# Optimisation of electrical network configuration: complexity and algorithms for ring topologies

Dominique Barth, Thierry Mautor, Arnaud de Moissac, Dimitri Watel, Marc-Antoine Weisser

# Optimisation of electrical network configuration: complexity and algorithms for ring topologies

Dominique Barth[a], Thierry Mautor[a], Arnaud de Moissac[b], Dimitri Watel[c,d], Marc-Antoine Weisser[e]

[a] *DAVID, University of Versailles-St Quentin, Versailles, 45 avenue des États-Unis, 78035, Versailles, France*
[b] *DCBrain, 55 Boulevard Vincent Auriol, 75013, Paris, France*
[c] *ENSIIE, 1 square de la Résistance, 91025, Evry, France*
[d] *SAMOVAR, Telecom SudParis, 9 Rue Charles Fourier, 91000, Évry, France*
[e] *LRI, CentraleSupelec, Paris-Saclay University, Rue Noetzlin, 91190, Gif-sur-Yvette, France*

**Abstract**

We consider power distribution networks containing source nodes producing electricity and nodes representing electricity consumers. These sources and these consumers are interconnected by a switched network. Configuring this network consists in deciding which switches are activated and the orientation of the links between these switches, so as to obtain a directed acyclic graph (DAG) from the producer nodes to the consumer nodes. This DAG is valid if the electric flow it induces satisfies the demand of each consumer without exceeding the production capacity of each source and the flow capacity of each switch. We show that the problem of deciding if such a valid DAG exists is NP-complete. In the case where such a valid DAG exists, we study the problem of determining a valid DAG that balances the ratio between the amount of electricity produced and the maximum production capacity for each source. We show that this minimization problem is also NP-complete in the general case but that it becomes polynomial in the case of ring network topologies.

*Keywords:* Complexity, Electrical network flow, Ring topology, Dynamic programming

## 1. Introduction

Reliability and resilience are two key features for modern a power grid. As the world becomes more and more electrical and the grids themselves more and more distributed, reliability and resilience are critical for consumers and grid managers. Reliability can be defined as the ability of the power system to deliver electricity in the quantity and with the quality demanded by users. Resilience is the ability of a system to recover and to reduce the magnitude and/or duration of disruptive events [16, 17, 21].

To be reliable and resilient, a grid needs to be in a configuration that can handle an outage and can converge to a new reliable and resilient configuration. In this context, a configuration consists in deciding which components (lines, sources, switches) have to be activated or not in the distribution network. In this paper, we present a way to configure a grid to find the more balanced configuration in terms of power load [9, 13, 22]. Indeed, when the grid is reconfigured after an outage, the electrical flow is switched to a new part of grid. If this new section is already heavily loaded, it can cause a snowball effect. With a well-balanced configuration, this risk is reduced. The fact that a well balanced grid suffers less electrical losses can be seen as a bonus.

In terms of optimisation, three main problems are usually studied when defining configurations of electrical distribution networks. Firstly, given a required load scenario, dimensioning the network infrastructure to reach flow and reliability constraints [3, 6, 18], with objective to minimize the cost of investment of electric lines and switches. Secondly, the reconfiguration of networks in case of failures, where two objectives are usually to be optimised [8, 12, 15], namely to maximize the restore load and to minimize the size of the sequence of operations to be done from the configuration before fails, mainly the switches activation, to reach the decided re-configuration ( these two objectives can be considered simultaneously [4, 7]). Finally,

planning schedules [2, 9, 14] consisting in a schedule of consecutive configurations for different consecutive scenarios of load requirements. Here again, optimisation criteria consists usually in minimizing the cost of activated components and/or the minimization of the size of the sequence of modifications between two consecutive configurations. In such planning schedules, vulnerability of configurations can be taken into account by considering various graph theory metrics [1]. Note that in this context, balanced planning schedules can also be provided by configuring the network in balanced subnetworks [21, 9, 12] or by considering balanced configurations in terms of power of loads [20].

Reconfiguration and planning problems are oftently considered through a graph theory point of view [18, 1]. Some solutions considers graph partitioning [12] and others graph covering by spanning subtrees or sub-DAGS [13]. In this paper, we focus on such approaches related to Steiner tree problems [10, 23] with a specific metric focusing balanced distribution of the required load on all the sources. Indeed in our context, the electric flow in a network is a direct consequence of the chosen configuration and the consumers [20]. Thus the objective is not to compute an electric flow in a graph (such as in [5]) but rather to determine the best spanning sub-DAG of the whole network optimizing the balance of proportional use of sources capacities.

