An 0-1 Integer Linear Programming approach to schedule outages of nuclear power plants.

V. Jost and D. Savourey

LIX CNRS, École Polytechnique Palaiseau
Chair X-CNRS-Microsoft Optimization for Sustainable Development
{vjost,savourey}@lix.polytechnique.fr

Abstract

We address the problem of planning outages of nuclear power plants submitted by EDF (Électricité De France) as the challenge EURO/ROADEF 2010. As our team won the first prize of the contest in the senior category, our approach may be of interest: it is conceptually simple, easy to program and computationally relatively fast. We present both our method and some ideas to improve it.

1 Introduction

The subject of the challenge EURO/ROADEF 2010 was proposed by EDF[1]. Nuclear power plants need to be stopped regularly for maintenance and refueling. The problem mainly consists, over a 5 years horizon, in deciding when to stop nuclear power plants and by how much to refuel them, so as to satisfy the demand, and in order to minimize the cost of production.

A nuclear plant works in a "cyclic" (although not periodic) way. Every cycle starts with an *outage* during which maintenance operations are performed, and more importantly, some nuclear combustible is renewed. This operation is called a *refueling*. Refuelings can only be performed when the

[1]Électricité de France : French electricity provider

plant is on outage. The amount of energy that a plant can produce with its current stock of combustible is called its (level of) *fuel*. Between two outages, a nuclear plant is considered in production. The production phase consists of two successive subphases and the transition between them only depends on the amounts of fuel remaining in the plant. The production starts with a *modulation phase*, during which the power delivered by the plant can be chosen. When the fuel is below the "zero boron" (denoted BO hereafter) threshold, the plant produces less than its nominal power, and its production is constrained to a value that depends only on its level of remaining fuel.

Power plants are grouped (geographically) in subsets which use the same resources (human and mechanical) during their outages. Therefore, the outages of grouped plants must satisfy some disjonctive constraints.

Finally the production of energy must be equal to the demand at each time step. Since the demand is known only through forecasts, which depend among other, on meteorological conditions, the demand is modeled by a set of independent scenarios. The production must then be equal to the demand at each time step in each scenario.

Energy is not produced only using nuclear plants, but other plants are more expansive and are used mainly during peaks of demands.

A simple model of these plants is included in the model of the challenge, but the reason for their presence is not to take operational decisions. They are present essentially to evaluate the cost induced by the decisions taken on nuclear plants.

The paper is organized as follows. The problem of the challenge is defined in Section 2. We review solution methods described in the litterature in Section 3. In Section 4, a global presentation of our approach is given. Section 5 focuses on the scheduling sub-problem and the way we modeled it. In Section 6, algorithms used to solve the refueling sub-problem are presented. Numerical results are provided in Section 7. Section 8 consists in an analysis of the instances: we present some features of the instances that help designing efficient ways to solve the real world cases of the problem (that is, for EDF in France). Finally, a discussion about our approach and the way it could be improved is given in Section 9.

2 Formal problem statement

We provide here a formal and detailed definition of subject of the challenge. We use almost exactly the same notations as those of the challenge, whose subject is available at `http://challenge.roadef.org/2010/`. We will use the following indices :

- t for a time period, in T;
- h for a week, in H;
- s for a scenario, in S;
- i for a nuclear power plant, in I;
- j for a non nuclear power plant, in J;
- k for an outage, in K.

Here are the inputs of the problem :

- $DEM^{t,s}$: demand of energy at time step t in scenario s;
- D : length of a time step;
- $PMAX_j^{t,s}$: maximum power of non nuclear power plant j during time step t in scenario s;
- $PMIN_j^{t,s}$: minimum power of non nuclear power plant j during time step t in scenario s;
- $C_j^{t,s}$: proportional production cost of non nuclear power plant j during time step t in scenario s;
- XI_i : initial fuel level of nuclear power plant i;
- $DA_{i,k}$: length of outage k of nuclear power plant i;
- $TO_{i,k}$: first possible week for outage k of nuclear power plant i;
- $TA_{i,k}$: last possible week for outage k of nuclear power plant i;

- $MMAX_{i,k}$: maximum modulation during cycle k of nuclear power plant i;
- $RMIN_{i,k}$: minimum load of fuel during cycle k of nuclear power plant i;
- $RMAX_{i,k}$: maximum load of fuel during cycle k of nuclear power plant i;
- Q_{ik} : refueling coefficient during cycle k of nuclear power plant i;
- $PMAX_i^t$: maximum power during time slot t of nuclear power plant i;
- $BO_{i,k}$: threshold on fuel level during cycle k of nuclear power plant i : when fuel level is above this threshold, the plant is in "modulation phase"; when fuel level is below this threshold, the plant is in "profile phase";
- $PB_{i,k}$: imposed increasing function defined on $[0, BO_{i,k}]$ to $[0, 1]$ for "profile phase" during cycle k of nuclear power plant i : values of $PB_{i,k}$ represent a proportion of maximum power imposed on production;
- ϵ : tolerance for tracking of the imposed decreasing profile;
- $SMAX_{i,k}$: maximum fuel level after refueling during cycle k of nuclear power plant i;
- $AMAX_{i,k}$: maximum fuel level before refueling during cycle k of nuclear power plant i;
- $C_{i,k}$: unit cost of fuel during cycle k of nuclear power plant i;
- $C_{i,T+1}$: unit cost at the end of time horizon of nuclear power plant i.

Now, we introduce the following variables :

- $ha(i, k)$: index of first week of outage in cycle k of nuclear power plant i;
- $r(i, k)$: reload of fuel during cycle k of nuclear power plant i;

- $p_1(j,t,s)$: production of non nuclear power plant j during time step t in scenario s;
- $p_2(i,t,s)$: production of nuclear power plant i during time step t in scenario s;
- $fuel(i,t,s)$: fuel level of nuclear power plant i during time step t in scenario s.

Additionally, $ec(i,k)$ will denote the set of time steps where nuclear power plant i is in production phase during cycle k; and $ea(i,k)$ will denote the set of weeks composing the outage k of nuclear power plant i.

We can now formulate precisely the problem by the following constraints.

Constraints CT1

First, the production must be equal to the demand for each time slot t in each scenario s :

$$\forall s \in S, t \in T, \quad \sum_j p_1(j,t,s) + \sum_i p_2(i,t,s) = DEM^{t,s}$$

Constraints CT2

The non nuclear production must be within bounds for each time slot t in each scenario s :

$$\forall s \in S, t \in T, j \in J, \quad PMIN_j^{t,s} \leq p_1(j,t,s) \leq PMAX_j^{t,s}$$

Constraints CT3

During every time step t of every scenario s where plant i is on outage, its production is equal to zero.

Constraints CT4

During every time step t of every scenario s where plant i is online, its production is positive or equal to zero.

Constraints CT5

The nuclear production must be under a maximal bound during modulation period :

$$\forall s \in S, t \in T, i \in I, k \in K, \quad (t \in ec(i,k)) \wedge (fuel(i,t,s) \geq BO_{i,k}) \Rightarrow p_2(i,t,s) \leq PMAX_i^t$$

Constraints CT6

When the fuel level is under a threshold $BO_{i,k}$, the production levels are imposed by a given production profile :

$$\begin{gathered}\forall s \in S, t \in T, i \in I, k \in K \\ (t \in ec(i,k)) \wedge (fuel(i,t,s) < BO_{i,k}) \Rightarrow \\ \Big(\text{if } fuel(i,t,s) \geq (PB_{i,k}(fuel(i,t,s)) \times PMAX_i^t) \times \\ \text{then} \\ (1-\epsilon)(PB_{i,k}(fuel(i,t,s)) \times PMAX_i^t) \leq p_2(i,t,s) \leq (1+\epsilon)(PB_{i,k}(fuel(i,t,s)) \times PMAX_i^t) \\ \text{else} \\ p_2(i,t,s) = 0\Big)\end{gathered}$$

Constraints CT7

The reloads of fuel must be within bounds for each outage :

$$\forall i \in I, k \in K \text{ s.t. } ha(i,k) \neq -1, RMIN_{i,k} \leq r(i,k) \leq RMAX_{i,k}$$

Constraints CT8

At first time step, fuel level is equal to XI_i :

$$\forall s \in S, i \in I, \quad fuel(i, 0, s) = XI_i$$

Constraints CT9

Fuel at time step $(t+1)$ is equal to fuel level at time step t minus the energy produced at time step t :

$$\forall s \in S, t \in T, i \in I, k \in K, \quad t \in ec(i,k) \Rightarrow fuel(i, t{+}1, s) = fuel(i, t, s) - p_2(i, t, s) \times D$$

Constraints CT10

A refueling is performed at each start of an outage :

$$\begin{aligned} &\forall s \in S, t \in T, i \in I, k \in K, \ \text{(t is the first time step of ea(i,k))} \\ &\quad \Rightarrow fuel(i, t+1, s) = ((Q_{i,k} - 1)/Q_{i,k})(fuel(i, t, s) - BO_{i,k-1}) + r(i,k) + BO_{i,k} \end{aligned}$$

Constraints CT11

Fuel levels must be under maximal bounds before and after each refuel :

$$\begin{aligned} &\forall s \in S, t \in T, i \in I, k \in K, (\text{ t is the first time step of ea(i,k))} \\ &\qquad \Rightarrow fuel(i, t, s) \leq AMAX_{i,k} \wedge fuel(i, t+1, s) \leq SMAX_{i,k} \end{aligned}$$

Constraints CT12

The allowed modulation is bounded for each production cycle :

$$\forall s \in S, i \in I, k \in K, \quad \sum_{t \in ec(i,k) \wedge fuel(i,t,s) \geq BO_{i,k}} [(PMAX_i^t - p_2(i, t, s)) \cdot D] \leq MMAX_{i,k}$$

Constraints CT13

It is possible not to schedule certain outages. In this case, we set $ha(i,k) = -1$.

