This article can be cited as G. El Khattabi, I. Benelallam and E. H. Bouyakhf, Solving Smart Grid Local Problems Based on Constraint Optimization, International Journal of Artificial Intelligence, vol. 18, no. 2, pp. 1-23, 2020.

Solving Smart Grid Local Problems Based on Constraint Optimization

Ghizlane El Khattabi[1], Imade Benelallam[1,2] and El Houssine Bouyakhf[1]

[1]LIMIARF Laboratory
Faculty of sciences
Mohammed V University
Rabat, Morocco
elkhattabi.ghizlane@gmail.com, bouyakhf@mtds.com

[2]SI2M laboratory
INSEA
Mohammed V University
Rabat, Morocco
imade.benelallam@ieee.org

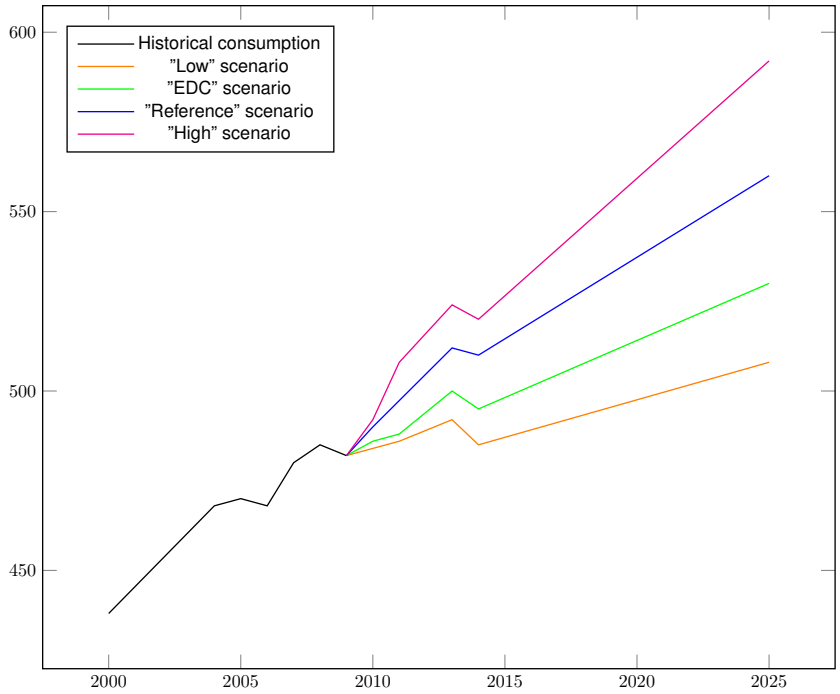
ABSTRACT

The appearance of several new electrical equipment types causes a strong demand for electricity. This brings cutouts causing very large losses. As well as the greenhouse gases emitted by the current plants can cause material and human damages. Smart Grids have emerged to meet these requirements. The various works, carried out up to now, architect the Smart Grids in three levels. i) The local level that can present a house, a building, a hospital or a factory ii) the microgrid level which serves as a bridge between the first and the third level and iii) the level of Transmission and Distribution (T&D) that serves to transmit and distribute the energy between the different locals. These works themselves have proposed fairly complex computer science methods that remain until now theoretical. Constraint Programming (CP) with its components is proved as a strong concept to represent mathematically several complex real problems. In this paper, we are going to use one of these components which is the Constraint Optimization Problem (COP), which represent each problem as variables, constraints and objective functions to optimize, in order to model and solve the local level and to optimize the energy distribution. Results exhibit the feasibility and strength of the proposed methods.

Keywords: Smart Grid, Electricity, Constraint Satisfaction Problem (CSP), Constraint Optimization Problem (COP), Artificial Intelligence.

2000 Mathematics Subject Classification: 97R40, 68N17, 46N10.

2012 ACM Computing Classification: Theory of computation $\rightarrow$ **Constraint and logic programming**; Theory of computation $\rightarrow$ Constructive mathematics; Theory of computatio $\rightarrow$ Mathematical optimization; **Computing methodologies** $\rightarrow$ **Planning and scheduling**; **Computing methodologies** $\rightarrow$ **Search methodologies**.



"Low" scenario: Assumptions reducing consumption
"High" scenario: Assumptions raising consumption
"EDC" scenario: Assumptions taking into account accelerating Energy Demand Control (EDC) through actions on consumer behavior.

Figure 1: Forecast of electricity consumption (source: Electricity Transmission Network in France)

1 Introduction

Power grids (Eyer and Corey, 2010) are infrastructures that route and distribute energy from suppliers (power plants) to consumers. The high electricity demand (Figure 1) will unable current electricity networks to meet all customer needs, because of the appearance of new electrical devices as electric cars and electric brushes. This will cause power cuts.
According to the US Department of Energy, the current cutouts cause an annual loss of $80 billion and the improvement of these networks could save between 46 and 117 billion dollars from 2010 to 2023, hence the apparition of Smart Grids.
Intelligent networks aspect was appeared by the automatic meter reading in 1980, the sending of statistics of daily consumption by meters in 1990 and the launching of Telegestore in 2000 (the first Smart Grid project linking 27 billion local).
The main objective of Smart Grids is to promote the use of renewable energies in order to reduce the overload made on power stations, to reduce the greenhouse gas emissions, to regulate the consumption, to satisfy consumers, balance supply and demand and to distribute energy without overcrowding. The use of renewable energy sources is not an obvious task, as these energies depend on several uncontrollable factors. For example, wind turbines depend on the wind and photovoltaics depend on the sun. In this case, the periods of the production of the energy does not necessarily correspond to the periods of the consumption peaks. Hence the necessity to store the energy to use in needs.

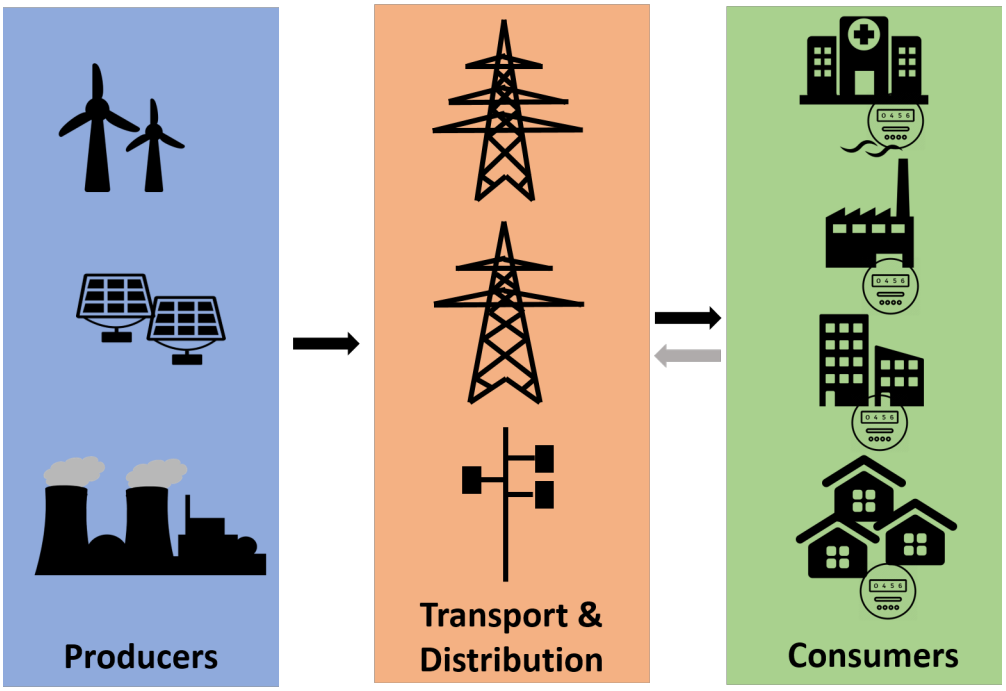


Figure 2: Main components of Smart Grids

A Smart Grid (Farhangi, 2009), (Gungor, Sahin, Kocak, Ergut, Buccella, Cecati and Hancke, 2011) is an electrical network which uses the Information and Communication Technology (ICT) tools to optimize the production, the consumption and the storage of energy as well as the losses. Smart Grid contains three main components. *i)* Consumers which regulate their own consumption by using smart meters (a new generation of meters), *ii)* producers which try to meet the needs of consumers in real-time, and *iii)* Entities which transport and distribute electricity between producers and consumers (Figure 2).

