# Model Driven Decision-Making Methods

Segurini Gloria [567352] g.segurini@studenti.unipi.it

University of Pisa
A.Y. 2022/2023

# Contents

**1 Introduction 1**

**2 Mixed-Integer Linear Programming formulation 1**

2.1 Problem description . . . 1

2.2 Model presentation and in-depth explanation . . . 2

**3 Instances generation 4**

3.1 Overview and additional parameters description . . . 4

3.2 Instance generation constraints declaration and explanation . . . 5

**4 Model implementation through CPLEX and solution's performance analysis 7**

4.1 CPLEX performance with a constant number of days . . . 7

4.2 CPLEX performance with distinct numbers of days . . . 10

**5 Performances improvement 12**

5.1 Model improvements . . . 12

5.1.1 Model improvement results . . . 13

5.2 Solver's settings improvement . . . 17

5.3 Conclusions . . . 18

# 1 Introduction

The present report shows the deployment of the Model Driven Decision-Making Methods project. The aim of the project is to put into practice the knowledge acquired during the course, developing a mathematical model to describe an optimization problem. In particular, the track of the current project regards the modeling of an electricity transmission system operator which has to plan power plant operations while satisfying all the demand at least cost.

# 2 Mixed-Integer Linear Programming formulation

## 2.1 Problem description

The track of the project is the following : *An electricity transmission system operator (TSO) is planning power plant operations for the next week, divided into seven 1-day periods, each of which has different demand requirements. The TSO has a fixed number of power plants of two types. Wind and solar power plants have a small per-unit generation cost, but supply differs on daily basis due to changing weather conditions and cannot be modified. They do not exceed 40% of the total installed power plant capacity. Thermal generators have higher per-unit generation cost, but with greater decision-making flexibility. In particular, the TSO can decide to keep such a power plant off in order to save money. However, it would need one full day to become operational at a fixed cost. If a thermal power plant is running, then there is a fixed cost and variable cost based on the amount of energy generated. There are also lower and upper limits on the energy output. A generator cannot be on/off for periods shorter than 2 days. The problem is to satisfy all the demand at least cost.*

The following tables show the indexes, variables and parameters that will be used in the model. Next to the variables and parameters it is showed the unit of measurement in gigawatt-hours.

| **Index** | **Description** | **Range** |
|---|---|---|
| g | day of the week | [1,...,7] |
| i | renewable power plant | [1,...,n] |
| j | thermal power plant | [1,...,m] |

Table 1: Indexes description

| **Variable** | **Type** | **Description** |
|---|---|---|
| $r_{gi} - GWh$ | continuous | daily renewable energy produced (wind and solar) of power plant $i$ |
| $t_{gj} - GWh$ | continuous | daily thermal energy produced for power plant $j$ |
| $y_{gj}$ | binary | *j-th* power plant switched on/off on *g-th* day |

Table 2: Variables description

| **Parameter** | **Type** | **Description** |
|---|---|---|
| $\mathbf{rcv_i}$ - $€/GWh$ | continuous | variable cost for renewable energy for power plant $i$ |
| $\mathbf{tcv_j}$ - $€/GWh$ | continuous | variable cost for thermal energy for power plant $j$ |
| $\mathbf{tcf_j}$ - € | continuous | fixed cost for thermal energy for power plant $j$ |
| $\mathbf{d_g} - GWh$ | continuous | daily demand of total energy (renewable and thermal energy) |

| | | |
|---|---|---|
| $\mathbf{c_{gi}} - GWh$ | continuous | daily capacity of renewable energy for power plant $j$ |
| $\mathbf{l_j} - GWh$ | continuous | lower bound to thermal energy for power plant $j$ |
| $\mathbf{u_j} - GWh$ | continuous | upper bound to thermal energy for power plant $j$ |

Table 3: Parameters description. The parameter $\mathbf{c_{gj}}$ refers to the changing weather conditions

## 2.2 Model presentation and in-depth explanation

1. $min \sum_{g=1}^{7}(\sum_{i=1}^{n} \mathbf{rcv_i} \cdot r_{gi} + \sum_{j=1}^{m} \mathbf{tcv_j} \cdot t_{gj} + \mathbf{tcf_j} \cdot y_{gj})$
2. $y_{gj} \in \{0, 1\}, \forall g = 1, ..., 7, \forall j = 1, ..., m$
3. $r_{gi} \geq 0, \forall g = 1, ..., 7, \forall i = 1, ..., n$
4. $t_{gj} \geq 0, \forall g = 1, ..., 7, \forall j = 1, ..., m$
5. $\mathbf{d_g} \leq \sum_{i=1}^{n} r_{gi} + \sum_{j=1}^{m} t_{gj}, \forall g = 1, ..., 7$
6. $r_{gi} \leq \mathbf{c_{gi}}, \forall g = 1, ..., 7, \forall i = 1, ..., n$
7. $\mathbf{l_j} \cdot y_{1j} \leq t_{1j}, \forall j = 1, ..., m$
8. $\mathbf{l_j} \cdot (y_{(g-1)j} + y_{gj} - 1) \leq t_{gj}, \forall g = 2, ..., 7, \forall j = 1, ..., m$
9. $t_{gj} \leq \mathbf{u_j} \cdot y_{gj}, \forall g = 1, ..., 7, \forall j = 1, ..., m$
10. $t_{gj} \leq \mathbf{u_j} \cdot y_{(g-1)j}, \forall g = 2, ..., 7, \forall j = 1, ..., m$
11. $y_{1j} = 1, \forall j = 1, ..., m$
12. $y_{2j} \geq 0, \forall j = 1, ..., m$
13. $y_{2j} \leq 1, \forall j = 1, ..., m$
14. $y_{gj} \geq y_{(g-1)j} - y_{(g-2)j}, \forall g = 3, ..., 7, \forall j = 1, ..., m$
15. $y_{gj} \leq 1 - y_{(g-2)j} + y_{(g-1)j}, \forall g = 3, ..., 7, \forall j = 1, ..., m$

Equation number 1 shows the objective function to be minimized.
As written in the track, we have to minimize the costs on seven one-day periods, so this is why the first sum goes from g = 1 to 7. The second summation sums all the variable unitary costs of renewable energy, that is to say wind and solar energy from $i = 1$ to $n$, multiplied by the amount of renewable energy produced. The second summation sums all the variable unitary costs of thermal energy from $j = 1$ to $m$ multiplied by the amount of thermal energy produced. The summation also includes the fixed costs multiplied by the binary variable $y$: in this way, the fixed costs will be equal to 0 when the thermal power plant $j$ is switched off on day $g$ and 1 otherwise.

Equation number 2 refers to the binary variable $y_{gj}$ modeling the switched-on power plant when the variable takes value 1 and 0 otherwise.

Equations number 3 and 4 refer to the fact that daily the amount of energy produced by each power plant cannot be negative.

Equation number 5 refers to the fact that, independently from the type of power plant, the total daily energy cannot be lower than the daily demand. Since the solver will try to reduce the costs (and so the produced energy) as much as possible, the constraint was set as an inequality and it was not necessary to set an exact equality.

Equation number 6 refers to the fact that wind and solar power plants' supply differs on a daily basis due to changing weather conditions. Since we are dealing with renewable energy, this constraint was set as an equality, because the produced energy will never be lower than the daily capacity $\mathbf{c_{gi}}$.

Equations number 7, 8, 9 and 10 aim to give lower and upper limits to the thermal energy output, taking into account that the power plants need one full day to become operational at a fixed cost. It is supposed that on the first day the thermal power plants already produce energy. This is modeled by the constraints 11, 7 and 9, which enable the parameters $\mathbf{l_j}$ and $\mathbf{u_j}$ and set a lower and upper bound to the energy output.

From the second day onwards, it will be necessary to also look at the state of the power plants on the previous day. The only possibility for the power plant to produce energy on the g-th day is to be on both the *(g-1)-th* day and the g-th. It is important to notice that if the power plant was switched-off on the *(g-1)-th* day and switched-on on the *g-th* day, equation number 10 sets the upper bound to 0 and equation number 9 sets it to $\mathbf{u_j}$. The solver will then set the upper bound to 0, so as to satisfy both inequalities. The same goes for the power plant switched-on on the *(g-1)-th* day and switched-off on the *g-th* day.