The following constraint defines the outages that must be done :

$$\forall (i,k) \in I \times K, TA_{i,k} \neq -1 \Rightarrow ha(i,k) \neq -1$$

In the case where an outage is not scheduled, the following outages can not be scheduled either.

Each scheduled outage must start within a given time window :

$$\forall (i,k) \in I \times K, \;\; TA_{i,k} \neq -1 \Rightarrow TO_{i,k} \leq ha(i,k) \leq TA_{i,k}$$

Outage k cannot start while outage $(k-1)$ is not over :

$$\forall (i,k) \in I \times K, k > 0, \;\; ha(i,k) \neq -1 \Rightarrow ha(i,k) \geq ha(i,k-1) + DA(i,k-1)$$

About constraints CT14,. . . ,CT21

The constraints CT14,. . . ,CT21 require specific input datas. All these datas must be considered relatively to the given type of constraint, even if their names are common from one type of constraints to another type. For example, the input Se_m in CT14 is not the same as the input Se_m in CT15. This notation ambiguity is inherited from the subject of the challenge, as it prevents from too heavy notations.

The set of constraints of each type is denoted by M_x where x is the index of constraint type. For example if $|M_{14}| = 5$, it means that there are 5 constraints of type $CT14$. Within a given type of constraints, $m \in M_x$ denotes the index of a specific constraint.

Constraints CT14

For some given groups of nuclear power plants, a minimum distance between each outage must be respected. When this distance is a negative number, this means a maximal overlap between outages. The inputs are :

- m : index of constraints of this type;

- A_m : set of nuclear power plants;
- Se_m : minimum spacing (when positive), maximum overlap (when negative).

$$\begin{gathered}\forall m \in M_{14}, \forall i, i' \in A_m, i \neq i', \forall k, k' \in K, \\ (ha(i,k) \neq -1 \wedge ha(i',k') \neq -1) \\ \Rightarrow (ha(i,k) - ha(i',k') - DA_{i',k'} \geq Se_m) \vee (ha(i',k') - ha(i,k) - DA_{i,k} \geq Se_m)\end{gathered}$$

Constraints CT15

For some given groups of nuclear power plants, during some specific time period, a minimum distance between each outage must be respected. When this distance is a negative number, this means a maximal overlap between outages. The inputs are :

- m : index of constraints of this type;
- A_m : set of nuclear power plants;
- Se_m : minimum spacing (when positive), maximum overlap (when negative).
- ID_m : beginning of the specific period;
- IF_m : end of the specific period.

$$\begin{gathered}\forall m \in M_{15}, \forall i, i' \in A_m, i \neq i', \forall k, k' \in K, \\ (ha(i,k) \neq -1 \wedge ha(i',k') \neq -1) \wedge \\ (ID_m - DA_{i,k} + 1 \leq ha(i,k) \leq IF_m) \wedge (ID_m - DA_{i',k'} + 1 \leq ha(i',k') \leq IF_m) \\ \Rightarrow (ha(i,k) - ha(i',k') - DA_{i',k'} \geq Se_m) \vee (ha(i',k') - ha(i,k) - DA_{i,k} \geq Se_m)\end{gathered}$$

Constraints CT16

For some given groups of nuclear power plants, a minimum distance between each starting dates of outage must be respected. The inputs are :

- m : index of constraints of this type;
- A_m : set of outages;
- Se_m : minimum spacing (must be positive).

$$\forall m \in M_{16}, \forall i, i' \in A_m, i \neq i', \forall k, k' \in K, \ \ (ha(i,k) \neq -1 \wedge ha(i',k') \neq -1) \Rightarrow \\ | \, ha(i,k) - ha(i',k') \, | \geq Se_m$$

Constraints CT17

For some given groups of nuclear power plants, a minimum distance between each ending dates of outage must be respected. The inputs are :

- m : index of constraints of this type;
- A_m : set of nuclear power plants;
- Se_m : minimum spacing (must be positive).

$$\forall m \in M_{17}, \forall i, i' \in A_m, i \neq i', \forall k, k' \in K, \ \ (ha(i,k) \neq -1 \wedge ha(i',k') \neq -1) \Rightarrow \\ | \, ha(i,k) + DA_{i,k} - ha(i',k') - DA_{i',k'} \, | \geq Se_m$$

Constraints CT18

For some given groups of nuclear power plants, a minimum distance between each starting dates and ending dates of outage must be respected. The inputs are :

- m : index of constraints of this type;

- A_m : set of nuclear power plants;
- Se_m : minimum spacing (must be positive).

$$\forall m \in M_{18}, \forall i, i' \in A_m, i \neq i', \forall k, k' \in K, \;\; (ha(i,k) \neq -1 \wedge ha(i',k') \neq -1) \Rightarrow \\ \mid ha(i,k) + DA_{i,k} - ha(i',k') \mid \geq Se_m$$

Constraints CT19

For some given group of nuclear power plants, for each week, the number of nuclear power plants that are in a specific time interval within the outage must be less than a maximal number. The inputs are :

- m : index of constraints of this type;
- A_m : set of outages;
- $L_{i,k,m}$: number of weeks after the start of the outage that defines the beginning of the specific period on which the constraint holds for outage k of nuclear power plant i;
- $TU_{i,k,m}$: length of the specific period on which the constraint holds for outage k of nuclear power plant i;
- Q_m : maximum number of unavailable nuclear power plants in their specific period.

$$\forall m \in M_{19}, \forall h \in H, \quad \sum_{(i,k) \in A_m \wedge ha(i,k) \neq -1} \mathbf{1}_{[ha(i,k)+L_{i,k,m}; ha(i,k)+L_{i,k,m}+TU_{i,k,m}[}(h) \leq Q_m,$$

where

$$\mathbf{1}_A(x) = \begin{cases} 1 \text{ if } x \in A \\ 0 \text{ if } x \notin A \end{cases}$$

Constraints CT20

For some given weeks, the number of nuclear power plants in a given subset that are in outage must be less than a maximal number. The inputs are :

- m : index of constraints of this type;
- A_m : set of outages;
- h_m : the week on which the constraint is defined;
- N_m : maximal number of plants.

$$\forall m \in M_{20}, \quad \sum_{(i,k)\in A_m \wedge ha(i,k)\neq -1} \mathbf{1}_{[ha(i,k);ha(i,k)+L_{i,k,m}+DA_{i,k}[}(h_m) \leq N_m,$$

where

$$\mathbf{1}_A(x) = \begin{cases} 1 \text{ if } x \in A \\ 0 \text{ if } x \notin A \end{cases}$$

Constraints CT21

For some given groups of nuclear power plants, for each time slot, the sum of the maximum production of nuclear power plants that are on outage must be less than a maximum value. The inputs are :

- m : index of constraints of this type;
- C_m : set of nuclear power plants;
- IT_m : a time period in weeks;
- $IMAX_m$: maximal value of offline power capacity.

$$\forall m \in M_{21}, \forall t \in h \text{ s.t. } h \in IT_m, \quad \sum_{i\in C_m \setminus (\exists k \setminus h\in ea(i,k))} PMAX_i^t \leq IMAX_m$$

Objective function

Finally, the goal is to minimize the sum of the average cost of production over all scenarios :

$$Minimize \sum_i \sum_k C_{i,k} \cdot r(i,k) + \frac{1}{S} \sum_s \left(\sum_t \left(\sum_j C_j^{t,s} \cdot p_1(j,t,s) \cdot D \right) - \sum_i C_{i,T+1} \cdot fuel(i,T,s) \right)$$

3 Literature review

Our approach to the problem of the challenge was to use an linear program with binary variables, indicating whether outages start at a given week or not. Background on linear progamming models and combinatorial optimization problems can be found in [10]. Reference work on linear programming and in particular time index formulations for scheduling problems include [11, 13].