The Smart Grid problem is modeled in (Ahat, Amor, Bui, Bui, Guérard and Petermann, 2013) by three major levels: *i)* local level or consumer, *ii)* microgrid level and *iii)* Transmission and Distribution (T&D) level. The local level represents the consumer. It includes all entities controlled by a smart meter. We can mention houses, factories, hospitals, buildings, etc. The microgrid level represents the intermediate between consumers and producers. Its function is to spread the energy while equilibrating between the demand and the supply. The T&D level represents the two other components of Smart Grid (producers and entities of transport and distribution). The main function of T&D is to transmit the energy offered by producers to microgrids. In this paper, we are interested in modeling and solving the local level since it is the level that requires more intelligence.

Constraint programming (CP) (Apt, 2003), (Rossi, Van Beek and Walsh, 2006) is aptly applicable to this problem, especially Constraint Satisfaction Problems (CSPs) and Constraint Optimization Problems (COPs). A CSP (Kumar, 1992), (Debruyne and Bessiere, 1997), (Brailsford, Potts and Smith, 1999) is a formalism which can model a real problem as a set of integer, real, boolean, or symbolic

variables, taking their values in a set of **domains**. These variables are linked between them

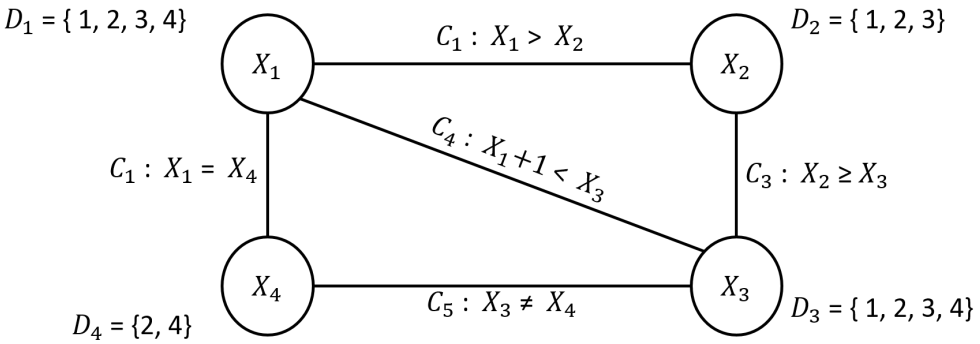


Figure 3: CSP example

by mathematical or conditional **constraints**. Figure 3 presents an example of a CSP problem. The refinement of a CSP solution is done via its modelization, not the algorithm, since once variables, domains, and constraints are identified, the resolution is done using one of the existing CSP algorithms, namely BackTracking (BT) (Kumar, 1992), Forward Checking (FC) (Bacchus and Grove, 1995), (Walsh, 2000), BackJumping (BJ) (Dechter and Frost, 2002) and Arc Consistency AC-1, AC-2, AC-3, AC-4, AC-2001 (Bessiere, 1991),(Mackworth, 1977),(Mohr and Henderson, 1986),(Bessiere, 1994),(Freuder, 1995) and Maintaining Arc Consistency (MAC) algorithm (Larrosa and Schiex, 2004).

There are many solvers implementing the last cited algorithms and allowing to solve CSP problems by just declaring their parameters (variables, domains, and constraints). among the existing solvers, we can point to SAT4J, Abscon, iZplus, Mistral, and Choco.

CSP has proven its capability to modelize several other real problems (Nadel, 1990) like Computer-Aided Design problems, scheduling and timetabling problems, optimization problems, and hardware configuration problems.

By definition, the Smart Grid Problem is an optimization problem. To this end, we are going to use the optimized version of Constraint Programming, which is the COP. A COP (Verfaillie, Lemaître and Schiex, 1996) is a CSP, plus one or more objective functions (other constraints) to optimize.

In ou last work (El Khattabi, Lahboub, El Houssine and Benelallam, 2018), we have modeled the Smart Grid local level using CSP,

by identifying the variables of the problem, the domain of each variable as well as the constraints in Choco solver. In this paper, we are going to improve the last work so as to solve the problem based on COP formalism, since the main problematic of Smart Grids consists of the optimization.

To this end, the paper is organized as follows: section 2 contains related work, section 3 defines and explains COP concept and presents, in details, our main contribution, section 4 shows the obtained experimental results, and section 5 presents the conclusion.

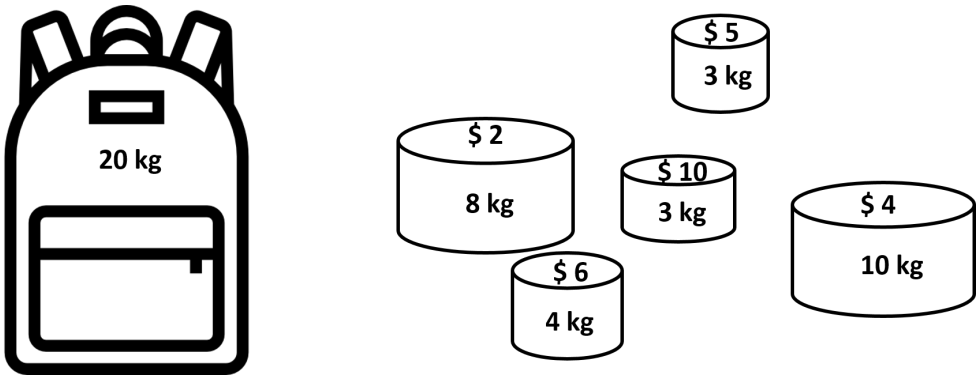


Figure 4: Knapsack problem example

2 Related Work

The proposition of computer science algorithms can somehow solve this problem. But the realization of a single algorithm to meet all the requirements of all levels is a very complex task or even impossible. Even in the case where researchers arrive to develop this kind of algorithm, it will make exponential complexity to meet all the needs of each consumer, at each local level and each microgrid. While the key identity of Smart Grids is the real-time response. For this reason, researchers interested in the development of Smart Grid algorithms develop an algorithm for each level (local, microgrid and T&D).

In (Marah and El Hibaoui, 2018), the authors proposed an algorithm for the local level and another for the T&D one. They modeled the local level as a "**Knapsack problem** (Ross and Tsang, 1989), (Chu and Beasley, 1998)" and proposed to solve it with a new version of the "**Branch and Bound**" algorithm(Kolesar, 1967), (Dyer, Kayal and Walker, 1984). While the algorithm used in the T&D level tries to transmit the energy in a distributed manner.

Figure 4 presents an example of Knapsack Problem. It contains a set of objects. Each object is identified with a value and a weight. The total weight exceeds the bag capacity. The Knapsack principle is to maximize the number of valued objects while not overtaking the bag capacity. If we project this problem on the Smart Grid, the bag models the local (house, building, ...) and the objects represent the local devices, the object value will be presented by a priority of the device and the object weight is the device consumption.

In the last cited work the possible priority values are "0" and "1". "0" to express that the device should be ON, and "1" to say that the device functioning is not so important.