Table 4 shows the possibile combinations of thermal power plant $j$ switched-on or off for two consecutive days and the relative upper and lower bounds to $\mathbf{t_{gj}}$. Obviously, when the power plant is switched-off for two consecutive days the lower bound will be negative by only considering constraint number 8, but it will become 0 by also taking into account constraint number 4.

| $y_{(g-1)j}$ | $y_{(g)j}$ | $t_{gj}$ l.b. | $t_{gj}$ u.b. |
|---|---|---|---|
| 0 | 0 | -$\mathbf{l_j}$ | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | $\mathbf{l_j}$ | $\mathbf{u_j}$ |

Table 4: Possible combinations of thermal power plant on/off on two consecutive days. The terms l.b. and u.b. stay respectively for lower and upper bound to $t_{gj}$. The values hold $\forall g = 2, ..., 7$.

Equations number 11, 12, 13, 14 and 15 model the constraint stating that a generator cannot be on/off for periods shorter than 2 days. The idea is to first set the $y$ variable for first day to be equal to 1 (see constraint 11).

In addition, since the basic assumption is that the thermal power plants were already switched-on the day before the beginning of the seven days of the problem, on the second day the power plants can either take value 0 or 1. This is modeled by constraints 12 and 13.

From the third day onwards, it was necessary to check for the different possible scenarios. Table 5 shows the possible combinations of the generator being switched on or off depending on two consecutive days and the relative value that the variable $y_{gj}$ must take in terms of upper and lower bounds (respectively $y_{gj}$ u.b. and $y_{gj}$ l.b.).

| $y_{(g-2)j}$ | $y_{(g-1)j}$ | $y_{gj}$ l.b. | $y_{gj}$ u.b. |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 |

Table 5: Possible combinations of thermal power plant on/off on two consecutive days. The terms l.b. and u.b. stay respectively for lower and upper bound to $y_{gj}$. The values hold $\forall g = 3, ..., 7$.

The first row of Table 5 models the case of two consecutive days on which the thermal power plant is off. In this case, the next variable $y_{2(g)p_2}$ can indifferently take value 0 or 1, since in this case the generator is off for a period not shorter than two days. The same reasoning can be done for the last line of Table 5, but in this case the the generator is on for a period not shorter than two days.

With respect to the second line of Table 5, since the generator was off the two previous days and on one day before the current day $g$, this means that the variable $y_{2(g)j}$ will have to take value 1, so that the generator is on for a period not shorter than two consecutive days. For this reason, the trick was to set both the upper and lower bounds equal to 1 through equations 12 and 13. The same holds for the third row of Table 5, where the generator is off for a period not shorter than two days and in this case both the upper and lower bounds are set = 0.

# 3 Instances generation

## 3.1 Overview and additional parameters description

The second part of the project was to recover or generate instances of the problem at hand.

In order to make the generated data consistent with the problem, it was necessary to take into account some additional constraints that are not explicitly stated by the text. Table 6 gives a brief description of the parameters used in the additional constraints below. All the ranges for the various parameters are explained in after the additional constraints statement. All parameters are continuous.

| **Parameter** | **Description** |
|---|---|
| $tCT$ - $GWh$ | Daily thermal total capacity summed up on all power plants |
| $\mathbf{d_g}$ - $GWh$ | daily demand of energy |
| $rCT$ - GWh | Daily renewable total capacity summed up on all power plants |
| $CT - GWh$ | Daily total capacity of total energy summed up on all power plants |
| $\gamma_i$ | *i-th* portion of rCT (in decimal value) generated from power plant $i$ |
| $coef_g$ | daily portion of capacity available on the basis of weather conditions |
| $c_{gi} - GWh$ | daily capacity of power plant $i$ on the basis of weather conditions |
| $\gamma_j$ | *j-th* portion of tCT (in decimal value) generated from power plant $j$ |

Table 6: Additional constraints' parameters description

It is important to notice that $tCT, rCT$ and $CT$ refer to the daily capacity, but the value is the same every day, because all power plants are in the same area. In addition, the $\gamma_i$ and $\gamma_j$ coefficients were used, because different power plants can have different sizes and they can contribute differently to the total capacity.

## 3.2 Instance generation constraints declaration and explanation

a) $tCT \geq \mathbf{d_g}, \forall g = 1, ..., 7$

b) $CT = rCT + tCT$

c) $rCT \leq 40\% CT$

- $tCT \geq 60\% CT$
- $CT \leq tCT/0.6$

d) $c_{gi} = \gamma_i \cdot rCT \cdot coef_g, \forall g = 1, ..., 7, \forall i = 1, ..., n$

e) $\mathbf{u_j} = \gamma_j \cdot tCT, \forall j = 1, ..., m$

f) $\mathbf{l_j} < \mathbf{u_j}, \forall j = 1, ..., m$

g) $\mathbf{rcv_i} < \mathbf{tcv_j}, \forall i = 1, ..., n, \forall j = 1, ..., m$

h) $coef_g \geq 0.2, \forall g = 1, ..., 7$

The instance generation constraint *a)* states that the thermal total capacity (that is the capacity on all power plants) must be sufficient to satisfy the daily demand. This was done, because it was necessary to make sure that constraint number 5 always holds. In fact, if the weather was rainy or cloudy, the produced renewable energy could be 0 every day in the worst case. The idea was to start off by taking into account the daily average request of energy $\mathbf{d_g}$ which was set equal to 51.7 GWh [1]. It was necessary to generate 7 instances.This was done by means of a uniform distribution with lower and upper limits respectively equal to 49.7 and 53.7, so as to have a $\pm 2$ interval with respect to 51.7. Then, in order to generate **tCT** it was used a uniform distribution with lower limit equal to the largest of the previously generated demands and upper limit equal to the largest demand + 5.

The instance generation constraint *b)* states that the total capacity is given by the sum of the renewable total capacity and the thermal total capacity.

The instance generation constraint *c)* states that the total capacity on all power plants of renewable energy do not exceed 40% of the total installed power plant capacity. Since the available data already generated was $tCT$, it is necessary to declare the complementary of this constraint, that is to say that $tCT$ must be $\geq 60\% CT$. This can be done, because constraint *b)* holds. This leads to $CT \leq tCT/0.6$ and this gives us an upper bound to CT.

On the other hand, since $CT = rCT + tCT$, it is necessary for CT to be $\geq tCT$ and this gives us a lower bound to CT. Therefore, CT was generated through a uniform distribution with the upper and lower bounds just mentioned. Finally, in order to generate $rCT$ it was sufficient to subtract $tCT$ to $TC$.

The instance generation constraint *d)* on the renewable energy explains the composition of $c_{gi}$. The multiplication $\gamma_i \cdot rCT$ gives us the maximum capacity of the *i-th* power plant. This result must be multiplied by $coef_g$, so as to model the various weather conditions and how they affect the daily capacity. For a better explanation, let us make a trivial example: if the number

[1] https://ambientenonsolo.com/i-consumi-energetici-dei-107-comuni-capoluogo-di-provincia-e-citta-metropolitana/

$n$ of renewable power plants is equal to 2 and the total capacity is 100 GWh, we could have $\gamma_1$ equal to $40\% rCT$ and $\gamma_2$ equal to $60\% rCT$. Therefore, we would have that the maximum capacity is equal to 40GWh for the first power plant and 60GWh for the second one. However, since this is the best case scenario, these capacities must be multiplied by a coefficient that models the weather conditions. In the worst case, such coefficient would be equal to 0.2 on the *g-th* day.

Both the various $\gamma_i$ and the coefficients $coef_g$ were generated through a uniform distribution with lower bound equal to 0 and upper bound equal to 1. The sum of all $\gamma_i$ is equal to 1.

The instance generation constraint *e)* describes the value of the parameter $\mathbf{u_j}$ present in constraints 9 and 10. Again, here the idea was to divide the thermal total capacity $tCT$ among $j$ different thermal power plants through the terms $\gamma_j$.

The instance generation constraint *f)* just ensures that the lower bounds to the thermal power plants are lower than the upper bounds. The lower bounds were created by multiplying the upper bounds for a coefficient equal to 0.1.

The instance generation constraint *g)* states that thermal generators have a higher per-unit generation cost with respect to renewable power plants. The idea was to first generate the costs for the renewable energy through a uniform distribution with lower bound equal to 16.700 €/GWh and upper bound equal to 28.070 €/GWh [2]. The same reasoning was done for the thermal variable costs, for which it was used a uniform distribution with lower bound equal to 90.790 €/GWh and an upper bound equal to 105500 €/GWh [3].