Solutions methods designed and tested specifically for the model of the challenge include:

- Local search [5]. Best reported results in the litterature, high quality solutions are obtained in a few minutes.
- Column Generation [12]. Second best results reported in the litterature. Requires more than several minutes to find good quality solutions.
- Local search and constraint programming [6]. Reported solutions around 1.3% above the best known solutions.
- Constraint programming [2]. Also proposes lower bound computations. Objective value of reported solutions is comparable with our method. On instances B and X, the best found lower bounds are more than 15% below the best found solutions.
- Benders Decomposition [9]. The reported implementation is good on small instances (set A) but not competitive on large scale instance (set B and X).

A similar problem was studied in [4], but with only one scenario, and one time step per week. The instances of the challenge ROADEF contain typically between 50 and 100 scenario and 7 to 21 time set per week, making the space of decision variables around 2000 bigger. Some technical constraints appear in the challenge version and not in [4] and vice-versa.

Although designed independently, our approach is very similar to the one of [4], as both highly rely on a binary linear formulation. Moreover, both approaches include production variables for each nuclear plant and each week.

The main difference between the design of our approach and the one of [4] seems that they treat exactly constraints bearing on one nuclear plant within their MILP, while we use only surrogate simplified constraints.

Also their approach did not scale up very well to the real world problem, maybe due to the presence of non-nuclear plants in their MILP, as well as the global objective function of the problem. On the contrary, we use a surrogate objective function that preserves the decomposability of the problem in subgroups of power plants that interact in the original model only through the constraints "production=demand".

Other studies on variants of the problem include hybridized local search and constraint programming approaches [8] and semi-definite relaxations [7].

4 Overview of the method

Conceptually, our approach consists in a hierarchical decomposition of the problem into three phases : we fix some variables before working on fixing subsequent other variables.

1. *ILP* : Fix the dates of outages (variables $ha(i,k)$),
2. *Dichotomy* : Fix the values of refueling (variables $r(i,k)$),
3. *Greedy* : Fix the production levels (variables $p_1(j,t,s)$ and $p_2(i,t,s)$).

The first phase is mainly done using an ILP, the second phase using a kind of dichotomised search on the values of refueling and the third, running greedily the dynamic of the nuclear plants over the time horizon. Hence, within this paper, "ILP" refers to the first phase while "Dichotomy" refers to the second phase. However, since we often need to check exactly the feasibility of the decisions taken, we use the "Greedy" subroutines in various places, and not only in the third phase.

4.1 Specifications, utilization and principle of the ILP

The ILP encodes exactly the constraints linking the dates of outages of various nuclear plants (CT14-21), as well as the bounds on these dates (CT13). The ILP also encodes constraints on minimum distances between pairs of outages of each given plant (see section 5.2). These (constraints on) distances are not given in the subject. We designed them as a way to aggregate approximately the constraints on the dynamic of production, interruptions and refueling in each nuclear plant (CT3-13) to help our ILP finding solutions that lead to the global feasibility of the problem.

The ILP uses binary temporal variables x_{ikh}, stating whether or not outage (i, k) starts at week h. In the subject, constraints (CT14-18) express incompatibility between pairs of outages (see Section 5). It turns out that the underlying incompatibility graphs have a kind of interval structure. Thus the graphs have a small number of cliques, on which we formulate packing constraints (see [10] for background on this topic). This leads to fewer constraints, which moreover provide a tighter linear relaxation than the constraints on pairs. Constraints (CT19-21) are essentially kept as in the subject.

The distance constraints being only necessary and not sufficient to satisfy (CT3-13), we run the Greedy routine at the end of the ILP. If the Greedy routine says that (CT3-13) can be satisfied, we go to the second phase. If not, we increase the minimum date ($TO_{i,k}$) on an outage on which we detected infeasibility and rerun the ILP.

The ILP needs an objective function to find appropriate outages dates that lead to a global solution with good objective value. As discussed in section 5.2, a binary variable p_{ih} serves as a rough estimate of whether plant i will be able to produce at week h or not (depending on the dates of outages of plant i). Having pre-computed some estimate val_{ih} on the economical value of producing on i during week h the ILP maximizes $\sum_i \sum_h val_{ih} p_{ih}$. Although the variables p_{ih} are the only ones in the objective functions, we only care about the variables x_{ikh} so as to fix the dates of outages $ha(i, k)$ after the solution of the ILP.

4.2 Specifications, utilization and principle of the Dichotomy

Once the dates of outage are fixed, we are looking for refueling values. We say that a set of refueling values $\{r(i,k)\}$ is feasible if we are able to find a global feasible solution using these values. Our goal is to maximize the ability to produce with nuclear plants while keeping a global feasible solution. As it seems hard to compute directly feasible refueling with high values, we use a kind of target-checker search on possible values. This method relies heavily on the quasi monotonic feasibility of refueling values: If $\{r(i,k)\}$ is an infeasible set and $r'(i,k) \geq r(i,k)$ for all (i,k), then $\{r'(i,k)\}$ is also very likely infeasible. The method relies also on the fact that we generally prefer higher refueling values than smaller ones. We therefore start with $low \leftarrow \{RMIN_{i,k}\}$. Then, given a feasible vector *low* of refueling values, we try to guess by how much we can increase them, thus obtaining another vector *high*. If *high* is feasible, $low \leftarrow high$ and *high* is again increased. If *high* is infeasible, $high \leftarrow (high + low)/2$. These rules are iteratively applied until *low* and *high* are almost equal (for all (i,k)). At the end we have a feasible set of refueling values, and we didn't find opportunities to improve any value in this set.

4.3 Specifications, utilization and principle of the Greedy

Our approach needs ways to make sure, during the first two phases, that we will be able to find feasible solutions based on the values we choose for dates and values of refueling. To do this, we need to choose production levels, either for each nuclear plant separately of for nuclear plants together. Therefore, two different greedy procedures are used.

The first one does not take into account the scenarios, that is to say there is no demand, and no modulation is required. This procedure is presented in details with Algorithm 1. It is used in order to check that chosen dates of outages for each plant are not violating the constraints involving only that plant. It is also used to compute how much modulation we can afford on a cycle of a plant, before conflicting with other constraints involving that plant.

The second procedure takes into account the demand, it chooses production levels for each scenario, by modulating if necessary. To do this, a rolling horizon procedure is used. For each scenario, we follow the dynamic of fuel for each nuclear power plant, producing at $PMAX_i^t$ in modulation phase and following the profile constraint otherwise. Whenever the production exceeds the demand in a scenario, we modulate on chosen power plants until the production equals the demand. We use modulation first on the plant for which we have the largest estimate of allowed modulation. If there exists a scenario and a time step on which more modulation would lead to violating some constraints but production is still above demand, the procedure answers “no”. This method is described more precisely with Algorithm 2.

5 Scheduling outages

5.1 Main variables of the ILP

We use the following binary decision variables

$$x_{ikh} = \begin{cases} 1 \text{ if the k-th outage of plant } i \text{ starts at week h,} \\ 0 \text{ otherwise.} \end{cases}$$

Variables x_{ikh} allow an enriched reformulation of the variables $ha(i,k)$. The main advantage of variables x_{ikh} is that disjunctive constraints (CT14-18) can be naturally expressed, ie without using “big M” tricks.