The paper is organized as follows. First, we define our modelization of power grid and the related computational problems. We proove that these problems are *NP*-complete. Finally, we provide a polynomial time algorithm for the restricted case of ring networks.

## 2. Modelization

### *2.1. Distribution network topology*

We consider a connected mixed graph $G = (V = S \cup W \cup P, E, A)$ in which vertices in $S$ represent *electrical sources*, $W$ *switches* and $P$ *consumers*, with $A$ a set of arcs and $E$ a set of edges. Set of arcs $A$ connect each vertex of $S$, as an initial extremity, to some vertices in $W$ being final extremities. Edges in $E$ interconnect vertices of $W$ and each vertex in $P$ is connected by an edge of $E$ to at least one vertex in $W$.

Each vertex $x \in S$ is characterized by a maximum production capacity denoted $Prod(x) > 0$. Each vertex $y \in P$ is characterized by a called power, denoted by $Pow(y)$. Each $w \in W$ is characterized by a flow capacity $Cap(w)$.

### *2.2. Activation and orientation of the network*

An **activation** of $G$ is a function $\alpha : W \to \{0, 1\}$. We denote by $W^1_\alpha$ the subset of vertices $x \in W$ such that $\alpha(x) = 1$. We define $G_\alpha = (V_\alpha, E_\alpha, A_\alpha)$ the subgraph of $G$ induced by $V_\alpha = S \cup W^1_\alpha \cup P$.

An **orientation** $\mathcal{O}_\alpha$ of $G_\alpha$ is a function associating each edge $[x, y] \in E_\alpha$ with a couple $(x, y)$ or $(y, x)$ corresponding to an orientation of this edge. We denote by $G^{\mathcal{O}_\alpha}$ the digraph obtained by applying such an orientation to $G_\alpha$.

Let $\alpha$ be an activation of $G$ and $\mathcal{O}_\alpha$ be an orientation of $G_\alpha$ such that $G^{\mathcal{O}_\alpha}$ is a Directed Acyclic Graph (DAG) whose leaves are vertices in $P$. Such a DAG is said to be **coherent** iff each vertex $v \in W^1_\alpha \cup P$ is included in at least one path from a vertex of $S$ to a vertex of $P$. Let us underline that to be coherent, the directed graph $G^{\mathcal{O}_\alpha}$ must be acyclic. It is not possible to have cycles of electricity.

### *2.3. Flow in a oriented and activated DAG*

Giving an activation $\alpha$ and an orientation $\mathcal{O}_\alpha$ of $G$ such that $G^{\mathcal{O}_\alpha}$ is coherent, we deterministically compute a flow $F$ in $G^{\mathcal{O}_\alpha}$ as follows.

For each vertex $y$ of $G^{\mathcal{O}_\alpha}$ (except the sources), let $\Gamma^+(y)$ (with cardinal $d^+(y)$) be its set of successors in $G^{\mathcal{O}_\alpha}$ ($\Gamma^-(y)$ and $d^-(y)$ for the predecessors). The flow on each arc $(x, y)$ of $G^{\mathcal{O}_\alpha}$ is

$$F(x,y) = \frac{\alpha_y + \sum_{z \in \Gamma^+(y)} F(y,z)}{d^-(y)}$$

where $\alpha_y = Pow(y)$ if $y \in P$, else $\alpha_y = 0$. Given that $G^{\mathcal{O}_\alpha}$ is coherent, $d^-(y) > 0$ for all $y \in W^1_\alpha \cup P$.

The flow coming out of $y$ (plus the possible power called in $y$) is distributed equitably over all the arcs entering $y$. Note that consequently, for a given activation and orientation, the flow is calculated by going up from the vertices of P and is unique.

Considering such a flow $F$, for each source or switch $x \in S \cup W^1_\alpha$, we note

$$Load(x) = \sum_{z \in \Gamma^+(x)} F(x,z).$$

By definition, since DAG $G^{\mathcal{O}_\alpha}$ is coherent then for each $w \in W^1_\alpha$, we have $Load(w) > 0$.

Such a DAG $G^{\mathcal{O}_\alpha}$ is **valid** iff it is coherent and

- for each $w \in W^1_\alpha$ we have $Load(w) \leq Cap(w)$,
- for each $s \in S$, we have $Load(s) \leq Prod(s)$.

