) \\ s.t.\ \nu_j \in \breve{\mathcal{V}}^t_{\mathcal{J}}(+)}} \sum_{p \in \mathcal{P}_{h_j}} q_p x_p \leqslant |\breve{\mathcal{V}}^t_{\mathcal{R}}(+)| - \sum_{\substack{j \in \mathcal{J}^{\mathcal{R}} \\ s.t.\ \nu_j \in \breve{\mathcal{V}}^t_{\mathcal{J}}(+)}} q_j \tag{38}$$

We now demonstrate how in some cases it is possible to strengthen the previous MCbM cut by reasoning about its composition. First, note that $|\breve{\mathcal{V}}^t_{\mathcal{R}}(+)| \neq |\mathcal{R}|$ because at least one of the cumulative constraints (6) would be unsatisfied otherwise. This also implies that $|\breve{\mathcal{V}}^t_{\mathcal{R}}(-)| \neq 0$. By definition of the max-flow/min-cut, the technicians associated with the set $\breve{\mathcal{V}}^t_{\mathcal{R}}(-)$ are either not connected to any other vertices or they are assigned to the jobs $j \in \mathcal{J}(\bar{x})$ such that $\nu_j \in \breve{\mathcal{V}}^t_{\mathcal{J}}(-)$, but they cannot be assigned to any of the jobs j such that $\nu_j \in \breve{\mathcal{V}}^t_{\mathcal{J}}(+)$. The latter means that either these technicians do not have the required skills to perform those jobs or they have at least one unavailability time period that prevent them to be assigned to those

jobs. We can deduce that inequality (37) is also valid for every potential job that overlaps time period t and that cannot be performed by any of the technicians associated with a vertex of set $\breve{\mathcal{V}}^t_{\mathcal{R}}(-)$. The MCbM cut (38) can therefore be rewritten as:

$$\sum_{h\in\mathcal{H}} \Psi^t_{\breve{\mathcal{V}}^t_{\mathcal{R}}(-)}(h) \sum_{p\in\mathcal{P}_h} q_p x_p \leqslant |\breve{\mathcal{V}}^t_{\mathcal{R}}(+)| - \sum_{\substack{j\in\mathcal{J}^{\mathcal{R}} \\ s.t.\ \nu_j\in\breve{\mathcal{V}}^t_{\mathcal{J}}(+)}} q_j \tag{39}$$

where $\Psi^t_{\breve{\mathcal{V}}^t_{\mathcal{R}}(-)}(h)$ is equal to 1 if and only if pattern h overlaps time period t and none of the technicians associated with the set $\mathcal{V}^t_{\mathcal{R}}(-)$ can be assigned to h.

Last but not least, it is noteworthy that the maximum cardinality b-matching problem has only to be solved for every time period t where at least one technician cannot be assigned to a job because of an unavailability time period occurring at a time period other than t. Otherwise, constraints (6) are necessary and sufficient condition of the existence of b-matchings with the desired cardinality.

3.2.3. Maximum-weight clique cuts

Another strategy to check that a given solution $\bar{x}$ to the restricted master problem $[RMP]$ leads to a feasible sub-problem relies on proving that it is impossible to assign the technicians to the tasks without violating the location-based incompatibilities. Since these constraints are defined by day, this search decomposes into $|\mathcal{D}|$ independent searches in which for each day $d \in \mathcal{D}$ we only consider the jobs of $\mathcal{J}(\bar{x})$ that overlap d. Moreover, since the daily location-based incompatibilities are checked individually for each technician, they impact the number of available technicians at each location. During the search, it is therefore necessary to take into account the skills required to perform the different jobs. For a fixed subset $\bar{\mathcal{S}} \subset \mathcal{S}$, we only consider the jobs $j \in \mathcal{J}(\bar{x})$ such that $\mathcal{S}_j \cap \bar{\mathcal{S}} \neq \varnothing$ and the technicians mastering at least one skill of this subset. This procedure increases the likelihood of finding violated daily location-based incompatibilities if the current solution to the restricted master problem does not lead to a feasible technician-to-task assignment sub-problem. This is particularly true when the ratio between the requirements and the number of available technicians varies widely across skills.

To look for violated constraints, we solve for each day d and for each subset $\bar{\mathcal{S}} \subset \mathcal{S}$ a maximum-weight clique problem in an undirected graph $\widetilde{G}^d_{\bar{\mathcal{S}}}(\bar{x})$. The graph $\widetilde{G}^d_{\bar{\mathcal{S}}}(\bar{x})$ is composed of:

- a set of vertices $\widetilde{\mathcal{V}}^d_{\bar{\mathcal{S}}}$ such that each vertex maps a job j of set $\mathcal{J}(\bar{x})$ that i) overlaps day d (i.e., $\mathcal{T}_d \cap \{S_j, ..., C_j\} \neq \varnothing$) and ii) requires at least one of the skill in $\bar{\mathcal{S}}$ (i.e., $\mathcal{S}_j \cap \bar{\mathcal{S}} \neq \varnothing$). We denote j_ν the job associated with vertex $\nu \in \widetilde{\mathcal{V}}^d_{\bar{\mathcal{S}}}$. We associate with every vertex ν a weight equal to the number of technicians q_{j_ν} required to perform job j_ν.
- a set of edges $\widetilde{\mathcal{U}}^d_{\bar{\mathcal{S}}}$ where for all vertices $\nu_1, \nu_2 \in \widetilde{\mathcal{V}}^d_{\bar{\mathcal{S}}}$:
  $(\nu_1, \nu_2) \in \widetilde{\mathcal{U}}^d_{\bar{\mathcal{S}}} \;\Leftrightarrow\; \nu_1 \neq \nu_2 \;\wedge\; \Big(\big(S_{j_{\nu_2}} \leqslant C_{j_{\nu_1}} \;\wedge\; S_{j_{\nu_1}} \leqslant C_{j_{\nu_2}}\big) \;\vee\; \sigma_{l_{j_{\nu_1}} l_{j_{\nu_2}}} = 0 \Big)$
  There exists an edge between two vertices ν_1 and ν_2 in $\widetilde{G}^d_{\bar{\mathcal{S}}}(\bar{x})$ if and only if a technician cannot be assigned to both jobs j_{ν_1} and j_{ν_2} with regard to constraints [C2] and [C3]. $\widetilde{G}^d_{\bar{\mathcal{S}}}(\bar{x})$ is a kind of sub-graph of graph G used to derive the clique constraints (24) in formulation $[SP_2(\bar{x})]$. The only difference comes from the insertion of jobs related to technician unavailability time periods.

Proposition 3.2 formally describes the link between the resolution of the technician-to-task assignment sub-problem and the resolution of maximum-weight clique problems.

Proposition 3.2. *The technician-to-task assignment sub-problem is feasible for a solution $\bar{x}$ to the restricted master problem $[RMP]$ if for each subset $\bar{\mathcal{S}} \subset \mathcal{S}$ of skills and for each day $d \in \mathcal{D}$ the maximum weight of a clique in graph $\widetilde{G}^d_{\bar{\mathcal{S}}}(\bar{x})$ is less than or equal to $|\mathcal{R}_{\bar{\mathcal{S}}}|$.*

Proof. For a fixed day d and a fixed subset of skills $\bar{\mathcal{S}}$, suppose by contradiction that the maximum weight of a clique in graph $\widetilde{G}^d_{\bar{\mathcal{S}}}(\bar{x})$ is strictly greater than $|\mathcal{R}_{\bar{\mathcal{S}}}|$. By construction of $\widetilde{G}^d_{\bar{\mathcal{S}}}(\bar{x})$, a technician cannot perform more than

one job among the jobs whose vertices belong to that clique. Since we need more technicians than those actually available, the technician-to-task assignment sub-problem is infeasible. □

Let us now assume a fixed day $d \in \mathcal{D}$ and a fixed subset of skills $\bar{\mathcal{S}}$. Let us also denote $\mathcal{C}^d_{\bar{\mathcal{S}}}(\bar{x})$ the set of vertices that belong to the maximum-weight clique of graph $\widetilde{G}^d_{\bar{\mathcal{S}}}(\bar{x})$. If the total weight of the vertices in $\widetilde{G}^d_{\bar{\mathcal{S}}}(\bar{x})$ is strictly greater than $|\mathcal{R}_{\bar{\mathcal{S}}}|$, the constraint that eliminates $\bar{x}$ is:

$$\sum_{\nu \in \mathcal{C}^d_{\bar{\mathcal{S}}}(\bar{x})} q_{j_\nu} \leqslant |\mathcal{R}_{\bar{\mathcal{S}}}| \tag{40}$$

The valid constraint (hereafter referred to as a maximum-weight (MWC) cut) that discards a solution $\bar{x}$ to the restricted master problem $[RMP]$ is therefore:

$$\sum_{\substack{j \in \mathcal{J}^{\mathcal{H}}(\bar{x}) \\ s.t.\ \nu_j \in \mathcal{C}^d_{\bar{\mathcal{S}}}(\bar{x})}} \sum_{p \in \mathcal{P}_{h_j}} q_p x_p \leqslant |\mathcal{R}_{\bar{\mathcal{S}}}| - \sum_{\substack{j \in \mathcal{J}^{\mathcal{R}} \\ s.t.\ \nu_j \in \mathcal{C}^d_{\bar{\mathcal{S}}}(\bar{x})}} q_j \tag{41}$$

This cut simply states that the number of technicians required by the jobs associated with the vertices of the clique has to be lower than the number of technicians mastering at least a skill in $\bar{\mathcal{S}}$. We derived this cut not only for the maximum-weight clique but also for all the cliques that have a weight greater than $|\mathcal{R}_{\bar{\mathcal{S}}}|$.

It is possible to tighten the MWC cut (41) by adding some additional patterns on its left side. More precisely, we proceed as follows. First, we consider the sub-graph $G^{sub}(\bar{x})$ of graph G (graph G is used to derive the clique inequalities in the formulation $[SP_2(\bar{x})]$) that includes a vertex for pattern $h \in \mathcal{H}$ if: i) $h \in \mathcal{H} \backslash \mathcal{H}(\bar{x})$ (i.e. $q_h(\bar{x}) = 0$), ii) h overlaps day d, iii) $s_h \in \bar{\mathcal{S}}$, and iv) in the case that a technician is assigned to h, he or she cannot be assigned to any job involved in $\mathcal{C}^d_{\bar{\mathcal{S}}}(\bar{x})$ with regard to constraints [C2] and [C3]. We then solve a maximum clique problem in sub-graph $G^{sub}(\bar{x})$. Let us denote $\mathcal{H}\left[\mathcal{C}^d_{\bar{\mathcal{S}}}(\bar{x})\right]$ the set of patterns associated with the vertices that are part of the maximum clique of $G^{sub}(\bar{x})$. Observing that a technician cannot be assigned to more than one of the jobs in $\mathcal{C}^d_{\bar{\mathcal{S}}}(\bar{x})$ or one of the patterns of $\mathcal{H}\left[\mathcal{C}^d_{\bar{\mathcal{S}}}(\bar{x})\right]$, we can rewrite the MWC cut (41) as follows:

$$\sum_{\substack{j \in \mathcal{J}^{\mathcal{H}}(\bar{x}) \\ s.t.\ \nu_j \in \mathcal{C}^d_{\bar{\mathcal{S}}}(\bar{x})}} \sum_{p \in \mathcal{P}_{h_j}} q_p x_p + \sum_{h \in \mathcal{H}\left[\mathcal{C}^d_{\bar{\mathcal{S}}}(\bar{x})\right]} \sum_{p \in \mathcal{P}_h} q_p x_p \leqslant |\mathcal{R}_{\bar{\mathcal{S}}}| - \sum_{\substack{j \in \mathcal{J}^{\mathcal{R}} \\ s.t.\ \nu_j \in \mathcal{C}^d_{\bar{\mathcal{S}}}(\bar{x})}} q_j \tag{42}$$

For efficiency consideration, we reduce the size of the sub-graph $G^{sub}(\bar{x})$ observing that it is sufficient to only consider one vertex for all the patterns that satisfy two conditions: same location and overlapping of the same portion of the day. We point out that this remark also applies for graph $\widetilde{G}^d_{\bar{\mathcal{S}}}(\bar{x})$.