Constraints CT13

Each outage (i,k) has at most one starting week h, which must be in within the bounds $[TO_{i,k}, TA_{i,k}]$:

$$\forall (i,k) \in I \times K, \sum_{h \in [TO_{i,k}, TA_{i,k}]} x_{ikh} \leq 1,$$

$$\forall (i,k) \in I \times K, \forall h \notin [TO_{i,k}, TA_{i,k}], x_{ikh} = 0.$$

If $k \geq 1$ and if the k-th outage of plant i starts at week h, then the $k-1$-th outage of plant i must occur, with a starting week not later than $h - DA_{i,k-1}$[2]
:

[2]The following constraints is written in the last paragraph, page 27 of the subject

$$\forall h \in H, \forall (i,k) \in I \times K, \sum_{h'=0}^{h-DA_{i,k-1}} x_{i,k-1,h'} \geq x_{ikh}.$$

Constraints CT14

These constraints consider outages as intervals $[ha(i,k), ha(i,k)+DA_{i,k}]$, and require separation of these intervals. We reformulate them as packing constraints:

$$\forall m \in M_{14}, \forall h \in H, \sum_{i \in A_m} \sum_{\substack{k \ s.t. \\ DA_{i,k}+Se_m-1 \geq 0}} \sum_{h'=h-DA_{i,k}-Se_m+1}^{h} x_{ikh'} \leq 1.$$

However, the above constraints are not sufficient, because outages for which $DA_{i,k} \leq -Se_m$ are not considered. If it exists, each such outage must not interfere with any of the packing described above.

$$\forall m \in M_{14}, \forall i \in A_m, \forall k \ \ s.t. \ \ DA_{i,k} \leq -Se_m, \forall h,$$

$$x_{ikh} + \sum_{i' \in A_m} \sum_{\substack{k' \ s.t. \\ DA_{i',k'}+Se_m-1 \geq 0}} \sum_{h'=h-DA_{i',k'}-Se_m+1}^{h+DA_{i,k}+Se_m-1} x_{i'k'h'} \leq 1.$$

Constraints CT15

For convenience, we introduce the following notations :

$$w_{ikm} = [ID_m - DA_{i,k} + 1, IF_m],$$

$$y_{iki'k'm}(h) = [h - DA_{i',k'} - Se_m + 1, h + DA_{i,k} + Se_m - 1].$$

We have :

$$\forall m \in M_{15}, \forall h \in H, \sum_{(i,k) \in A_m} \sum_{\substack{h' \in [h-DA_{i,k}-Se_m+1,h] \\ h' \in w_{ikm}}} x_{ikh'} \leq 1.$$

Then, the second part of (CT15) can be expressed as :

$$\forall m \in M_{15}, \forall i \in A_m, \forall k \;\; s.t. \;\; DA_{i,k} \leq -Se_m, \forall h,$$
$$x_{ikh} + \sum_{i' \in A_m} \sum_{\substack{k' \;\; s.t. \\ DA_{i',k'} + Se_m - 1 \geq 0}} \sum_{\substack{h' \in y_{iki'k'm}(h) \\ h' \in w_{i'k'm}}} x_{i'k'h'} \leq 1.$$

Constraints CT16

For each week h, the number of decoupling dates in interval $[h, h + Se_m - 1]$ must be smaller or equal to 1 :

$$\forall m \in M_{16}, \forall h \in H, \sum_{(i,k) \in A_m} \sum_{h'=h}^{h+Se_m-1} x_{ikh'} \leq 1.$$

Constraints CT17

For each week h, we consider the interval $[h, h + Se_m - 1]$ and impose that at most one coupling date belongs to this interval. It is equivalent to say that at most one decoupling date belongs to $[h - DA_{i,k}, h - DA_{i,k} + Se_m - 1]$. Thus, we have :

$$\forall m \in M_{17}, \forall h \in [0, H + \max_{ik} DA_{i,k} - 1], \sum_{(i,k) \in Am} \sum_{h'=h-DA_{i,k}}^{h-DA_{i,k}+Se_m-1} x_{ikh'} \leq 1.$$

Constraints CT18

Again, we consider interval $[h, h + Se_m - 1]$. There must be at most one coupling date or decoupling within this interval. Outage (i, k) couples or decouples in $[h, h + Se_m + 1]$ iff $ha(i, k) \in [h - DA_{i,k}, h + Se_m - 1]$. Thus, (CT18) can be rewritten as :

$$\forall m \in M_{18}, \forall h \in [0, H + \max_{ik} DA_{i,k} - 1], \sum_{(i,k) \in A_m} \sum_{h'=h-DA_{i,k}}^{h+Se_m-1} x_{ikh'} \leq 1.$$

Constraints CT19

There must be at most Q_m decoupling dates within every intervals $[h' + L_{i,k,m}, h' + L_{i,k,m} + TU_{i,k,m}]$:

$$\forall m \in M_{19}, \forall h \in H, \sum_{(i,k)\in A_m} \sum_{h'=h-L_{i,k,m}-TU_{i,k,m}+1}^{h-L_{i,k,m}} x_{ikh'} \leq Q_m.$$

Constraints CT20

For a given week h_m, the number of outages overlapping this week must be smaller than N_m :

$$\forall m \in M_{20}, \sum_{(i,k)\in A_m} \sum_{h'=h_m-DA_{i,k}+1}^{h_m} x_{ikh'} \leq N_m.$$

Constraints CT21

The total power of nuclear plants on outage cannot exceed a given threshold :

$$\forall m \in M_{21}, \forall h \in IT_m, \forall t \in h, \quad \sum_{i\in C_m} \sum_{k\in K} \sum_{h'=h-DA_{i,k}+1}^{h} PMAX_i^t \cdot x_{ikh'} \leq IMAX_m.$$

5.2 Minimum distances between the outages within each plant

The variables x do not allow to express the constraints concerning the production of nuclear plants. Indeed, constraints (CT3-11) imply that, even if refueling at the minimum allowed $RMIN_{i,k}$ and producing at $PMAX_i^t$, the plant needs several weeks for its fuel level to fall below $AMAX_{i,k+1}$.

Let $i \in I$ and $C \subseteq K$, we say that a set of week of outage $\{ha(i,k)\}_{k\in C}$ is "feasible" if there exists values $\{ha(i,k)\}_{k\in K\setminus C}$ such that there exists fuel, production and refueling levels such that plant i satisfies the constraints (involving only itself) (CT3-13).

It turns out that a set of outages dates $\{ha(i,k)\}_{k\in K}$ might be infeasible, while fixing all but one of these values might still be feasible. One might then wish to write constraints on $\{ha(i,k) | k \in C \subseteq K\}$ "as they are needed" (for example, when the current solution violates them). But this would lead to an iterative approach calling the ILP solver as a subroutine many times.

Concrete values in the data set suggest that feasible outage dates are highly (and mainly) constrained by minimum distances between (consecutive) pairs. In other words, under constraints involving pairs of outages of a plant, a set of value $\{ha(i,k)\}_{k\in K}$ is often feasible or requires shifting some outages to the future by only one week to become feasible.

Our trade-off is thus to rely only on constraints involving pairs. To do so, we pre-compute exactly the values (with $k < k'$):

$$earl(i,k,k_2,h) := \text{ first week } h_2 \text{ for which } x_{ikh} \text{ and } x_{ik_2h_2} \text{ are feasible together}$$

These values (*earl*) are then all encoded in our ILP, with the following constraints:

$$\forall h \in H, \forall (i,k), \forall k' > k,\ x_{ikh} + \sum_{h'=0}^{earl(i,k,k_2,h)} x_{i,k_2,h_2} \leq 1.$$

The values $earl(i,k,k',h)$ are deduced on each plant i independently, using a dynamic program to first compute the following quantities.

Let $minFl(i,k,h)$ be the minimum fuel level that we can achieve at week h, assuming that outage k has been completed (that is, with $h \geq ha(i,k)+DA_{i,k}$). Let $minFuel(i,k,k_2,h,h_2)$ be the minimum reachable fuel level at week h_2 such that k_2 is the last completed outage, and such that outage k has started at week h.

Assuming these values are computed, one deduces easily earliest feasible week of an outage and minimum distances between outages of one plant. This amounts to ask whether *minFl* and *minFuel* are lower than *AMAX*. Notice first that computing values *minFuel* knowing the values *minFl* is very similar to computing *minFl* starting from week 0 with fuel level XI_i.

The rest of section 5.2 describes the dynamic program. Constraint (CT10) implies that the dynamic of refueling is non-linear (although affine):

if $t-1$ is the first time step of outage (i, k), then the fuel level at time t is given by $fuel(i, t) = \Psi_{i,k}(fuel(i, t-1)) + r(i, k)$ with

$$\Psi_{i,k}(f) := \frac{Q_{i,k} - 1}{Q_{i,k}}.(f - BO_{i,k-1}) + BO_{i,k},$$

(where $Q_{i,k}$ is equal to 3 or 4 in practice).

Constraint (CT6) says that, if the fuel level is less than a threshold $BO_{i,k}$, then, until the plant enters an outage, the production is constrained to a proportion of its capacity $PMAX_i^t$. This proportion decreases together with the fuel level, and is given by a (piecewise-linear decreasing function $PB_{i,k}$). We simplify, and indeed, over-constrain (CT10) (which normally allows a tiny margin) into:

$$p(i, t) = PMAX_i^t * PB_{i,k}(fuel(i, t)).$$

To achieve $minFl(i, k, h)$, we assume that each refueling is done at $RMIN_{i,k}$ and that consumption is always equal to $PMAX_i^t$ in modulation phase. This second hypothesis is false in general because using modulation to adjust sharply the fuel level before entering the profile phase might help decreasing it at the end of the profile phase. Anyway, although not exact, these hypothesis are almost exact given the order of magnitude of the numbers in the data set.

