The CP Shortcut: Solving the High-Power Pump Activation Problem without the Overkill

Mohamed-Anass Gallass
UCLouvain, Belgium

Philippe Greiner
Technord, Belgium

Antoine Tuerlinckx
Technord, Belgium

Hélène Verhaeghe
UCLouvain, Belgium

Abstract

Constraint Programming (CP) has been around for decades, yet it remains largely unknown in industry. When faced with combinatorial optimization problems, industry practitioners not knowledgeable in CP techniques often resort to more creative but not necessarily adequate solutions.

This paper is the result of an actual case study brought by Technord, an industry consultant. The problem at hand is the optimization of the activation schedule for high-power pumps in a water treatment facility under fluctuating energy costs. The schedule was previously generated using Discrete Particle Swarm Optimization (DPSO). This method struggled with the increasing complexity of volatile market signals and strict operational constraints. Our simpler model, developed in Python using the CPMpy library, formalizes the problem as a CP model. Our experiments demonstrate the benefits of our model's simplicity compared to the DPSO solution. This use case also showcases the importance of the accessibility of constraint modelling solutions for less knowledgeable practitioners.

2012 ACM Subject Classification Computing methodologies → Optimization algorithms; Computing methodologies → Combinatorial algorithms

Keywords and phrases Application, Industrial Problem, Modelling

Digital Object Identifier 10.4230/LIPIcs.CP.2026.55

Category Short Paper

Related Version Code repository: `https://github.com/anassgallass/Pump-CP-Shortcut`

1 Introduction

Constraint Programming (CP) has been around for several decades [15]. However, it remains less known, especially in industry, despite numerous successful industrial applications [1, 2, 9, 16, 19, 21], particularly in scheduling. This paper presents an industry case.

The rapid decarbonization of the European energy sector has introduced significant volatility into the electrical grid. As the grid mix moves away from traditional baseload generation, electricity prices have become strongly correlated with renewable availability, serving as a critical signal for grid stability. In this environment, industries need to adapt to maintain operational cost-efficiency and support grid reliability.

This paper addresses the scheduling of high-power pump operations at an actual water treatment facility. The constraint model presented here integrates the physical infrastructure limits and industrial safety margins. The total cost of the operations has to be minimized, given time-varying day-ahead energy prices. Typically, approaches to solve similar problems include Mixed-Integer Linear Programming (MILP) formulations or meta-heuristics. It seems Constraint Programming approaches were never proposed. At the facility, the old naive, unoptimized approach was improved some time ago using Discrete Particle Swarm Optimization (DPSO) by the consultancy agency addressing the problem.

The design of the constraint-based solution proposed follows a CPMpy-based [7] approach targeted at industrial users seeking to try constraint-based technologies to solve their problems without requiring extensive expertise. The solution was designed in accordance with the hypothesis, parameters, and demands provided initially by the facility.

The paper is organized as follows: **(i)** first, it recalls the related work about the problem, **(ii)** then the facility and the problem are presented, **(iii)** third, the former solutions are presented, **(iv)** then, our proposed CP model is detailed, and **(v)** finally, the results of a few experiments on historical data are discussed.

2 Related works

The problem at hand falls within the diversity of Pump Scheduling Problems (PSP). This family of problems involves determining when to operate pumps to efficiently meet fluctuating water demands while respecting operational constraints. Multiple variants of the problem exists as described by the survey [12].

In [4], the authors study pump scheduling for transferring water from a low- to a high-altitude tank under participation in the day-ahead energy market, incorporating price uncertainty and stochastic water demand. The problem is solved daily to plan electricity purchases and again in real time for pump operation, using an MILP formulation. Metaheuristic approaches have also been explored; for example, [20] applies Simulated Annealing and Hybrid Genetic Algorithms to schedule pumps for both reservoir filling and water intake to the treatment unit. In contrast, the present use case differs in several key aspects, including known water demand, a different facility configuration, distinct operational constraints (e.g., human-operated pumps), and imposed pump priority rules. As a result, these existing models cannot be directly applied.

Another problem of the family is the optimal water flow in distribution networks, which involves operating pumps and valves across tanks, reservoirs (inputs), and consumers (outputs). Studies such as [22] and [18] formulate it as a mixed-integer nonlinear program (MINLP) to minimize pump energy costs. To reduce complexity, approximation [3], linearization [17], and relaxation [5] techniques have been proposed.

3 Description of the Water Treatment Facility and the Problem

The water treatment facility (Fig. 1) operates as follows. First, it pumps non-drinkable river water. The water is treated in the treatment unit. The now potable water is then stored in buffer tanks. Before being sent to the water distribution network, the water is moved to a larger reservoir at a higher altitude to facilitate injection into the network. The transfer between the tanks is performed using high-power pumps.

The operation of the facility follows a given number of constraints and hypotheses:

- *Production*: Each day, the network issues an order for w units of water. The treatment unit is thus set up to treat $\frac{w}{24}$ units of water per hour, or $\frac{w}{24*4}$ units of water per 15 min.
- *Human operators*: The activation/deactivation of the pumps is processed by human operators according to a schedule. A maximum of M operations is allowed each day, and operations are scheduled in 15-minute increments.
- *Buffer tanks*: The 2 tanks at the facility are required to remain filled between a min $V_{\min}$ and a max $V_{\max}$ volume. As they are linked, one can assume a single hypothetical tank.