Last but not least, to avoid overloading our algorithm, we solve the maximum-weight clique problem only if: i) the sum of the weights of the vertices is greater than the number of available technicians and ii) there exists during a particular day at least two jobs that do not overlap and are executed at incompatible locations. Otherwise, the cumulative constraints (6) ensure for each day and for each subset of skills the non-existence of a clique with a weight strictly greater than the number of available technicians. We use the algorithms introduced in (Östergård 2001) and (Östergård 2002) for solving the maximum clique and maximum-weight clique problems.

Since the approximations to the sub-problem described in Section 3.2.2 and in Section 3.2.3 can be decomposed into a series of small problems, we can potentially identify multiple subsets of plans that cause the infeasibility of the technician-to-task assignment sub-problem. This usually leads to the generation of multiple cuts, which is known to significantly improve the efficiency of a cut generation process. We can also think to run the resolution of those small problems in parallel.

For the sake of clarity, we provide three examples in Appendix B to illustrate how we build up the different cuts previously described. For the approximations described in 3.2.2 and 3.2.3, the first two examples show

that there is no strict dominance of one over the other one (i.e., neither of the MCbM and MWC cuts are the strongest cuts). The third example is meant to illustrate a case where the two previous approximations do not find any cut although the technician-to-task assignment sub-problem is infeasible.

3.3. The algorithm: general structure

The decomposition described in section 3.1 can be seen as a Benders decomposition of the problem. To efficiently solve the problem while exploiting this decomposition, one can easily distinguish two different implementation approaches. The first approach is the classical implementation of *Benders decomposition*, which relies on Kelley's cutting plane algorithm. The method successively solves a restricted master problem and one (or multiple) sub-problem(s). New constraints, referred as Benders feasibility cuts, are introduced in the restricted master problem if the sub-problem(s) is (are) infeasible. When the sub-problem(s) is (are) not only constraint satisfaction problem(s), we may also have to generate Benders optimality cuts. The method iterates between a restricted master problem and the sub-problem(s) until it converges or concludes that there is no solution. This method has been successfully applied to (the list is not exhaustive): power systems (Shahidehopour and Fu 2005), hub network design (Cordeau et al. 2006), cargo shipping (Agarwal and Ergun 2008).

One noticeable disadvantage of the classical implementation of Benders decomposition is that the restricted master problem is repeatedly solved to optimality. Moreover, the time needed to solve the restricted master problem tends to increase with each iteration. To overcome this drawback, as an alternative approach, Benders cuts can be generated on the fly while solving a restricted master problem. The sub-problem is then solved during the search for a solution to the master problem. More specifically, at each integer node of the branch-and-bound tree, the corresponding solution is sent to the sub-problem in order to generate the Benders cuts. This implementation approach is referred to as a *Benders-based branch-and-cut* algorithm in (Naoum-Sawaya and Elhedhli 2010) or as a *branch-and-Benders-cut method* in (Gendron et al. 2014). It has been used to solve several types of problems: hub location (De Camargo et al. 2011), production routing under demand uncertainty (Adulyasak et al. 2015), location-design (Gendron et al. 2014), facility location and network design (Naoum-Sawaya and Elhedhli 2010), and hop-constrainted survivable network (Botton et al. 2013). Botton et al. (2013) reported a significant improvement using this alternative approach instead of the classical implementation of Benders decomposition, while Gendron et al. (2014) outlined the benefits of the Benders-based branch-and-cut in terms of solution quality, scalability, and robustness.

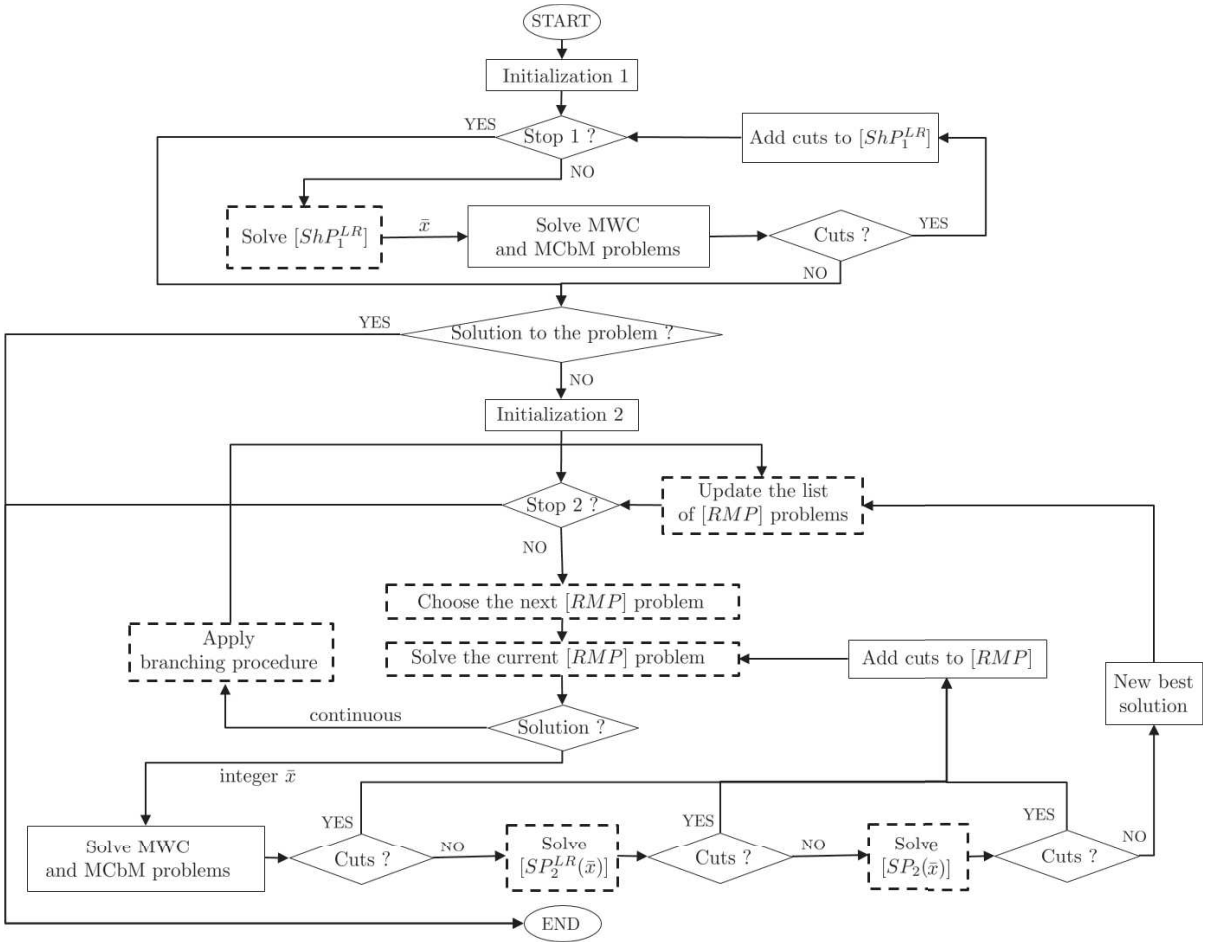
The difference between the two implementation approaches previously described is rather similar to the difference between two methods primarily used for linear programming (LP) and CP hybridization: the *Logic-based Benders decomposition* introduced by Hooker (2000) and the *branch-and-check* (B&C) framework described by Thorsteinsson (2001). Similarly to the Benders-based branch-and-cut, the B&C framework method solves only once a restricted master problem. Thorsteinsson (2001) applied it to a planning and scheduling problem, while Sadykov (2008) used it for a complex scheduling problem on a single machine. We refer the reader to (Beck 2010) for a more comprehensive survey.

Since the master problem is a pure ILP model, its resolution is very likely to be time-consuming. Therefore, it seems better suitable to solve the problem with a B&C approach[5]. Moreover, one drawback of applying the classical implementation of Benders decomposition to our problem is that a feasible solution (and optimal) is only obtained at the end, whereas a B&C approach may provide feasible solutions throughout the resolution of the master problem. Figure 4 outlines the general structure of our two-stage method:

- Stage 1 (solve a linear relaxation of $[P]$)):
  The purpose of this first stage is to generate potential useful MCbM and MWC cuts while working with a much easier problem. To this end, we consider $[ShP_1^{LR}]$ the linear relaxation of $[ShP_1]$ ("Initialization 1"). Using a LP solver, we solve the problem to optimality ("Solve $[ShP_1^{LR}]$"). We then solve the approximations to the sub-problem described in Sections (3.2.2) (the maximum cardinality b-matching becomes a fractional maximum cardinality b-matching) and (3.2.3) (in this stage, we only derive a

[5]Although we do not use constraint programming, we choose this terminology since the technician-to-task assignment sub-problem is a feasibility test, and the approximations to this sub-problem yields something rather similar to filtering algorithms.

Figure 4: Flow chart of the B&C approach .



NB: All the steps denoted with a rectangular dash-line box are performed by an ILP solver

cut for the maximum-weight clique) for the continuous solution $\bar{x}$ of $[ShP_1^{LR}]$ ("Solve MWC and MCbM problems"). If we generate some MCbM and/or MWC cuts, we add them to $[ShP_1^{LR}]$ ("Add cuts to $[ShP^{LR}]$") and we re-optimize this problem. Otherwise, we stop this first stage.

Last but not least, to avoid wasting too much time generating cuts that may not be all useful, we choose arbitrary to stop the resolution ("Stop 1") after the first 100 iterations if it is not stopped sooner ("Cuts ? No"). Note also that this stage could have been performed at the root-node of the search tree defined for the second stage. However, our aim is to take advantage of the preprocessing techniques embedded in ILP solvers.

- Stage 2 (Solve the master problem $[P]$):
  In the second stage, we first check ("solution to the problem ?") if the solution $\bar{x}$* obtained during the previous stage is feasible for $[P]$ (i.e., if $\bar{x}$* is integer and leads to a feasible technician-to-task assignment sub-problem). If it is not the case, we solve the master problem by a branch-and-cut method implemented in a commercial solver. We initialize the restricted master problem $[RMP]$ with all the cuts that have been previously generated ("Initialization 2") and we forward it to the ILP solver. As long as the solution computed by the solver is continuous ("Solution ? continuous"), we let it make its own branching decisions to produce integer solutions ("Apply branching procedure"), its own exploration of the search tree ("Choose the next $[RMP]$ problem"), and we let it use its own techniques to compute feasible solutions and generic cuts which help to reduce the list of active nodes ("Update the list of $[RMP]$ problems"). For every integer solution $\bar{x}$ to the current restricted master problem $[RMP]$ ("Solution ? integer"), we check if $\bar{x}$ is feasible regarding the technician-to-task assignment sub-problem. We start by solving the maximum cardinality b-matching and the maximum-weight clique problems ("Solve MWC and MCbM problems"). If it produces at least one MCbM or MWC cut, we discard the current solution $\bar{x}$ by adding the generated cut(s) to $[RMP]$ ("Add cuts to $[RMP]$"). Otherwise, we solve the LP formulation $[SP_2^{LR}(\bar{x})]$ ("Solve $[SP_2^{LR}(\bar{x})]$"). If we identify a violated BF cut of type (33), we add it to $[RMP]$ ("Add cuts to $[RMP]$"). Otherwise, we cannot directly conclude to the feasibility of the technician-to-task assignment sub-problem before solving the ILP formulation $[SP_2(\bar{x})]$ ("Solve $[SP_2(\bar{x})]$"). If the solution has a strictly positive cost, we generate a CB cut of type (31) and add it to the restricted master problem ("Add cuts to $[RMP]$"). Otherwise, we conclude that $\bar{x}$ is a new feasible solution to our problem ("New best solution"). Note that the branch-and-bound scheme ensures that $\bar{x}$ is strictly better than the best previous solution. We end up this phase ("Stop 2") with the optimal solution to $[P]$ or with a feasible solution if a time limit has been reached. Note that if no solution is found within the time limit, one can always consider the feasible solution in which all the tasks are postponed.