One concludes that, in order to compute $minFl(i, k, h)$ we shall use the following equations describing fuel levels and power.

$$p(i, t) = \begin{cases} 0 & \text{if plant } i \text{ is in outage at } t, \\ PMAX_i^t & \text{if } fuel(i, t) \geq BO_{i,k}, \\ PMAX_i^t.PB_{i,k}(fuel(i, h)) & \text{if } fuel(i, t) < BO_{i,k}; \end{cases} \tag{1}$$

$$fuel(i, t) = \begin{cases} XI_i & \text{if t=0 ,} \\ \Psi_{i,k}(fuel(i, t-1)) + RMIN_{i,k} & \text{if outage } k \text{ starts at } t-1, \\ fuel(i, t-1) - p(i, t-1).D & \text{otherwise.} \end{cases} \tag{2}$$

Of course, to achieve $minFl(i, k, h)$, we still need to choose the dates of outages. $minFl(i, k, h)$ is set to $+\infty$ at the initialization of the dynamic program. It may remain infinite, indicating that is is impossible to complete outage k by week h. We start with initial conditions $minFl(i, 0, 0) := XI_i$.

We compute the following values (as they are needed and without storing them): let $conso(i, k, h-1, x)$ be the quantity of fuel consumed during all time steps of week $h-1$ under equations (1) and (2), assuming that plant i is in cycle k, and with fuel level x at the beginning of that week.

$minFl(i, k, h)$ satisfies a Bellman relation. It is the minimum between two values:

$$\begin{cases} minFl(i, k, h-1) - conso(i, k, h-1, minFl(i, k, h-1)), \\ \Psi_{i,k}(minFl(i, k-1, h-DA_{i,k})) + RMIN_{i,k}. \end{cases} \tag{3}$$

The first case is always acceptable, but the second can be chosen only if both $minFl(i, k-1, h-DA_{i,k}) \leq AMAX_{i,k}$ and $TO_{i,k} \leq h - DA_{i,k} \leq TA_{i,k}$.

Concerning $minFuel(i, k, k_2, h, h_2)$, we don't go into computational details here. Let us just mention that $minFuel(i, k, k_2, h, h_2)$ satisfies a relation quasi-identical to (3) as it is the minimum between:

$$\begin{cases} minFuel(i, k, k_2, h, h_2-1) - conso(i, k_2, h_2-1, minFuel(i, k, k_2, h, h_2-1)), \\ \Psi_{i,k}(minFuel(i, k, k_2-1, h, h_2-DA_{i,k_2})) + RMIN_{i,k_2}. \end{cases} \tag{4}$$

5.3 Computing an approximate objective function

One first issue that we had to solve, before evaluating costs, is to evaluate the nuclear availability at each time step, or at least at each week. We choose to evaluate the availability of each plant for each week. Actually, we assume here that a plant either produce at *PMAX* or is offline (hence neglecting modulation and profile phase effects !).

Let the following binary variable be

$$p_{ih} = \begin{cases} 1 \text{ if we forecast a production (at } PMAX \text{) on plant } i \text{ at week h,} \\ 0 \text{ if we forecast no production on } i \text{ at } h. \end{cases}$$

Contrary to variables x, the values of variables p are not intended to be used outside of the ILP. They just help defining (and hence finding) the quality of outages dates.

Variables p must be constrained to 0 while on outage:

$$\forall i \in I, \forall h \in H, \quad p_{ih} + \sum_k \sum_{h'=h+1-DA_{i,k'}}^{h} x_{ikh'} \leq 1.$$

p should be set to 0 in case of fuel shortage. Concerning fuel shortage, we assume that each cycle starts at level $RMAX_{i,k'}$ and that the fuel is consumed at $PMAX_{i,t}$. It is therefore immediate to pre-compute the number $N_{ikh'}$ of weeks during which we can produce at $PMAX$ before fuel stock falls to 0. The following constraints is added to the ILP :

$$\forall i \in I, \forall h \in H, \quad p_{i,h} \leq \sum_k \sum_{h'=h-N_{ikh'}}^{h-DA_{i,k}} x_{ikh'}. \tag{5}$$

Now that we have an estimate of the availability of plant i at week h, we can express an objective function :

$$Maximize \sum_h \sum_i DEM_h.p_{ih}, \tag{6}$$

where DEM_h is the average demand at week h, *i.e.* $DEM_h = \frac{1}{S} \sum_s \sum_{t \in h} DEM^{t,s}$.

Many hypothesis made in this section seem unjustified oversimplifications of the problem. Notice however the various advantages of the approach. Thanks to the objective function, the ILP is decomposable on subsets of plants that are not interacting through (CT14-21). Assuming fuel levels of $RMAX$ after outage is an under approximation of the fuel level we can ensure, but often, the maximum fuel allowed ($SMAX$) is not much higher than $RMAX$. Moreover, when $RMAX - RMIN \gg 0$, the objective function tends to help satisfying distance constraint discussed in section 5.2, because scheduling the next outage much before the end of the N_{ikh} allowed weeks results in a loss in the objective function.

6 Production and refueling levels

In all this section, we consider that variables $ha(i,k)$ are already fixed, so they could be considered as implicit inputs of the algorithms discussed here.

6.1 Fuel Levels without modulation

Algorithm 1 computes bounds on fuels levels for a given set of outages and refueling values, without taking into account the demand.

It consists in unrolling time for each power plant, producing at maximal level or following the profile production constraint. Refueling levels are set by the input. During this loop, several bounds on fuel levels are computed and stored :

- $amaxGap_{ik}$: difference between $AMAX_{i,k}$ and fuel level $fuel[i]$ just before the refueling $r(i,k)$;
- $smaxGap_{ik}$: difference between $SMAX_{i,k}$ and fuel level $fuel[i]$ just after the refueling $r(i,k)$;
- $nomodfuel_{ik}$: fuel level $fuel[i]$ at the end of cycle (i,k) ($nomodfuel$ stands for "no modulation").

6.2 Exact production level affectation

Algorithm 2 affects power production levels for each power plant for a given scenario s. Outage dates and refuelings are known. It is a rolling horizon procedure, very similar to Algorithm 1. The main difference is that the demand of scenario s is taken into account. Thus, we can not affect production levels independently on each power plant. Each time step is decomposed in two phases : (1) maximal or profile levels affectation for each power plant ; (2) if production exceeds demand, resolution by modulation. Several additional indicators on the ability for a power plant to modulate are also maintained. In particular, *modMaxToEnd* is computed using *amaxGap* and *smaxGap* provided by Algorithm 1, and correponds to the maximal possible modulation allowed in the current cycle taking into account current *MMAX* and all future bounds *AMAX* and *SMAX*. In case where phase (2) can not decrease the production down to the demand (*i.e.* no more modulation is allowed), the algorithm stops and return "false". When this has been done for every time step, a new rolling time loop is done to complete production with type 1 power plants to fill the demand.

Algorithm 1: *bounds*$(r(i,k))$

```
Data: r(i,k), i ∈ I, k ∈ K
Result: amaxGap_ik, smaxGap_ik, nomodFuel_ik, i ∈ I, k ∈ K, a set of power
        plant indexes toReschedule
begin
    for i ∈ I do
        initialize state[i], cycle[i], fuel[i] ;
    for i ∈ I do
        for t ∈ T do
            if state[i] = MOD then
                prod[i] ← PMAX_i^t ;
            if state[i] = PROF then
                prod[i] ← PB_{i,k}(fuel[i]) ;
            if state[i] = OUT then
                if t = ha(i,k) · T/H then
                    compute amaxGap_ik ;
                    if fuel[i] violates AMAX_{i,cycle[i]} then
                        toReschedule += {i} ;
                    update fuel[i] with r(i,k) ;
                    compute smaxGap_ik ;
                    if fuel[i] violates SMAX_{i,cycle[i]} then
                        toReschedule += {i} ;
                prod[i] ← 0 ;
            fuel[i] ← fuel[i] − prod[i] · D ;
            if t + 1 = ha(i,k + 1) · T/H then
                nomodfuel_ik ← fuel[i] ;
            update state[i], cycle[i] for next time step ;
```