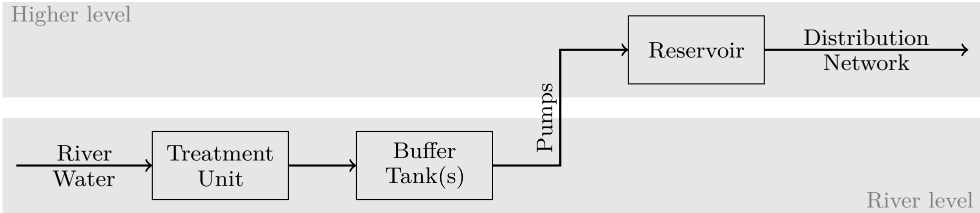


Figure 1 Schema of the water treatment plant

- *Reservoir*: The reservoir always contains sufficient water to meet the day's demand and has sufficient capacity to receive the day's production. Producing the same amount ordered each day helps maintain this hypothesis, allowing for any pump schedule.
- *Pump specifications*: There are 7 pumps. Each pump has its own power and flow. Pumps are either on or off; their flow cannot be modulated. Pumps are sorted in decreasing order of efficiency. When k pumps are activated, they are always the first k pumps in this order. Due to pipeline capacity, at most P_{limit} pumps can be active simultaneously. Given P_{usable}, the number of pumps not in maintenance, $P = \min(P_{limit}, P_{usable})$ is the number of pumps available. Also, to avoid premature pump wear and grid instability, the facility requires that setups remain unchanged for at least 30 minutes.
- *Energy costs, duration, and start of a schedule*: The current solution starts its computation at 16 h and is required to provide a plan for the next 24 h. It uses 40 h of prices provided by the facility (forecasted from the day-ahead market). Schedules are currently computed for 40 h, but only the first 24 h are actually used. The goal is to already account for the energy cost trend for the day ahead, thereby providing a longer-term view (useful for assessing whether it is better to keep the buffer tank at a lower or higher level at the end of the day). As settings do not change during a 15-min time slot, a fixed price for each 15-minute time slot is preprocessed from the real-time forecasted price.
- *Robustness of the plan*: Data have been collected in real time for the past months. They show that there is sometimes a small delay of up to 1-2 minutes between the planned and actual activation/deactivation of the pump. Moreover, pumps may take a few minutes to reach full flow upon activation and to fully stop. The buffer tank levels have been computed to accommodate small discrepancies arising from the application of the plan and the modelling hypotheses adopted. Given the available historical data, one can confidently conclude that these threshold levels are effective and that any plan that respects them will enable the facility to operate in a feasible manner.

The main decision in day-to-day operations is the operation of the pumps that move water from the buffer tanks to the reservoir. These are the equipment that requires the most energy in the facility, as they must move large volumes of water uphill. Because energy costs vary throughout the day, different schedules can result in substantial cost differences.

The problem is thus to decide when to activate/deactivate the pumps, to move the w units produced that day from the buffer tanks into the reservoir while optimizing pump operating costs by accounting for the day's evolving electricity prices.

This leads to the definition of the optimization problem at hand: finding the cost-optimal activation schedule for the high-power pumps subject to the treatment facility's operating constraints, given fluctuating energy costs.

Summary of the parameters. The model uses the following parameters:

- The plan is H hours long ($H = 40$), with $T = \frac{H}{0{,}25}$ 15-minutes time slots to plan.

- The mean price of the forecast over time slot t is π_t (€/kWh).
- The maximal volume of the tanks is V and safety min and max volumes $V_{\min}$ and $V_{\max}$ ($0 < V_{min} < V_{max} < V$) and the initial volume of water in the buffer V_0.
- The quantity of water produced over time slot t is w_t.
- The number of available pumps for the day is P ($P = 4$ in normal conditions (NC)).
- The maximum pump switches (on or off) on a 1-day span is M ($M = 24$ in NC).
- The flow parameter vector $f[k]$, with $k \in \{0, ..., P\}$, contains the quantity of water pumped in 15 min, given k active pumps. It is precomputed using physical formulas.
- The power parameter vector $p[k]$, with $k \in \{0, ..., P\}$, contains the power consumed in 15 min, given k active pumps. It is precomputed as the cumulative sum of the powers.

4 History of the solutions in place

4.1 Old Regime

The initial solution stemmed from the former electricity cost system. Prior to the increase in photovoltaic electricity production, electricity costs were billed under a bi-timing system. During the day, when people were active and most industries were operating, electricity costs were higher. During the night, when people were sleeping and many industries were not working, the cost was low. This stemmed from the fact that electricity production relied primarily on nuclear reactors. This type of electricity production is very difficult to modulate on a day-to-day basis, and even impossible on an hour-to-hour basis, leading to constant energy production and a price incentive for the public to use electricity when it is less likely to be used. The initial solution was to allow the buffer tank to fill during the day, pumping as little as possible to the reservoir at that time, while emptying the buffer tank at night. However, once photovoltaic solutions were democratized, energy production became less stable, and the bi-timing system did not align demand with production. From that time onward, electricity was billed at prices that fluctuated over time. This motivated the water treatment facility to adopt a different solution.