With respect to the fixed costs, it was decided to take an average production of 134.06 €/KW per year (as a result of the mean between the costs of geothermal, biomass and nuclear energy) [4]. Since this is a per-year value, it was necessary to divide it by 365, for a total of 0.3673 €/KW per-day. This value was then multiplied by 1.000.000 for a cost of 367.300 € for 1 GW of energy in one day. It was then necessary to know the capacities of each of our thermal power plants, which are the various $\mathbf{u_j}$. Since this last term refers to the capacity of a power plant in a day, that is to say in 24 hours, it must be divided by 24 so as to find the power. By multiplying the result by 367.300 we finally find the fixed cost of the power plant $j$. Obviously, this value varies on the basis of the power plant, because these can have different characteristics such as the size.

The instance generation constraint *h)* was added so that renewable generators can always produce at least 20% of their total capacity.

The following Table shows a very small instance of data generation: by fixing $g$ equal to 3, even if there is a small number of power plants the instance gets a size of 3* $max\{n, m\}$ . For this reason, in the following instance $n$ and $m$ were set equal to 2 and the number of days was also set equal to 3, just to give an idea of the data generation process.

[2]https://www.fossilconsulting.com/blog/operations/power-industry-economics/

[3]https://www.fossilconsulting.com/blog/operations/power-industry-economics/

[4]https://www.statista.com/statistics/519144/power-plant-operation-and-maintenance-costs-in-the-us-by-technology/

| **Parameter** | **Values** |
|---|---|
| **n** | 2 |
| **m** | 2 |
| **g** | 3 |
| $\mathbf{d_g}$ list | [50.29787441928239, 53.695299734338604, 52.14962920059396] |
| $\mathbf{c_{gi}}$ list | [[5.601140231395193, 6.595491517515427, 2.8226100529195213], [18.26458771107128, 21.50703755711552, 9.204163216041916]] |
| $\mathbf{u_j}$ list | [23.00030716704262, 32.96710693942776] |
| $\mathbf{l_j}$ list | [2.300030716704262, 3.296710693942776] |
| $\mathbf{rcv_i}$ list | [25539.191177559318, 23928.57456717387] |
| $\mathbf{tcv_j}$ list | [100181.14191919251, 99132.79358135158] |
| $\mathbf{tcf_j}$ list | [352000.53, 504534.1] |
| **tCT** | [55.96741410647038] |
| **rCT** | [29.329477550246132] |
| $\gamma_\mathbf{i}$ list | [0.23469387755102042, 0.7653061224489796] |
| $\gamma_\mathbf{j}$ list | [0.410958904109589, 0.589041095890411] |

Table 7: Data generation instance

## 4 Model implementation through CPLEX and solution's performance analysis

The mathematical model was implemented through CPLEX, since it is one of the most powerful and widely used solvers. The Python script used to generate the instances was adapted so as to give a *.dat* file in output, which was used on the CPLEX IDE. The *.mod* file was also built. The calculations were performed on a computer running Windows 11 Pro with 8GB RAM and processor Inter Core i5. CPLEX's version is 22.1.1.0 .

In order to test the capabilities of the solver, it was decided not to pry on the value of the objective function, because it is related to many variables. In fact, the value of the objective function will decrease on the basis of the demand and the variable and fixed costs. Also the coefficient $\mathbf{c_{gi}}$ changes the value, because if the weather conditions are adverse, the renewable power plants' capacity will decrease, and this means that the demand will be satisfied by using the thermal energy, which has a higher per-unit costs in addition to the fixed costs. Obviously, the costs also change the value of the objective function. Therefore, it was decided to test the solver's breakpoint in terms of size of the problem. For Table 8, the lowest gap reached by the solver is always 0,01%.

### 4.1 CPLEX performance with a constant number of days

The results of the solution time with the various sizes of $n$ and $m$ are shown in Table 8. All the configurations were run with the default CPLEX's settings and with $g$ = *7*.

| ***column index*** | ***n,m*** | ***Initial gap*** | ***Solution time*** | ***Initial obj. fun. value*** | ***Final obj. fun. value*** |
|---|---|---|---|---|---|
| 1 | 6, 6 | 73,35% | 0,06 sec. | 3,96221e+07 € | 28579964.437 € |
| 2 | 40,50 | 87,09% | 0,62 sec. | 4,22880e+07 € | 34905352.82 € |
| 3 | 50,40 | 86,26% | 1,61 sec. | 4,17469e+07 € | 40271049.149 € |
| 4 | 50, 50 | 87,83% | 2,05 sec. | 4,10124e+07 € | 32063760.306 € |

| | | | | | |
|---|---|---|---|---|---|
| 5 | 500, 500 | 87,85% | 2.02 sec. | 4,08229e+07 € | 31712803.651 € |
| 6 | 694,934 | 89,86% | 7.41 sec. | 4,00389e+07 € | 27136561.056 € |
| 7 | 1000, 1000 | 86,40% | 5,30 sec. | 4,23099e+07 € | 40270501.015 € |
| 8 | 1768,1885 | 84,42% | 78,20 sec. | 3,40876e+07 € | 32454627.488 € |
| 9 | 2000, 2000 | 84,78% | 56,12 sec. | 3,13207e+07 € | 29923825.773 € |
| 10 | 2819,2788 | 86,66% | 71,06 sec. | 3,67614e+07 € | 36104228.23 € |
| 11 | 3000,3000 | 85,23% | 13,78 sec.. | 3,75754e+07 € | 37243595.828 € |
| 12 | 9028,9452 | 88,24% | 1726,98 sec. | 4,45347e+07 € | 38157385.215 € |

Table 8: CPLEX solution time with various $n$ and $m$, $g = 7$

First of all, it is worth noticing that sometimes the solution time decreases with higher values of $n$ and $m$ and also by increasing the number of days. In this regard, the following figure shows the solver's progression with the settings of line 11 of Table 8.

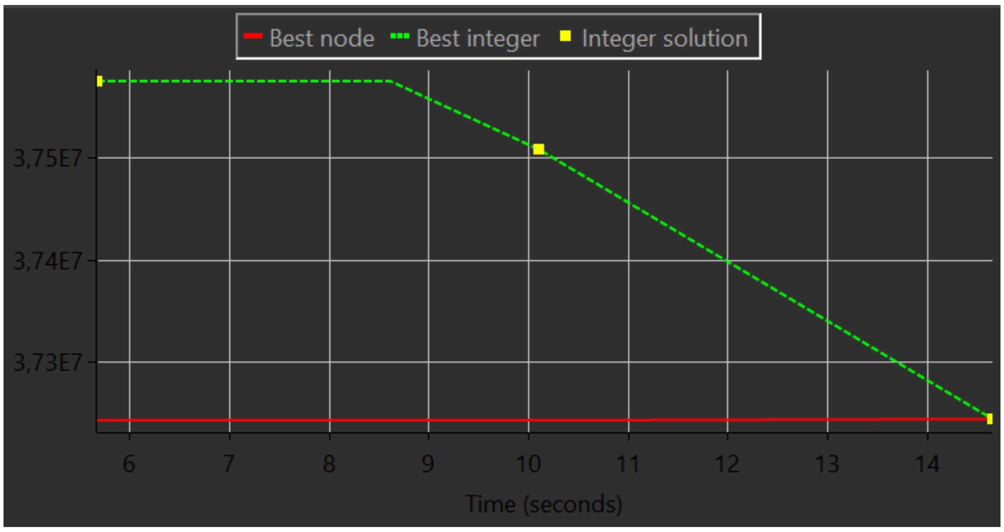


Figure 1: CPLEX's progression with $n = 3000$, $m = 3000$ and $g = 7$

Despite a higher value of $n$ and $m$ this configuration has better solution time than the instance of line 9 of Table 8, whose progression is visible in the following figure. It is evident that Figure 1 has a faster convergence to the optimal solution. A possible explanation could be that this happens because of particularly favourable initial solutions.