In (Marah and El Hibaoui, 2018), the algorithm used to resolve the local level is a merge of the "Branch Bound" algorithm and their own algorithm, namely the "Priority Management Algorithm". The latter gives the priority to devices with "0" priority to use the energy and subtracts the sum of their energy consumptions from the total energy. The rest of the energy is given to the Branch and Bound algorithm as an input, to distribute it among the other devices, by maximizing the profit.

In (Ahat et al., 2013), authors treat Smart Grid as an Optimization Problem whose objective is to manage the consumer side, by regularizing the consumption curve and to synchronize data for effective results, by running each station and each piece of equipment in the same time.

To this end, they modeled the local problem as a Knapsack problem too. For the microgrid level, authors proposed to use "Game networks" (La Mura, 2000) which is a formalism in game theory (Osborne and Rubinstein, 1994), (Myerson, 2013) used to permit the players to play multiple games at the same time. For the T&D level, they proposed to use Max Flow (Shiloach and Vishkin, 1982) (Boykov and Kolmogorov, 2004) and Equilibrium algorithms (Florian and Hearn, 1995) to avoid routing and congestion problems. These algorithms are known in electric networks.
The modeling works done so far are all theoretical. In our last work (El Khattabi et al., 2018), we modeled, implemented and solved some locals' examples, which gave satisfactory results. We modeled the local level using CSP formalism while dividing the day into multiple slots (n slots means that the computation is done for every $\frac{24}{n}$ hours.), wherein the variables (Every variable represents a set of variables. One for each slot) are:

1. The local consumption of the period i:

   $$X_{c_i} \in [C_{min}, C_{max}]$$

   such that C_{min} (respectively C_{max}) is the minimum (respectively maximum) consumption of the local to treat;

2. The predicted amount of energy for each slot i:

   $$E_i \in \{E_i\}$$

3. The amount of energy we will recover from the power plant at the slot i:

   $$X_{Ep_i} \in [0, E_Pmax]$$

   such that E_Pmax is the maximum energy produced by electrical power plants;

4. The amount of energy we will recover from renewable energies' sources at the slot i:

   $$X_{Er_i} \in [0, Re_i]$$

   Re_i is the predicted amount of energy, offered by renewable energies;

5. The energy amount to store in the battery at the slot i:

   $$X_{Eb_i} \in [0, B]$$

   such that B is the storage capacity of the battery;

6. The functioning state of each device j at the slot i:

   $$X_{ON_{i,j}} \in \{0, 1\}$$

Devices that have only $\{1\}$ in the domain must always be ON, while devices with domain $\{0, 1\}$ may be running as they may not be;

7. The consumption of each device j:

$$C_j/j \in \{1, 2, \cdots, n\}$$

And the constraints are:

C1 The local consumption should be less than or equal to the expected consumption at any slot i

$$X_{c_i} \leq E_i/\forall i \in \{1, 2, \cdots, s\} \tag{2.1}$$

C2 The total consumption is equal to the sum of consumptions of running devices

$$X_{c_i} = \sum_{j=1}^{n} C_j \times X_{ON_{i,j}} \tag{2.2}$$

C3 The used energy (from power plant X_{Ep_i}, renewable energies' sources X_{Er_i} and battery X_{Eb_i}) should be greater than or equal to the total consumption X_{c_i}

$$X_{Ep_i} + X_{Er_i} + X_{Eb_i} \geq X_{c_i}/\forall i \in \{1, 2, ..., s\} \tag{2.3}$$

C4 The energy offered by the power plant should be equal to '0' when the energy produced by renewable energies' sources is sufficient

$$X_{Er_i} \geq X_{c_i} \Rightarrow X_{Ep_i} = 0 \tag{2.4}$$

C5 If the energy produced by sources of renewable energies is insufficient (even after adding the energy stored in the battery) it will be stored in the battery.

$$X_{Er_i} < X_{c_i} \vee X_{Er_i} + X_{Eb_i} < X_{c_i} \Rightarrow X_{Eb_{i+1}} = X_{Eb_i} + X_{Er_i} \tag{2.5}$$

C6 Otherwise (ie. the sum of the energy produced by sources of renewable energies and that stored in the battery is sufficient), The battery will contain the rest.

$$X_{Er_i} + X_{Eb_i} \geq X_{c_i} \Rightarrow X_{Eb_{i+1}} = X_{Eb_i} + X_{Er_i} - X_{c_i} \tag{2.6}$$

This modeling has been declared using the Choco solver which uses a smart combination of CSP algorithms, according to the variables' types and the constraints to solve this problem as fast and satisfactory as possible.
To have a look of how algorithms solve CSP problems, let take the most used CSP algorithm which is AC-3 (Algorithm 1). This algorithm makes the difference between unary and binary constraint. The unary constraint is a constraint on a single variable ($x_1 \leq 2$). Whereas binary one is a constraint on two variables ($x_1 = x_2$). This algorithm brows the set of variables X and

Algorithm 1 AC-3 algorithm' pseudo-code

```
1: function AC-3
2:     Given a constraint network CN = (X, D, C_u, C_b)        ▷ X: set of variables
                                                               ▷ D: set of domains,
                                                               ▷ C_u: set of unary constraints
                                                               ▷ C_b: set of binary constraints
3:     for each x ∈ X do
4:         for each v ∈ D_x do
5:             if x = v doesn't satisfy one constraint among C_u then
6:                 D_x = D_x − v
7:                 if D_x = ∅ then return failure
8:                 end if
9:             end if
10:        end for
11:    end for
12:    Q ← {(x_i, x_j)|(x_i, x_j) ∈ X}        ▷ List initialized by 2 variables constrained by C_b
13:    while Q ≠ ∅ do
14:        Q ← Q − {x_i, x_j}
15:        if Revise(x_i, x_j) then
16:            if D_{x_i} = ∅ then return failure
17:            else
18:                Q ← Q ∪ {(x_k, x_i)|(x_k, x_i) ∈ X, k ≠ j}
19:            end if
20:        end if
21:    end while
22: end function

23: function REVISE((x_i, x_j))
24:    REVISE ← false
25:    for each value v_i ∈ D_{x_i} do
26:        if ∄ v_j ∈ Dx_j, such compatible(v_i, v_j) then
27:            D_{x_i} = D_{x_i} − a
28:            REVISE ← true
29:        end if
30:    end for
        return REVISE
31: end function
```

checks for each variable x if there are values in its domain that do not satisfy the set of unary constraints C_u. if so, it deletes them from the x domain. For binary constraints, AC-3 initializes a list with a couple of variables (x_i, x_j) linked by a binary constraint in C_b and browses the domains of the two variables. For each value v_j from x_j domain, it deletes all values, from x_i domain, that are inconsistent with the instantiation $(x_j = v_j)$.
The algorithm ends in two cases. The first case when it filters all couples of variables without emptying any domain, in this instance, the problem is solvable and we can use, for example, the backtracking basic algorithm to find an instantiation as soon as possible. In the second case, when it finds an empty domain. In that case, the problem is insolvable.
This algorithm as well as the other CSP algorithms terminate since the number of variables, domains' values and constraints are finite.
In case the problem consists of finding a solution that optimizes a set of functions. The algorithm looks for all the solutions that satisfy the constraints and keep the ones that optimize the functions. The choice of the optimized solution/solutions is done either with the algorithm itself or with one of the metaheuristic methods mentioned in Related Work section.
Smart Grid is, by definition, an optimization problem. This is why we are going to model the local level using the COP formalism. But before, we are going to import improvements to our latest CSP modeling.