4.2 DPSO solution

The solution proposed by the consulting firm is based on a meta-heuristic algorithm called Discrete Particle Swarm Optimization (DPSO). In this meta-heuristic framework, a population (*swarm*) of candidate solutions (*particles*) explores the T-dimensional search space, where T represents the number of time intervals in the planning horizon. Each particle i at iteration k is defined by its position vector $x_i(k) \in \mathbb{N}^T$, representing a specific pumping schedule, and its velocity vector $v_i(k) \in \mathbb{R}^T$.

The initial swarm x_0 is initialized by sampling n particles using a Probability Density Function (PDF). This PDF is the result of translating electricity price volatility, reflecting the economic preference for pumping.

During the algorithms, particles follow trajectories influenced by their own historical success and the collective performance of the swarm. The velocity is influenced by an inertia weight, acceleration coefficients, a stochastic component, the particle's best-so-far solution, the swarm's best-so-far solution, and the number of authorized pumps. To ensure that each particle corresponds to a pump planning (i.e., that the vector is composed of integer values from 0 to P, a rounding and a clipping operation are applied.

Since DPSO does not allow direct enforcement of constraints, constraints are integrated into the objective function. The objective is composed of three elements:

- First, the monetary cost of the solution, computed given the schedule and the price.
- Secondly, a penalty counting the number of slots violating the safety thresholds $V_{min/max}$.
- Third, a penalty counting the number of switches in the plan.

Over multiple iterations, the swarm converges to schedules that minimize electricity costs.

In production at the plant, DPSO solutions are computed in two steps. First, at 16:00 each day, the algorithm runs for 10 min to generate an initial solution with a 40-h horizon. The first 2 hours of this solution (corresponding to operations from 16:15 to 18:00 included) are sent to the operators. During these 2 hours, the algorithm continues to run with the same parameters to improve the solution for the remainder of the plan. The next 22 h of the plan are sent to the operators (operation from 18:15 to 16:00 the next day). At 16:00 the next day, a new schedule is computed based on the updated price information.

5 Model of the Problem

The first step in solving the problem is to reformulate it as a constrained optimization problem. This section explains the formalism constraint by constraint.

Decision Variables. For each time slot $t \in \{0, ..., T-1\}$, the main goal is to determine the number of active pumps $n_{active}[t]$. The domains of these variables consist of the values from 0 (i.e., all pumps are deactivated) to P (i.e., all pumps are active). Variables $q[t]$ are defined to model the outflow of water from the buffer tanks to the reservoir. Their domains span from 0 (i.e., no pumps so no flow) to $f[P]$ (i.e., maximum number of pumps so maximal flow). A third set of variables, $V[t]$, is used to model the volume in the buffer tank at the start of the time slot t. Their domains span from 0 (i.e., empty tanks) to V (i.e., tank completely full). $V[T]$ represent the water volume at the end of the plan.

Water flow. The quantity of water moved by the active pump during time slot t can thus be constrained by an element constraint: $q[t] = f[n_{active}[t]] \quad \forall t \in \{0, ..., T-1\}$. This leads to the constraint computing the evolution of the volume of the combined buffer tanks as $V[t+1] = V[t] + w_t - q[t] \quad \forall t \in \{0, ..., T-1\}$. The initial volume is specified by $V[0] = V_0$.

Reservoir constraints. Due to the small operating errors, the volumes in the tank can actually go a bit under/above the minimal/maximal threshold. The model should thus allow the situation (e.g., if the levels are so when the computation for the next day starts). Constraints are added to force the return to normal levels in that situation: $(V[t] > V_{\max}) \rightarrow (V[t] > V[t+1])$ and $(V[t] < V_{\min}) \rightarrow (V[t] < V[t+1])$.

Pumps switches. The next constraint ensure that the number of operation in one day don't exceed the maximum allowed: $\sum_{i=j}^{j+24\times4-1} |n_{active}[i] - n_{active}[i+1]| \leq M$ for all time slots j considered as the beginning of a day ($j \in \{0, (24 \times 4)\}$ in our settings). Also, settings are not allowed to change again if they were modified the previous time slot, leading to the constraint: $(n_{active}[t-1] \neq n_{active}[t]) \rightarrow (n_{active}[t] = n_{active}[t+1])$.

Objective. The objective is to minimize the overall cost of the operation of the schedule, i.e., the sum of the cost for each time slot: $\min Cost = \min \sum_{t=0}^{T-1} \pi_t \times p[n_{active}[t]]$.

5.1 Finding the Right Solving Tool

Finding the right solver for a problem is often difficult, especially for problems that are not closely related to known ones. Modelling libraries, such as CPMpy [7], PyCSP [11], MiniZinc [13], and Essence [6], can simplify the process for users who are less knowledgeable about constraint-solving techniques. Such libraries allow the user to formulate their problem as a high-level constraint model first. Then, the user can select from a collection of supported

	Demands (m^3/day)			Prices (c€/kWh)			
	Min	Mean	Max	Min	Mean	Max	
May	82.000	99.258	117.000	-1,00	6,04	9,98	avg demand, low prices
July	62.000	75.290	127.000	4,06	8,39	17,72	low demand, high prices
December	127.000	143.161	157.000	6,08	8,54	11,9	high demand, avg prices

Table 1 Characteristics of the three months (year 2025) of data used for the simulations. The negative cost stems from the combination of a high photovoltaic panel density and a bright month.