4. Computational experiments

4.1. Instances

We report computational results on the 160-instance testbed proposed by Froger et al. (2016b). These instances were randomly generated from the insight on wind prediction and maintenance operations that they obtained from their collaborators in the wind industry. They considered time horizons of different lengths (5 days and 10 days with 2 or 4 time periods per day), different number of tasks (20, 40, 80), and different number of skills (1 or 3). For each combination of parameters, they generated two categories of instances: 5 instances with a tight technicians-to-work ratio (i.e., technicians can perform the majority of the tasks during the planning horizon, but they are not guaranteed to be enough to perform all the tasks), and 5 instances with a regular technicians-to-work ratio (i.e., technicians can perform all the tasks during the planning horizon). They referred to the former as type A and to the latter as type B. They obtained 32 family of instances as representative as possible of the wide range of situations which may occur. The cost of postponement is set in every instance in such a way that postponing a task in non-profitable. In the following, we refer to each family of instance with symbol "a_b_c_d_e" where a, b, c, d, and e refer to the number of time periods in the planning horizon, the number of time periods within a day, the number of

skills, the number of tasks, and to the technicians-to-work ratio, respectively. For a thorough discussion on the instance generation process the reader is referred to Appendix D.

4.2. Results

We implemented our algorithms using *Java 8 (JVM 1.8.0.25)*. We rely on *Gurobi 6.5.1* for solving LP and ILP models. We ran our experiments on a Linux 64 bit-machine, with an Intel(R) Xeon(R) X5675 (3.07Ghz) and 12GB of RAM. We set a 3-hour time limit to solve the different instances (notice that all CPU times are reported in seconds and rounded to the closest integer). In order to assess the quality of our results, we computed the gap with respect to the optimal solution when it is known, or to the best upper bound retrieved by the solver over all the tests reported in this paper.

4.2.1. Exact approaches

We tested four different approaches to exactly solve our maintenance scheduling problem. We present the computational results for the direct resolution of the two different ILP formulations of the problem ($[P_1]$ and $[P_2]$). To quantify the relevance and the contribution of the problem-specific cuts introduced in Section 3.2, we present the computational results of the B&C approach presented in Section 3.3 without ($\overline{B\&C}$) and with ($B\&C$) the MCbM and MWC cuts. These two approaches only differ according to how we discard an infeasible solution $\bar{x}$ to the restricted master problem $[RMP]$. In Table 1, we report the average, over all the instances belonging to the same family or sharing a common characteristic, of: the gap (Gap), the solution time (Time), and the percentage of tasks scheduled in the best solution (%S). We also report the number of optimal solutions found within the time limit (#Opt). In order to have a meaningful comparison, the average solution time only takes into account those instances for which an optimal solution has been found within the time limit. Similarly, the average gap and percentage of tasks scheduled (i.e., not-postponed) takes into account only the instances which are not optimally solved. Indeed, since in our instances postponing a task is non-profitable and heavily penalized, a large gap is often related to a low percentage of tasks scheduled during the time horizon. This allows a better understanding of the results. Notice that on average 99% of the tasks are scheduled in the optimal or best-known solutions for our testbed.

First, we observe that formulation $[P_2]$ seems to outperform formulation $[P_1]$. Nonetheless, the comparison between these two formulations is quite difficult as the best model regarding the gap and solution time if it is not optimally solved can vary within a family from one instance to another one. We also notice that the average gap is still considerable for the majority of the family of instances (the solver fails to schedule a large proportion of the tasks), and when optimality is reached, it is on average with a significant solution time. It is not very surprising as the formulations involves a large number of binary variables and constraints. We therefore reach the following conclusion: directly solving the ILP formulations with a commercial solver is not a suitable exact approach for the problem.

Second, we can state that the performance of the B&C approach is strongly correlated with the cuts generated from the approximations to the technician-to-task assignment sub-problem. Indeed, including the problem-specific cuts allows us to find the optimal solution on 63 additional instances. On the remaining instances, it also significantly reduces the gap from around 4.0%. This highlights the weakness of the generic Benders cuts, as opposed to the apparent strength of the problem-specific cuts.

Third, we observe that the B&C approach outperforms by far the direct resolution of ILP formulations, and this for every family of instances. Indeed we are able to solve to optimality 80% of the instances and, in this case, the solution time is importantly reduced (around 3 minutes on average). Moreover, the overall average gap when optimality is not reached is relatively small (1.7%).

Lastly, our results suggest that the number of skills does not have a significant impact on the difficulty of the instances (although we observe that instances with 3 skills appear to be easier to solve). This may be explain because there are less symmetries among technicians and a shorter number of feasible configurations to schedule the tasks. On the other hand, the number of tasks seems to have an impact on the difficulty of the instances when the technicians-to-work ratio is tight. This can be explained by the higher difficulty of finding a maintenance plan when considering more tasks. For the instances with a regular technicians-to-work ratio, although the number of plans is larger when considering more tasks, the number of patterns

Table 1: Detailed computational results for the different exact approaches.

Family	[P1]				[P2]				B&C				B&C			
	Gap	%S	#Opt	Time	Gap	%S	#Opt	Time	Gap	%S	#Opt	Time	Gap	%S	#Opt	Time
10_2_1_20_A	-	-	5/5	598	-	-	5/5	152	-	-	5/5	2,272	-	-	5/5	**5**
10_2_1_20_B	-	-	5/5	12	-	-	5/5	37	-	-	5/5	2	-	-	5/5	**1**
10_2_1_40_A	0.01%	100%	2/5	1,813	0.01%	100%	1/5	7	0.70%	100%	2/5	3,379	-	-	**5/5**	7
10_2_1_40_B	-	-	5/5	205	-	-	5/5	121	-	-	5/5	8	-	-	5/5	**1**
10_2_3_20_A	-	-	5/5	2,635	-	-	5/5	1,574	1.0%	97%	2/5	2,871	-	-	5/5	**162**
10_2_3_20_B	-	-	5/5	31	-	-	5/5	18	-	-	5/5	94	-	-	5/5	**2**
10_2_3_40_A	-	-	5/5	3,220	-	-	5/5	3,996	2.2%	98%	2/5	4,040	-	-	5/5	**17**
10_2_3_40_B	-	-	5/5	180	-	-	5/5	295	-	-	5/5	18	-	-	5/5	**1**
20_2_1_40_A	2.4%	98%	2/5	8,074	1.4%	98%	2/5	2,858	6.4%	96%	1/5	7,301	-	-	**5/5**	230
20_2_1_40_B	0.01%	100%	4/5	232	-	-	5/5	2,078	-	-	5/5	695	-	-	5/5	**4**
20_2_1_80_A	436%	0%	0/5	-	334%	20%	0/5	-	8.2%	96%	0/5	-	0.02%	100%	**4/5**	300
20_2_1_80_B	318%	50%	3/5	1,485	229%	49%	3/5	3,823	0.13%	100%	4/5	580	-	-	**5/5**	5
20_2_3_40_A	1.25%	99%	1/5	322	1.2%	99%	3/5	5,534	4.6%	95%	1/5	1,136	2.1%	98%	**4/5**	40
20_2_3_40_B	-	-	5/5	376	-	-	5/5	155	-	-	5/5	525	-	-	5/5	**3**
20_2_3_80_A	257%	20%	0/5	-	156%	39%	0/5	-	3.3%	98%	0/5	-	-	-	**5/5**	51
20_2_3_80_B	-	-	5/5	3,415	196%	50%	3/5	2,456	0.02%	100%	4/5	163	-	-	**5/5**	5
20_4_1_20_A	1.8%	96%	0/5	-	1.3%	97%	0/5	-	2.1%	95%	0/5	-	2.2%	95%	**3/5**	1,715
20_4_1_20_B	-	-	5/5	204	-	-	5/5	405	-	-	5/5	131	-	-	5/5	**2**
20_4_1_40_A	264%	0%	0/5	-	61%	75%	0/5	-	8.6%	93%	0/5	-	1.2%	98%	**2/5**	1,586
20_4_1_40_B	174%	49%	1/5	5,208	107%	74%	1/5	1,968	0.3%	100%	4/5	630	-	-	**5/5**	6
20_4_3_20_A	1.2%	98%	0/5	-	2.4%	95%	4/5	5,005	3.1%	95%	0/5	-	2.0%	95%	**4/5**	237
20_4_3_20_B	0.01%	100%	4/5	52	-	-	5/5	113	3.1%	95%	4/5	134	-	-	5/5	**9**
20_4_3_40_A	373%	20%	0/5	-	5.3%	95%	0/5	-	6.7%	94%	0/5	-	0.85%	98%	**1/5**	8,888
20_4_3_40_B	1.5%	99%	2/5	6,112	0.03%	100%	3/5	2,108	0.29%	100%	1/5	1,608	-	-	**5/5**	11
40_4_1_40_A	352%	0%	0/5	-	106%	76%	0/5	-	15.7%	89%	0/5	-	**2.1%**	98%	0/5	-
40_4_1_40_B	1,594%	40%	0/5	-	3.0%	98%	0/5	-	0.7%	100%	0/5	-	-	-	**5/5**	31
40_4_1_80_A	4,948%	0%	0/5	-	4,948%	0%	0/5	-	16.1%	90%	0/5	-	**1.5%**	99%	0/5	-
40_4_1_80_B	331%	0%	0/5	-	331%	0%	0/5	-	1.8%	99%	0/5	-	-	-	**5/5**	89
40_4_3_40_A	1,087%	20%	0/5	-	4.6%	96%	0/5	-	14.6%	90%	0/5	-	**1.5%**	99%	0/5	-
40_4_3_40_B	477%	20%	0/5	-	0.84%	99%	2/5	2,118	0.3%	100%	0/5	-	-	-	**5/5**	36
40_4_3_80_A	2,813%	0%	0/5	-	2,727%	18%	0/5	-	14.3%	90%	0/5	-	**2.3%**	98%	0/5	-
40_4_3_80_B	3,899%	0%	0/5	-	3,899%	0%	0/5	-	0.96%	100%	0/5	-	-	-	**5/5**	86

Characteristics		[P1]				[P2]				B&C				B&C			
		Gap	%S	#Opt	Time	Gap	%S	#Opt	Time	Gap	%S	#Opt	Time	Gap	%S	#Opt	Time
$\lvert\mathcal{S}\rvert =$	1	854%	35%	32/80	1,108	621%	61%	34/80	1,035	6.7%	95%	36/80	957	1.6%	98%	**64/80**	179
	3	1,036%	39%	37/80	1,677	982%	61%	43/80	1,896	4.9%	96%	29/80	722	1.7%	98%	**64/80**	186
$\frac{\lvert\mathcal{T}\rvert}{\lvert\mathcal{D}\rvert} =$	2	179%	56%	57/80	1,383	144%	60%	57/80	1,461	3.9%	97%	51/80	982	1.0%	99%	**78/80**	49
	4	1,197%	30%	12/80	1,555	1,014%	61%	20/80	1,757	6.5%	95%	14/80	380	1.7%	98%	**50/80**	390
Type =	*A*	878%	38%	20/80	2,618	759%	65%	25/80	2,838	7.8%	94%	13/80	3,106	1.7%	98%	**48/80**	456
	B	1,059%	35%	49/80	921	802%	53%	52/80	913	0.8%	100%	52/80	288	-	-	**80/80**	18
All		940%	37%	69/160	1,413	773%	61%	77/160	1,538	5.7%	96%	65/160	852	1.7%	98%	**128/160**	182

does not grow proportionally, which results in a moderately more complicated technician-to-task assignment sub-problem. Moreover, our exact approaches suit better to instances with 2 time periods per day as the solution time is lower and the number of optimal solutions is larger than when considering 4 time periods per day. A plausible explanation is that the daily location-based incompatibilities are more binding on instances with more time periods per day. Indeed, a larger number of periods provide a wider choice of task starting times and therefore more opportunities to move technicians between locations during a single day. Instances with 4 time periods per day also have a larger number of plans and patterns; this may also explain their higher difficulty. In conclusion, according to our experiments, the difficulty of an instance increases with the number of time periods per day and the tightness of the technicians-to-work ratio.