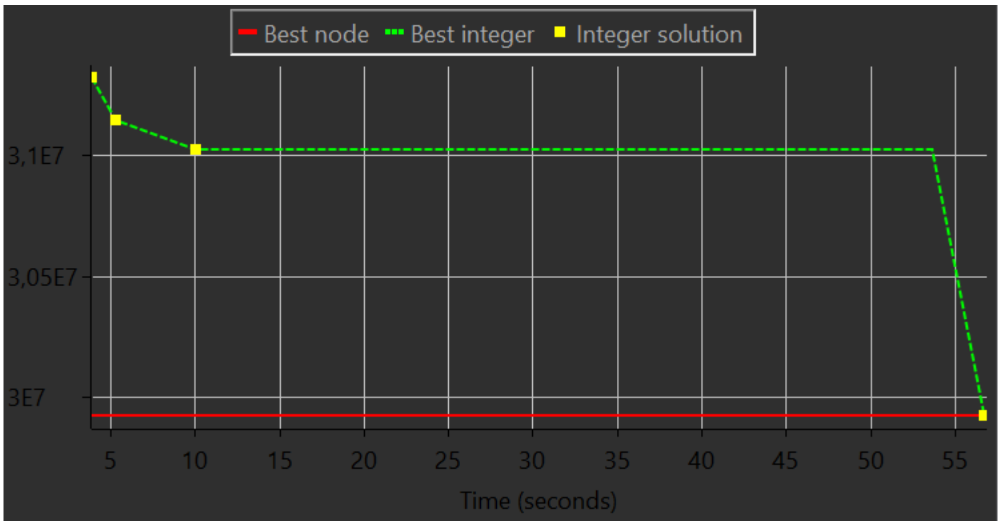


Figure 2: CPLEX's progression with $n = 2000$, $m = 2000$ and $g = 7$

As a final consideration about the isntances of Table 8, despite the high value of $n$ and $m$ of Table 8, the solution time does not increase too much; the highest one is the one of line 12, where the number of power plants was set really high. However, even in this worst case the solution time is more than acceptable. In order to give an idea of the solver's convergence speed, the following Figure shows the CPLEX's performance with the smallest configuration ($n = 6$, $m = 6$, $g = 7$).

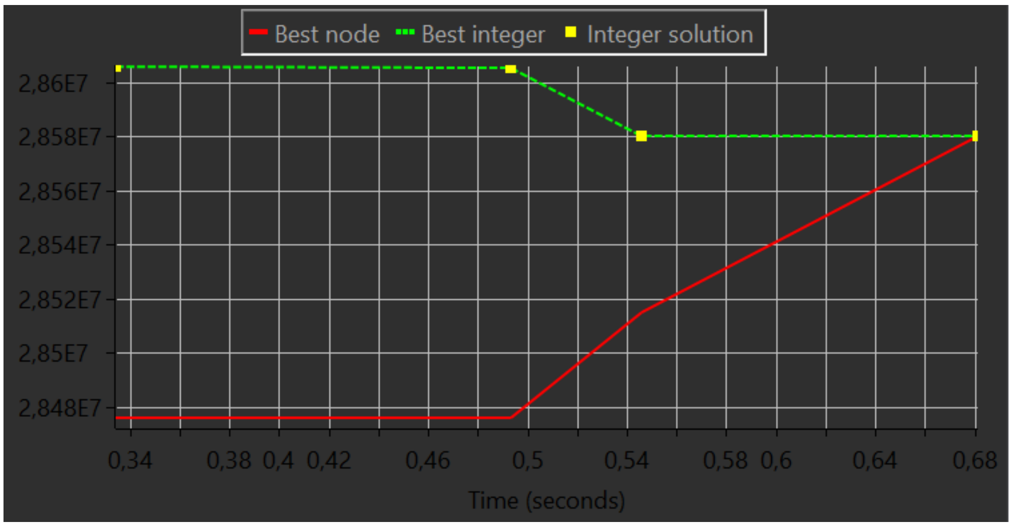


Figure 3: CPLEX's progression with emphn $= 6$, $m = 6$, $g = 7$

The solution converges really fast to the optimal value.

The following figure shows instead the solver's performance with the configuration of line 12 of Table 8.

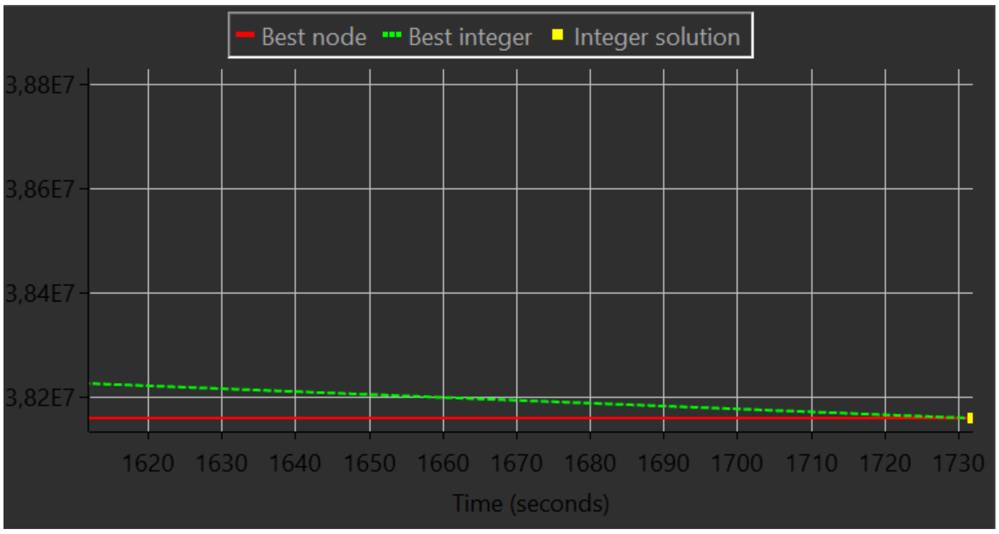


Figure 4: CPLEX's progression with emphn $= 9028$, $m = 9452$, $g = 7$

In this case, it is possible to notice that the solver is able to reach the optimal solution, but is a lot slower than the instance of Figure 3. In particular, the log file shows that the gap was first stuck at 0,99% and could no longer decrease. Thus, a restart was necessary. The algorithm may have spent a lot of time without finding significant improvements in the solution, which could be due to poor starting conditions or getting trapped in a local minimum. Restarting can help in exploring new regions of the solution space. The gap finally decreased to 0,01%.

Another interesting feature about CPLEX is that it gives information about the cuts applied. The following ones are the most interesting that are applied in general: Cover cuts, Flow cuts, Mixed integer rounding cuts, Gomory fractional cuts.

## 4.2 CPLEX performance with distinct numbers of days

The second option was to test the solver by varying the number of days of the power plants operations. The results are shown in Table 9.

| ***column index*** | ***n,m*** | ***g*** | ***Initial gap*** | ***Solution time*** | ***Initial obj. fun. value*** | ***Final obj. fun. value*** |
|---|---|---|---|---|---|---|
| 1 | 6,6 | 15 | 71,73% | 0,16 sec. | 8,48603e+07 € | 72674947.628 € |
| 2 | 40,50 | 15 | 93,97% | 5,02 sec. | 9,24129e+07 € | 82874250.345 € |
| 3 | 130, 146 | 15 | 94,23% | 11,34 sec. | 7,49302e+07€ | 67664839.367 € |
| 4 | 295,228 | 30 | 97,32% | 32,64 sec. | 1,96761e+08 € | 152735859.614 € |
| 5 | 337,309 | 50 | 98,62% | 85,03 sec. | 3,16058e+08 € | 217721698.164 € |
| 6 | 304,325 | 100 | 99,17% | 190,83 sec. | 6,53834e+08 € | 521808368.488 € |
| 7 | 449,438 | 150 | 99,46% | 411,19 sec. | 9,76661e+08 € | 783116485.181 € |
| 8 | 566,593 | 200 | 99,54% | 274,41 sec. | 1,26216e+09 € | 1131934524.624 € |
| 9 | 179,184 | 300 | 99,76% | 4437,09 sec. | 2,06362e+09 € | 1407155332.081 € |
| 10 | 214,233 | 300 | 99,84% | 5428,66 sec. | 2,10682e+09 € | 1,24629e+09 € |
| 11 | 613,750 | 300 | 99,72% | 8811,41 sec. | 2,01842e+09 € | 1,67102e+09 € |