3 Solving Smart Grid Local Problems based on COP Formalism

The optimization problem attracts a big emphasis on different domains and applications such as the fuzzy cognitive maps (Vaščák, 2012), the optimization of faults after their detection and insulation (Precup, Angelov, Costa and Sayed-Mouchaweh, 2015), Ideal gas (Shams, Rashedi, Dashti and Hakimi, 2017), traffic lights (Gil, Johanyák and Kovács, 2018), fuzzy controlled servo systems (Precup and David, 2019), and Constraint Programming (Hentenryck and Michel, 2009; Hooker, 2002; Crown, Buyukkaramikli, Thokala, Morton, Sir, Marshall, Tosh, Padula, Ijzerman, Wong et al., 2017).
The Smart Grid Problem is an optimization problem, especially its local level. Many parameters have to be optimized such as the maximization of the number of functional devices in a given local and the favor of devices with higher priority, as well as the minimization of the lag between consumption and forecasting.
Since we have already proposed a CSP modelization of this problem, we are going to extend the modelization to the optimized version of Constraint Programming which is COP.
A COP(Verfaillie et al., 1996) is a CSP plus one or many objective functions to optimize, namely to minimize or to maximize. In a formal way, a COP is defined by the quadruple (X, D, C, O) such as:

- $X = \{X_1, X_2, ..., X_n\}$: the set of variables
- $D = \{D_1, D_2, ..., D_n\}$: the set of domains. Each variable X_i takes its values in the domain D_i.
- $C = \{C_1, C_2, ..., C_m\}$: the set of constraints

- $O = \{O_1, O_2, ..., O_p\}$ $/O_i : D_1 \times D_2 \times ... \times D_n \rightarrow \mathbb{R}, \forall i \in \{1, 2, ..., n\}$: the set of objective functions. Each objective function o_i is a function to maximize or to minimize.

All CSP algorithms can solve COP problems , by looking for all CSP solutions and keep one that optimize the objective functions. In addition, there are other metaheuristic methods like Genetic Algorithms (GAs) (Homaifar, Qi and Lai, 1994), Simulated annealing (Fleischer, 1995), Tabu search (Glover and Laguna, 1998) and SNOPT (Gill, Murray and Saunders, 2005). The difference between CSP algorithms and metaheuristic methods is that the optimum found, using the metaheuristics, is not always global because they do not browse all solutions, contrary to CSP algorithms.
As mentioned in section 2, we have already proposed a CSP modeling of the Smart Grid local level (El Khattabi et al., 2018), by defining the variables $X = \{X_{C_i}, E_i, X_{E_{p_i}}, X_{E_{r_i}}, X_{E_{b_i}}, X_{ON_{i,j}}, C_j, \}$, the domains as well as the constraints $C = \{C_1, C_2, C_3, C_4, C_5, C_6\}$.
In this paper, we are going to *i)* improve the latter work and *ii)* extend it to a COP problem.
For the improvements, we have made the following changes:

- The variables $X_{ON_{i,j}}, \forall i, j$ that represent the devices' priorities. They could take either "0" or "1" as in all of the above works. They presented, as a matter of fact, the operating state of the devices.

  In this paper, we keep this variable. Whereas now, it represents the state of the device (it always has $\{0, 1\}$ as domain);

- We have added other data that represents the real priority of each device ($\{0,1,2,3,...\}$). To express the need to operate one device over another. This is not a variable, but rather a datum since it is frozen by the user at the start;

- The variables $E_i, s\forall i$ which represented the forecasts of the total consumption, the energy offered by the power plant and also that offered by renewable energies' sources are no longer variables but datums since the statistics are done beforehand;

- We added B the battery capacity as data;

- We considered the battery capacity within the constraints C_5 and C_6;

- we added a constraint to take into consideration the priority of devices. A lower priority device can function just after functioning a higher priority one, or in other cases. We will discuss all data, variables, constraints and objective functions in details, in subsections 3.1 and 3.2.

As mentioned in section 3, COP is a CSP + one or more objective functions to optimize. So, it consists of the quadruplet (Variables, Domains, Constraints, Objective functions). In order to complete COP modelization, we added four objective functions to:

- promote the operation of devices with a higher priority;

- maximize the number of functional devices;

- minimize the lag between consumption and forecasting;

In addition to the four components of COP (Variables, domains, constraints, and objective functions), we have added a fifth component for the data offered by users that is necessary for our proposed modeling.

3.1 Data

Any Smart Grid local problem is compiled by a set of data:

- The number of devices

$$n$$

- The number of slots, of which we want to plan consumption

$$s$$

- The predicted amount of energy for each slot i

$$E_i/i \in \{1, 2, \cdots, s\}$$

- The predicted amount of energy, offered by renewable energies' sources for each slot i

$$Re_i/i \in \{1, 2, \cdots, s\}$$

- The consumption of each device j

$$C_j/j \in \{1, 2, \cdots, n\}$$

- The priority of each device j

$$P_j/j \in \{1, 2, \cdots, n\}$$

- The maximum energy the power plant can produce

$$E_{Pmax}$$

- The battery capacity

$$B$$

3.2 Model

Variables & Domains

The set of variables is the same as that of the previous contribution minus E_i, the predicted amount of energy for the slot i, and C_j, the consumption of the device j. These two last variables are treated as data in this contribution.

$$X = \{X_{c_i}, X_{E_{p_i}}, X_{E_{r_i}}, X_{E_{b_i}}, X_{ON}\}$$

,
The differences also arise in the domains of the variables X_{C_i} and X_{ON}.

- The local consumption of the period i can take values from 0 to the predicted energy amount

$$X_{c_i} \in [0, E_i]$$

- The functioning state of each device j at the slot i can take either 0 as value or 1:

$$X_{ON_{i,j}} \in \{0, 1\}$$

Constraints

The set of constraints is

$$C = \{C_1, C_2, C_3, C_4, C_5, C_6, C_7\}$$

such as C_1, C_1, C_2, C_4, C_4 are still the same as the last works. While C_5 and C_6 are changed to consider the battery capacity. Whereas C_7 is a new constraint added in this contribution to treat the priorities of the devices:

C5 Before storing the energy produced by sources of renewable, the battery capacity should be checked. If it is exceeded, we keep the maximum energy which is the battery capacity itself.

$$X_{Er_i} < X_{c_i} \vee X_{Er_i} + X_{Eb_i} < X_{c_i} \Rightarrow \begin{cases} X_{Eb_{i+1}} = X_{Eb_i} + X_{Er_i}, & \text{if } X_{Eb_i} + X_{Er_i} \leq B \\ X_{Eb_{i+1}} = B, & \text{else} \end{cases} \quad (3.1)$$

C6 The same treatment is done against the new quantity to store in the battery.

$$X_{Er_i} + X_{Eb_i} \geq X_{c_i} \Rightarrow \begin{cases} X_{Eb_{i+1}} = X_{Eb_i} + X_{Er_i} - X_{c_i}, & \text{if } X_{Eb_i} + X_{Er_i} - X_{c_i} \leq B \\ X_{Eb_{i+1}} = B, & \text{else} \end{cases} \quad (3.2)$$

C7 If the sum of the consumptions of devices having the highest priority (”1”), does not exceed the forecasts, then all these devices will be functional. For the other priorities, the same test will be done but by comparing the consumptions of devices with the remaining energy, after subtracting the energy used to operate the previous devices from the predicted energy.