**Problem** VALID.
**Given** a mixed graph $G = (V = S \cup W \cup P, E, A)$, three functions *Prod*, *Cap* and *Pow*.
**Question :** Does there exist an activation $\alpha$ and an orientation $\mathcal{O}_\alpha$ of $G_\alpha$ such that $G^{\mathcal{O}_\alpha}$ is a DAG and is valid?

### *2.4. Network optimization*

To evaluate the quality of a valid pair of orientation and activation $(\alpha, \mathcal{O}_\alpha)$ of a graph $G$, we consider the following metric of the DAG $G^{\mathcal{O}_\alpha}$. This metric measures the maximum gap of solicitation rate of the sources, knowing that the more these rates are balanced, the more the network is resistant to a reconfiguration in case of failures [9, 13].

The **load reserve** of $G^{\mathcal{O}_\alpha}$ is defined by

$$Res(G^{\mathcal{O}_\alpha}) = \max_{\{s,s'\} \subset S} \left| \frac{Load(s)}{Prod(s)} - \frac{Load(s')}{Prod(s')} \right|.$$

As we will see in Theorem 1, the Problem VALID is *NP*-complete. The corollary is that the problem of finding an activation and an orientation minimizing the load reserve is not approximable. In this context, to focus on this optimisation problem, we only consider in the following valid instances, i.e., instances for which the answer to Problem VALID is positive.

The problem we focus on is then the following :

**Problem** RES_CHG.
**Given** A valid mixed graph $G = (V = S \cup W \cup P, E, A)$, three functions *Prod*, *Cap* and *Pow* and an integer $K$.
**Question :** Does there exist an activation $\alpha$ and an orientation $\mathcal{O}_\alpha$ of $G_\alpha$ such that $G^{\mathcal{O}_\alpha}$ is a valid DAG and $Res(G^{\mathcal{O}_\alpha}) \leq K$?

Note that the optimisation related to this decision problem is a minimization problem.

## 3. Complexity of Problems VALID and RES_CHG

### *3.1. Problem VALID is NP-complete*

In this section we prove the Problem VALID is *NP*-Complete even in the restricted case where the capacities of the switches are arbitrarily large or in the case where the production capacities of the sources are arbitrarily large.

**Lemma 1.** *Problem VALID belongs to NP.*

*Proof.* Given an activation $\alpha$ and an orientation $\mathcal{O}_\alpha$, we can, in polynomial time, determine if $G^{\mathcal{O}_\alpha}$ is a DAG; check that, for every node $v \in P \cup W^1_\alpha$, at least one source fills $v$; compute the flow from the leaves to the sources and then check the capacity and production constraints. Thus, it is possible to check if a DAG is valid in polynomial time. □

We now give a polynomial reduction from (3-SAT) to (VALID) restricted to the case where each switch has a capacity arbitrarily large, *i.e.* for any switch $w$, all the called power can go through $w$: $Cap(w) \geq \sum_{p \in P} Pow(p)$.

Let $\mathcal{I} = (x_1, x_2, \ldots, x_n, \varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_m)$ be an instance of (3-SAT) where $x_i$ are boolean variables and $C_i$ are disjunctive clauses with 3 literals. We build an instance $\mathcal{J}$ of (VALID) from $\mathcal{I}$, *i.e.* a mixed graph $G = (S \cup W \cup P, E, A)$ and three functions *Prod*, *Cap* and *Pow*.

We define two useful parameters to simplify the calculations: $\beta = 10m+1$ and $\gamma = n \cdot (6\beta + 6m) - 4m - 3$.