		AR		DPSO		Or-Tools		CP Optimizer		Gurobi	
		24 h	40 h	24 h	40 h	24 h	40 h	24 h	40 h	24 h	40 h
May	S1	3036	7160	2773	6744	2260	5681	2300	5728	**2255**	**5659**
	S2					2282	**5731**	2309	5787	**2280**	5746
	S3					2350	5898	2376	5930	**2306**	**5819**
Jul.	S1	3317	7219	2391	5736	2234	**5133**	2235	5138	**2226**	**5133**
	S2					2240	**5271**	2256	5284	**2233**	5278
	S3					**2281**	**5244**	2296	5271	2332	5337
Dec.	S1	6473	12840	6085	12232	5976	11830	6009	11874	**5970**	**11817**
	S2					**5931**	**11943**	5968	12006	5933	11947
	S3					5887	11706	5928	11747	**5884**	**11698**

Table 2 Mean cost (€/day) over the full month for each scenario and each model. Cost over the first 24 h actually used and cost over the full computed plan

solvers across various solving technologies (CP, SAT, MIP, ...). Using such tools can greatly facilitate benchmarking, allowing one model to test multiple paradigms simultaneously.

In this case, the model was implemented in CPMpy, leveraging the common use of Python for the current solution and data-collection platform. Moreover, the company wishes to use a commercial-grade solver for the final solution. This drove benchmarking using Gurobi [8] (motivated by the use of MILP in [4]), CP Optimizer [10], and Or-Tools [14].

6 Experiments

The results are compared with two baselines. First, baseline *AR* estimates of a very naive solution in which we assume the availability of a hypothetical pump that we could set up at the exact flow rate of the production of the day. In practice, the power required by the plan on a given day is summed and averaged over the schedule. Second, baseline *DPSO* corresponds to the DPSO solution engineered by the consulting firm. Experiments are conducted in scenarios replicating actual facility conditions, using the facility's actual data for the parameters. The experiments were conducted on three months of real data, selected to showcase diverse situations. The characteristics of the three months are displayed in Tab. 1. The experiments were run on a 32-core Intel Broadwell Virtual Machine with 32 GiB of RAM, using CPMpy version 0.10.0, Or-Tools version 9.14.6206, Gurobi version 13.0.1, and CP Optimizer version 22.1.1, and 8 threads per solver. Constraint-based solutions were run with a 10-minute timeout. The CPMpy model also includes information (*hints*, supported by Or-Tools and Gurobi) to guide the search: if prices are high, no pumping; if prices are low, maximum pumping; and if prices are average, one pump is activated.

6.1 Plant at full capacity

The first three scenarios simulate normal operating conditions for the plant (i.e., no equipment under maintenance, no personnel shortage). This is the most flexible situation. *DPSO* was

	DPSO	Or-Tools			
	Normal conditions	Normal conditions	3 pumps	18 switches	12 switches
May	2267	1843	1877	1866	1901
Jul.	3562	3475	3677	3476	3615
Dec.	7054	6762	6954	6774	6853

Table 3 Mean cost (€/day) over the first 5 days of each month in special situations (24 h)

computed as they would do at the plant. The initial state (i.e., the number of pumps currently activated and the buffer tank volume) at the beginning of each simulated month is obtained from historical data. Simulations are done for the three full months, and results over these scenarios are presented in Tab. 2.

- **Scenario 1 (S1): Day-by-day.** In this first scenario, we simulate each day independently using the same input. This input is the final state of the previous day of the DPSO. The goal is to compare one run at a time.
- **Scenario 2 (S2): Day-by-day with final state.** The second scenario uses the setup of the first, with an additional constraint to achieve a buffer tank state similar to that at the end of the day (within an ϵ tolerance). The goal is to compare on a run-by-run basis, avoiding potential bias arising from the tank's final state (it is less costly to keep the buffer full at the end of a day).
- **Scenario 3 (S3): Continuous use.** For the third scenario, the constraint solvers are executed as in actual production. The plant's historical initial state is used to initialize the first day of the simulation, and, between days, the final state of the previous day is reused for the next. The water that did not move in the previous plan is thus moved in the next plan, eliminating potential bias. The goal is to simulate the solution in production.

The results of S1 indicate that constraint-based methods are more effective for solving the problem. S2 shows that there is indeed a small bias at the end of the schedule, and that optimal solutions likely set the buffer's final state to a higher volume than the DPSO. However, this effect is partially mitigated by using only the first 24 hours. The last scenario, S3, solves multiple optimizations chained together. For May and July, S2 (using the DPSO states at the end of the plan for the next day) yields a better solution than S1. However, the opposite holds in December. This indicates that it would likely be worth fine-tuning the solution with a dynamic end-of-the-plan target range, potentially depending on factors such as the month or the expected solar energy. For May and July, the cost of the first 24h is less than half of the full plan. This is because the schedule starts at 16:00, so the last 16 hours of the schedule are from 16:00 to 8:00, when prices are highest during the day. During these months, the influence of photovoltaic power on the grid is greatest, contributing to the imbalance between the first part of the schedule and the remainder. Solutions of S1 also display a tendency to have a full buffer at the end of the plan. This tendency is mitigated by the use of the first 24 hours.

6.2 Plant at lower capacity

The plant sometimes runs at lower capacity. This situation includes cases where some pumps are undergoing maintenance (changes to the values of P and the p and f vectors), or when fewer people are available to operate the facility (e.g., holidays), thereby reducing the number of allowed pump switches too (changes to the M value).