Algorithm 2: $affectScenar(s)$

Data: $amaxGap_{ik}, smaxGap_{ik}, r(i,k), i \in I, k \in K$, scenario index s
Result: boolean value and $p_2(i,t,s)$ and $p_1(j,t,s)$ for $i \in I, j \in J, t \in T, s \in S$

```
begin
    for i ∈ I do
        initialize state[i], cycle[i], fuel[i], modMaxToEnd[i] ;
    for t ∈ T do
        dem ← DEM^{t,s} ;
        for j ∈ J do
            p_1(j,t,s) ← PMIN_j^{t,s} ;
            dem ← dem − p_1(j,t,s) ;
        compute puissMax (available nuclear production at time t) ;
        if puissMax > dem then
            toModul ← puissMax − dem ;
            for i ∈ I do
                capaModul[i] ← min(PMAX_i^t, modMaxToEnd[i]) ;
            while toModul > 0 do
                choose i* such that state[i*] = MOD and
                capaModul[i*] > 0 and modMaxToEnd[i*] is maximal;
                if there is no candidate for i* then
                    return false ;
                modul[i*] ← min(toModul, capaModul[i*]) ;
                update modMaxToEnd[i*], capaModul[i*] ;
                toModul ← toModul − modul[i*] ;
        for i ∈ I do
            if state[i] = MOD then
                p_2(i,t,s) ← PMAX_i^t − modul[i];
            if state[i] = PROF then
                p_2(i,t,s) ← PB_{i,k}(fuel[i]) ;
            if state[i] = OUT then
                p_2(i,t,s) ← 0 ;
            update fuel[i], cycle[i], state[i], modMaxToEnd[i] for next
            time step ;
    fill demand for each time step t with type 1 power plants sorted
    by increasing cost C_j^{t,s} ;
    return true ;
```

Algorithm 3: $fillIdleProd(r(i,k))$

Data: $r(i,k), i \in I, k \in K$
Result: new values for $r(i,k), i \in I, k \in K$
begin
    for $k \in K$ **do**
        for $i \in I$ **do**
            if *outage* (i,k) *exists and* $nomodfuel_{ik}$ *is null* **then**
                increase $r(i,k)$ so that fuel level becomes null at the very end of cycle (i,k) ;
                adjust $r(i,k)$ to satisfy constraints on $SMAX_{i,k}$ and $RMAX_{i,k}$;
        Recompute bounds on fuel levels with Algorithm 1

6.3 Avoiding lack of fuel

It can happen that a power plant is on fuel shortage (much) before an outage, although the preceding refueling could be increased. Algorithm 3 aims at avoiding these situations. We are given refuelings $r(i,k), i \in I$ and fuel bounds $nomodfuel_{ik}, i \in I, k \in K$. Considering cycles increasingly, this algorithm checks if the fuel level is null at the end of cycle (i,k), with the estimator $nomodfuel_{ik}$. If this quantity is null, $r(i,k)$ is increased so that the fuel level becomes null only at the end of cycle (i,k). Bounds on fuel level are recomputed along and at the end of this method, using Algorithm 1.

6.4 Increasing refueling

We would like to maximize the amount of refuelings over all nuclear power plants, without breaking the feasibility of the solution. Because it may be hard to ensure this feature, we use an optimistic algorithm that increase refuelings, and we will deal with the feasibility later. Increasing refuelings is done in Algorithm 4. We consider iteratively every cycle. For each cycle, all nuclear power plant are treated independently. Given a power plant i and a cycle k, a quantity add_{ik} is computed and added to $r(i,k)$. To compute add_{ik}, we need to fulfil the following requirements :

- do not violate $SMAX_{i,k'}$, for $k' \geq k$;

Algorithm 4: *incRefuelings*$(r(i,k))$

Data: $r(i,k), i \in I, k \in K$
Result: new values for $r(i,k), i \in I, k \in K$
begin
  for $k \in K$ **do**
    for $i \in I$ **do**
      if *outage* (i,k) *exists* **then**
        Compute maximal value for add_{ik} (quantity of fuel to add to $r(i,k)$) so that : no fuel bounds constraints are violated within the current and the following cycles ; fuel at the end of cycle (i,k) is no more than $BO_{i,k}$;
        $r(i,k) \leftarrow r(i,k) + add_{ik}$;
    Recompute bounds on fuel levels with Algorithm 1

- do not violate $AMAX_{i,k'}$, for $k' > k$;
- do not violate $RMAX_{i,k'}$, for $k' \geq k$;
- $nomodfuel_{ik} \leq BO_{i,k}$;
- maximize add_{ik}.

As we have an heuristic evaluation of fuel levels at the end of every cycle with *nomodfuel* values, we are able to estimate the maximal amount of fuel that we can add to $r(i,k)$ without violating bound constraints over *AMAX*, *SMAX* and *RMAX*. The target being at $BO_{i,k}$ at the end of cycle (i,k), we just assume that power plant i produces at *PMAX* at each time step of cycle k. We thus obtain a quantity of fuel needed at the beginning of cycle (i,k).

After each power plant has been considered for cycle k, we recompute indicators on gaps to fuel bounds using Algorithm 1.

6.5 Optimal refueling : Up&Down refuelings

Algorithm 5 just tries to affect production levels in every scenario, and return true iff a global feasible solution has been found.

Algorithm 5: $affectPowerOk(r(i,k))$

Data: $r(i,k), i \in I, k \in K$
Result: a boolean indicating if power affectation is feasible

```
begin
    for s ∈ S do
        if affectScenar(s) is FALSE then
            return FALSE ;
    return TRUE ;
```

Algorithm 6: $upAndDown(r(i,k))$

Data: $r(i,k), i \in I, k \in K$
Result: new values for $r(i,k), i \in I, k \in K$

```
begin
    low ← r(i,k) ;
    high ← incRefuelings(r(i,k)) ;
    while low ≉ high do
        if affectPowerOk(high) then
            low ← high ;
            high ← IncRefuelings(high) ;
        else
            high ← middle(low, high) ;
```

Algorithm 6 is aimed to find the largest values of refueling such that a feasible solution for the general problem has been found with these values. It maintains two sets of refueling values : *low* and *high*. The *low* set is always feasible. The *high* set is a simple increasing based on the *low* set. The algorithm checks iteratively whether the *high* set is feasible or not. In the positive case, *low* takes values from *high* and a new *high* set is computed from this new *low* set, using Algorithm 4. In the negative case, *low* does not change and *high* is decreased. The algorithm runs until *low* and *high* are "close enough", or a time limit constraint has been reached. At the end of this algorithm, the *low* set is always feasible.

7 Numerical Results

The program was evaluated on instances published by EDF. We provide results for the version discussed in this paper (even for instances A, for which we were evaluated on a preliminary version). A global evaluation is described in table 7. Gap of x to y is defined as $100(x-y)/y$. Challenge's Best is the best value of solution found within the challenge (which concerns only qualification results for instances A). CPU time is computed before printing the solution in a file (which can take several minutes for B and X instances.). Computational times are missing for X instances, since these were run by EDF, with feedback only on the value of the solutions. Our average gap to the best solutions founds on instances B and X (which was the ranking criterion of the challenge) is $3,9$ percents.

We worked with the following PC-configuration: 4 processors Dual Core AMD Opteron(tm) Processor 275 (2.2GHz, 1Mo cache), 16Go of RAM, ILOG CPLEX 12.1. One of the eight cores was used.

8 Analysis of the data

The subject of the challenge in full generality is NP-Hard. However there are structural properties that can be (robustly) found in all the 16 instances proposed by EDF. These properties are important to state precisely and formally, because they (might) help finding efficient resolution techniques. We therefore discuss the most structuring properties we were able to find.

Instance	Objective Value (Euros)	Gap to Challenge's Best (%)	CPU time (mm:ss)
A0	8.73566e+12	1,2	00:00
A1	1.69937e+11	0,05	00:08
A2	1.46383e+11	0,2	00:16
A3	1.5501e+11	0,3	00:15
A4	1.12643e+11	0,9	00:29
A5	1.26876e+11	0,8	00:47
B6	8.6874e+10	4,1	01:56
B7	8.36859e+10	3,1	02:20
B8	8.7091e+10	6,3	04:31
B9	8.68907e+10	6,3	14:49
B10	8.1594e+10	3,7	04:28
X11	8.06461e+10	3,1	
X12	7.88175e+10	1,6	
X13	8.0582e+10	5,4	
X14	7.94392e+10	4,3	
X15	7.6063e+10	1,3	

Table 1: Evaluation of the challenge ROADEF version of our program.

Pre-processing PMIN. Studying the subject only and not the instances, observe that minimum power of type 1 plants can be assumed to be 0. Indeed, as a pre-processing, for all (j, t, s) one can decrease $PMIN_j^{t,s}$, $PMAX_j^{t,s}$ and $DEM^{t,s}$ by $PMIN_j^{t,s}$. To ensure that the objective value invariant by this pre-processing, we also need to increase the cost function by $D/S \cdot \sum_{j,t,s} PMIN_j^{t,s}$. When comparing demand to nuclear availability, this pre-processing is highly recommended.

