A branch-and-check approach to solve a short-term maintenance scheduling problem faced by the onshore wind industry

A. Froger[1,2], M. Gendreau[3], J.E. Mendoza[2], E. Pinson[1], L-M Rousseau[3]

[1] Université Bretagne Loire, Université Catholique de l'Ouest, LARIS EA7315, Angers, France
eric.pinson@uco.fr

[2] Université François-Rabelais de Tours, CNRS, LI EA 6300, OC ERL CNRS 6305, Tours, France
{aurelien.froger,jorge.mendoza}@univ-tours.fr

[3] CIRRELT et Département de Mathématiques et de Génie Industriel, Polytechnique Montréal, Montréal, Canada
michel.gendreau@cirrelt.ca,louis-martin.rousseau@cirrelt.net

Mots-clés : *scheduling, integer linear programming, branch-and-check, cuts generation.*

1 Introduction

Although maintenance scheduling in the electricity industry has been widely studied, we observe that the contributions of Operations Research essentially target thermal and nuclear power plants [5]. 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 and the constantly-decreasing cost of turbines, wind energy is currently the world's fastest-growing source of electricity. In this context, efficient wind turbine maintenance planning and scheduling becomes a critical tool to prevent unnecessary loss of production and excessive operational costs. We discuss here a new and challenging short-term maintenance scheduling problem rising in onshore wind farms. Our primary contribution is to propose an efficient exact approach for solving this problem.

2 Problem statement

The aim of the problem is to schedule a set of non-preemptive maintenance tasks during a discrete and finite planning horizon (that spans over several days) in order to maximize the revenue generated by the electricity production of a set of wind turbines. We estimate the revenue according to the forecasted wind speed. The tasks have different impacts on the availability of the turbines to produce electricity. Some tasks shut down one (or more) turbine(s) since the task starts until the task ends. Some tasks shut down the turbines when the technicians are effectively working on the task, but not necessarily during the nights or the weekends they overlap. Other tasks do not have any impact on electricity production.

To execute the maintenance operations, we have a finite set of multi-skilled technicians. Each task requires technicians mastering a specific skill. 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. 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. We assume that all the technicians work the same shift, which is a common practice in this industry. Each technician has also an individual availability schedule 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.

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 multiple days. 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. Additionally, some subsets of tasks cannot overlap due, for instance, to the use of disjunctive resources, an interference (e.g., two tasks cannot be executed on the same turbine at the same time), or managerial preferences.

One particularity of this problem is the possibility to postpone the scheduling of some tasks until the next planning horizon. When a task is postponed, it penalizes the objective. The value of this penalty is fixed according to the relative degree of priority of the task. This priority depends on reliability consideration (the more a maintenance operation is delayed, the higher is the probability of failure) and contract commitments. If the penalties are high enough, postponing a task is just triggered to overcome a possible lack of technicians. 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). 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. By polynomially reducing the cumulative scheduling problem – known to be NP-complete in the strong sense [1] – to the decision problem[1] associated with WTMSP, we prove that WTMSP is strongly NP-hard.

3 The decomposition approach

To tackle WTMSP, we present an exact approach that takes advantage of the intrinsic decomposition of the problem into a restricted master problem (RMP) and a sub-problem (SP). Solving RMP consists in scheduling the tasks (or to decide to postpone them) in order to maximize the difference between the revenue from the wind electricity production and the postponing penalties. In this problem, technician considerations have been partially dropped. From a solution to RMP, SP checks if there exists an assignment of the technicians to the tasks complying with the daily location-based incompatibilities and coping with individual resource availability time periods. It is worth noting that SP is NP-complete. This can be proved by its equivalence to a L-coloring problem. More precisely, we associate a vertex with each *job* (either a task or an unavailability time period of a technician) and consider an edge between two vertices if the underlying jobs overlap or cannot be performed by the same technician. We choose a different color for each technician and associate with each vertex a list of all its possible colors. The aim of our approach is to design a coordination procedure between RMP and SP.

To efficiently solve WTMSP while exploiting the decomposition previously described, we generate cuts on the fly while solving an integer linear programming (ILP) formulation of RMP. More specifically, at each integer node of the branch-and-bound tree, we check the feasibility of

[1] 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.