For each variable $x_i$ $(1 \leq i \leq n)$, we add a tree containing 13 nodes to $G$ (this tree is drawn in Figure 1): five switchs $\{w^2_{x_i}, w^1_{x_i}, w_{x_i}, w^1_{\bar{x}_i}, w^2_{\bar{x}_i}\} \subset W$; five sources $\{s^2_{x_i}, s^1_{x_i}, s_{x_i}, s^1_{\bar{x}_i}, s^2_{\bar{x}_i}\} \subset S$ with the following respective production capacities $(4\beta, \beta + m, \gamma + \beta + m, \beta + m, 4\beta)$; and three consumers $\{p^2_{x_i}, p_{x_i}, p^2_{\bar{x}_i}\} \subset P$ with the following called powers $\{4\beta, \gamma, 4\beta\}$. We link those nodes with 5 arcs of $A$ and 7 edges of $E$: five arcs $(s^2_{x_i}, w^2_{x_i})$, $(s^1_{x_i}, w^1_{x_i})$, $(s_{x_i}, w_{x_i})$, $(s^1_{\bar{x}_i}, w^1_{\bar{x}_i})$ and $(s^2_{\bar{x}_i}, w^2_{\bar{x}_i})$, three edges $[w^2_{x_i}, p^2_{x_i}]$, $[w_{x_i}, p_{x_i}]$ and $[w^2_{\bar{x}_i}, p^2_{\bar{x}_i}]$ and four edges $[w^2_{x_i}, w^1_{x_i}]$, $[w^1_{x_i}, w_{x_i}]$, $[w_{x_i}, w^1_{\bar{x}_i}]$ and $[w^1_{\bar{x}_i}, w^2_{\bar{x}_i}]$.

For each clause $C_j = (l_1 \vee l_2 \vee l_3)$ $(1 \leq j \leq m)$, we add two switches $\{w_{C_j}, w^0_{C_j}\} \subset P$; a source $s^0_{C_j}$ with production capacity $Prod(s^0_{C_j}) = \gamma + 2$; two consumers $\{p_{C_j}, p^0_{C_j}\} \subset P$ with the following respective called powers $(4, \gamma)$; an arc $(s^0_{C_j}, w^0_{C_j})$, three edges $[w_{C_j}, p_{C_j}]$, $[w^0_{C_j}, p^0_{C_j}]$ and $[w^0_{C_j}, w_{C_j}]$ and three other edges $[w_{l_i}, w_{C_j}]$ for each $i \in [\![1; 3]\!]$ (one for each literal of the clause).

Let us show some useful properties of a feasible solution of instances build using the former reduction. Let $\alpha$ and $\mathcal{O}_\alpha$ be valid activation and orientation of $G$, if such a solution exists. Figure 1 illustrates such a gadget with all the consequences of those properties.

**Property 1.** *For all $i \in [\![1; n]\!]$, $\alpha(w^2_{x_i}) = \alpha(w_{x_i}) = \alpha(w^2_{\bar{x}_i}) = 1$ and the orientation $\mathcal{O}_\alpha$ directs the edges $[w^2_{x_i}, p^2_{x_i}]$, $[w_{x_i}, p_{x_i}]$ and $[w^2_{\bar{x}_i}, p^2_{\bar{x}_i}]$ from the switch to the consumer nodes. The same property occurs for all $j \in [\![1; m]\!]$ for $w_{C_j}$, $w^0_{C_j}$, $p_{C_j}$ and $p^0_{C_j}$ ($\alpha(w_{C_j}) = \alpha(w^0_{C_j}) = 1$, orientations $(w_{C_j}, p_{C_j})$ and $(w^0_{C_j}, p^0_{C_j})$).*

*Proof.* It is obvious, otherwise, no source fills one of the consumer nodes $p^2_{x_i}$, $p_{x_i}$, $p^2_{\bar{x}_i}$, $p_{C_j}$ or $p^0_{C_j}$. □

**Property 2.** *For all $j \in [\![1; m]\!]$, $\mathcal{O}([w_{C_j}, w^0_{C_j}]) = (w^0_{C_j}, w_{C_j})$.*

*Proof.* Both $w^0_{C_j}$ and $w_{C_j}$ are active (Prop. 1). Let us suppose that the edge is orientated $(w_{C_j}, w^0_{C_j})$. In this case, $s^0_{C_j}$ produces a flow of $\frac{\gamma}{2}$ for the consumer $p^0_{C_j}$. The total productions of all the other sources is $n \cdot (\gamma + 11\beta + 3m) + (m-1) \cdot (\gamma + 2)$. The total called power not filled by $s^0_{C_j}$ is at least $n \cdot (\gamma + 8\beta) + m \cdot (4 + \gamma) - \frac{\gamma}{2}$. By the conservation of the flow and as all the consumers must be filled, we must satisfy the following