In Table 2, we present a brief description of the average number of cuts generated during the execution of the B&C approach for each testbed. The average, over all the instances sharing a common characteristic, of the total number of cuts (#Cuts) is decomposed into the CB, BF, MCbM, and MWC cuts. Detailed results for each family of instances are available in Appendix C. First, we observe that, on average, 90% of the cuts are problem-specific cuts whereas the other 10% are generic Benders cuts. These results naturally show that the approximations are not always able to identify the infeasibility of the technician-to-task assignment sub-problem. However, when $|\mathcal{S}| = 1$, it is noteworthy that solving the maximum cardinality b-matching and maximum-weight clique problems allow almost always to identify the infeasibility of the sub-problem. We observe that we generate more cuts for instances with 3 skills, 4 time periods per day and a tight technicians-to-work ratio. This is due to the largest number of patterns in the first two cases and to the fact that there is less potential configuration to schedule the tasks in the last case. Notice that we never generate CB cuts. Actually, we observe that the optimal solution to the relaxed sub-problem $[SP_2^{LR}(\bar{x})]$ is most of the time integer although the constraint matrix is not totally unimodular[6]. We can also note that we generate only few MCbM cuts. On the contrary, we generate many MWC cuts. The reason for this is that the restricted master problem has no information about the daily location-based incompatibilities at the beginning of the optimization. It is then more likely that these specific constraints are not satisfied by the solutions to the restricted master problem.

Table 2: Description of the average number of cuts generated in the B&C approach.

Characteristic		#Cuts	CB	Other cuts		
				BF	MCbM	MWC
$\lvert\mathcal{S}\rvert$ =	1	102	0	0.3	3	98
	3	224	0	28	10	186
$\frac{\lvert\mathcal{T}\rvert}{\lvert\mathcal{D}\rvert}$ =	2	56	0	9	3	44
	4	270	0	19	10	241
Type =	*A*	262	0	24	10	229
	B	63	0	4	4	56
All		163	0	14	7	142

Furthermore, we notice that all the components of the B&C approach have a favorable trade-off between their efficiency and the time spent on it. Since the relaxation of the problem considered in the first stage only contains continuous variables and no (or few) cuts, this first stage does not require too much time: on average 1% of the CPU time. Notice also that the limit on the number of iterations during this stage is never reached in our experiments. The results also show that solving the restricted master problem in the B&C approach is the most time-consuming part of the second-stage. Indeed, this component is responsible on average for 99% of the CPU time. This compares with the negligible time spent on solving, for a solution $\bar{x}$ to $[RMP]$, the formulations $[SP_2^{LR}(\bar{x})]$ and $[SP_2(\bar{x})]$ with the commercial solver, or the approximations to the sub-problem.

4.2.2. A cooperative approach

As a second part of our experiments, we tested the use of the CPLNS introduced in (Froger et al. 2016b) along with the B&C approach. More specifically, the idea is to run the CPLNS in parallel with the algorithm.

[6]We found an instance - not part of our testbed - and a solution $\bar{x}$ of $[RMP]$ where the optimal value of $[SP_2^{LR}(\bar{x})]$ is equal to 0 whereas the optimal value of $[SP_2(\bar{x})]$ is equal to 1.

If the current solution to the CPLNS is better than the best solution found so far, we provide this solution to the solver. If this improves the current lower bound of the ILP solver it may help to prune some nodes in the branch-and-bound tree. This idea comes from the observation that for the large-sized instances the solver has sometimes trouble with finding good quality solutions. Table 3 summarizes the results. The last columns report the average gap for the best solution found by the CPLNS with the same execution time as the B&C approach (i.e., we stop the CPLNS when the B&C approach finds the optimal solution or when it reaches the time limit). More detailed results for each family of instances are available in Appendix C. Since no additional instances are solved to optimality, we conclude that running the CPLNS in parallel with the B&C approach has no significant effect on its efficacy and efficiency (even if the gap is reduced by 0.4% for the instances for which optimality is not reached). Above all, these results allow us to state that, if we are given a 3-hour time limit, the B&C approach significantly outperforms the CPLNS with an average difference of around 2% between the two gaps. It is mainly due to the difficulty of the CPLNS in scheduling some tasks when the technicians-to-work ratio is tight. It also highlights the fact that the metaheuristic may often be trapped in local optima. However, one may find a smaller gap between the efficiency of the CPLNS and the B&C approach if one imposes another time limit. Lastly, we observe that the characteristics that make the instances difficult to solve by the B&C approach are the same for the CPLNS.

Table 3: Aggregated computational results of the B&C approach coupled with the CPLNS.

Characteristics		**B&C**			**CPLNS**	
		Gap	**#Opt**	**Time**	**Gap**[1]	**Gap**[2]
$\lvert\mathcal{S}\rvert =$	1	1.6%	64/80	183	3.7%	1.2%
	3	1.0%	64/80	195	2.7%	1.0%
$\frac{\lvert\mathcal{T}\rvert}{\lvert\mathcal{D}\rvert} =$	2	0.09%	78/80	38	3.7%	0.90%
	4	1.4%	50/80	399	3.2%	1.4%
Type =	*A*	1.3%	48/80	445	3.2%	1.9%
	B	-	80/80	20	-	0.61%
All		1.3%	128/160	179	3.2%	1.1%

[1] Takes into account the instances where the time limit is reached in the B&C approach.

[2] Takes into account the instances solved to optimality by the B&C approach.

5. Conclusions and research perspectives

In this study, we have proposed a branch-and-check approach to solve a challenging maintenance scheduling problem rising in the onshore wind industry. This exact method takes advantage of the decomposition of the problem into a task scheduling problem and a technician-to-task assignment sub-problem. For each selection of plans, we actually check the existence of an assignment of the technicians to the scheduled tasks that copies with the availability of every single technician and that meets the travel limitations imposed on each day. Since the ILP formulation of the sub-problem does not possess the integrity property, we use the concept of combinatorial Benders cuts to invalidate infeasible maintenance plans to the restricted master problem, while trying to identify violated classical Benders feasibility cuts beforehand. However the key part of the algorithm comes from the approximations to the technician-to-task assignment sub-problem as a series of maximum cardinality b-matching and maximum-weight clique problems. Indeed, according to the experiments that we conducted on randomly generated instances with technical expertise, the resulting problem-specific cuts proves to be very effective speeding up the convergence of the B&C approach. This latter method finds optimal solutions in short execution times for the large majority of the instances or delivers high-quality integer solutions in those instances in which optimality is not reached. It significantly outperforms the direct resolution of ILP models as well as, in a certain context, a state-of-the-art metaheuristic approach.

In this research we considered the deterministic version of the problem. An interest perspective is to consider the inherent uncertainty on wind speed forecasts. Future research will focus on solving the resulting stochastic optimization problem using stochastic programming and/or robust optimization techniques.

References

Adulyasak, Y., Cordeau, J.-F., and Jans, R. (2015). Benders Decomposition for Production Routing Under Demand Uncertainty. *Operations Research*, 63(4):851–867.

Agarwal, R. and Ergun, Ö. (2008). Ship scheduling and network design for cargo routing in liner shipping. *Transportation Science*, 42(2):175–196.

Baptiste, P., Le Pape, C., and Nuijten, W. (1999). Satisfiability tests and time-bound adjustmentsfor cumulative scheduling problems. *Annals of Operations research*, 92:305–333.

Beck, J. (2010). Checking-Up on Branch-and-Check. *Principles and Practice of Constraint Programming CP 2010 SE - 10*, 6308:84–98.

Biro, M., Hujter, M., and Tuza, Z. (1992). Precoloring extension. i. interval graphs. *Discrete Mathematics*, 100(1):267–279.

Botton, Q., Fortz, B., Gouveia, L., and Poss, M. (2013). Benders decomposition for the hop-constrainted survivable network design problem. *INFORMS Journal On Computing*, 25(1):13–26.

Bron, C. and Kerbosch, J. (1973). Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577.

Codato, G. and Fischetti, M. (2006). Combinatorial Benders' Cuts for Mixed-Integer Linear Programming. *Operations Research*, 54(4):756–766.

Cordeau, J.-F., Pasin, F., and Solomon, M. (2006). An integrated model for logistics network design. *Annals of Operations Research*, 144(1):59–82.

De Camargo, R., De Miranda, G., and Ferreira, R. (2011). A hybrid Outer-Approximation/Benders Decomposition algorithm for the single allocation hub location problem under congestion. *Operations Research Letters*, 39(5):329–337.

Ding, F., Tian, Z., and Jin, T. (2013). Maintenance modeling and optimization for wind turbine systems: A review. In *2013 International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (QR2MSE)*, pages 569–575.

Froger, A., Gendreau, M., Mendoza, J., Pinson, E., and Rousseau, L.-M. (2016a). Maintenance scheduling in the electricity industry: A literature review. *European Journal of Operational Research*, 251(3):695–706.

Froger, A., Gendreau, M., Mendoza, J., Pinson, E., and Rousseau, L.-M. (2016b). Solving a wind turbine maintenance scheduling problem. Under review.

Gendron, B., Scutellà, M., Garroppo, R., Nencioni, G., and Tavanti, L. (2014). A Branch-and-Benders-Cut Method for Nonlinear Power Design in Green Wireless Local Area Networks. Technical Report CIRRELT-2014-42, CIRRELT.

Hooker, J. (2000). *Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction*. Wiley.

Hooker, J. N. and Ottosson, G. (2003). Logic-based Benders decomposition. *Mathematical Programming*, 96(1):33–60.

Irawan, C., Ouelhadj, D., Jones, D., Stålhane, M., and Sperstad, I. (2016). Optimisation of maintenance routing and scheduling for offshore wind farms. *European Journal of Operational Research*.

Jensen, T. and Toft, B. (2011). *Graph coloring problems*, volume 39. John Wiley & Sons.

Kovács, A., Erds, G., Viharos, Z., and Monostori, L. (2011). A system for the detailed scheduling of wind farm maintenance. *CIRP Annals - Manufacturing Technology*, 60(1):497–501.

Laporte, G. and Louveaux, F. V. (1993). The integer l-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, 13(3):133 – 142.

Naoum-Sawaya, J. and Elhedhli, S. (2010). An interior-point Benders based branch-and-cut algorithm for mixed integer programs. *Annals of Operations Research*, 210(1):33–55.

Östergård, P. (2001). A new algorithm for the maximum-weight clique problem. *Nordic Journal of Computing*, 8(4):424–436.

Östergård, P. (2002). A fast algorithm for the maximum clique problem. *Discrete Applied Mathematics*, 120(1):197–207.

Sadykov, R. (2008). A branch-and-check algorithm for minimizing the weighted number of late jobs on a single machine with release dates. *European Journal of Operational Research*, 189(3):1284–1304.

Shahidehopour, M. and Fu, Y. (2005). Benders decomposition: applying benders decomposition to power systems. *Power and Energy Magazine, IEEE*, 3(2):20–21.

Thorsteinsson, E. (2001). Branch-and-check: A hybrid framework integrating mixed integer programming and constraint logic programming. In Walsh, T., editor, *Principles and Practice of Constraint Programming CP 2001*, volume 2239 of *Lecture Notes in Computer Science*, pages 16–30. Springer Berlin Heidelberg.