The next scenarios (Tab. 3) are performed for the first 5 days of each selected month. They are used to evaluate the robustness of the constraint solution. The situations tested are (i) only 3 pumps available instead of 4, (ii) only 18 switches, and (iii) only 12 switches.

In situations with less flexibility, the mean daily cost increases as expected. One can also observe that the constraint-based approach is highly efficient, as even in these restricted settings it yields costs that are mostly lower than those of an unrestricted DPSO.

6.3 Discussion

These experiments also aimed to answer multiple questions posed by the consulting agency that originated the DPSO solution. First, whether a constraint-based solution was better than the current one in place. For this question, the answer is clear. The constraint-based solution achieves a clear lower cost while being allowed less time to compute in all tested scenarios. With only 10 min of computation, the constraint methods achieved daily operational savings of a few hundred euros, corresponding to 4 to 17% of the current solution operation cost. Secondly, whether there was a difference in performance between the constraint-based technologies (i.e., is one better). In Tab. 2, one can see that Gurobi (MILP) and Or-Tools (CP-SAT) are relatively close to each other. The mean cost over the three months in scenario 3 (the more realistic scenario) is nearly identical for the two solvers. CP Optimizer is slightly less good for the same allowed time, probably due to the non-support of the hints. This led us to conclude that, for this problem, Or-Tools and Gurobi are good candidates, and either could be selected by the facility to integrate into their process. Thirdly, whether the constraint-based solutions were robust to changes in the parameters. As shown by the experiments, situations with fewer resources are handled as efficiently as the normal condition. The constraint-based solution is thus also suited to this solution. Fourthly, whether it was possible to design a solution that quickly provides a new good plan that can be used at the next allowed switch slot without further adjustments. As parameters sometimes change abruptly (e.g., unplanned water demand changes, pump failures), the facility seeks an exploitable solution that can be computed in 10 minutes. Because the DPSO solution was usable but not sufficiently good after 10 min, the two-stage computation was required. As the CP solution is already better than the DPSO after only 10 min of computations (and proven optimal for up to 3 days/month) there was no need to set up a two stages computation as it was better suiting the facility requirements.

7 Conclusion

This paper presents an industry use case. The optimization problem presented is the one faced by a water treatment facility. It consists of optimizing the cost of the schedule of the activation of its high-power pumps under operational constraints to move water from a buffer tank to a higher-altitude reservoir. A step-by-step construction of the problem is presented, followed by the use of a modelling library to identify the adequate constraint techniques to solve it. Our results show that this transition to a constraint-based approach significantly enhances both economic performance and operational robustness compared to the previous heuristic methods. This study provides a scalable blueprint for any industrial asset with inherent buffering capacity, demonstrating how existing physical storage can enable flexible, high-power processes and facilitate more resilient participation in the energy transition.

This model was conceived with the facility's hypotheses and operational constraints in mind. Further work on this problem would include challenging some of these to further optimize the operational cost.

References

1 Valentin Antuori, Damien T Wojtowicz, and Emmanuel Hebrard. Solving the agile earth observation satellite scheduling problem with cp and local search. In *31st International Conference on Principles and Practice of Constraint Programming (CP 2025)*, volume 340, pages 3–1, 2025.

2 Ignace Bleukx, Ryma Boumazouza, Tias Guns, Nadine Laage, and Guillaume Poveda. Modeling and explaining an industrial workforce allocation and scheduling problem. *Leibniz International Proceedings in Informatics*, 340, 2025.

3 Gratien Bonvin and Sophie Demassey. Extended linear formulation of the pump scheduling problem in water distribution networks. In *International Network Optimization Conference, June 12-14, 2019, Avignon, France.*, pages 13–18, 2019.

4 Ioannis Boukas, Elodie Burtin, Antonio Sutera, Quentin Gemine, Bernard Pevee, and Damien Ernst. Exploiting the flexibility potential of water distribution networks: A pilot project in belgium. *IEEE Transactions on Smart Grid*, 15(1):394–404, 2023.

5 Dariush Fooladivanda and Joshua A Taylor. Energy-optimal pump scheduling and water flow. *IEEE Transactions on Control of Network Systems*, 5(3):1016–1026, 2017.

6 Alan M Frisch, Warwick Harvey, Chris Jefferson, Bernadette Martínez-Hernández, and Ian Miguel. Essence: A constraint language for specifying combinatorial problems. *Constraints*, 13(3):268–306, 2008.

7 Tias Guns. Increasing modeling language convenience with a universal n-dimensional array, cppy as python-embedded example. In *Proceedings of the 18th workshop on Constraint Modelling and Reformulation at CP (Modref 2019)*, volume 19, 2019.

8 Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2026. URL: `https://www.gurobi.com`.

9 Dongyun Kim, Yeonjun Choi, Kyungduk Moon, Myungho Lee, Kangbok Lee, and Michael L Pinedo. Iterated greedy constraint programming for scheduling steelmaking continuous casting. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 477–492. Springer, 2023.

10 Philippe Laborie, Jérôme Rogerie, Paul Shaw, and Petr Vilím. Ibm ilog cp optimizer for scheduling: 20+ years of scheduling with constraints at ibm/ilog. *Constraints*, 23(2):210–250, 2018.