Device	d_1	d_2	d_3	d_4	d_5	
Priority	1	2	1	3	2	O_3
X_{ON}	**1**	**1**	**1**	**0**	**0**	**4**
X_{ON}	1	1	0	1	0	6
X_{ON}	1	1	0	0	1	5
X_{ON}	1	0	1	1	0	5
X_{ON}	**1**	**0**	**1**	**0**	**1**	**4**
X_{ON}	1	0	0	1	1	6

Table 1: O_3 values in different scenarios

$$\begin{cases} \begin{cases} \sum_j C_j \leq E_i/ \\ C_j \; is \; the \; consumption \; of \; devices \; with \; priority \; 1 \end{cases} \\ \textbf{or} \\ \begin{cases} \forall p \in \{2, \cdots, p_{max}\}, \forall i \in \{1, 2, \cdots, s\} \\ \sum_j C_j \leq E_i - \sum_l C_l/ \\ C_j \; is \; the \; consumption \; of \; devices \; with \; priority \; p \\ C_l \; is \; the \; consumption \; of \; devices \; with \; priority \; p-1 \end{cases} \end{cases} \rightarrow \begin{cases} \forall k \in \{1, \cdots, n\}/P_k = p, \\ X_{ON_k} = 1 \end{cases} \tag{3.3}$$

Objective Functions

O1 Maximize the number of devices running, at each slot j

$$Min(\sum_{j=1}^{s} m - \sum_{i=1}^{n} X_{ON_{i,j}}) \tag{3.4}$$

O2 Minimize losses by minimizing the lag between total consumption and forecast, at each slot i

$$Min(\sum_{i=1}^{s} E_i - X_{c_i}) \tag{3.5}$$

O3 The constraint C7 is used to distribute the energy in priority order. In the case where the remaining energy suffices all the devices of the same priority, it is distributed between them. But in the case where the remaining energy is no longer sufficient, there is no constraint to control its distribution. The purpose of this objective function is therefore to distribute this energy by maximizing the number of functional devices with highest priority.

$$Min(\sum_{j=1}^{s} \sum_{i=1}^{n} P_i \times X_{ON_{i,j}}) \tag{3.6}$$

The more the priority p is high, the higher the term $P_i \times \sum_{i=1}^{n} X_{ON_{i,j}}$ is (p is highest $\rightarrow$ devices with p are the lowest priority ones). So, to give the hand to the higher priority devices before the lower priority ones, we will try to minimize this term. In the example

Slots	1	2	3	4	5	6	7	8	9	10
E	4561	4424	3961	3958	4139	4445	4879	5216	5459	5543
Re	2000	4500	4000	3400	3500	3600	4000	4100	4200	4400

Table 2: Predictions

shown in table 1, we suppose to have 5 devices with different priorities 1, 2 or 3. The example shows six energy distribution scenarios and calculates the objective function O_3, for each scenario. It shows that the two scenarios where O_3 is minimal are the best.

O4 In some examples, where the number of higher priority devices is very high compared to the number of lower priority devices, the objective function O_3 is insufficient. To this end, we have to increase the coefficient associated with the priority ($10^{priority}$) and link it with the number of functional devices with priority $priority$ compared to the number of all functioning ones.

$$Min(\sum_{j=1}^{s} \frac{\sum_{i=1}^{n} P_i \times 10^i \times X_{ON_{i,j}}}{\sum_{k=1}^{n} X_{ON_{k,j}}}) \tag{3.7}$$

In order to prove the efficiency of our modelization, we have to demonstrate that either theoretically or experimentally. Since the problem is real, we have to experiment our proposition in different realistic instantiations, in a real local, with a given number of slots, a given number of devices and some realistic predicted energy values.

4 Experimental Results

To evaluate the performance of our new contribution and study its complexity, we implemented it using Choco 4 solver (Narendra, Rochart and Lorca, 2008), (Prud'homme, Fages and Lorca, 2014) and applied it on local problems characterized by $< n, s, E, Re, C, P, B >$.

- n is the number of devices, which takes values from $\{2, 4, 6, 8, 10, 12\}$;
- s is the number of slots. It varies from 1 to 10 by 1 as step;
- E is the set of predicted consumed energy values of each slot (Table 2);
- Re is the set of values of predicted energy offered by renewable energies' sources of each slot (Table 2);
- C is the set of consumptions of devices;
- P is the set of priorities of devices. Table 3 shows the names of the devices that can be integrated into the local, the capacity of each device and its priority.
- B is the capacity of the battery. it is equal to 4000

Devices	TV	Fr	L	VC	HD	EO	ER	WM	D	REF	PC	H
C	190	3900	200	600	100	2050	160	1200	50	175	160	1400
P	1	1	2	2	3	1	3	2	1	2	2	1

TV: **T**elevision; **HD**: **H**air **D**ryer; **D**: **D**ishwasher;
Fr: **Fr**eezer; **EO**: **E**lectric **O**ven; **REF**: **R**efrigerator;
L: **L**amp; **ER**: **E**lectric **R**azor; **PC**: **P**ortable **C**omputer;
VC: **V**acuum **C**leaner (VC); **WM**: **W**ashing **M**achine ; **H**: **H**eating.

Table 3: Consumptions and priorities of used devices

As we have defined in subsection 3.2, Smart Grid local problems are optimization problems with 4 objective functions to minimize.

So, in addition to the quality of the returned solutions, the performance of the contribution will also be measured by the objective functions' values, so as to confirm, they are really minimal or not. But first, let study the complexity of each objective function separately.

Figure 5 shows the variation of the four objective functions O_1, O_2, O_3 and O_4 depending on the number of slots s and the number of devices n established in the local. The problems given to Choco can give several optimized solutions. In that case, the taken value is the average ($\frac{O_i}{The\ number\ of\ solutions}$, i = 1, 2, 3 or 4). In the case of $n > 1$, we take also the average of each objective function values ($\frac{O_i}{n}$).

The figure shows that the first objective function O_1 does not depend on the number of slots. It is still constant when the number of devices is fixed. Which is normal, because the objective of O_1 is to maximize the number of the running devices (minimizing the recall between the existing devices and the functioning ones). The variation of O_1 according to the number of devices can be considered linear except when n is equal to 6 because the six first devices include three devices having the higher priority, but cannot be served all in the same time, because the sum of their consumptions surpasses the provided value. So only two devices will have the priority to function, and the remaining values will be distributed among the lowest priority devices while the remaining energy amount is low.

The variation of O_2 is linear too according to the number of devices and the number of slots. When the devices are more numerous, O_2 becomes minimal. Which is normal, because O_2 aims to minimize the lag between the local consumption and the broadcast. When we have a bit number of devices, even we make all devices ON, we still far from the broadcast. The last point is a highlight of O_2.

For O_3, it is linear too, according to the two parameters. It becomes more important when the number of slots and the number of devices increase. This is very normal according to its expression. It is an increasing function, summing the priorities of devices which are active. So the more we have active devices, the more their priorities are considered. The values of objective functions will indeed be divided by the number of slots to recover their averages. So the fact that the O_3 expression depends on the number of slots does not explain why it is growing when the number of slots is larger. The real reason is to have more energy stored in the battery and so the possibility of operating more devices. The same for O_4 which can be an