Appendix A. Complement: sub-problem and complexity

Appendix A.1. Equivalence to the L-coloring problem

To assess the complexity of the technician-to-task assignment sub-problem, we prove its equivalence to the L-coloring problem

First, let us associate a color $color_r$ to every technician $r \in \mathcal{R}$. For a given solution $\bar{x}$ to the restricted master problem $[RMP]$, let us also introduce the undirected graph $\ddot{G}(\bar{x})$ composed of:

- a set of vertices $\ddot{\mathcal{V}}$
  - For each job $j \in \mathcal{J}(\bar{x})$, we add q_j vertices in $\ddot{\mathcal{V}}$. Parameter j_ν denotes the job associated with a vertex $\nu \in \ddot{\mathcal{V}}$ and $\ddot{\mathcal{V}}_j$ the set of vertices associated with job j. Denoting L_ν the set of colors associated with vertex ν, we define $L_\nu = \{color_r\}_{r \in \mathcal{R}_j}$.
- a set of edges $\ddot{\mathcal{U}}$ defined such that $\forall \nu_1\, \nu_2 \in \ddot{\mathcal{V}}$:
  $(\nu_1, \nu_2) \in \ddot{\mathcal{U}} \Leftrightarrow \nu_1 \neq \nu_2 \ \wedge \ \Big(\big(S_{j_{\nu_2}} \leqslant C_{j_{\nu_1}} \ \wedge \ S_{j_{\nu_1}} \leqslant C_{j_{\nu_2}} \big) \ \vee \ \sigma_{l_{j_{\nu_1}} l_{j_{\nu_2}}} = 0 \Big)$
  (There exists an edge between two vertices ν_1 and ν_2 in $\ddot{G}(\bar{x})$ if and only if a technician cannot be assigned to both jobs j_{ν_1} and j_{ν_2} with regard to constraints [C2] and [C3])

We prove in Proposition B.1 the equivalence between the technician-to-task assignment sub-problem and the L-coloring problem in graph $\ddot{G}(\bar{x})$.

Proposition B.1. *Let $\bar{x}$ be a solution to the restricted master problem $[RMP]$. The technician-to-task assignment sub-problem for $\bar{x}$ is equivalent to the L-coloring problem in graph $\ddot{G}(\bar{x})$.*

Proof. Assume that we know a feasible solution to the technician-to-task assignment sub-problem. This solution directly yields the list $\mathcal{R}_j^{ass}$ of technicians assigned to every job $j \in \mathcal{J}(\bar{x})$ (in that solution). We can then build a solution to the L-coloring problem by iterating through the vertices of $\ddot{G}(\bar{x})$. More specifically, for each vertex $\nu \in \ddot{\mathcal{V}}$, we pick a technician $r \in \mathcal{R}_{j_\nu}^{ass}$ (and remove it from this set) and color the vertex ν with $color_r$. By construction of the graph, we are ensured that for a job j the set $\mathcal{R}_j^{ass}$ becomes empty only when every vertex of $\ddot{\mathcal{V}}_j$ is colored. We are also ensured that the graph is L-colorable since each vertex $\nu \in \ddot{\mathcal{V}}$ picks up an admissible color in L_ν.

Alternatively, assume that we know a solution c to the L-coloring problem in graph $\ddot{G}(\bar{x})$. This solution directly yields the list $\mathcal{R}_j^{ass}$ of technicians assigned to every job $j \in \mathcal{J}(\bar{x})$. More specifically, for each vertex $\nu \in \ddot{\mathcal{V}}$, we add to $\mathcal{R}_j^{ass}$ the technician $r \in \mathcal{R}$ such that $color_r = c(\nu)$. We then induce the underlying assignment of the technicians to the plans selected in the solution to the restricted master problem. By construction of the graph $\ddot{G}(\bar{x})$, the assignments satisfy the location-based incompatibility constraints and copy with individual technician availability time periods. This produces a feasible solution to the technician-to-task assignment sub-problem. □

Since the graph coloring problem is a specific case of the L-coloring problem, the strong NP-completeness of the former (Jensen and Toft 2011) implies the strong NP-completeness of the latter. We can therefore state that the technician-to-task assignments sub-problem is NP-complete in the strong sense. It is noteworthy that in the case of interval graphs, the L-coloring problem stays NP-complete (Biro et al. 1992) although the graph coloring problem becomes polynomial.

Appendix B. Illustrative examples

Appendix B.1. Example 1

This example (referred to as *Example 1*) illustrates thoroughly how we build the different cuts described in the report.

We consider in this example a fixed time horizon made up of 8 time periods of identical length ($\mathcal{T} = \{1, 2, ..., 8\}$) and partitioned into two days: time periods 1 to 4 belong to day 1 (i.e., $\mathcal{T}_1 = \{1, 2, 3, 4\}$) and time periods 5 to 8 to day 2 (i.e., $\mathcal{T}_2 = \{5, 6, 7, 8\}$). We have three different locations denoted as l_1, l_2, and l_3 (i.e., $\mathcal{L} = \{l_1, l_2, l_3\}$). Locations l_2 and l_3 cannot be visited by a technician within the same day (i.e., $\sigma_{l_2 l_3} = 0$). We do not define any other daily location-based incompatibilities. We consider 4 tasks to schedule ($\mathcal{I} = \{A, B, C, D\}$), 3 technicians ($\mathcal{R} = \{r_1, r_2, r_3\}$), and 3 skills ($\mathcal{S} = \{s_1, s_2, s_3\}$). The characteristics of the tasks and the technicians are defined in Table B.4a and in Table B.4b. For the sake of simplicity, we do not explicitly introduce all the parameters defining an instance of the problem, we introduce only those which are useful for the illustration of the cut generation process.

Table B.5 shows a given solution to the restricted master problem $[RMP]$ in which no cuts have been previously added. According to the selected plans, the table reports the starting and completion time periods of each task as well as the number of technicians required to perform every task. We refer to this solution using symbol $\bar{x}$. Note that $\bar{x}$ satisfies the cumulative constraints (6) of the restricted master problem[7].

Table B.4: Data of Example 1.

$\mathcal{I}$	l_i	s_i
A	l_1	s_1
B	l_2	s_1
C	l_1	s_2
D	l_3	s_3

(a) Characteristics of the tasks.

$\mathcal{R}$	$\{s \in \mathcal{S} \mid \zeta_{rs} = 1\}$	unavailability time period
r_1	$\{s_1\}$	at location l_3 during time period 8
r_2	$\{s_1, s_2, s_3\}$	–
r_3	$\{s_1, s_3\}$	–

(b) Characteristics of the technicians.

Table B.5: A solution to the restricted master problem for Example 1.

$\mathcal{I}$	Selected plan p	S_p	C_p	q_p	$\mathcal{R}_p$
A	p_A	2	5	1	$\{r_1, r_2, r_3\}$
B	p_B	4	7	2	$\{r_2, r_3\}$
C	p_C	7	8	1	$\{r_2\}$
D	p_D	1	3	2	$\{r_2, r_3\}$

In the following, symbol u_{r_1} refers to the job associated with the unavailability time period of technician r_1. Observing that u_{r_1} and p_B both overlaps day 2 and are defined at two incompatible locations l_2 and l_3, we can deduce that technician r_1 cannot be assigned to plan p_B (although he or she has the required skill).

Note that in solution $\bar{x}$ we have as many patterns as selected plans, and therefore we introduce as many jobs as plans. Job j_A refers then to plan p_A, job j_B refers then to plan p_B, and so on.

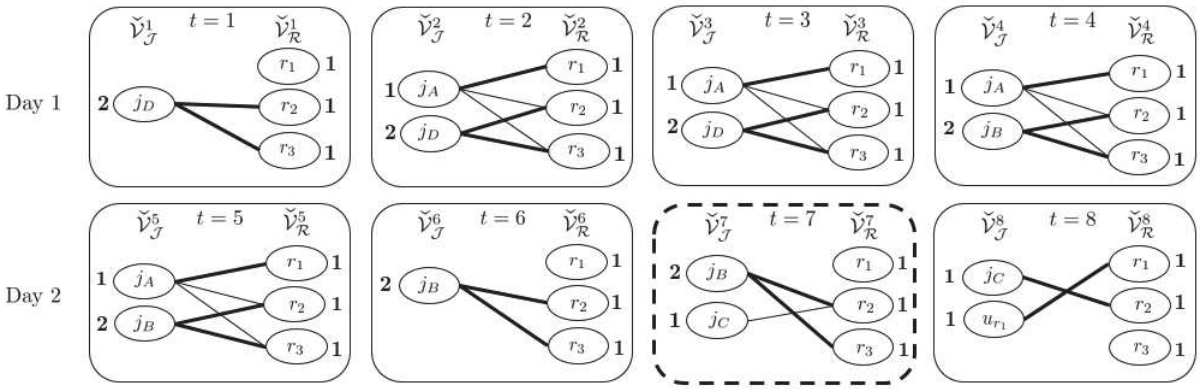
First, let us look at the potential generation of MCbM cuts. Figure B.5 describes graph $\breve{G}^t(\bar{x})$ for each time period of the planning horizon for Example 1.

For time period $t = 7$, the value of the maximum flow problem in graph $\widehat{G}^7(\bar{x})$ is equal to 2 and so is the maximum cardinality of a b-matching in $G^7(\bar{x})$. Since $q_{j_B} + q_{j_C} = 3$, $\bar{x}$ is an infeasible solution to the whole problem. We then compute the minimum cut in graph $\widehat{G}^7(\bar{x})$ (see Figure B.6).

From the general expression (38), we build the MCbM cut (B.1).

$$2x_{p_B} + 1x_{p_C} \leqslant 2 \tag{B.1}$$

[7]For instance, at time period 7 we have seven cumulative constraints. Plugging in the values of the variables, we obtain $2 \leqslant 3$, $1 \leqslant 1$, $0 \leqslant 2$, $3 \leqslant 3$, $2 \leqslant 3$, $1 \leqslant 2$, $3 \leqslant 3$ when $\bar{\mathcal{S}}$ is respectively equal to $\{s_1\}$,$\{s_2\}$,$\{s_3\}$,$\{s_1, s_2\}$,$\{s_1, s_3\}$,$\{s_2, s_3\}$,$\{s_1, s_2, s_3\}$



NB: in bold a solution to the maximum cardinality b-matching problem, a rectangular dash-line box means that no b-matching with the desired cardinality can be found

Figure B.5: Structure of graphs $\breve{G}^t(\bar{x})$ for Example 1 and different values of $t \in \mathcal{T}$.

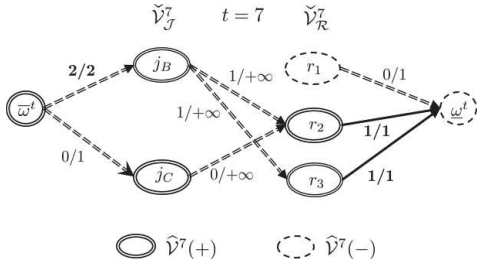


Figure B.6: Minimum cut in graph $\widehat{G}^7(\bar{x})$ for Example 1.

Second, let us look at the potential generation of MWC cuts. For illustration purposes, Figure B.7 depicts $\widetilde{G}^d_{\bar{\mathcal{S}}}(\bar{x})$ for every day $d \in \{\text{day 1, day 2}\}$ and every subset $\bar{\mathcal{S}} \subset \mathcal{S}$ of skills. Looking at Figure B.7, one can see that the maximum-weight clique is strictly greater than the number of available technicians in 4 different cases. From the general expression (41), we build the MWC cuts (B.2) and (B.3).

$$1x_{p_A} + 2x_{p_B} + 1x_{p_C} \leqslant 3 \tag{B.2}$$

$$2x_{p_B} + 1x_{p_C} \leqslant 2 \tag{B.3}$$

The cut (B.2) is built from the clique computed either in $\widetilde{G}^1_{\{s_1,s_2,s_3\}}(\bar{x})$ or in $\widetilde{G}^1_{\{s_1,s_3\}}(\bar{x})$. In the same way, the cut (B.3) is built from the clique computed either in $\widetilde{G}^2_{\{s_1,s_2,s_3\}}(\bar{x})$ or in $\widetilde{G}^2_{\{s_1,s_2\}}(\bar{x})$.