Quasi-monotonic feasibility of refuelings. The main structuring characteristic of data is that non-nuclear availability is large enough so that it always allows to match demand, whatever the nuclear power available. On the other hand, too much nuclear availability might require too much modulation and hence lead to global infeasibility. These two properties almost imply that, given fixed outage dates, if $\{r(i,k)\}$ is infeasible and $RMIN_{i,k} \leq r(i,k) \leq r'(i,k)$ for all (i,k) then $\{r'(i,k)\}$ is also infeasible. Indeed this statement is formally false because increasing refuelings shift the entrance into profile phase to the future, possibly allowing modulation for some time steps. However, to the best of the little experiments we have done regarding this issue, this counter-proof doesn't apply in practice.

Approximation and evaluation of costs. Nuclear production costs are roughly between 15 and 21, while type 1 plants costs typically vary between 10 and 10k. Most of the time, the most expansive nuclear plant is cheaper than the cheapest non-nuclear one, but this is not always true. The costs of type 1 plants are indicated in Figures 1, 2 and 3. Type 1 completion costs are convex piecewise linear for each scenario and time step. Equivalently, associated marginal costs are non-increasing step functions (of nuclear power available). One should be careful on his hypothesis when aggregating on scenarios and time steps. However, if we rely on the Figure 2, that is, if nuclear availability is assumed uniform in a given week, we observe that marginal cost is constant (for small value on the x-axis), then piecewise linear, then strictly convex. This implies that the cost can be sharply approximated on some interval, by an affine or a quadratic function. Further Log(-log) plots should provide more insight on the nature of the curves 1 and 2 for $x \geq 25000$. For future investigations we recommend to include (an evaluation) the cost of nuclear power within such plots.

More importantly, one should use approximate cost functions that zoom on the range of likely (or possible) nuclear capability of each week.

Laminarity and decomposability of nuclear plant interactions. Insightfully, one can draw the hyper-graph of constraints (CT14-21). To do so, associate a vertex with each nuclear plant, and an hyper-edge (A_m) with each constraint. On all instances B, we observe that this hyper-graph is roughly the same (after relabelling the plants). The reason is that that sets A_m rely on the geographical positions of French nuclear plants. The hyper-graph has a laminar structure (for any 2 pair of edges A_m and $A_{m'}$, either $A_m \cap A_{m'} = \emptyset$ or $A_m \subseteq A_{m'}$ or $A_{m'} \subseteq A_m$). To visualize the hypergraph of constraints, one can just draw a map of France, put one node for all the 58 reactors, and draw a circle for all reactors belonging to a nuclear site. Then for nuclear sites having more than 2 reactors, match the reactors in pairs. Also, an hyperedge may merge 2 nuclear sites that are geographically close to each other.

In particular, often constraints (CT14) involve only 2 plants with large Se_m, hence strongly linking these 2 plants. The connected components of the hyper-graph are of size 2, 4 or 6 (with a total of more than 50 plants), yielding around 9 components. This implies that demand constraints (CT1) are the only one linking these 9 subgroups of nuclear plants in the model, so that just dualizing (CT1) allows to split the problem in much smaller parts. We used this non-connectivity of the hyper-graph only in the rescheduling phase of the ILP: if the solution of the ILP doesn't allow to find feasible production plans for a subset I^* of plants, then we rerun the ILP only with plants connected to at least one plant in I^*.

9 Ideas for future work

When participating in such a challenge, it is hard to explore all mathematical and algorithmic ideas that come to mind. When studying the subject,

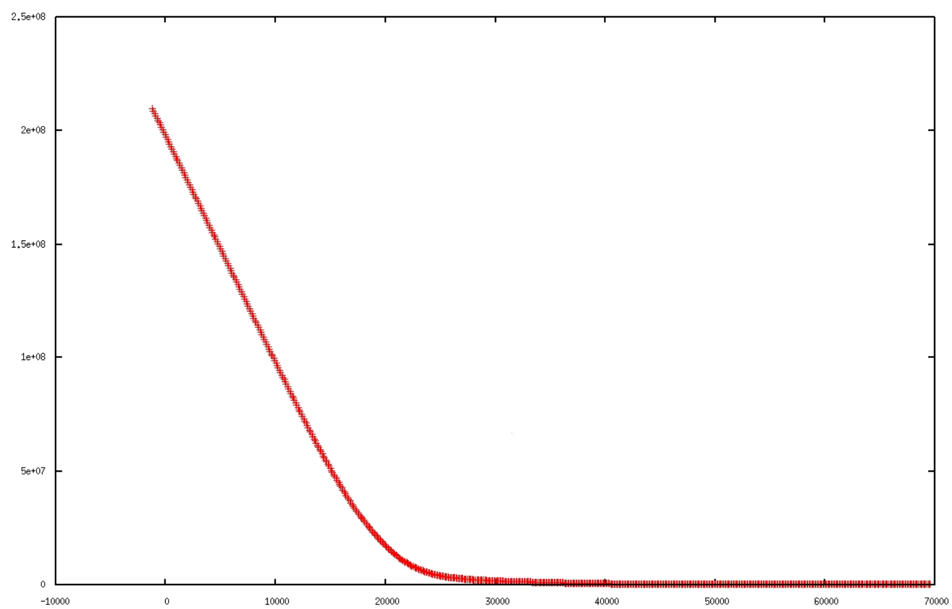

Figure 1: Cost of completion by non-nuclear plants as a function of total nuclear availability, averaged on all steps and scenarios of week 63, instance B6.

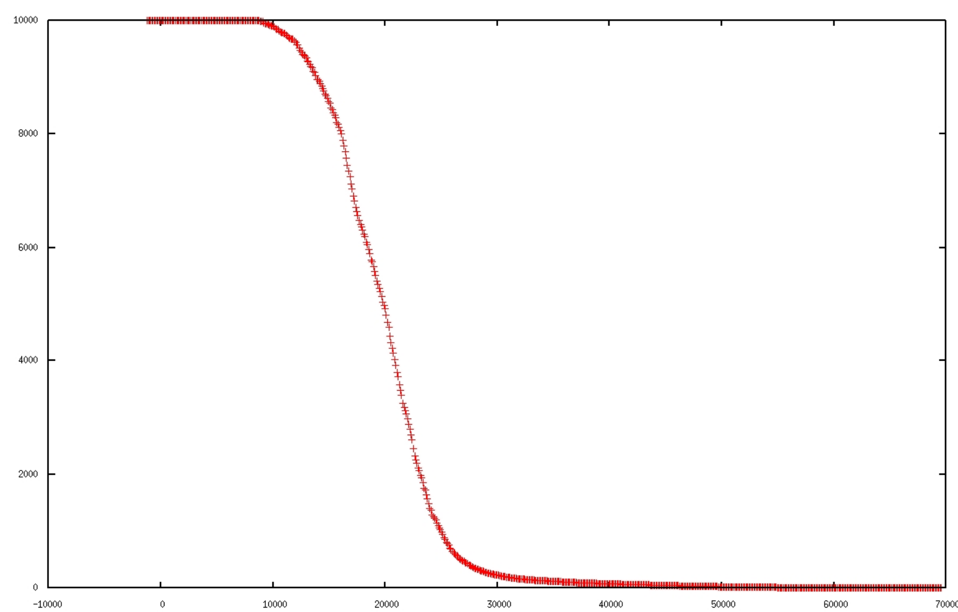

Figure 2: Marginal cost of completion by non-nuclear plants as a function of total nuclear availability, averaged on all steps and scenarios of week 63, instance B6.

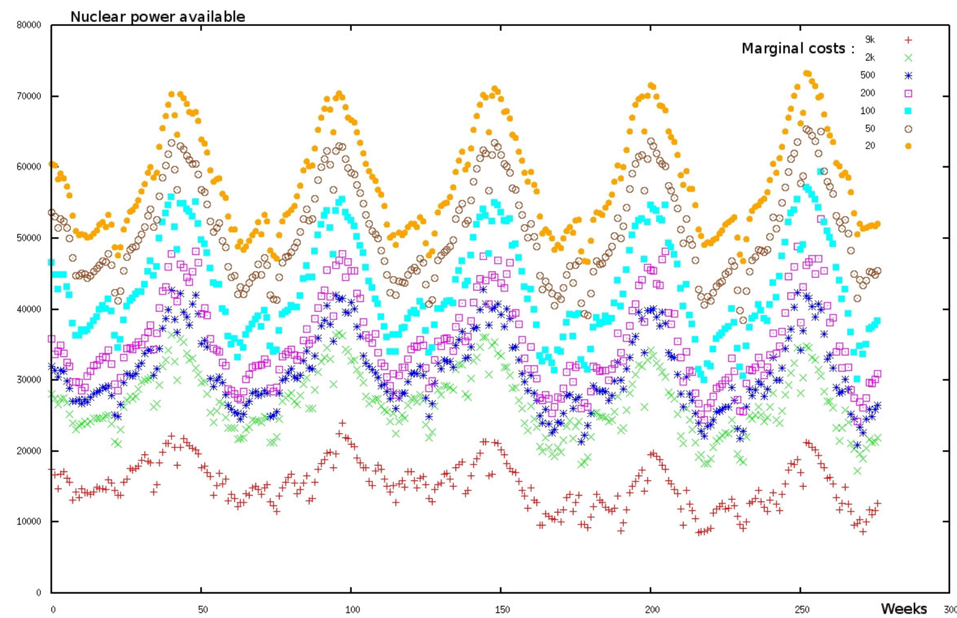

Figure 3: Marginal cost of completion by non-nuclear plants as a function of the week (x-axis) and nuclear availability, averaged on all steps and scenarios, instance B6.

and even the instance, it is very hard to guess whether an approach will be fast, robust and/or competitive. We hope that sub-problems will be studied thoroughly so has to provide insight and subroutines aiming at building bricks on which smart global approaches could rely. We want to share our ideas concerning such issues in this section.