11 Christophe Lecoutre and Nicolas Szczepanski. Pycsp3: Modeling combinatorial constrained problems in python. *arXiv preprint arXiv:2009.00326*, 2020.

12 Helena Mala-Jetmarova, Nargiz Sultanova, and Dragan Savic. Lost in optimisation of water distribution systems? a literature review of system operation. *Environmental modelling & software*, 93:209–254, 2017.

13 Nicholas Nethercote, Peter J Stuckey, Ralph Becket, Sebastian Brand, Gregory J Duck, and Guido Tack. Minizinc: Towards a standard cp modelling language. In *International conference on principles and practice of constraint programming*, pages 529–543. Springer, 2007.

14 Laurent Perron and Frédéric Didier. Cp-sat. URL: `https://developers.google.com/optimization/cp/cp_solver/`.

15 Francesca Rossi, Peter Van Beek, and Toby Walsh. *Handbook of Constraint Programming*. Elsevier, 2006.

16 Andreas Schutt, Matteo Cardellini, Jip J Dekker, Daniel Harabor, Marco Maratea, and Mauro Vallati. Constraint-based in-station train dispatching. In *31st International Conference on Principles and Practice of Constraint Programming (CP 2025)*, pages 33–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2025.

17 Yu Shao, Xinhong Zhou, Tingchao Yu, Tuqiao Zhang, and Shipeng Chu. Pump scheduling optimization in water distribution system based on mixed integer linear programming. *European Journal of Operational Research*, 313(3):1140–1151, 2024.

18 Byron Tasseff, Russell Bent, Carleton Coffrin, Clayton Barrows, Devon Sigler, Jonathan Stickel, Ahmed S Zamzam, Yang Liu, and Pascal Van Hentenryck. Polyhedral relaxations

for optimal pump scheduling of potable water distribution networks. *INFORMS Journal on Computing*, 36(4):1040–1063, 2024.

19 Charles Thomas and Pierre Schaus. A constraint programming approach for aircraft disassembly scheduling. In *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 211–220. Springer, 2024.

20 Túlio P Vieira, Paulo EM Almeida, Magali RG Meireles, and Marcone Jamilson Freitas Souza. Use of computational intelligence for scheduling of pumps in water distribution systems: a comparison between optimization algorithms. In *2018 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE, 2018.

21 Nils Weidmann. Multi-league sports scheduling with team interdependencies: An optimization model. In *31st International Conference on Principles and Practice of Constraint Programming (CP 2025)*, pages 37–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2025.

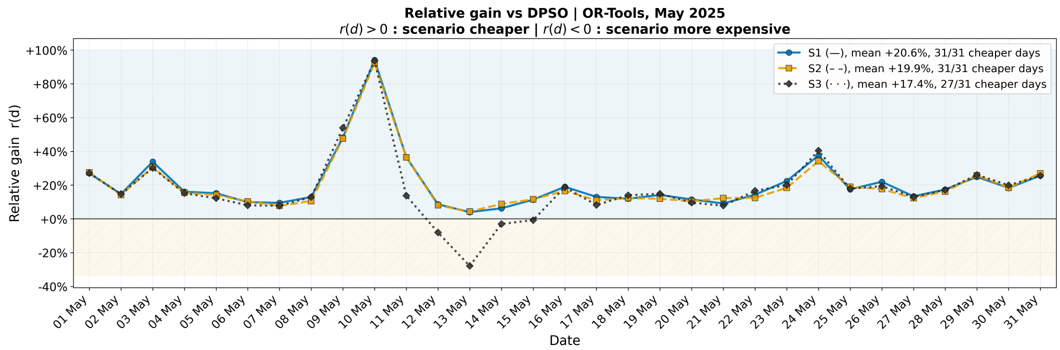
22 G Yu, RS Powell, and MJH Sterling. Optimized pump scheduling in water distribution dystems. *Journal of optimization theory and applications*, 83(3):463–488, 1994.

8 Annexes

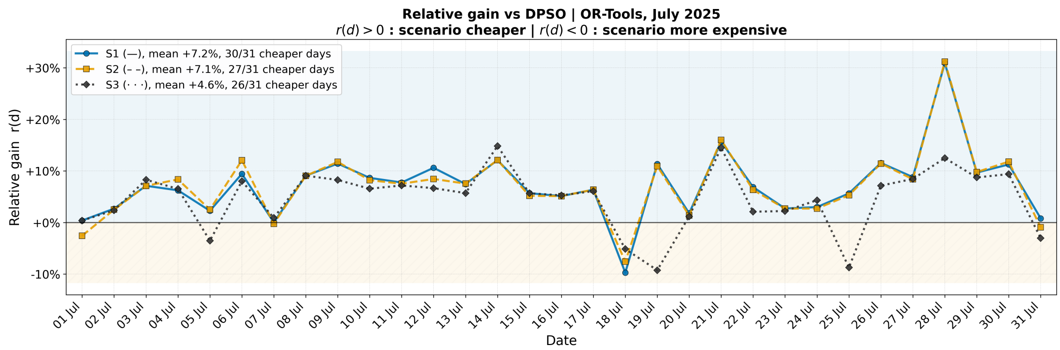
These annexes contain other visualizations of the results.

Figure 2 displays, for the solver Or-Tools, for the three months, and for the three scenarios, the relative gain (cost of the plan for the day divided by the cost of the DPSO plan for the day) of each day.