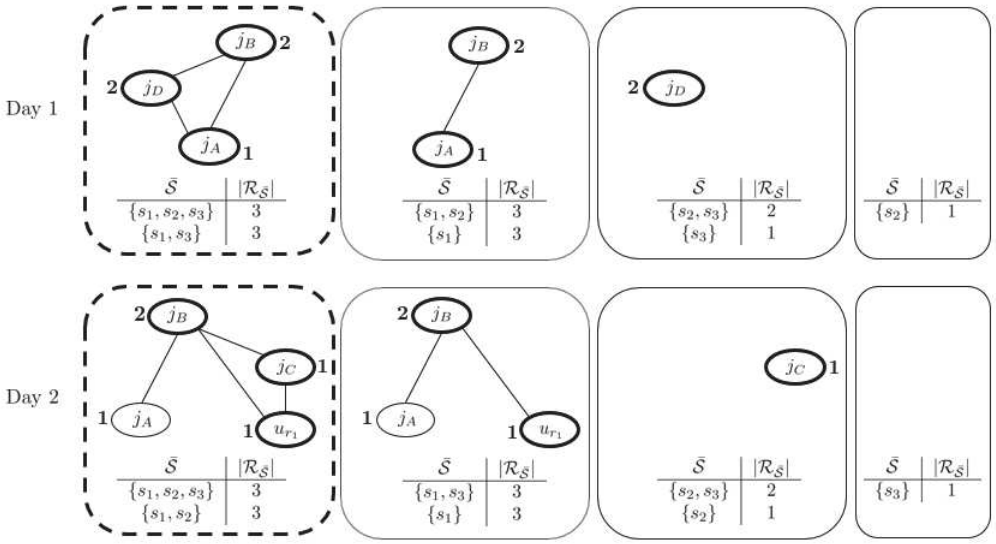
Third, let us solve the formulation $[SP_2^{LR}(\bar{x})]$ with a commercial solver. Since its optimum value is strictly greater than zero (equal to 2), we identify the violated BF cut (B.4).

$$2x_{p_B} + x_{p_C} + 2x_{p_D} \leqslant 3 \tag{B.4}$$

Fourth, let us directly solve the ILP formulation $[SP_2(\bar{x})]$. As with the linear relaxation, the optimum value is equal to 2. We then generate the CB cut (B.5).

$$x_{p_A} + x_{p_B} + x_{p_C} + x_{p_D} \leqslant 3 \tag{B.5}$$

Table B.6 reports the infeasible sections of plans discarded by the MCbM, MWC, BF and CB cuts. We denote each selection of plans using a four dimensional vector where the first, second, third and fourth coordinate refers to the plan selected for task A, task B, task C and task D, respectively. Symbol "..." simply



NB: in bold a solution to the maximum-weight clique problem, a rectangular dash-line box means that the weight of this clique is strictly greater than the maximum allowed

Figure B.7: Structure of $\widetilde{G}^d_{\bar{\mathcal{S}}}(\bar{x})$ for Example 1

means that the infeasibility of the plans selection holds for any executing plan selected at the corresponding coordinate.

Table B.6: The infeasible plans selections discarded in Example 1.

Plans selection	CB	BF	MCbM	MWC
(p_A, p_B, p_C, p_D)	✓	✓	✓	✓
$(p_A, p_B, p_C, ...)$			✓	✓
$(p_A, p_B, ..., p_D)$		✓		✓
$(..., p_B, p_C, p_D)$		✓	✓	✓
$(..., p_B, p_C, ...)$			✓	✓
$(..., p_B, ..., p_D)$		✓		✓

For the cut (B.1), notice that we can build a stronger MCbM cut of type (39) as described in Section 3.2.2. Indeed, we can add to the left hand side of the cut (B.1) all the patterns overlapping time period 7 to which technician r_1 cannot be assigned (this is the only technician associated with a vertex of set $\breve{\mathcal{V}}^t_{\mathcal{R}}(-)$).

For the MWC cuts (B.2) and (B.3), we can build stronger MWC cuts of type (42) as described in Section 3.2.3. To strengthen MWC cut (B.2), we consider the sub-graph $G^{sub}(\bar{x})$ of G that includes the vertices linked to: i) patterns at location l_1 overlapping time periods 3 and 4, ii) patterns at location l_2 overlapping time periods 4, and iii) patterns at location l_3 overlapping at least time period 2, 3 or 4. Indeed, one technician cannot be assigned to any of the previous patterns if he or she is assigned to pattern p_A, p_B or p_D. We then solve a maximum clique problem in this sub-graph, and we add to the left hand side of the MWC cut (B.2) all the plans associated with the patterns involved in the maximum clique. We proceed on a similar way for cut (B.3) by considering the sub-graph $G^{sub}(\bar{x})$ of G that includes the vertices linked to: i) patterns at location l_1 overlapping time periods 7 and 8, ii) patterns at location l_2 overlapping at least time period 7 or 8, and iii) patterns at location l_3 overlapping at least time period 8. Indeed, one technician cannot be assigned to any of the previous patterns if he or she is assigned to pattern p_B, p_D or to the job u_{r_1}. Again, we solve a maximum clique problem in this sub-graph, and we add to the left hand side of the MWC cut (B.3) all the plans associated with the patterns involved in the maximum clique. In this example, it is worth noting that when building the sub-graph, we do not pay a special attention to the skill associated with the patterns because the MWC cuts (B.2) and (B.3) have been computed with $\bar{\mathcal{S}} = \mathcal{S}$. Otherwise, only

the patterns having their skill in $\bar{\mathcal{S}}$ can be added to the left hand side of the cuts. (since the right hand side of the cuts is based on the total number of technicians mastering at least one skill of $\bar{\mathcal{S}}$).

Example 1 illustrates a case where the approximation described in Section 3.2.3 dominates the approximation described in Section 3.2.2 (i.e., a case where the MWC cuts are stronger than the MCbM cuts)

Appendix B.2. Example 2

We introduce now a second example (referred to as *Example 2*) to illustrate a case where the approximation described in Section 3.2.2 dominates the approximation described in Section 3.2.3 (i.e., a case where the MCbM cuts are stronger than the MWC cuts)

In this example, we consider a fixed time horizon made up of 4 time periods ($\mathcal{T} = \{1, 2, 3, 4\}$) and partitioned into two days: time periods 1 and 2 belongs to day 1 and time periods 3 and 4 belongs to day 2. We consider 1 location ($\mathcal{L} = \{l\}$), 2 tasks to schedule ($\mathcal{I} = \{E, F\}$), 1 skill ($\mathcal{S} = \{s\}$) and 2 technicians ($\mathcal{R} = \{r_4, r_5\}$). The technician r_2 is unavailable during time periods 1 and 3. The characteristics of the tasks and the technicians are summarized in Table B.7a and in Table B.7b. We denote with symbol u_{r_5} the jobs associated with the unavailability time periods of technician r_5.

Table B.7: Data of Example 2.

(a) Characteristics of the tasks.

$\mathcal{I}$	l_i	s_i
E	l	s
F	l	s

(b) Characteristics of the technicians.

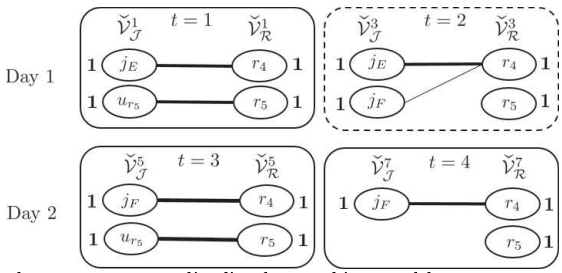
$\mathcal{R}$	$\{s \in \mathcal{S} \mid \lambda_{rs} = 1\}$	unavailability time periods
r_4	$\{s\}$	–
r_5	$\{s\}$	at location l during time periods 1 and 3

Table B.8 shows a given solution to the restricted master problem $[RMP]$ in which no cuts have been previously added. According to the selected plans, the table reports the starting and completion time periods of each task as well as the number of technicians required to perform every task. Note that $\bar{x}$ satisfies the cumulative constraints (6) of the restricted master problem.

Table B.8: A solution to the restricted master problem for Example 2.

$\mathcal{I}$	Selected plan p	S_p	C_p	q_p	$\mathcal{R}_p$
E	p_E	1	2	1	$\{r_4\}$
F	p_F	2	4	1	$\{r_4\}$

First, let us look at the potential generation of MCbM cuts. Figure B.8 describes graph $\breve{G}^t(\bar{x})$ for each time period of the horizon for Example 2.



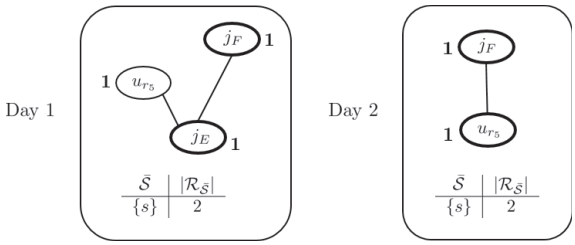
NB: in bold a solution to the maximum cardinality b-matching problem, a rectangular dash-line box means that no b-matching with the desired cardinality can be found

Figure B.8: Structure of graphs $\breve{G}^t(\bar{x})$ for Example 2 and different values of $t \in \mathcal{T}$.

Since technician r_2 cannot be assigned to task B because of its personal availability schedule, the maximum cardinality of a b-matching at time period 2 is equal to 1 whereas the tasks scheduled during this time period require a total of two technicians. The following MCbM cut is then produced:

$$x_{p_E} + x_{p_F} \leqslant 1 \tag{B.6}$$

Second, let us look at the potential generation of MWC cuts. Figure B.9 depicts $\widetilde{G}^d_{\mathcal{S}}(\bar{x})$ for every day $d \in$ {day 1, day 2}.



NB: in bold a solution to the maximum-weight clique problem

Figure B.9: Structure of $\widetilde{G}^d_{\mathcal{S}}(\bar{x})$ for Example 2

We immediately see that solving maximum-weight clique problems does not enable us to produce any MWC cut for this example. Therefore, Example 2 illustrates a case where the approximation described in Section 3.2.2 dominates the approximation described in Section 3.2.3.

Appendix B.3. Example 3

This third example (referred to as *Example 3*) is meant to illustrate a case where the problem-specific approximations do not find any cut although the technician-to-task assignment sub-problem is infeasible.

We consider in this example a fixed time horizon of 8 time periods ($\mathcal{T} = \{1, ..., 8\}$) and partitioned into two days: time periods 1 to 4 belongs to day 1 and time periods 5 and 8 belongs to day 2. We consider 1 location ($\mathcal{L} = \{l\}$), 4 tasks to schedule ($\mathcal{I} = \{G, H, I, J\}$), 3 skills ($\mathcal{S} = \{s_6, s_7, s_8\}$) and 2 technicians ($\mathcal{R} = \{r_6, r_7\}$). The characteristics of the tasks and the technicians are summarized in Table B.9a and in Table B.9b.

Table B.9: Data of Example 3.

(a) Characteristics of the tasks.

$\mathcal{I}$	l_i	s_i
G	l	s_6
H	l	s_6
I	l	s_7
J	l	s_8

(b) Characteristics of the technicians.

$\mathcal{R}$	$\{s \in \mathcal{S} \mid \lambda_{rs} = 1\}$	unavailability time periods
r_6	$\{s_6\}$	–
r_7	$\{s_6, s_7, s_8\}$	–

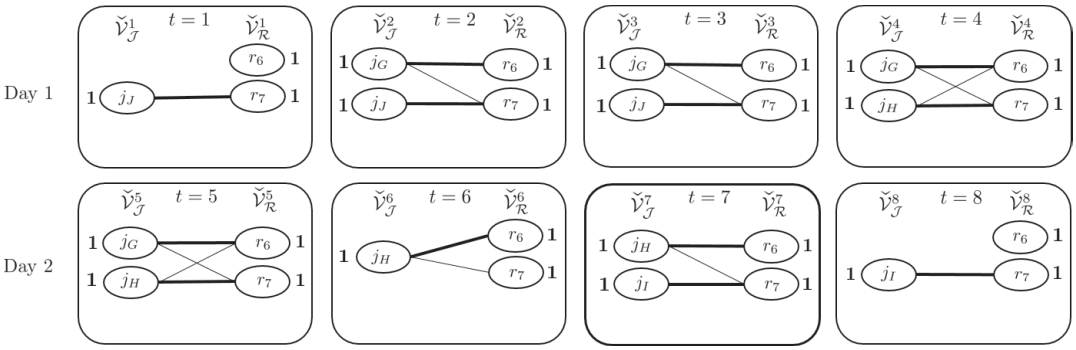
Table B.10 shows a given solution to the restricted master problem $[RMP]$ in which no cuts have been previously added. According to the selected plans, the table reports the starting and completion time periods of each task as well as the number of technicians required to perform every task. Again, it is easy to verify that $\bar{x}$ satisfies the cumulative constraints (6) of the restricted master problem.