The main characteristic of our approach is its hierarchical decomposition nature. This kind of strategy might be considered inefficient and not noble. But we gave some arguments showing that subsequent choices of variables might be evaluated approximately in compact implicit ways. Moreover, since other resolution techniques are successfully addressed by other competing teams, we prefer to stick to our approach in this discussion.

Assuming that the hierarchical structure is preserved, there are several issues that need to be addressed to judge the ability of our decomposition to provide solutions with highly competitive objective value.

9.1 Improving the ILP

Implicit refueling and fuel stocks at the end of cycles. Although simplistic and preliminary, our work challenges the idea that refueling values should be intimately studied with dates of outages. The hypothesis that fuel stock after an outage is limited to only RMAX should be improvable using (one) more dates of outage than only the last one (as done in Equation (5)).

Modulation and profile phases. Using variables p_{ikh} (production within each cycle) instead of just p_{ih}, there might be some ways to express the possibility of modulation. Moreover, assuming that the profile phase is convex, there might be ways to impose it using only linear inequalities, that is, only with upper bounds on the production. Notice that modulation and profile destroy the assumption that production is binary and constant within each week. Still it seems hard not to express modulation and profile without expliciting dates at which fuel reaches BO. Morevoer as long as our algorithm for power affectation does not modulate for objective value, we have no reason to allow the ILP to modulate.

Evaluating objective costs within the ILP. The problem with the variables x_{ikh} is that they don't allow to reformulate easily (an approximation of) the objective function of the subject. To evaluate the cost of a schedule, we should compute the amount of nuclear power dedicated to each pair (t, s). This last information seems too detailed, because it yields too many variables. One may wish to compute only an approximation of the completion cost by non-nuclear plants for each week (aggregated on steps and scenarios). Such values are plotted in section 8. We tried such an approach by approximating, for each week, the function of Figure 1 piecewise-linearly. Indeed, we tried both upper and under approximations, using exponential thresholds (like 20,50,100,200,500,2k,9k) on the derivative (*i.e.* marginal costs), as indicated in Figure 3. Because the function of Figure 1 is convex, minimizing a piecewise linear approximation can be expressed with linear inequalities only. The method works, indeed, it works so well that the optimal value found by CPLEX was very close to the exact value of the final solution. However the time taken by CPLEX on instance B9 to find an optimal solution was around 40 minutes. Moreover and more surprisingly, the quality of the solution was not really improved (it was a little better or worse depending on the instances). So we closed this direction of research during the challenge. The reason why this nice idea didn't bring improvements in the solution seems to be that something else is too roughly taken into account, and should be improved before thinking about cost evaluation within the ILP.

Polyhedral study and pre-processing constraints. We tried to provide to CPLEX strong constraints, but further studies on the linear relaxation of our formulation are required to judge its quality. Constraints (CT21) are a priori highly redundant and naively written, bunch of constraints (CT21) should be advantageously reformulated in the format of (CT20) by studying subset of plants that conflict together.

Decomposability of the ILP Assuming we use non-correlating objective functions (such as Equation 6), the ILP could be submitted to CPLEX for each connected components one at a time (or in parallel).

Using constraint based and local search solvers. After all, why solving the ILP at optimality while we know it is only an approximation of the global problem ? One might wish to apply other resolution paradigms originating from the SAT community like pseudo-boolean programming [3]. Local search [1] seems also a promising way in order to find good solutions faster.

9.2 Improving the refueling and the power assignment

Modulation for value. Our approach doesn't look at future demand when dealing with a given scenario at a given time step. We only use modulation for feasibility.

Faster greedy procedures. Because computational time was not a key point that needed improvement, we made intensive use of nuclear power affectation in order to check feasibility. We clearly recompute again and again similar affectations when adjusting the refueling values. In some instances, the time to adjust refueling is therefore substantial. In order to accelerate recognizing set of refueling that are infeasible, it would be useful to identify scenarios and step windows that are the most critical. Recomputing only (or first) on these critical points would be natural.

Smarter view on good refueling. One thing that we overlooked is that although $r(i,k)$ cannot be increased, fuel level might be far below $BO_{i,k}$ at the end of cycle (i,k), yet it might be possible to increase $r(i,k-1)$. More generally, our view on refueling would benefit visualizing many outputs and comparing with better solvers to evaluate how much we lose at this point. This idea can be generalized easily, as we explain now.

9.3 Creating collaboration between solvers of the challenge.

Following our decomposition, one might wish to evaluate each competing solver of the challenge not on the global aspect of the subject, but only in solving sub-problems. The following sketch of protocol should provide an easy framework. Given solvers S_1 and S_2, and given an instance I of the problem, let S_1 solve I and obtain solution I_1. Gather part of the values in I_1 (like $\{ha(i,k)\}$ and/or $\{r(i,k)\}$). Fix these values in I to obtain a more constrained instance I'. Run S_1 and S_2 on I' and compare speeds and values. We can have up to 3 solvers collaborating this way, fixing sequentially

$\{ha(i,k)\}$, $\{r(i,k)\}$, $\{p_2(i,t,s)\}$ and $\{p_1(j,t,s)\}$. Unfortunately, notice that this idea will not be so easy to use if we want to fix other critical informations (like time index of entrance into profile phase) because such informations are not designed to be encoded in the format of inputs.

References

[1] T. Benoist, B. Estellon, F. Gardi, R. Megel, and K. Nouioua. Localsolver 1. x: a black-box local-search solver for 0-1 programming. *4OR: A Quarterly Journal of Operations Research*, pages 1–18, 2011.

[2] F. Brandt, R. Bauer, M. Völker, and A. Cardeneo. A constraint programming-based approach to a large-scale energy management problem with varied constraints. *Journal of Scheduling, to appear*, 2012.

[3] N. Eén and N. Sörensson. Translating pseudo-boolean constraints into sat. *Journal on Satisfability, Boolean Modeling and Computation*, 2:1–25, 2006.

[4] F. Fourcade, E. Johnson, M. Bara, and P. Cortey-Dumont. Optimizing nuclear power plant refueling with mixed-integer programming. *European journal of operational research*, 97(2):269–280, 1997.

[5] F. Gardi and K. Nouioua. Local search for mixed-integer nonlinear optimization: a methodology and an application. *Evolutionary Computation in Combinatorial Optimization*, pages 167–178, 2011.

[6] S. Godskesen, Th. Jensen, N. Kjeldsen, and R. Larsen. Solving a real-life, large-scale energy management problem. *Journal of Scheduling, to appear*, 2012.

[7] A. Gorge, A. Lisser, and R. Zorgati. Stochastic nuclear outages semidefinite relaxations. *Computational Management Science*, pages 1–17, 2012.

[8] M. Khemmoudj, M. Porcheron, and H. Bennaceur. When constraint programming and local search solve the scheduling problem of electricité de france nuclear power plant outages. *Principles and Practice of Constraint Programming-CP 2006*, pages 271–283, 2006.

[9] R.M. Lusby, L.F. Muller, and B. Petersen. A solution approach to the roadef/euro 2010 challenge based on benders' decomposition. *http://orbit.dtu.dk/en/, Report 18.2010*, 2010.

[10] G.L. Nemhauser and L.A. Wolsey. *Integer and combinatorial optimization*, volume 18. Wiley New York, 1988.

[11] M. Queyranne and A.S. Schulz. *Polyhedral approaches to machine scheduling*. TU, Fachbereich 3, 1994.

[12] A. Rozenknop, R. Wolfler Calvo, L. Alfandari, D. Chemla, and L. Letocart. Solving eletricity production planning by column generation. *Journal of Scheduling, to appear*, 2012.

[13] J.M. van den Akker, C.A.J. Hurkens, and M.W.P. Savelsbergh. Time-indexed formulations for machine scheduling problems: Column generation. *INFORMS J. Comput.*, 12(2):111–124, 2000.