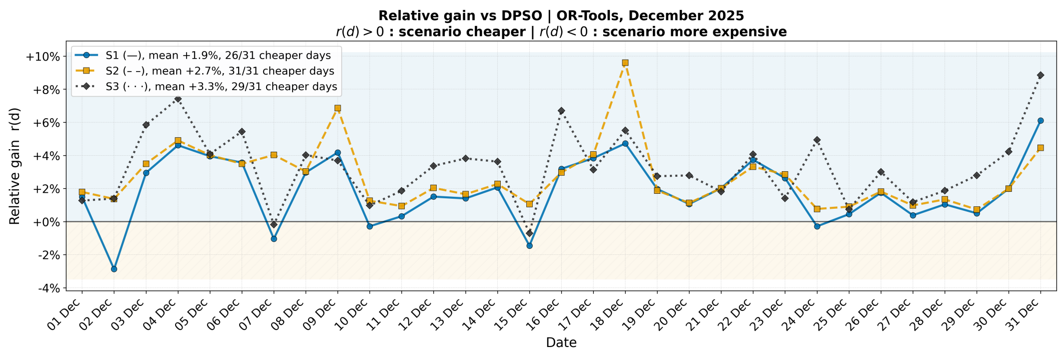
Figure 3 displays on the same graph all the pump schedule for each month for DPSO and the three constraint-based methods. It also shows the mean schedule. Figure 4 shows for each month the prices of each days. The mean value is also shown. By looking at the two series of graphs, we see that the solutions of the constraint-based model negatively correlate more with the prices than the DPSO solution. When prices are high, the pumps in the



(a) Relative gain for May 2025



(b) Relative gain for July 2025



(c) Relative gain for December 2025

Figure 2 Relative gain of the three scenarios comparing the model on Or-Tools and the DPSO

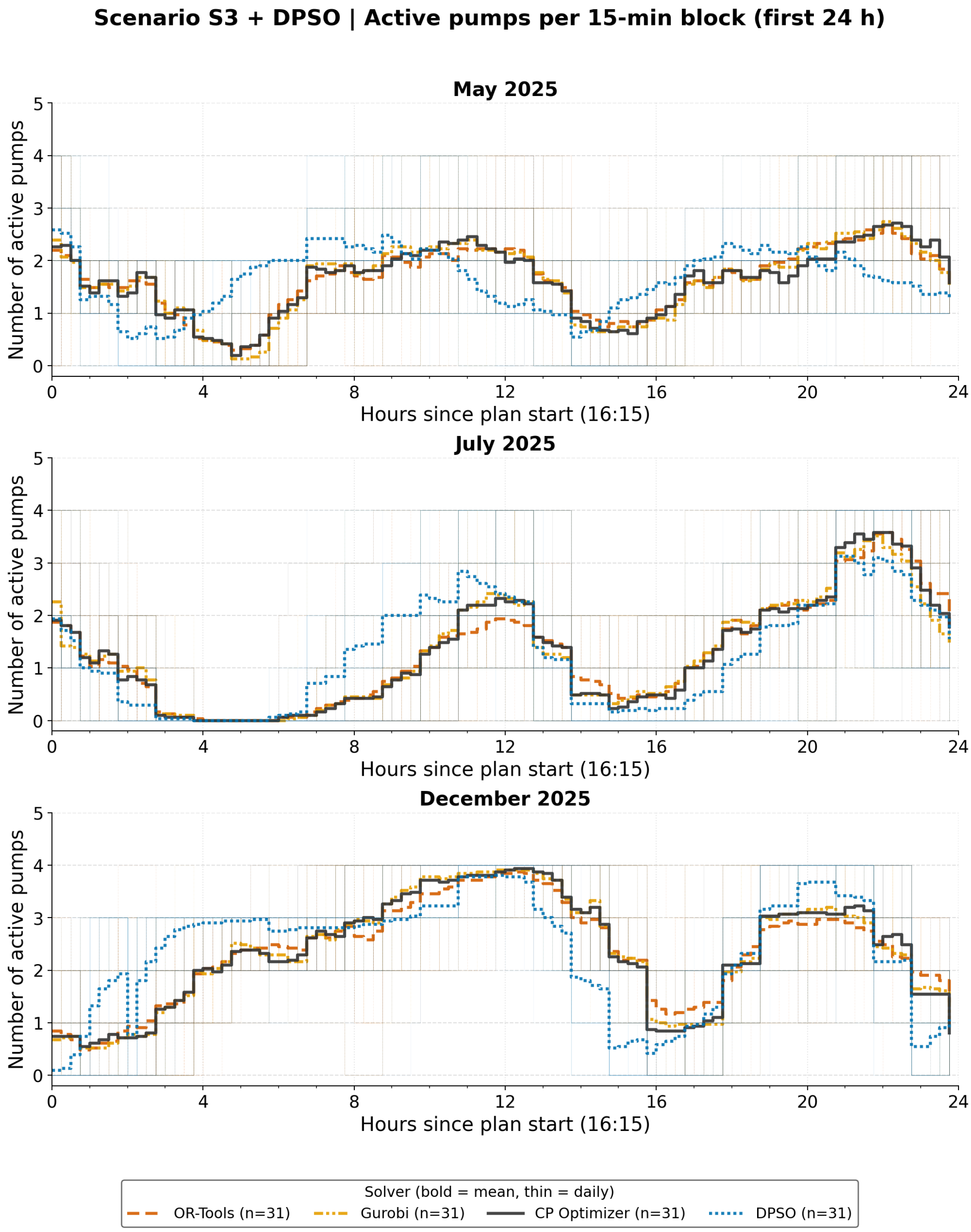


Figure 3 Mean activation of the pumps over the months on scenario 3 for the three solvers

constraint-based solutions are low. When the prices are low, the pumps are high. This is an expected trend of an optimal solution. The DPSO solution is less aligned with the cost trend. A Pearson test (Tab. 4) formally confirms the negative correlation between the scheduling and the price cost for any solutions. The constraint-based solutions (more optimal ones) shows a higher negative correlation. This confirms the trend and validate the choice of the hints selected to guide the search.