Table B.10: A solution to the restricted master problem for Example 3.

$\mathcal{I}$	Selected plan p	S_p	C_p	q_p	$\mathcal{R}_p$
G	p_G	2	5	1	$\{r_6, r_7\}$
H	p_H	4	7	1	$\{r_6, r_7\}$
I	p_I	7	8	1	$\{r_7\}$
J	p_J	1	3	1	$\{r_7\}$

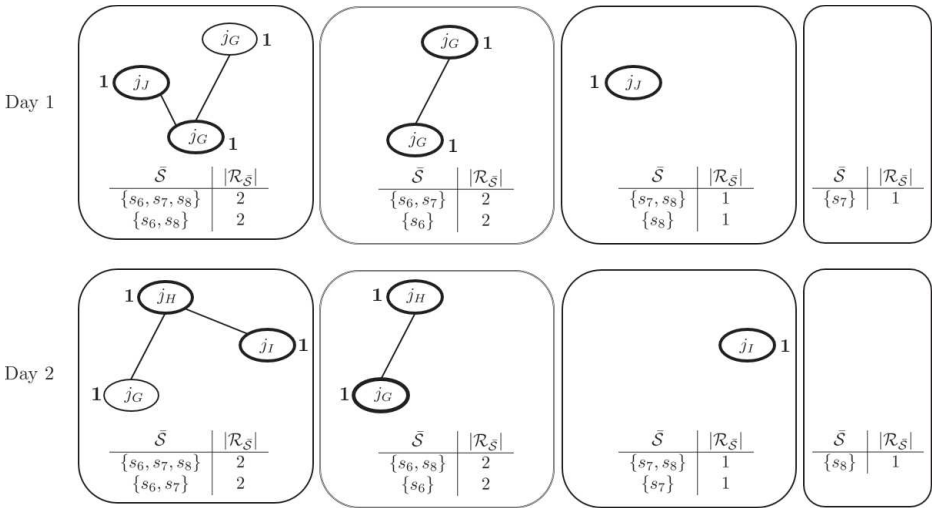
First, let us look at the potential generation of MCbM cuts. Figure B.10 describes graph $\breve{G}^t(\bar{x})$ for each time period of the horizon for Example 3. We observe that solving maximum cardinality b-matching problems does not enable us to produce any MCbM cut for this example.

Second, let us look at the potential generation of MWC cuts. Figure B.11 depicts $\widetilde{G}^d_{\mathcal{S}}(\bar{x})$ for every day $d \in \{day1, day2\}$. Similarly, no MWC cuts are produced from this approximation.



NB: in bold a solution to the maximum cardinality b-matching problem

Figure B.10: Structure of graphs $\check{G}^t(\bar{x})$ for Example 3 and different values of $t \in \mathcal{T}$.



NB: in bold a solution to the maximum-weight clique problem

Figure B.11: Structure of $\tilde{G}^d_{\mathcal{S}}(\bar{x})$ for Example 3

However, it is easy to see that the technician-to-task assignment sub-problem does not admit any solution. This comes from two observations. First, it is clear that technician r_7 has to perform tasks I and J. Second, tasks G and H cannot be performed by the same technician since they overlap. The same holds for tasks G and J as well as for tasks H and I. The technician-to-task assignment sub-problem therefore does not admit any solution since technician r_7 cannot perform either tasks G, I and J, or tasks H, I and J.

The resolution of formulation $[SP_2^{LR}(\bar{x})]$ gives an optimum value strictly greater than zero (equal to 1). We then identify the violated BF cut (B.7) (which has here the same expression as a CB cut).

$$x_{p_G} + x_{p_H} + x_{p_I} + x_{p_J} \leqslant 3 \tag{B.7}$$

This example illustrates a case where no MCbM and/or MWC cuts are identified although the technician-to-task assignment sub-problem is infeasible.

Appendix C. Detailed experimental results

Table C.11: Description of the average number of cuts generated in the B&C approach.

Family	All	CB	Other cuts		
			BF	MCbM	MWC
10_2_1_20_A	21	0	0	2	19
10_2_1_20_B	10	0	0	1	9
10_2_1_40_A	36	0	0	0.8	35
10_2_1_40_B	12	0	0	0.6	12
10_2_3_20_A	153	0	94	9	50
10_2_3_20_B	28	0	7	6	15
10_2_3_40_A	66	0	16	9	41
10_2_3_40_B	17	0	0.2	2	15
20_2_1_40_A	67	0	0	1	66
20_2_1_40_B	28	0	0	1	27
20_2_1_80_A	105	0	0	3	101
20_2_1_80_B	18	0	0	1	17
20_2_3_40_A	144	0	17	7	120
20_2_3_40_B	40	0	5	2	32
20_2_3_80_A	118	0	6	5	107
20_2_3_80_B	31	0	0.6	0.8	30
20_4_1_20_A	94	0	0	5	90
20_4_1_20_B	26	0	0	3	23
20_4_1_40_A	202	0	5	12	185
20_4_1_40_B	57	0	0	3	54
20_4_3_20_A	314	0	34	16	263
20_4_3_20_B	84	0	21	13	50
20_4_3_40_A	392	0	38	26	329
20_4_3_40_B	96	0	20	10	66
40_4_1_40_A	229	0	0	3	226
40_4_1_40_B	91	0	0	2	88
40_4_1_80_A	480	0	0	5	476
40_4_1_80_B	151	0	0	4	146
40_4_3_40_A	672	0	65	19	588
40_4_3_40_B	144	0	9	6	129
40_4_3_80_A	1,106	0	102	33	971
40_4_3_80_B	184	0	5	5	174

Table C.12: Detailed computational results on the testbed of the B&C approach coupled with the CPLNS.

Family	**B&C**			**CPLNS**	
	Gap	**#Opt**	**Time**	**Gap**[1]	**Gap**[2]
10_2_1_20_A	-	5/5	9	-	2.6%
10_2_1_20_B	-	5/5	3	-	0.48%
10_2_1_40_A	-	5/5	8	-	3.1%
10_2_1_40_B	-	5/5	3	-	0.52%
10_2_3_20_A	-	5/5	101	-	1.1%
10_2_3_20_B	-	5/5	3	-	0.09%
10_2_3_40_A	-	5/5	13	-	2.5%
10_2_3_40_B	-	5/5	3	-	0.44%
20_2_1_40_A	-	5/5	115	-	0.82%
20_2_1_40_B	-	5/5	5	-	0.22%
20_2_1_80_A	0.02%	4/5	288	5.3%	1.1%
20_2_1_80_B	-	5/5	7	-	0.16%
20_2_3_40_A	0.16%	4/5	35	2.2%	0.28%
20_2_3_40_B	-	5/5	4	-	0.07%
20_2_3_80_A	-	5/5	57	-	0.72%
20_2_3_80_B	-	5/5	6	-	0.15%
20_4_1_20_A	2.1%	3/5	1,130	2.1%	0.29%
20_4_1_20_B	-	5/5	4	-	1.2%
20_4_1_40_A	1.3%	2/5	2,188	4.4%	5.1%
20_4_1_40_B	-	5/5	8	-	2.7%
20_4_3_20_A	2.0%	4/5	340	2.0%	4.3%
20_4_3_20_B	-	5/5	11	-	1.0%
20_4_3_40_A	0.64%	1/5	9,416	3.0%	2.4%
20_4_3_40_B	-	5/5	13	-	1.8%
40_4_1_40_A	2.0%	-	-	2.9%	-
40_4_1_40_B	-	5/5	30	-	0.19%
40_4_1_80_A	1.4%	-	-	4.5%	-
40_4_1_80_B	-	5/5	108	-	0.22%
40_4_3_40_A	0.51%	-	-	1.1%	-
40_4_3_40_B	-	5/5	36	-	0.33%
40_4_3_80_A	1.8%	-	-	4.3%	-
40_4_3_80_B	-	5/5	70	-	0.20%

[1] Takes into account the instances for which the time limit is reached in the *B&C* approach.
[2] Takes into account the instances solved to optimality by the *B&C* approach.

Appendix D. Instance generation

An instance of the problem is primarily characterized by:

- a finite time horizon (a finite number of time periods)
- a number of time periods per day (yielding the number of days)
- a set of locations (wind farms + home depot)
- a set of wind turbines distributed over the wind farms
- a set of maintenance tasks to perform at the different locations and that impact the availability of the turbines
- a set of technicians to perform the tasks
- wind speed for each time period and location
- postponing penalties

The generator is based on the following parameters:

- $n_{\mathcal{T}}$, $n_{\mathcal{D}}$, $n_{\mathcal{I}}$, $n_{\mathcal{S}}$ (length of time horizon, number of days, number of wind farms, number of tasks, and number of skills)
- $Dn^{\mathcal{L}}$: probability distribution of the number of locations
- Dl^{xy}: probability distribution of the coordinates associated with each location
- $Dn^{\mathcal{L}}_{\mathcal{W}}$: probability distribution of the number of turbines per location
- $Dn^{\mathcal{W}}_{\mathcal{I}}$: probability distribution of the number of tasks per turbine
- Δl^{min}: minimum distance between two locations
- Δr^{max}: maximum distance between two locations such that they can be visited by the a technician during the same day
- K: set of all types of preventive tasks that we consider
- $p(k)$: probability of generating a task of type $k \in K$
- $Di_{impact}(k)$: probability distribution of the impact of each type of preventive task on the wind turbines
- $Di_{dur}(k)$: probability distribution of the duration of each type of preventive task
- $Di_{req}(k)$: probability distribution of the number of technicians that can perform each type of preventive task during any time period
- $Dr_{\#skills}$: probability distribution of the number of skills mastered by a technician
- $Dr_{\mathbb{P}(unv)}$: probability that a technician has some unavailability time periods during the time horizon
- $Dr_{\#unv}$: probability distribution of the number of time periods during which a technician is unavailable
- $Dr_{d^{unv}}$: probability distribution of the duration of the unavailability of a technician (in man-hours)
- Dw_{power}: probability distribution of the nominal power (in kW) of each turbine
- $\widehat{\phi}$: average wind speed on each wind farm

- Υ_{max}^{safety}: maximum wind speed allowed to perform a task
- Δl^{max}: maximum distances for the spatial correlation of the wind speed
- δ: number of values used in the moving average for the time-wise dependency between the wind speeds
- α: correlation factor between wind speed

We generate an instance following multiple steps. First of all, the length of time horizon, the number of days, the number of tasks, and the number of skills are input values. This yields directly the set $\mathcal{T}$ of time periods and the set $\mathcal{D}$ of days.

We then start the generation of an instance by building the set $\mathcal{L}$ of locations whose cardinality is set by sampling the $Dn^{\mathcal{L}}$ distribution. According to the distance Δl^{min}, we then generate the coordinates of each location by sampling the Dl^{xy} distribution. Based on these coordinates and on the distance Δr^{max}, we compute the parameters $(\delta_{ll'})_{(l,l')\in\mathcal{L}^2}$ that enable to define the daily location-based incompatibility constraints.