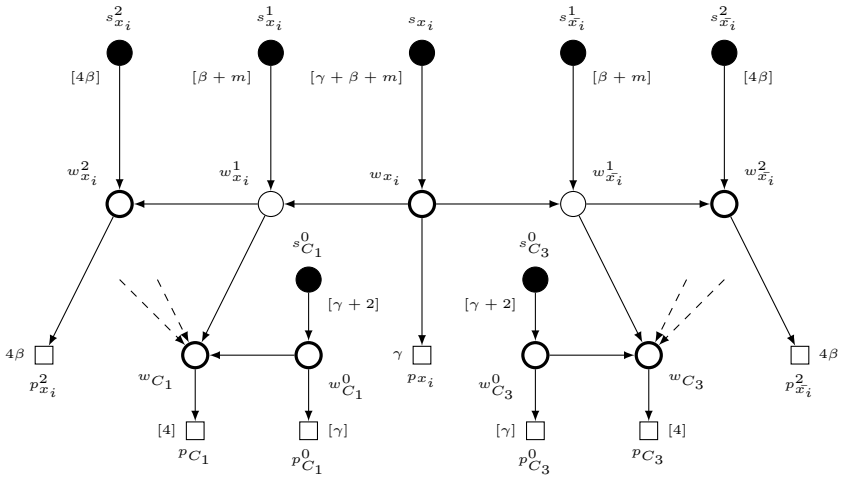


Figure 1: Gadget associated with a variable $x_i$ in a reduction from (3-SAT) to (VALID). The literal $x_i$ is contained in the clause $C_1$ and the negative literal $\bar{x_i}$ is contained in the clause $C_3$. The black nodes are the sources, the square nodes are the consumers and the others are the switches. Next to each node is written inside square brackets its capacity (for the sources) or its called power (for the consumers). Note that each of the nodes $w_{C_1}$ and $w_{C_3}$ is linked to two other nodes in the graph (the 2 other literals of the clause). On the gadget are also drawn the necessary activations and orientations of that instance in case it is valid. An activated node is drawn thick. Other nodes may or may not be activated.

constraint:

$$\begin{aligned}
n \cdot (\gamma + 11\beta + 3m) + (m-1) \cdot (\gamma + 2) &\geq n \cdot (\gamma + 8\beta) + m \cdot (4 + \gamma) - \frac{\gamma}{2} \\
n \cdot (3\beta + 3m) + (m-1) \cdot (\gamma + 2) &\geq m \cdot (4 + \gamma) - \frac{\gamma}{2} \\
n \cdot (3\beta + 3m) + (m-1) \cdot 2 &\geq \gamma + 4m - \frac{\gamma}{2} \\
n \cdot (6\beta + 6m) + (m-1) \cdot 4 - 8m &\geq \gamma \\
n \cdot (6\beta + 6m) - 4m - 4 &\geq \gamma
\end{aligned}$$

However $\gamma = n \cdot (6\beta + 6m) - 4m - 3 > n \cdot (6\beta + 6m) - 4m - 4$. Consequently, the property is proved. □

**Property 3.** *For all $i \in [\![1; n]\!]$, if $\alpha(w^1_{x_i}) = 1$ and $Load(w^1_{x_i}) > 0$ then, $\mathcal{O}([w^2_{x_i}, w^1_{x_i}]) = (w^1_{x_i}, w^2_{x_i})$ and, for all $j$ such that $C_j$ contains the litteral $x_i$, $\mathcal{O}([w^1_{x_i}, w_{C_j}]) = (w^1_{x_i}, w_{C_j})$. The same property occurs for $\bar{x_i}$.*

*Proof.* If $\mathcal{O}([w^2_{x_i}, w^1_{x_i}]) = (w^2_{x_i}, w^1_{x_i})$ and $Load(w^1_{x_i}) > 0$ then $Load(w^2_{x_i}) > 4\beta$ and all that flow must come from $s^2_{x_i}$. This is not possible as $Prod(s^2_{x_i}) = 4\beta$. Consequently, if $\alpha(w^1_{x_i}) = 1$ and $Load(w^1_{x_i}) > 0$, then the edge $[w^1_{x_i}, w^2_{x_i}]$ is directed to $w^2_{x_i}$ and $Load(w^1_{x_i}) \geq 2\beta$.
If we now assume that, for some $C_j$ containing the litteral $x_i$, $\mathcal{O}([w^1_{x_i}, w_{C_j}]) = (w_{C_j}, w^1_{x_i})$, then $Load(w_{C_j}) \geq 4 + \frac{2\beta}{d^-(w^1_{x_i})} \geq 4 + \frac{2\beta}{m+2}$. By Property 2, a part of that flow comes from $s^0_{C_j}$ through $w^0_{C_j}$. Consequently, $Load(s^0_{C_j}) = Load(w^0_{C_j}) = \gamma + \frac{Load(w_{C_j})}{d^-(w_{C_j})} \geq \gamma + \frac{Load(w_{C_j})}{4} \geq \gamma + \frac{\beta}{2(m+2)}$. However, as $\beta > 10m$ and $Prod(s^0_{C_j}) = \gamma + 2$, the source $s^0_{C_j}$ cannot satisfies its production capacity constraint. □