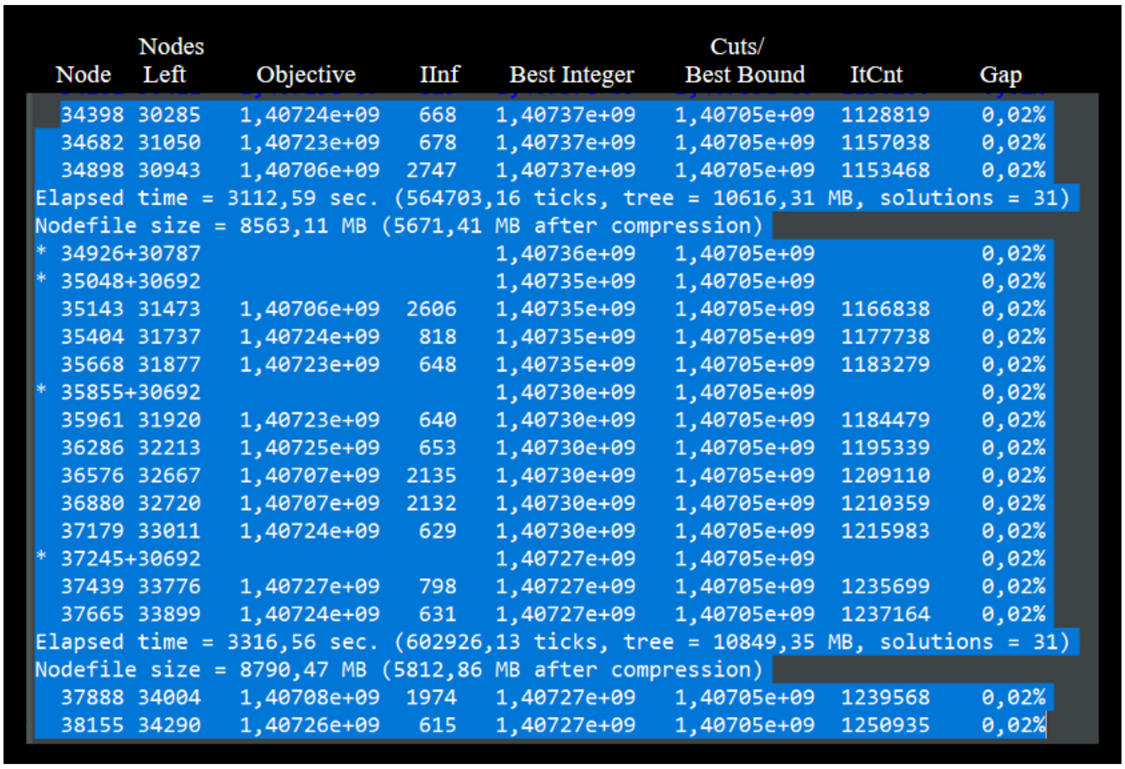
Table 9: CPLEX solution time with various $n$, $m$ and $g$

From the results shown in Table 9 it is clear that by increasing a lot the number of days, the solver begins to encounter difficulties. In fact, the computational time needed to find a solution starts to grow significantly in the configuration of line 9 of Table 9.

With respect to this configuration, from the log file it is evident that the solver employs much time by decreasing the gap from 0,02% to 0,01%.

In particular, the progress is the following:

- Before 256 seconds, the solver decreases the gap really fast from 99,76% down to 0,04%
- Between 256 and 317 seconds, the solver decreases the gap from 0,04% down to 0,02%.
- Between 317 and 376 seconds CPLEX is stuck with a gap of 0,02% and performs a restart. The best possible explanation by looking at the log file is that CPLEX is exploring new nodes, without coming up with a better solution.
- After the restart, that is to say between 317 and 6000 seconds CPLEX keeps exploring the search tree with a gap of 0,02%. In particular, the *Objective* value equal to 1,40705e+09 € starts to increase progressively, while the *Best Integer value* remains equal to 1,40737e+09 €. After 3111 seconds the *Objective* value reaches 1,40724e+09 € and the *Best Integer value* is still 1,40737e+09 €. From this point onwards, the *Best Integer value* begins to swing between values that reach a lower bound equal to 1,40706e+09 € and an upper bound equal to 1,40737e+09 €. In the meanwhile, the *Best Integer value* progressively decreases. An instance of this behaviour of the solver is shown in the following Figure.



```
        Nodes                                         Cuts/
   Node  Left     Objective    IInf   Best Integer   Best Bound    ItCnt     Gap
  34398 30285   1,40724e+09   668   1,40737e+09   1,40705e+09  1128819    0,02%
  34682 31050   1,40723e+09   678   1,40737e+09   1,40705e+09  1157038    0,02%
  34898 30943   1,40706e+09  2747   1,40737e+09   1,40705e+09  1153468    0,02%
Elapsed time = 3112,59 sec. (564703,16 ticks, tree = 10616,31 MB, solutions = 31)
Nodefile size = 8563,11 MB (5671,41 MB after compression)
* 34926+30787                       1,40736e+09   1,40705e+09             0,02%
* 35048+30692                       1,40735e+09   1,40705e+09             0,02%
  35143 31473   1,40706e+09  2606   1,40735e+09   1,40705e+09  1166838    0,02%
  35404 31737   1,40724e+09   818   1,40735e+09   1,40705e+09  1177738    0,02%
  35668 31877   1,40723e+09   648   1,40735e+09   1,40705e+09  1183279    0,02%
* 35855+30692                       1,40730e+09   1,40705e+09             0,02%
  35961 31920   1,40723e+09   640   1,40730e+09   1,40705e+09  1184479    0,02%
  36286 32213   1,40725e+09   653   1,40730e+09   1,40705e+09  1195339    0,02%
  36576 32667   1,40707e+09  2135   1,40730e+09   1,40705e+09  1209110    0,02%
  36880 32720   1,40707e+09  2132   1,40730e+09   1,40705e+09  1210359    0,02%
  37179 33011   1,40724e+09   629   1,40730e+09   1,40705e+09  1215983    0,02%
* 37245+30692                       1,40727e+09   1,40705e+09             0,02%
  37439 33776   1,40727e+09   798   1,40727e+09   1,40705e+09  1235699    0,02%
  37665 33899   1,40724e+09   631   1,40727e+09   1,40705e+09  1237164    0,02%
Elapsed time = 3316,56 sec. (602926,13 ticks, tree = 10849,35 MB, solutions = 31)
Nodefile size = 8790,47 MB (5812,86 MB after compression)
  37888 34004   1,40708e+09  1974   1,40727e+09   1,40705e+09  1239568    0,02%
  38155 34290   1,40726e+09   615   1,40727e+09   1,40705e+09  1250935    0,02%
```

Figure 5: CPLEX's log file of the instance number 9 of Table 9, with$n$ = 179, $m$ = 184 and $g$ = 300

- In the end, the solver is able to finally reduce the gap to 0,01% : the *Objective* value decreases to 1,40722e+09 € and the *Best Integer value* is 1,40716e+09 €, as reported in Table 9.

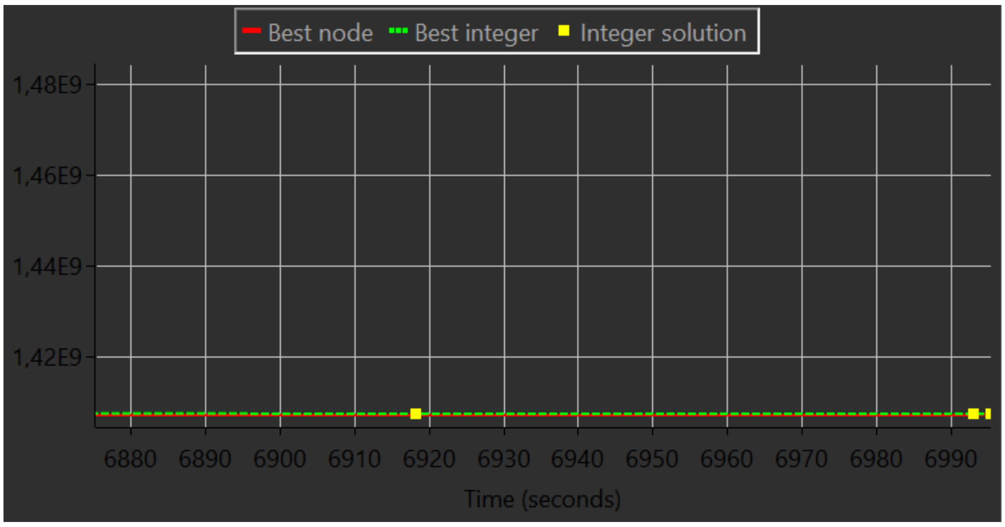


Figure 6: CPLEX's progression $n$ = 179, $m$ = 184 and $g$ = 300

With respect to the instance number 10 of Table 9, CPLEX gives an out-of-memory error. The following warning arises: MIP starts not constructed because of out-of-memory status. In fact, the solution log file is empty and for this last instance the value of the column *Final obj. fun. value* is the last *Best Integer* value of CPLEX's engine log file. However, the solver is still able to reach a 0,01% final gap.