Afterwards, we built the set $\mathcal{W}$ of wind turbines. To this end, according to the target number of tasks, we start by generating a number of wind turbines per locations by sampling the $Dn_{\mathcal{W}}^{\mathcal{L}}$ distribution. For each location where there is at least one wind turbine (i.e., this location is a wind farm), we then generate a nominal power by sampling the Dw_{power} distribution and we set the nominal power P_w equal to this latter value for each wind turbine $w \in \mathcal{W}$ of the wind farm.

After that, we call procedure **genTasks**() to create the set $\mathcal{I}$ of tasks. Notice that for each task $i \in \mathcal{I}$ we build the set $\mathcal{M}_i$ of execution modes such that it meets the two following requirements:

- $\forall m, m' \in \mathcal{M}_i,\ q_{im} \neq q_{im'}$,
- $\forall m, m' \in \mathcal{M}_i,\ q_{im} < q_{im'} \rightarrow d_{im} > d_{im'}$.

Arbitrarily, we build $ov(\mathcal{I})$ considering that overlapping tasks are forbidden on the same turbine. Notice that, according to some experts in the field, it is reasonably realistic to only consider these subsets. After the generation of the tasks, we generate the set $\mathcal{R}$ of technicians using procedure **genTechnicians**().

The last part of the generator concerns the parameters related to the objective function. For the sake of convenience, we introduce the set $\mathcal{T}^+$ of all time periods formed by the union of set $\mathcal{T}$ and the set of rest time periods that occur between each day. More specifically, we include a rest time period after every $\frac{|\mathcal{T}|}{|\mathcal{D}|}$ consecutive time periods of $\mathcal{T}$.

As it concerns the wind speed at hub height, the main purpose is to use realistic values. First, we generate wind speed $\bar{\phi}_l^t$ for every location $l \in \mathcal{L}$ and every time period $t \in \mathcal{T}^+$ using a Rayleigh distribution with a scale parameter equal to $\widehat{\phi}\sqrt{\frac{2}{\pi}}$ (so that the expected wind speed is $\widehat{\phi}$). Since space correlation can be significant, we compute a corrected wind speed $\bar{\bar{\phi}}_l^t$ for every location l and every time period t as follows:

$$\bar{\bar{\phi}}_l^t = \frac{\sum\limits_{\substack{l'\in\mathcal{L} \\ s.t.\ \Delta_{ll'}<\Delta l^{max}}} (\Delta l^{max} - \Delta_{ll'})\, \bar{\phi}_{l'}^t}{\sum\limits_{\substack{l'\in\mathcal{L} \\ s.t.\ \Delta_{ll'}<\Delta l^{max}}} (\Delta l^{max} - \Delta_{ll'})}.$$

Wind speeds were generated independently from a time period to another one. However, this time-wise independence assumption is unlikely to be verified in practice. To smooth out the speed-values, we use a

Procedure genTasks

$\mathcal{I} \leftarrow \emptyset$
for $i \in \{1, ..., n_{\mathcal{I}}\}$ **do**
  ; /* Creation of a new task i */
  Associate randomly a wind turbine to task i by sampling the $Dn_{\mathcal{I}}^{\mathcal{W}}$ distribution
  Define the type $k \in K$ of the task according to the probabilities $p(k)$
  Define the impact of the task on the wind turbines by sampling the $Di_{impact}(k)$ distribution
  Draw randomly the skill s_i required by task i from the set $\mathcal{S}$
  Set the minimal (q_i^{MIN}) and the maximal (q_i^{MAX}) numbers of technicians that can perform task i at any given time period by sampling the $Di_{req}(k)$ $Dn_{\mathcal{I}}^{\mathcal{W}}$
  Generate a task duration d_i by sampling the $Di_{dur}(k)$ distribution
  $n_{\mathcal{M}_i} \leftarrow q_i^{MAX} - q_i^{MIN} + 1$
  $\mathcal{M}_i \leftarrow \emptyset$
  ; /* d_i^{prev}: duration of the last executing mode created for task i */
  for $m \in \{1, ..., n_{\mathcal{M}_i}\}$ **do**
    Create executing mode m for which task i requires $q_i^{\mathcal{M}}$ technicians and lasts $d_i^{\mathcal{M}}$ time periods with:
      $q_i^{\mathcal{M}} \leftarrow q^{MAX} - m + 1$
    ; /* We assume that the duration of a working day is 8 hours. */
      $d_i^{\mathcal{M}} = \max(d_i^{prev} + 1, \lfloor \frac{d_i |\mathcal{T}|}{8 |\mathcal{D}| q_i^{\mathcal{M}}} + 0.5 \rfloor$
    Add the created executing mode to $\mathcal{M}_i$
    $d_i^{prev} \leftarrow d_i^{\mathcal{M}}$
  end
  Add the created task i to $\mathcal{I}$
end

δ-weighted moving average that yields wind speed ϕ_l^t according to the following formula:

$$\phi_l^t = \frac{\bar{\phi}_l^t + \sum_{t'=\max(0,t-\delta)}^{\max(0,t-1)} \alpha^{t-t'} \phi_l^{t'}}{1 + \sum_{t'=\max(0,t-\delta)}^{\max(0,t-1)} \alpha^{t-t'}}.$$

The resulting values are rounded to the nearest tenth. From our perspective, they compare well to realistic data.

Afterwards, for each task $i \in \mathcal{I}$ and every time period $t \in \mathcal{T}$, we compute the binary parameter $\widetilde{\vartheta}_i^t$ equal to 1 if and only if $\phi_l^t < \Upsilon_{max}^{safety}$ (i.e. the task i can be scheduled during time period t according to safety concerns). Arbitrarily, we set each parameter ϑ_i^t equal to 1 for every task i and every time period t. We point out here that this choice makes the instances more complicated to solve as there is a wide flexibility to schedule the maintenance operations. This also matches field observations.

The last step consists in computing the revenue value g_w^t for every wind turbine $w \in \mathcal{W}$ during each time period $t \in \mathcal{T}^+$. We compute the revenue from the nominal power P_w of the wind turbine and from the wind speed ϕ_l^t. We also use an estimation $hours(t)$ of the number of hours during every time period t. More specifically, we compute the revenue g_w^t generated by each turbine $w \in \mathcal{W}$ that is available during time period $t \in \mathcal{T}$ as follows:

Procedure genTechnicians

Let d^{unv} be the average number of time periods during which a technician is not available according to $Dr_{\#unv}$ and $Dr_{d^{unv}}$.
$\mathcal{R} \leftarrow \varnothing$
for $s \in \{1, ..., n_{\mathcal{S}}\}$ **do**
    ; /* compute the average total request of the tasks RS_s^{avg} */
    $RS_s^{avg} = \sum_{\substack{i \in \mathcal{I} \\ s_i = s}} \frac{1}{|\mathcal{M}_i|} \sum_{m \in \mathcal{M}_i} q_{im}$
    ; /* n_s minimum number of technicians mastering skill s */
    $n_s \leftarrow \epsilon \cdot \frac{RS_s^{avg}}{d^{unv}}$
    for $r \in \{1, ..., n_s\}$ **do**
        Create a technician mastering skills s and generate his or her unavailability time periods by sampling the $Dr_{\#unv}$ and $Dr_{d^{unv}}$ distributions
        Add this technician to $\mathcal{R}$
    end
end
for $r \in |\mathcal{R}|$ **do**
    Sample the $Dr_{\#skills}$ distribution to generate the number of skills mastered by technician r
    According to the previous value, generate additional skills for technician r
end

$$g_w^t = 0.08 \cdot P_w \cdot hours(t) \cdot CF(\phi_{l_w}^t).$$

where

- 0.08: is an approximation to the selling price in euros of 1 kWh of wind energy (this selling price is guaranteed for the next 10 years in France).
- $hours(t)$: estimation of the number of hours during time period $t \in \mathcal{T}^+$
- P_w: nominal power of wind turbine $w \in \mathcal{W}$
- $\phi_{l_w}^t$: wind speed during time period t at the location of turbine $w \in \mathcal{W}$
- $CF(\phi)$: the ratio of the net electricity generated according to a wind speed equal to ϕ to the energy that could have been generated at full-power operation (this ratio is given by a piecewise linear function estimated from real data)

Finally, we compute a single postponing penalty set equal for each task. This penalty is equal to the maximum loss of revenue that can be generated by the scheduling of a task of $\mathcal{I}$ plus one. With this definition, we almost always (if not always) ensure that postponing a task is non-profitable. With this penalty we therefore almost ensure to schedule the maximum number of tasks according to the total number of technicians and their availability. This is quite in line with the practice in the field.

Table D.13 presents the detail parameter setting used in the generation process.

Table D.13: Detail parameter setting used by the instance generator

Parameters	**Values**							
$(n_{\mathcal{T}}, n_{\mathcal{D}})$	$(10,5)$		$(20,5)$		$(20,10)$		$(40,10)$	
$n_{\mathcal{I}}$	20	40	20	40	40	80	40	80
$n_{\mathcal{S}}$	1 *or* 3							
$Dn^{\mathcal{L}}$	$\mathcal{U}(\{5,...,10\})$	$\mathcal{U}(\{8,...,12\})$	$\mathcal{U}(\{5,...,10\})$	$\mathcal{U}(\{8,...,12\})$	$\mathcal{U}(\{8,...,12\})$	$\mathcal{U}(\{12,...,20\})$	$\mathcal{U}(\{8,...,12\})$	$\mathcal{U}(\{12,...,20\})$
Dl^{xy}	$\mathcal{U}(\{0,...,80\}) \times \mathcal{U}(\{0,...,80\})$							
$Dn^{\mathcal{L}}_{\mathcal{W}}$	$\mathcal{U}(\{1,2,...,12\})$							
$Dn^{\mathcal{W}}_{\mathcal{I}}$	$\mathcal{U}(\{0,1,2\})$							
K	main maintenance tasks				inspection, gearbox oil change, other operations...			
$p(k)$	0.5				0.5			
$Di_{impact}(k)$	$\mathcal{U}(\{ST, BL\})$				$\mathcal{U}(\{NST, ST, BL\})$			
$Di_{dur}(k)$	$\mathcal{U}(\{20,40,60,80\})$				$\mathcal{U}(\{8,16\})$			
$Di_{req}(k)$	$2,3,4$				$2,3$			
$Dr_{\#skills}$	$\mathcal{U}(\{1,...,\lvert S\rvert\})$							
$Dr_{\mathbb{P}(unv)}$	0.25							
$Dr_{\#unv}$	1							
Dr_{dunv}	$\mathcal{U}(\{1,...,2\})$							
ϵ	1 *or* 1.2							
Dw_{power}	$\mathcal{U}(\{2000,2500,3000\}$							
Δl^{min}	5							
Δr^{max}	25							
$\hat{\phi}$	6							
Υ^{safety}_{max}	12							
Δl^{max}	15							
α	0.9							
δ	1		2		1		2	
$hours(t)$	$hours(t) = \begin{cases} 14 & \text{if } t \text{ is a rest time period} \\ 5^* & \text{if } t \text{ is not a rest time period and } \frac{\lvert\mathcal{T}\rvert}{\lvert\mathcal{D}\rvert} = 2 \\ 2.5^* & \text{if } t \text{ is not a rest time period and } \frac{\lvert\mathcal{T}\rvert}{\lvert\mathcal{D}\rvert} = 4 \end{cases}$							
$CF(\phi)$	piecewise linear approximation with the following 9 points							

ϕ	0	3.5	5.5	7	12.5	14	24.9	25	30
$CF(\phi)$	0	0	0.1	0.23	0.91	1	1	0	0

***** We consider that the turbine is stopped slightly longer than the duration of a task (that has an impact on its execution) because the turbine has often to be stopped a bit earlier than the starting time of this task .

NS No turbines are shut down during the execution of the task.

ST The task impacts only the turbine on which it is executed. This turbine is stopped during the execution of the task (not during the rest time periods the task overlaps).

BL The task impacts only the turbine on which it is executed. This turbine is stopped during the execution of the task and during the rest time periods the task overlaps.

Remark: The model can also consider tasks that shut down multiple turbines in a wind farm (e.g., the maintenance of the wind farm substation), but this is extremely rare in practice.