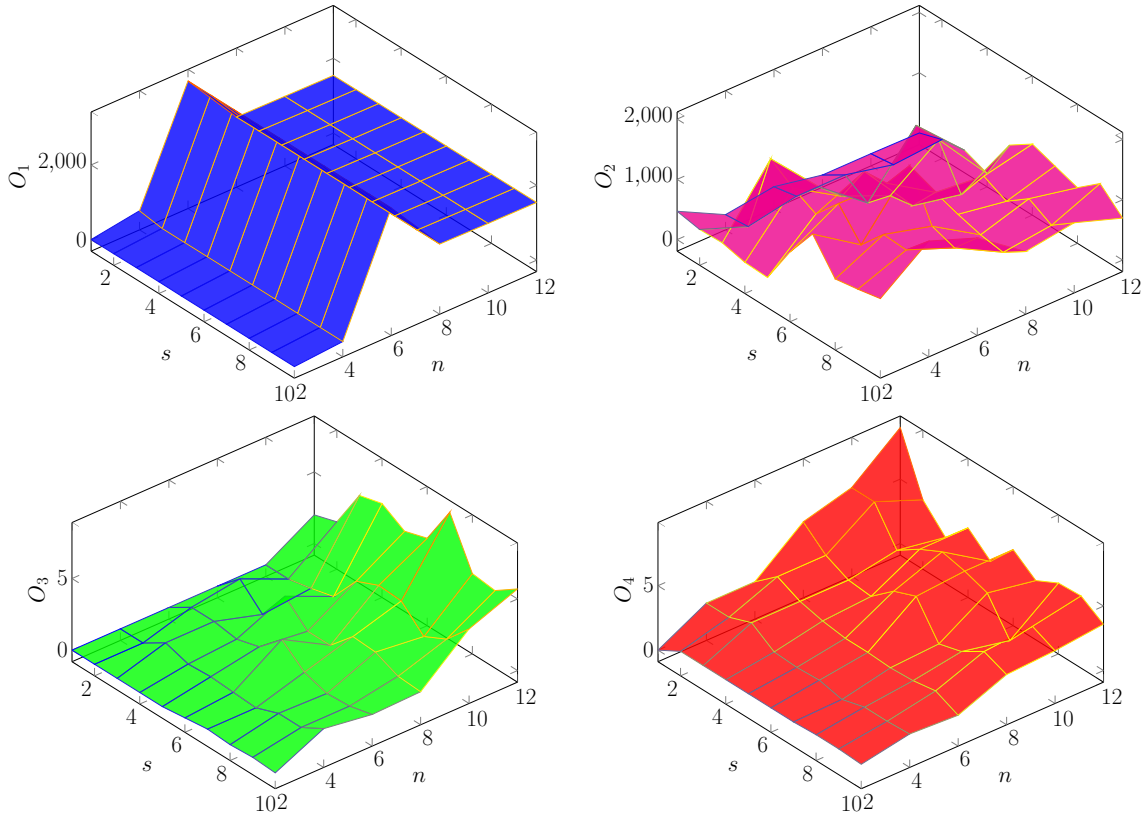


Figure 5: Objective functions

additional function of O_3. Except that it depends inversely on the number of active devices. So, contrary to O_3, which is increasing, O_4 is decreasing against the number of devices.

After evaluating each objective function separately, figures 6, 7 and 8 show how much their optimizations make the difference. According to figure 5, the behavior of a local depends on the devices it contains. So, we chose 3 values of n to scan all types of problems. A small value n = 1 (Figure 6), a medium value n = 5 (Figure 7) and another large n = 10 (Figure 8).

We can check that a function is minimal after optimization, only when we compare it with its values when the optimization is not carried out. Such a COP problem returns multiple non-optimal solutions. So, to compare the value of the objective function of the optimized solutions with the others which are not, we have to select some general cases (i.e. 4 or 5 cases). For this purpose, we select the cases when we optimize the other objective functions others than the one to assess. Assuming that there will be no values greater than those returned by these cases, where the resolution process makes more effort, to optimize the other objective functions.

To this end, we are going to evaluate each objective function in 5 cases. The first case concerns the optimization of the objective function itself, the second, the third and the fourth case relate to the optimization of the three other functions and the fifth case refers to the optimization of the four objective functions concurrently, based on the principle that the four cases present four possible behaviors of the function without optimization (minimization).

For the first objective function O_1, it has the same behavior in the three cases (n=1, n=5 and n=10). Optimizing this function affects the solution. It is better than all values of other solutions'