About instance number 11 of Table 9, the solver employs more than two hours for reaching a deadlock phase where it is not able to decrease the gap below 1,91%. CPLEX's file *Solutions* is therefore empty, so also in this case the value of the column *Final obj. fun. value* is the last *Best Integer* value.

Finally, in the following section the aim will be to implement some ways to improve the solver's performance on the isntances of Table 9 in terms of solution time and/or objective function value.

## 5 Performances improvement

### 5.1 Model improvements

The first model improvement idea is to use the same criteria of the column generation approach, where a subset of variables on which to optimize is first selected. However, since the renewable power plants are the ones with the lowest per-unit cost, the solver always produces as much energy as possible with them. This means that selecting the subset of variables $r_{gi}$ and optimizing on them would make the solver set them equal to the parameter $\mathbf{c_{gi}}$. For this reason, the variable $r_{gi}$ was directly put equal to $\mathbf{c_{gi}}$ and the following constraints

3. $r_{gi} \geq 0, \forall g = 1, ..., 7, \forall i = 1, ..., n$

6. $r_{gi} \leq \mathbf{c_{gi}}, \forall g = 1, ..., 7, \forall i = 1, ..., n$

were removed. For this reason, the new objective function is now the following :

1. $min \sum_{g=1}^{7}(\sum_{i=1}^{n} \mathbf{rcv_i} \cdot \mathbf{c_{gi}} + \sum_{j=1}^{m} \mathbf{tcv_j} \cdot t_{gj} + \mathbf{tcf_j} \cdot y_{gj})$

The second model improvement idea was to set the following constraint

5. $\mathbf{d_g} \leq \sum_{i=1}^{n} r_{gi} + \sum_{j=1}^{m} t_{gj}, \forall g = 1, ..., 7$

as an equality one, avoiding producing more energy than needed. It is true that the optimization problem concerns a minimization objective function, but an equality constraint may help the solver in cutting off non-optimal solutions.

The third model improvement idea was to add the following constraints:

16. $\sum_{j=1}^{m} y_{gj} \geq 1, \forall g = 1, ..., 7$ This constraint states that every day at least one thermal power plant is switched on. It is possible to state it by construction of the data generation process.

17. $\sum_{g=1}^{7} \sum_{j=1}^{m} y_{gj} \leq g * m$ This constraint states that even if all the thermal power plants are switched on every single day, they cannot exceed g*m.

18. $\sum_{j=1}^{m} y_{gj} \leq m, \forall g = 1, ..., 7$ This constraint states that every day the number of the switched on power plants cannot exceed their quantity.

### 5.1.1 Model improvement results

As declared in the previous section, the idea now is to test the solver's performance both in terms of objective function value (since the data are the same used in Table 9) and solution time. The following table shows the results of the instances with just the first model improvement presented in section 5.1, that is to say by removing constraints 3 and 6 and by setting the new objective function.

| ***column index*** | ***n,m*** | ***g*** | ***Initial gap*** | ***Solution time*** | ***Initial obj. fun. value*** | ***Final obj. fun. value*** |
|---|---|---|---|---|---|---|
| 1 | 6,6 | 15 | 71,15% | 0,05 sec. | 8,48603e+07 € | 72674947.628 € |
| 2 | 40,50 | 15 | 92,91% | 8,38 sec. | 9,24129e+07 € | 82874086.899 € |
| 3 | 130, 146 | 15 | 89,01% | 9,02 sec. | 9,28759e+07 | 67666466.488 € |
| 4 | 295,228 | 30 | 94,70% | 11,22 sec. | 1,96761e+08 € | 152738157.509 € |
| 5 | 337,309 | 50 | 93,23% | 55,81 sec. | 3,16058e+08 € | 217727435.755 € |
| 6 | 304,325 | 100 | 96,93% | 633,41 sec. | 6,53834e+08 € | 521810175.908 € |
| 7 | 449,438 | 150 | 97,12% | 395,88 sec. | 9,76661e+08 € | 783113108.512 € |
| 8 | 566,593 | 200 | 98,90% | 783,84 sec. | 1,26216e+09 € | 1131931248.180 € |
| 9 | 179,184 | 300 | 95,66% | 3959,36 sec. | 2,06362e+09 € | 1407179300.964 € |
| 10 | 214,233 | 300 | 94,03% | 4848,64 sec. | 2,10682e+09 € | 1,24625e+09 € |
| 11 | 613,750 | 300 | 99,72% | 8819,41 sec. | 2,01842e+09 € | 1,67102e+09 € |

Table 10: CPLEX solution time with various $n$, $m$ and $g$ with model change number 1

In general, the changes to the model have had a positive impact on the solver's performance. In fact, some objective functions' values have decreased with respect to the results of Table 9 as well as some solution times. Since the renewable energy costs are lower than the thermal ones, this translated into a lower objective function value. However, in some cases the *Final obj. fun. values* have increased. This can be caused by the fact that in this way the solver is forced to use as much renewable energy as possible and exploit the entire capacity of a certain power plant on a certain day $c_{gi}$.

Furthermore, both the solutions of Table 10 and those of Table 11 are certified as optimal; the reason can be that the difference between the two objective function values is in the range of the tolerance of the solver, which is 1.0E-4 by default.

Afterwards, it was tried to see the solver's behaviour just with the implementation of the second model improvement, that is to say by setting constraint 5 as an equality one. The following Table shows the results.

| ***column index*** | ***n,m*** | ***g*** | ***Initial gap*** | ***Solution time*** | ***Initial obj. fun. value*** | ***Final obj. fun. value*** |
|---|---|---|---|---|---|---|
| 1 | 6,6 | 15 | 43,35% | 0,03 sec. | 7,07600e+07 € | 69239742.053 € |
| 2 | 40,50 | 15 | 93,35% | 4,66 sec. | 8,30973e+07 € | 81854168.261 € |
| 3 | 130, 146 | 15 | 92,07% | 7,86 sec. | 6,55700e+07 € | 62657181.824 € |
| 4 | 295,228 | 30 | 96,64% | 23,56 sec. | 1,52757e+08 € | 147444625.402 € |
| 5 | 337,309 | 50 | 98,12% | 75,22 sec. | 2,11780e+08 € | 200315739.381 € |
| 6 | 304,325 | 100 | 98,98% | 1318,80 sec. | 5,22302e+08 € | 507028741.63 € |
| 7 | 449,438 | 150 | 99,34% | 315,48 sec. | 7,82255e+08 € | 760165676.526 € |
| 8 | 566,593 | 200 | 99,88% | 707,33 sec. | 1,14519e+09 € | 1123767292.161 € |
| 9 | 179,184 | 300 | 99,66% | 4268,41 sec. | 1,39672e+09 € | 1322281749.44 € |

| | | | | | | |
|---|---|---|---|---|---|---|
| 10 | 214,233 | 300 | 99,77% | 6104,01 sec. | 1,21974e+09 € | 1123234801.438 € |
| 11 | 613,750 | 300 | 99,72% | 8811,41 sec. | 2,01842e+09 € | 1,67102e+09 € |

Table 11: CPLEX solution time with various $n$, $m$ and $g$ with model change number 2

With respect to the results of Table 9, the performances are much better both in terms of solution time and objective function value. Just the instances number 6, 8 and 10 have a higher solution time. Furthermore, this time instance number 10 has finished and has a lower objective function value with respect to the original one. It is natural that the objective function values have decreased: this time, the produced energy must be just the right amount to satisfy the demand.