**Property 4.** *For all $i \in [\![1; n]\!]$, if $\alpha(w^1_{x_i}) = 1$ then $\mathcal{O}([w_{x_i}, w^1_{x_i}]) = (w_{x_i}, w^1_{x_i})$. The same property occurs for $\bar{x_i}$.*

*Proof.* If we assume the contrary, then by Properties 2 and 3, all the flow going through the arc $(w^1_{x_i}, w_{x_i})$ comes from the source $s^1_{x_i}$. Thus $Load(s^1_{x_i}) \geq 2\beta + \frac{\gamma}{3}$. However, as $\beta > m$ and $Prod(s^1_{x_i}) = \beta + m$, the source $s^1_{x_i}$ cannot satisfies its production capacity constraint.

□

The only decision that should be made is to activate or not the nodes $w^1_{x_i}$ and $w^1_{\bar{x_i}}$.

**Property 5.** *It is not possible to activate $w^1_{x_i}$ and $w^1_{\bar{x_i}}$ at the same time.*

*Proof.* If $w^1_{x_i}$ and $w^1_{\bar{x_i}}$ were activated at the same time, $Load(w^1_{x_i}) \geq 2\beta$ and $Load(w^1_{\bar{x_i}}) \geq 2\beta$ due to the flows going to $w^2_{x_i}$ and $w^2_{\bar{x_i}}$. As the switch $w_{x_i}$ is activated, half of these values is sent by $w_{x_i}$ (the other half is produced by the sources $s^1_{x_i}$ and $s^1_{\bar{x_i}}$). However $w_{x_i}$ also sends $\gamma$ units of flow to $p_{x_i}$. Consequently $Load(s_{x_i}) = Load(w_{x_i}) \geq 2\beta + \gamma > \beta + m + \gamma$: the capacity constraint is not satisfied for $s_{x_i}$. Thus it is not possible to activate $w^1_{x_i}$ and $w^1_{\bar{x_i}}$ at the same time. □

**Lemma 2.** *Given a boolean formula $\varphi$ and an instance $\mathcal{J}$ of (VALID) built from $\varphi$ using the former reduction. If $\varphi$ is satisfiable then $\mathcal{J}$ is valid.*

*Proof.* If the formula $\varphi$ can be satisfied then there exists a truth affectation of the variables. We activate $w^1_{x_i}$ if $x_i$ if true and $w^1_{\bar{x_i}}$ otherwise.

For each clause $C_j = (l_1 \vee l_2 \vee l_3)$, at least one of the three literals is true. Let $l_i$ be that literal, then $s^1_{l_i}$ fills $p_{C_j}$. Each other consumer node $p_{x_i}$, $p^2_{x_i}$ and $p^2_{\bar{x_i}}$ is filled at least by the corresponding source in the gadget of $x_i$.

The capacity constraint is satisfied for all the nodes.

- $Load(s^0_{C_j}) \leq \gamma + \frac{Load(w_{C_j})}{2} \leq \gamma + 2 = Prod(s^0_{C_j})$;
- $Load(s^2_{x_i})$ and $Load(s^2_{\bar{x_i}})$ are either $2\beta$ or $4\beta$ and $Prod(s^2_{x_i}) = Prod(s^2_{\bar{x_i}}) = 4\beta$.
- If $\alpha(w^1_{x_i}) = 1$, then $Load(s^1_{x_i}) \leq \frac{2\beta+2m}{2} = Prod(s^1_{x_i})$. Similarly for $w^1_{\bar{x_i}}$.
- Finally $Load(s_{x_i}) \leq \gamma + \beta + m = Prod(s_{x_i})$ as $w^1_{x_i}$ and $w^1_{\bar{x_i}}$ cannot be activated at the same time.