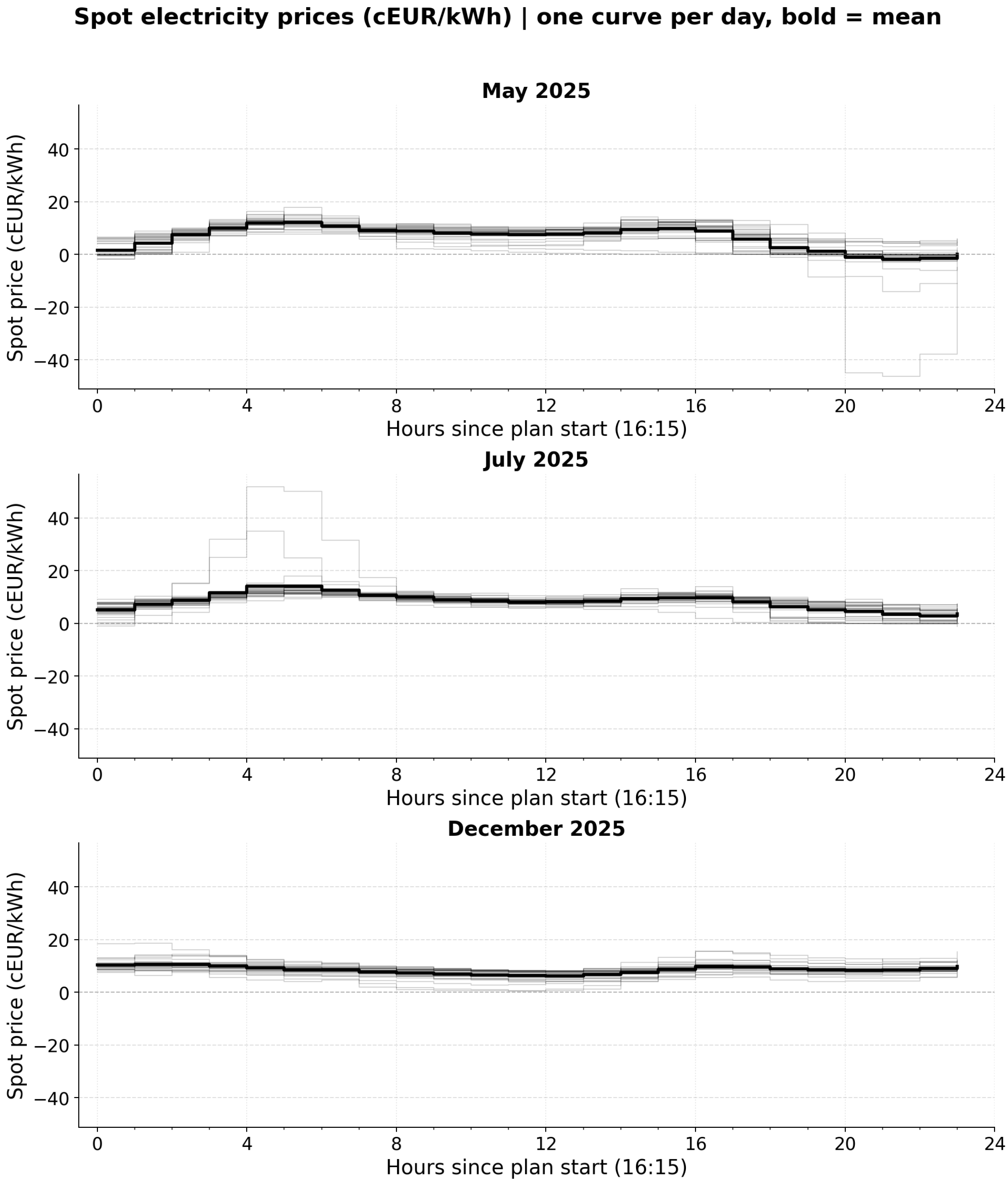


Figure 4 Mean cost of the energy over the months

	DPSO	Or-Tools	CP Optimizer	Gurobi
May	$-0,2 \pm 0,182$	$-0,543 \pm 0,094$	$-0,529 \pm 0,123$	$\mathbf{-0,545 \pm 0,098}$
Jul.	$-0,637 \pm 0,063$	$-0,752 \pm 0,062$	$\mathbf{-0,773 \pm 0,074}$	$-0,733 \pm 0,080$
Dec.	$-0,547 \pm 0,128$	$-0,808 \pm 0,106$	$-0,788 \pm 0,106$	$\mathbf{-0,816 \pm 0,099}$

Table 4 Pearson correlation test for the scenario 3 for all the solvers