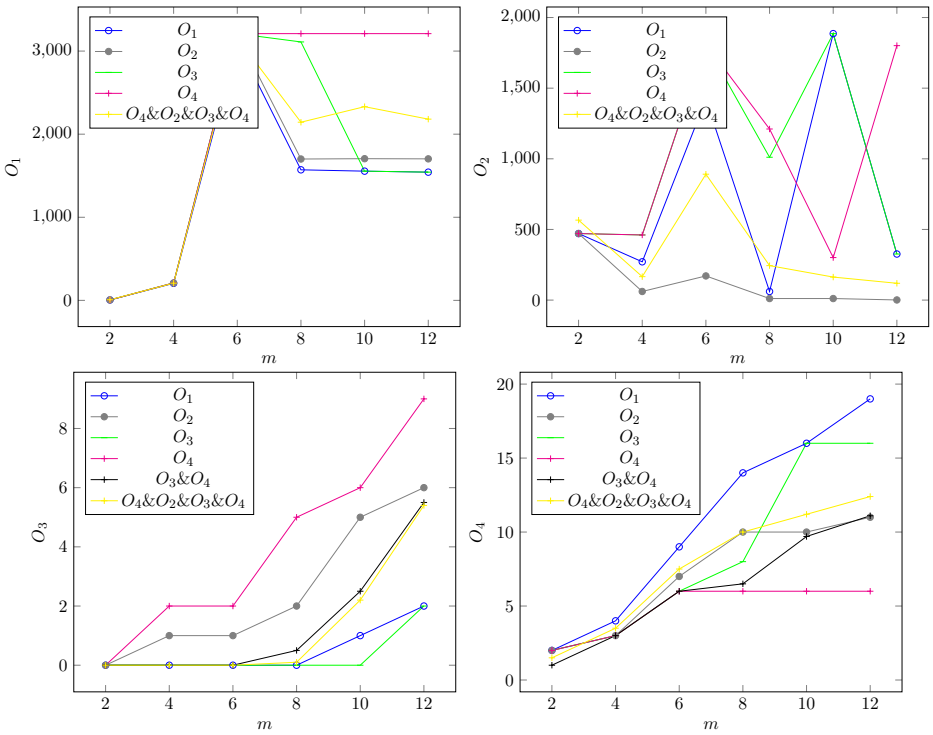


Figure 6: The influence of objective functions' optimization when n = 1

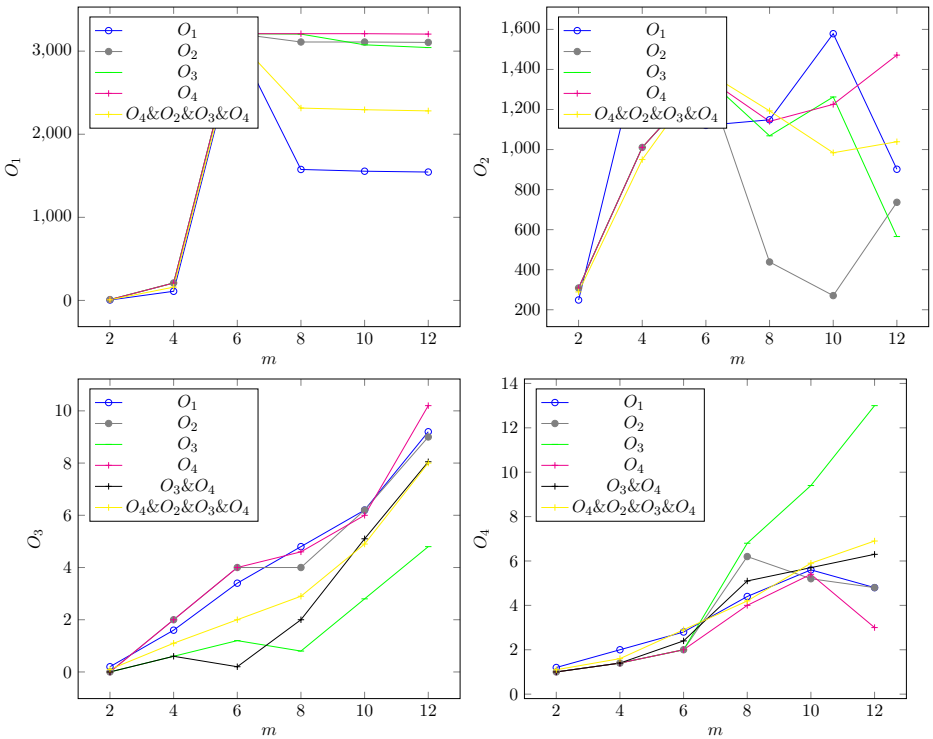


Figure 7: The influence of objective functions' optimization when n = 5

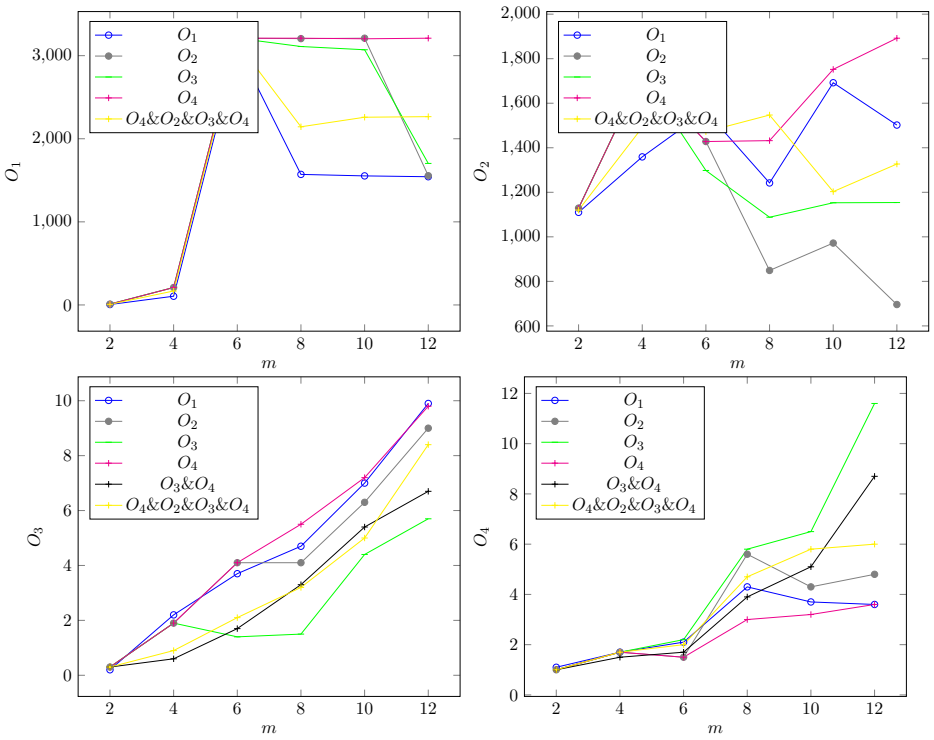


Figure 8: The influence of objective functions' optimization when n = 10

O_1, especially when the number of devices is large (the difference is very important).

Unlike O_1, O_2 has not the same behavior when the number of slots is changed. For $n = 1$, the optimization of O_2 makes a very important difference. It is very better than other cases. It can be considered stable according to the number of devices, contrary to other instances. For $n = 5$ and $n = 10$ it behaves in the same manner. It is also the best but the optimization impact can be seen when the number of devices is more important.

For O_3 and O_4, They behave, for the three values of n, in the same way. The optimization becomes more and more important whenever the number of devices increases.

For the three values of n, we tried to see the impact of optimizing all objective functions concurrently on O_1, O_2, O_3 and O_4. The figures show that the obtained results are so satisfying. They are all near to the objective function optimization itself.

To give the obtained results a life, we taken one of the processed problems randomly and shown the values taken by the defined variables.

Tables 4 and 5 show the variables' values without (Table 4) and with (Table 5) optimizing all function objectives concurrently. The two tables show that all defined constraints are satisfied. The local consumption never exceeds forecast and it is equal to the sum of consumptions of running devices, the use of energy produced by renewable energy sources is the most favored and the unused energy is stored in the battery.

i	X_{c_i}	E_i	Production			Devices											Objective functions			
			X_{Ep_i}	X_{Er_i}	X_{Eb_i}	TV	Fr	L	VC	HD	EO	ER	WM	D	REF	PC	H	O_1	O_2	O_3	O_4
1	2000	4561	0	2000	0				x								x	10	2561	3	105
2	3900	4424	0	4500	600		x											11	524	1	10
3	600	3961	0	4000	4000				x									11	3361	2	200
4	3450	3958	0	3400	50			x			x		x					9	508	5	136.7
5	0	4139	0	3500	3500													12	4139	0	-
6	3600	4445	0	3600	3550	x					x		x			x		8	845	6	105
7	4090	4879	0	4000	90	x	x											10	489	2	10
8	600	5216	0	4100	3590				x									11	4616	2	200
9	4225	5459	0	4200	25			x	x		x		x		x			7	1234	10	162
10	5200	5543	0	4400	800		x			x			x					9	343	6	1070
																		9.8	1962	3.7	222

Table 4: One found solution using our previous work (El Khattabi et al., 2018)

i	X_{c_i}	E_i	Production			Devices											Objective functions			
			X_{Ep_i}	X_{Er_i}	X_{Eb_i}	TV	Fr	L	VC	HD	EO	ER	WM	D	REF	PC	H	O_1	O_2	O_3	O_4
1	4275	4561	2275	2000	0		x	x							x			**9**	**286**	5	136.7
2	4140	4424	0	4500	360	x	x							x				**9**	**284**	3	**10**
3	1840	3961	0	4000	2520	x		x						x			x	**8**	**2121**	5	**73.3**
4	3785	3958	0	3400	2135						x				x	x	x	**8**	**173**	6	**105**
5	4110	4139	0	3500	1525		x					x		x				**9**	**29**	5	**1006**
6	3950	4445	0	3600	1175				x	x	x		x					**8**	**495**	8	802.5
7	4595	4879	595	4000	0		x	x				x			x	x		**7**	**284**	9	722
8	4850	5216	0	4100	750		x	x	x	x				x				**7**	**336**	9	684
9	2335	5459	0	4200	2615				x			x			x		x	8	3124	**8**	855
10	5425	5543	1025	4400	0	x		x			x	x	x	x	x		x	**4**	**118**	13	455
																		7.7	728	7.1	484.9

Table 5: One found solution using our current work

In table 4, the solution does not take devices' priorities into consideration. However, table 5 shows that devices with higher priority run before the lowest ones (when the produced energy is sufficient).
We must recall that O_1 aims to maximize the number of running devices compared to the existing ones. So, the values shown in the table represent, more precisely, the number of nonfunctional devices. It is noticed that is almost always minimal when O_1 is optimized and, admittedly, the number of devices in the second case is great than the first one (table 4).
For O_2, which aims to minimize the recall between the consumed energy and the forecasted one, it is also almost minimal when it is optimized.
Unlike O_1 and O_2, the values O_3 and O_4 are not minimal, even with optimization. This is due to the number of functional devices. When O_1 is minimal, the number of running devices is large. So O_3, which compute the priorities of running devices, will be large too. In table 4 , O_3 is better, because the number of devices is low. This can be proved by the solution of the ninth slot (s = 9). In this instance, O_1 is not minimal but O_3 is.
For O_4 which will be considered as an O_3 complement. Effectively, we notice that half of the values are low in table 5 compared to table 4, and that is outstanding by the good distribution of energy among devices, which takes into consideration the priority of devices.

5 Conclusion

In this paper, we have treated the Smart Grid local problem, as a Constraint Optimization Problem. It is a decisive problem which is the main component of Smart Grid. The quantity of parameters to be considered makes the problem purpose very complex. The existing resolution methods are NP-hard and remain only theoretical. In our contribution, we have modeled and implemented the proposed model to validate it. The found results prove the feasibility, the intelligence and the strength of our approach. As perspectives, we plan to solve the whole Smart Grid problem, using Distributed Constraint Satisfaction Problem (DisCSP) (Yokoo, 2001) and Distributed Constraint Optimization Problem (DCOP), since the problem can be summed as a set of local problems.