Finally, it was tried to test the solver's performance with the introduction of the new constraints 16, 17 and 18, which are quite trivial, but they were added so as to perform some kind of additional cuts that could help the solver eliminate some non-optimal solutions earlier. In order to test each constraint's contribution, they were tested separately. The following Table shows the results with just the introduction of constraint number 16.

| ***column index*** | ***n,m*** | ***g*** | ***Initial gap*** | ***Solution time*** | ***Initial obj. fun. value*** | ***Final obj. fun. value*** |
|---|---|---|---|---|---|---|
| 1 | 6,6 | 15 | 71,73% | 0,06 sec. | 8,48603e+07 € | 72674947.628 € |
| 2 | 40,50 | 15 | 94,09% | 3,81 sec. | 9,42756e+07 € | 82874203.052 € |
| 3 | 130, 146 | 15 | 94,38% | 11,12 sec. | 9,53188e+07 € | 67665010.639 € |
| 4 | 295,228 | 30 | 97,36% | 10,66 sec. | 1,99544e+08 € | 152736693.594 € |
| 5 | 337,309 | 50 | 98,64% | 91,88 sec. | 3,20780e+08 € | 217727735.346 € |
| 6 | 304,325 | 100 | 99,17% | 142,14 sec. | 6,57486e+08 € | 521812671.41 € |
| 7 | 449,438 | 150 | 99,46% | 401,62 sec. | 9,78441e+08 € | 783125174.503 € |
| 8 | 566,593 | 200 | 99,54% | 271,98 sec. | 1,26323e+09 € | 1131935104.19 € |
| 9 | 179,184 | 300 | 99,76% | 4254,84 sec. | 2,06832e+09 € | 1407190166.179 € |
| 10 | 214,233 | 300 | 99,84% | 7114,36 sec. | 2,11092e+09 € | 1246245515.556 € |
| 11 | 613,750 | 300 | 99,72% | 6090,89 sec. | 2,02134e+09 € | 1,67093e+09 € |

Table 12: CPLEX solution time with various $n$, $m$ and $g$ with the introduction of constraint 16

In general, the objective function values of Table 12 are higher than in Table 9, but the solution times are lower. It is important to notice that instance number 10 has finished the computation, leading to a lower objective function value than the one without model improvements. The solver is not yet able to finish instance number 11, leading to a 1.90% gap. The *Solutions* file is empty. However, the *Final obj. fun. value* is better than the one of Table 9. It is possible to think that this constraint makes the solver produce higher solution times, because we are imposing that every day, at least one thermal power plant must be switched on. However, by construction of the data-generation process it is not possible to satisfy the whole demand with just the renewable energy. Therefore, the cause of higher objective function values cannot be the additional constraint itself.

The following Table shows the result with the introduction of constraint 17.

| ***column index*** | ***n,m*** | ***g*** | ***Initial gap*** | ***Solution time*** | ***Initial obj. fun. value*** | ***Final obj. fun. value*** |
|---|---|---|---|---|---|---|
| 1 | 6,6 | 15 | 71,73% | 0,11 sec. | 8,48603e+07 € | 72674947.628 € |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 40,50 | 15 | 93,97% | 7,58 sec. | 9,24129e+07 € | 82874250.346 € |
| 3 | 130, 146 | 15 | 94,38% | 11,12 sec. | 9,53188e+07 € | 67665010.639 € |
| 4 | 295,228 | 30 | 97,32% | 31,16 sec. | 1,96761e+08 € | 152735859.614 € |
| 5 | 337,309 | 50 | 98,62% | 87,33 sec. | 3,16058e+08 € | 217721698.164 € |
| 6 | 304,325 | 100 | 99,17% | 171,06 sec. | 6,53834e+08 € | 521808368.488 € |
| 7 | 449,438 | 150 | 99,46% | 373,45 sec. | 9,76661e+08 € | 783116485.181 € |
| 8 | 566,593 | 200 | 99,54% | 247,12 sec. | 1,26216e+09 € | 1131934524.624 € |
| 9 | 179,184 | 300 | 99,76% | 4447,89 sec. | 2,06362e+09 € | 1407155332.081 € |
| 10 | 214,233 | 300 | 99,84% | 4066,17 sec. | 2,10682e+09 € | 1,24629e+09 € |
| 11 | 613,750 | 300 | 99,72% | 1883,91 sec. | 2,01842e+09 € | 1,68229e+09 € |

Table 13: CPLEX solution time with various $n$, $m$ and $g$ with the introduction of constraint 17

In general, the solution times of the model with the addition of constraint 17 are lower than the original ones, while the objective function values are unchanged. While in this case in configuration 10 CPLEX is able to decrease the gap to 0,01%, it is not able to do the same with configuration 11, where the gap is stuck at 2,56% and after only 1/8 of the original solution time, CPLEX gives an out-of-memory error.

Finally, the following table shows the solver's performances with just the introduction of constraint 18.

| ***column index*** | ***n,m*** | ***g*** | ***Initial gap*** | ***Solution time*** | ***Initial obj. fun. value*** | ***Final obj. fun. value*** |
|---|---|---|---|---|---|---|
| 1 | 6,6 | 15 | 43,35% | 0,06 sec. | 7,41952e+07 € | 72674947.628 € |
| 2 | 40,50 | 15 | 92,21% | 3,81 sec. | 8,41177e+07 € | 82873303.776 € |
| 3 | 130, 146 | 15 | 85,54% | 8 sec. | 7,05773e+07 € | 67665223.8 € |
| 4 | 295,228 | 30 | 93,41% | 23,59 sec. | 1,58049e+08 € | 152737742.231 € |
| 5 | 337,309 | 50 | 90,66% | 65,92 sec. | 2,29191e+08 € | 217728793.379 € |
| 6 | 304,325 | 100 | 96,26% | 162,98 sec. | 5,37074e+08 € | 521815559.285 € |
| 7 | 449,438 | 150 | 96,51% | 288,58 sec. | 8,05213e+08 € | 783123061.317 € |
| 8 | 566,593 | 200 | 99,17% | 748,41 sec. | 1,15336e+09 € | 1131931860.267 € |
| 9 | 179,184 | 300 | 93,95% | 4681,34 sec. | 1,48162e+09 € | 1407190166.179 € |
| 10 | 214,233 | 300 | 99,64% | 3581,47 sec. | 1,34273e+09 € | 1,24628e+09 € |
| 11 | 613,750 | 300 | 98,38% | 2223,98 sec. | 1,70319e+09 € | 1,67372e+09 € |

Table 14: CPLEX solution time with various $n$, $m$ and $g$ with the introduction of constraint 18

The above results show, in general, higher objective function values and lower solution times in respect of the ones of Table 9. With respect to the most expensive configurations, CPLEX runs out of memory in case of configuration number 10, but is able to lower the gap to 0,01%. With respect to configuration number 11, already after 1/4 of the original time CPLEX gives an out of memory error and the final objective function value (which is not the one in the solutions file, but the last best integer value) is higher than the original one.

Now the idea is to have an overview of the full situation, so as to decide which model improvements to add and move to the next section with the final model. The following Table shows the big picture of the various model changes results in terms of solution time and objective function value with respect to configurations 1 to 11. An up-oriented arrow indicates that the configuration has a higher solution time or objective function value with respect to the original one (see Table 9 and the other way round goes for a down-oriented arrow. The symbol "==" indicates that the value is equal to the original one. *Model change 1* refers to the removal of constraints 3 and 6 and change of the objective function. *Model change 2* refers to constraint 5 set as an equality. *Model change 3 A*, *Model change 3 B* and *Model change 3 C* refer respectively to the addition of constraint 16, 17 and 18.

| | | **Model change 1** | **Model change 2** | **Model change 3 A** | **Model change 3 B** | **Model change 3 C** |
|---|---|---|---|---|---|---|
| **1** | **Sol. time** | ↓ | ↓ | ↓ | ↓ | ↓ |
| | **Final obj. fun. value** | == | ↓ | == | == | == |
| **2** | **Sol. time** | ↑ | ↓ | ↓ | ↑ | ↓ |
| | **Final obj. fun. value** | ↓ | ↓ | ↓ | == | ↓ |
| **3** | **Sol. time** | ↓ | ↑ | ↓ | ↓ | ↓ |
| | **Final obj. fun. value** | ↑ | ↓ | ↑ | ↑ | ↑ |
| **4** | **Sol. time** | ↓ | ↓ | ↓ | ↓ | ↓ |
| | **Final obj. fun. value** | ↑ | ↓ | ↑ | == | ↑ |
| **5** | **Sol. time** | ↓ | ↓ | ↑ | ↑ | ↓ |
| | **Final obj. fun. value** | ↑ | ↓ | ↑ | == | ↑ |
| **6** | **Sol. time** | ↑ | ↑ | ↓ | ↓ | ↓ |
| | **Final obj. fun. value** | ↑ | ↓ | ↑ | == | ↑ |
| **7** | **Sol. time** | ↓ | ↓ | ↓ | ↓ | ↓ |
| | **Final obj. fun. value** | ↓ | ↓ | ↑ | == | ↑ |
| **8** | **Sol. time** | ↑ | ↑ | ↓ | ↓ | ↑ |
| | **Final obj. fun. value** | ↓ | ↓ | ↑ | == | ↓ |
| **9** | **Sol. time** | ↓ | ↑ | ↓ | ↑ | ↑ |
| | **Final obj. fun. value** | ↑ | ↓ | ↑ | == | ↑ |
| **10** | **Sol. time** | ↓ | ↑ | ↑ | ↓ | ↓ |
| | **Final obj. fun. value** | ↓ | ↓ | ↓ | == | ↓ |
| **11** | **Sol. time** | ↑ | == | ↓ | ↓ | ↓ |
| | **Final obj. fun. value** | == | == | ↓ | ↑ | ↑ |