SP for the current solution (to RMP). We introduce cuts in RMP to discard solutions that are infeasible for the whole problem. This implementation approach is referred to as a *Benders-based branch-and-cut* algorithm in [6], and it is also similar to the *branch-and-check* (B&C) framework described in [7] (even if it is primarily used for linear programming and constraint programming hybridization). We choose to refer to our method as a B&C approach. This approach has the advantage to provide feasible solutions throughout the resolution of RMP. It also overcomes the drawback to solve the ILP model several times, which would be too time-consuming. 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 efficiency of the cut generation process relies primarily on identifying the reason(s) of the infeasibility of SP in order to discard the largest set of infeasible solutions to RMP. Therefore, at each integer node of the branch-and-bound, we proceed as follows to check the feasibility of the associated solution:

1. We start by solving two approximations of SP:
   For each time period t, we solve a maximum-cardinality b-matching (MCbM) problem in a bipartite graph where the first group of vertices represents the tasks running during t, the second group of vertices represents the available technicians during t, and there exists an edge between two vertices if the underlying technician can be assigned to the corresponding task. If during a time period, the maximum cardinality of a b-matching is strictly shorter than the total number of requested technicians, then SP is infeasible. In that case, we generate a cut (hereafter referred to as a MCbM cut) applying the max-flow/min-cut theorem. Sometimes, it is possible to strengthen the previous MCbM cut by reasoning about its composition. This potentially allows to invalidate more infeasible solutions to RMP.

   For each day d and each subset of skills $\bar{\mathcal{S}}$, we solve a maximum-weight clique (MWC) problem in a graph where the vertices represent a *job* overlapping d and requiring technicians mastering a skill in $\bar{\mathcal{S}}$. A job is either a task or a technician's unavailability time period. The weight of each vertex is equal to the number of technicians needed to perform the job. We consider an edge between two vertices if the underlying jobs overlap or cannot be performed by the same technicians as regards to the daily location-based incompatibilities. If there exists in this graph a clique with a weight strictly greater than the total number of technicians mastering a skill in $\bar{\mathcal{S}}$, then SP is infeasible. In that case, we introduce a cut (hereafter referred to as a MWC cut) that states that the number of technicians required by the jobs associated with the vertices of the clique has to be lower than the total number of technicians mastering at least a skill in $\bar{\mathcal{S}}$. We show that it is possible to tighten this cut by adding some additional jobs on its left hand side. Notice that some arguments help us to work with relatively small-sized graphs.

2. If no cuts have been generated during the first stage, we solve the continuous relaxation of an ILP formulation of SP. If we identify a violated Benders feasibility cut, we add it to the restricted master problem. Otherwise, since the constraint matrix of the model is not totally unimodular; we then solve the ILP formulation of SP. If this states that we cannot find a feasible assignment of the tasks to the technicians, we generate a combinatorial Benders cut [2] to invalidate the current solution to RMP. Otherwise, the branch-and-bound scheme ensures that the current solution is strictly better than the best previous solution.

To reduce the size of the branch-and-bound tree, we also generate MCbM and MWC cuts at the root node (as long as the continuous solution leads to the generation of at least one cut). We refer the reader to [4] for detailed explanations of the method (e.g., models, expression of the cuts).

Notice that the scheme of our method (and the previous mentioned cuts) could apply (although not providing a framework) to other scheduling problems, especially those where cumulative constraints have to be considered for resources that may travel between locations.

4 Results

Since our problem is new to the maintenance scheduling literature, no publicly available benchmarks exists. We therefore took advantage of our close collaboration with companies specializing on wind predictions, wind turbine maintenance, and maintenance scheduling software, to get inside knowledge on how real-data for the problem looks like. Based on this knowledge, we built an instance generator that we believe captures reality with a good degree of accuracy.

We used our generator to build a 160-instance testbed. For each instance, we consider time horizons of different lengths (10, 20, or 40), different number of time periods per day (2 or 4), different number of tasks (20, 40, or 80), and different number of skills (1 or 3).

We observe that the B&C approach outperforms by far the direct resolution of ILP formulations. Indeed we are able to solve to optimality 80% of the instances and, in this case, the solution time is short (around 3 minutes on average). Moreover, the overall average gap when optimality is not reached in the 3-hour time limit (this only happens for the large-sized instances) is relatively small (1.7%). We observe that the performance of the B&C approach is strongly correlated with the cuts generated from the approximations to SP. Indeed, including the MCbM and MWC 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%. According to our experiments, the difficulty of an instance increases with the number of time periods per day, the tightness of the technicians-to-work ratio, and, to a less extent, the number of tasks.

We also tried to take advantage of solutions provided by a previously developed metaheuristic [3], a constraint programming-based large neighborhood search (CPLNS). The idea is to run the CPLNS in parallel with the algorithm. 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. While this has no significant effect on the efficacy and efficiency of the B&C, the results of our tests allow us to state that for instances with a regular technicians-to-work ratio the B&C approach outperforms the CPLNS.

References

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

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

[3] A. Froger, M. Gendreau, J.E. Mendoza, E. Pinson, and L-M. Rousseau. Solving a wind turbine maintenance scheduling problem. Under review.

[4] A. Froger, M. Gendreau, J.E. Mendoza, E. Pinson, and L-M. Rousseau. A branch-and-check approach to solve an onshore wind turbine maintenance scheduling problem. Technical report CIRRELT-2016-62, CIRRELT, *Available at* https://www.cirrelt.ca/DocumentsTravail/CIRRELT-2016-62.pdf, 2016.

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

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

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