References

Ahat, M., Amor, S. B., Bui, M., Bui, A., Guérard, G. and Petermann, C. 2013. Smart grid and optimization, *American Journal of Operations Research* **3**(1A): 196–206.

Apt, K. 2003. *Principles of constraint programming*, Cambridge university press.

Bacchus, F. and Grove, A. 1995. On the forward checking algorithm, *International Conference on Principles and Practice of Constraint Programming, Cassis, France*, Springer, pp. 292–309.

Bessiere, C. 1991. Arc-consistency in dynamic constraint satisfaction problems., *AAAI, Anaheim, California*, Vol. 91, pp. 221–226.

Bessiere, C. 1994. Arc-consistency and arc-consistency again, *Artificial intelligence* **65**(1): 179–190.

Boykov, Y. and Kolmogorov, V. 2004. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision, *IEEE Transactions on Pattern Analysis & Machine Intelligence* (9): 1124–1137.

Brailsford, S. C., Potts, C. N. and Smith, B. M. 1999. Constraint satisfaction problems: Algorithms and applications, *European journal of operational research* **119**(3): 557–581.

Chu, P. C. and Beasley, J. E. 1998. A genetic algorithm for the multidimensional knapsack problem, *Journal of heuristics* **4**(1): 63–86.

Crown, W., Buyukkaramikli, N., Thokala, P., Morton, A., Sir, M. Y., Marshall, D. A., Tosh, J., Padula, W. V., Ijzerman, M. J., Wong, P. K. et al. 2017. Constrained optimization methods in health services research—an introduction: report 1 of the ispor optimization methods emerging good practices task force, *Value in health* **20**(3): 310–319.

Debruyne, R. and Bessiere, C. 1997. Some practicable filtering techniques for the constraint satisfaction problem, *In Proceedings of IJCAI'97, Nagoya, Aichi, Japan*, Citeseer, pp. 412–417.

Dechter, R. and Frost, D. 2002. Backjump-based backtracking for constraint satisfaction problems, *Artificial Intelligence* **136**(2): 147–188.

Dyer, M., Kayal, N. and Walker, J. 1984. A branch and bound algorithm for solving the multiple-choice knapsack problem, *Journal of computational and applied mathematics* **11**(2): 231–249.

El Khattabi, G., Lahboub, S., El Houssine, B. and Benelallam, I. 2018. Contribution to modeling smart grid problem with the constraint satisfaction problem formalism, *Proceedings of the 2nd Mediterranean Conference on Pattern Recognition and Artificial Intelligence (MedPRAI), Rabat, Morocco*, ACM, pp. 58–62.

Eyer, J. and Corey, G. 2010. Energy storage for the electricity grid: Benefits and market potential assessment guide, *Sandia National Laboratories* **20**(10): 5.

Farhangi, H. 2009. The path of the smart grid, *IEEE power and energy magazine* **8**(1): 18–28.

Fleischer, M. 1995. Simulated annealing: past, present, and future, *Winter Simulation Conference Proceedings, Hyatt Regency Crystal City, Arlington, VA*, IEEE, pp. 155–161.

Florian, M. and Hearn, D. 1995. Network equilibrium models and algorithms, *Handbooks in Operations Research and Management Science* **8**: 485–550.

Freuder, E. C. 1995. Using inference to reduce arc consistency computation, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI'95). 23, 28, 33*, Morgan Kaufmann Publishers Inc., pp. 592–598.

Gil, R. P. A., Johanyák, Z. C. and Kovács, T. 2018. Surrogate model based optimization of traffic lights cycles and green period ratios using microscopic simulation and fuzzy rule interpolation, *International Journal of Artificial Intelligence* **16**(1): 20–40.

Gill, P. E., Murray, W. and Saunders, M. A. 2005. Snopt: An sqp algorithm for large-scale constrained optimization, *SIAM review* **47**(1): 99–131.

Glover, F. and Laguna, M. 1998. Tabu search, *Handbook of combinatorial optimization*, Springer, pp. 2093–2229.

Gungor, V. C., Sahin, D., Kocak, T., Ergut, S., Buccella, C., Cecati, C. and Hancke, G. P. 2011. Smart grid technologies: Communication technologies and standards, *IEEE transactions on Industrial informatics* **7**(4): 529–539.

Hentenryck, P. V. and Michel, L. 2009. *Constraint-based local search*, The MIT press.

Homaifar, A., Qi, C. X. and Lai, S. H. 1994. Constrained optimization via genetic algorithms, *Simulation* **62**(4): 242–253.

Hooker, J. N. 2002. Logic, optimization, and constraint programming, *INFORMS Journal on Computing* **14**(4): 295–321.

Kolesar, P. J. 1967. A branch and bound algorithm for the knapsack problem, *Management science* **13**(9): 723–735.

Kumar, V. 1992. Algorithms for constraint-satisfaction problems: A survey, *AI magazine* **13**(1): 32–32.

La Mura, P. 2000. Game networks, *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence, San Francisco, CA*, Morgan Kaufmann Publishers Inc., pp. 335–342.

Larrosa, J. and Schiex, T. 2004. Solving weighted csp by maintaining arc consistency, *Artificial Intelligence* **159**(1-2): 1–26.

Mackworth, A. K. 1977. Consistency in networks of relations, *Artificial Intelligence* **8**(1): 99–118.

Marah, R. and El Hibaoui, A. 2018. Algorithms for smart grid management, *Sustainable cities and society* **38**: 627–635.

Mohr, R. and Henderson, T. C. 1986. Arc and path consistency revisited, *Artificial intelligence* **28**(2): 225–233.

Myerson, R. B. 2013. *Game theory*, Harvard university press.

Nadel, B. A. 1990. *Some applications of the constraint satisfaction problem*, Wayne State University, Department of Computer Science.

Narendra, J., Rochart, G. and Lorca, X. 2008. Choco: an open source java constraint programming library, *CPAIOR, Paris, France*, Vol. 8, pp. 1–10.

Osborne, M. J. and Rubinstein, A. 1994. *A course in game theory*, MIT press.

Precup, R.-E., Angelov, P., Costa, B. S. J. and Sayed-Mouchaweh, M. 2015. An overview on fault diagnosis and nature-inspired optimal control of industrial process applications, *Computers in Industry* **74**: 75–94.

Precup, R.-E. and David, R.-C. 2019. *Nature-Inspired Optimization Algorithms for Fuzzy Controlled Servo Systems*, Butterworth-Heinemann.

Prud'homme, C., Fages, J.-G. and Lorca, X. 2014. Choco documentation, *TASC, INRIA Rennes, LINA CNRS UMR* **6241**: 64–70.

Ross, K. W. and Tsang, D. H. 1989. The stochastic knapsack problem, *IEEE Transactions on communications* **37**(7): 740–747.

Rossi, F., Van Beek, P. and Walsh, T. 2006. *Handbook of constraint programming*, Elsevier.

Shams, M., Rashedi, E., Dashti, S. and Hakimi, A. 2017. Ideal gas optimization algorithm, *International Journal of Artificial Intelligence* **15**(2): 116–130.

Shiloach, Y. and Vishkin, U. 1982. An o (n2log n) parallel max-flow algorithm, *Journal of Algorithms* **3**(2): 128–146.

Vaščák, J. 2012. Adaptation of fuzzy cognitive maps by migration algorithms, *Kybernetes* **41**(3–4): 429–443.

Verfaillie, G., Lemaître, M. and Schiex, T. 1996. Russian doll search for solving constraint optimization problems, *AAAI/IAAI, Portland, Oregon*, Vol. 1, pp. 181–187.

Walsh, T. 2000. Sat v csp, *International Conference on Principles and Practice of Constraint Programming*, Springer, pp. 441–456.

Yokoo, M. 2001. Distributed constraint satisfaction problem, *Distributed Constraint Satisfaction*, Springer, pp. 47–54.