Table 15: Overview of model changes results with respect to Table 9

As shown in Table 15, the best model's performances are given with the introduction of Model change number 2 and Model change 3 B. In fact, with model change number 2 almost all the objective function values decrease and with model change 3 B they are the same, but the solution times are lower. Therefore, it was tried to see how the solver performs with both the model changes, so constraint number 5 was set as an equality the additional constraint number 17 was added. The model's performances are shown in the following Table.

| *column index* | *n,m* | *g* | *Initial gap* | *Solution time* | *Initial obj. fun. value* | *Final obj. fun. value* |
|---|---|---|---|---|---|---|
| 1 | 6,6 | 15 | 42,04% | 0,12 sec. | 7,41952e+07 € | 72674947.6286 € |
| 2 | 40,50 | 15 | 93,38% | 5,47 sec. | 8,41177e+07 € | 82873303.776 € |
| 3 | 130, 146 | 15 | 92,41% | 12,38 sec. | 7,05773e+07 € | 67666378.036 € |
| 4 | 295,228 | 30 | 99,13% | 13,27 sec. | 1,66220e+08 € | 152736125.224 € |
| 5 | 337,309 | 50 | 99,46% | 80,56 sec. | 2,50842e+08 € | 217729459.128 € |
| 6 | 304,325 | 100 | 99,74% | 1117 sec. | 5,55426e+08 € | 5,21826e+08 € |
| 7 | 449,438 | 150 | 99,83% | 424,55 sec. | 8,32948e+08 € | 83117345.344 € |
| 8 | 566,593 | 200 | 99,88% | 1021,03 sec. | 1,15915e+09 € | 1131921109.100 € |
| 9 | 179,184 | 300 | 99,66% | 5108,81 sec. | 1,48162e+09 € | 1407182406.758 € |
|  | 214,233 | 300 | 99,64% | 4112,47 sec. | 1,34273e+09 € | 1,24639e+09 € |
| 11 | 613,750 | 300 | 97,36% | 2223,98 sec. | 1,70319e+09 € | 1,67482e+09 € |

Table 16: CPLEX's performance with the candidate final model

With respect to configuration 6, the solver is not able to finish and gives an out of memory error. The final gap is 0,01%. In general, the gaps are higher than the original ones. This means that the initial solution space is not as good as before. The performances are also worse in terms of objective function value with respect to the ones of Table 9. Therefore, the final model used for the rest of the analysis is the one with Model change number 2, that is to say with constraint 5 set as an equality one.

## 5.2 Solver's settings improvement

The idea of the present section is to check if further model improvements are possible in terms of solution time and/or objective function value. The first idea was to manipulate the CPLEX parameters *cplex_tuningmeasure* and *cplex_tuningrepeat.* This advanced tool prompts CPLEX to run tuning tests with different parameter settings automatically. By setting *cplex_tuningmeasure* to *average* the solver uses the mean average of time to compare different parameter sets over a suite of models, while setting the parameter to *minmax* lead the solver to use a minmax approach to compare the time of different parameter sets over a suite of models. Also, the *cplex_tuningrepeat* parameter was set equal to 3, so as to perform multiple runs. However, the model results were worse than the ones of Table 11.

Furthermore, several start algorithms parameters were tested. in particular, the *cplex_startalg* parameter can take the following values: Primal Simplex, Dual Simplex, Network Simplex, Barrier, Sifting, Concurrent (Dual, Barrier, and Primal in opportunistic mode; Dual and Barrier in deterministic mode). The idea was to check if the solver was able to start from a good initial solution space with a low initial gap. Still, the performances did not improve.

By examining the Engine log file, it was noticeable that, in particular, from instance number 5 onwards, the solver employs much time in the solving the problem, as if it was stuck on the best node without being able to furtherly improve the performance. In the IBM documentation there was an interesting source [5] about this trouble, which suggests to use a strong branching approach. As suggested by the source, the *VarSel* parameter was set equal to 3. Nonetheless, the performances did not improve significantly.

[5] https://www-eio.upc.es/lceio/manuals/cplex-11/html/usrcplex/solveMIP25.html656567

Moreover,it was decided to test the behaviour of the solver by manipulating the parameter *cplex*, which ontrols trade-offs between speed, feasibility, optimality, and moving bounds in MIP. All such values were tested, without particularly fevourable results.

The last attempt was to use aggressive settings for cut generation. In particular, the generation strategy of Clique cuts, Cover cuts, Mixed-Integer Rounding cuts and Disjunctive cuts was set to *very aggressive*. The results are the following.

| ***column index*** | ***n,m*** | ***g*** | ***Initial gap*** | ***Solution time*** | ***Initial obj. fun. value*** | ***Final obj. fun. value*** |
|---|---|---|---|---|---|---|
| 1 | 6,6 | 15 | 43,35% | 0,05 sec. | 7,07600e+07 € | 69239742.053 € |
| 2 | 40,50 | 15 | 93,35% | 10 sec. | 8,30973e+07 € | 81854111.006 € |
| 3 | 130, 146 | 15 | 92,07% | 15,33 sec. | 6,55700e+07 € | 62656498.678 € |
| 4 | 295,228 | 30 | 96,64% | 73,42 sec. | 1,52757e+08 € | 147443910.105 € |
| 5 | 337,309 | 50 | 98,12% | 147,55 sec. | 2,11780e+08 € | 200314641.21 € |
| 6 | 304,325 | 100 | 98,98% | 305,09 sec. | 5,22302e+08 € | 507036652.491 € |
| 7 | 449,438 | 150 | 99,34% | 651,70 sec. | 7,82255e+08 € | 760159675.881 € |
| 8 | 566,593 | 200 | 99,88% | 1359,97 sec. | 1,14519e+09 € | 1123765768.662 € |
| 9 | 179,184 | 300 | 99,66% | 10121,78 sec. | 1,39672e+09 € | 1322277994.387 € |
| 10 | 214,233 | 300 | 99,66% | 8709,39 sec. | 1,39672e+09 € | 1322282141.12 € |
| 11 | 613,750 | 300 | 99,91% | 3204,22 sec. | 1,67705e+09 € | 1613127705.060 € |

Table 17: CPLEX's performance with generation strategy of Clique cuts, Cover cuts, Mixed-Integer Rounding cuts and Disjunctive cuts was set to *very aggressive*

In the end, despite general higher solution times, the final objective function values are much lower than those of Table 11, with the exception of instances number 10 and 6. Another worth-to-notice particular is that in the very first working minutes the solver spends much time in the generation of disjunctive cuts. Disjunctive cuts are employed when dealing with sets of variables that are mutually exclusive, meaning they cannot simultaneously take on non-zero values. It is possible that they result particularly helpful in the present scenario where there are important constraints about the possibility to start producing energy only after one full operational day and the limit that a generator cannot be on/off for periods shorter than 2 days.

Furthermore, this time it was possible to terminate both instance number 10 and instance number 11. In fact, as suggested by some IBM documentation [6], another improvement was to set CPLEX's *workmem* parameter equal to 15000.0 instead of its default value 2048. These values are in MB and refer to the central memory available that the solver can use for working storage. The *cplex_memoryemphasis* parameter was also set to its non-default value, that is to say that memory reduction was enabled. This parameter directs CPLEX that it should conserve memory where possible. This time, no out-of-memory error occurred.

## 5.3 Conclusions

This study led to the construction of a model for analyzing the costs of energy production. The model works well, both in terms of computational time and likelihood: it is possible to increase the number of power plants and the number of days, making it applicable in different situations.

[6]https://www-eio.upc.es/lceio/manuals/cplex-11/html/usrcplex/solveLP16.html675271

The analysis carried out to try to optimize performance has led to considerable improvements, especially in the last subsection, where the results were further boosted by the customization of some CPLEX settings.

Thanks to the present project it was possible to concentrate on modeling skills so as to build a solid optimization model and focus on the data generation process, which helped in a more in-depth understanding of the model, since the generated data must be consistent with the proposed scenario. Finally, the model implementation through CPLEX allows to become handy with the use of the solver and the relative performance analysis was useful for checking the solver's bottleneck so as to implement the necessary improvements.

In the end, CPLEX turned out to be a very powerful tool for solving and ameliorating optimization problems.