□

**Lemma 3.** *If $\mathcal{J}$ is valid then $\varphi$ is satisfiable.*

*Proof.* If there exists a valid activation $\alpha$ and a valid orientation $\mathcal{O}_\alpha$ of $G$, then we define the following truth affectation: $x_i$ is true if and only if $w^1_{x_i}$ is activated.

Each consumer node is filled by at least one source. The consumer $p_{C_j}$ cannot be completely filled by $s^0_{C_j}$, otherwise $Load(s^0_{C_j})$ would be $\gamma + 4$, greater than $Prod(s^0_{C_j}) = \gamma + 2$.

Thus, for each clause $C_j = (l_1 \vee l_2 \vee l_3)$, there exists a path in $G^{\mathcal{O}_\alpha}_\alpha$ from some source to $p_{C_j}$ and that path necessarily goes through one of the nodes $w^1_{l_i}$. Consequently, that literal is true and the clause is also true: the formula is satisfied by this assignment. □

**Theorem 1.** *The Problem VALID is NP-complete, even if the maximum capacity of the switches is arbitrarily large.*

*Proof.* The proof is a direct consequence of lemmas 1, 2 and 3 □

**Corollary 1.** *The Problem VALID is NP-complete, even if the maximum capacity of the switches is arbitrarily large.*

*Proof.* By Theorem 1, the problem VALID is NP-Complete. We can easily transform an instance of VALID to an instance where all the sources have an arbitrarily large production capacity by adding, for each source $s$, a new switch $w_s$ with capacity $Cap(w_s) = Prod(s)$. We then delete the edges incident to $s$ and link $w_s$ to $s$ and each previous neighbour of $s$. Finally, we can now arbitrarily increase the value of $Prod(s)$ without changing the feasible flows. □

Relaxing the constraints on the capacity and production makes the Problem VALID easy to solve. We can produce a solution to such instances by choosing an orientation of the edges from the sources to the producers. With only one of these constraints, the problem VALID becomes NP-Complete.

*3.2. Problem RES_CHG is NP-complete*

**Theorem 2.** *Problem RES_CHG is NP-complete.*

*Proof.* Problem RES_CHG is clearly in NP, since checking whether an activation and an orientation answer positively to the question can be done in polynomial time.

Consider the Set-Cover problem of sets which instance is a set $\epsilon$ with cardinal $M \geq 2$, a set of subsets $R$ of $\epsilon$ and an integer $k$. The question is to decide if there are (at most) $k$ subsets of $R$ whose union is $\epsilon$. This problem is known as NP-complete [11].

From such an instance $(\epsilon, R, k)$, we define an instance of $RES\_CHG$, i.e., a graph $G = (S \cup W \cup P, E, A)$ and three functions *Prod*, *Cap* and *Pow* as follows (this graph is illustrated on Figure 2). Let us consider $\epsilon = \{e_1, \ldots, e_M\}$ and $R = \{r_1, \ldots, r_N\}$.

- $S = \{S_1, S_2\}$ with maximum production capacities $Prod_1 = Prod_2 = N \times (M + 1)$.
- $P$ is the union of $\epsilon$ and of a set of new vertices $\Pi = \{p_1, \ldots, p_N\}$. We fix $Pow(e_i) = N$ for each vertex $e_i \in \epsilon$ and $Pow(p_j) = 2$ for each vertex $p_j \in \Pi$.
- $W$ is the union of $R$ and of a set of new vertices $\mu = \{w_1, \ldots w_N\}$ with each vertex of $R$ with capacity greater than $Prod_1 + Prod_2$ and $cap(w_i) = 2$ for each $w_i \in \mu$.

Moreover,

- There is an arc from $S_1$ to each vertex in $R$ and from $S2$ to each vertex in $\mu$.
- There is an edge $[p_i, w_i]$ and an edge $[r_i, w_i]$ for each $1 \leq i \leq N$. Note that since each vertex in $p_i \in \Pi$ has to be satisfied, then in any valid DAG obtained from $G$ each vertex $w_i \in \mu$ has to be activated and edges $[p_i, w_i]$ are oriented from $w_i$ to $p_i$. Thus, for each $w_i$ we have $Load(w_i) = Pow(p_i) = 2$.
- For each vertex $r_i \in R$, there is an edge $[r_i, e_j]$ for each $e_j \in \epsilon$ such that $e_j \in r_i$.

We define $K = \frac{N(M-2)+2k}{N(M+1)}$.

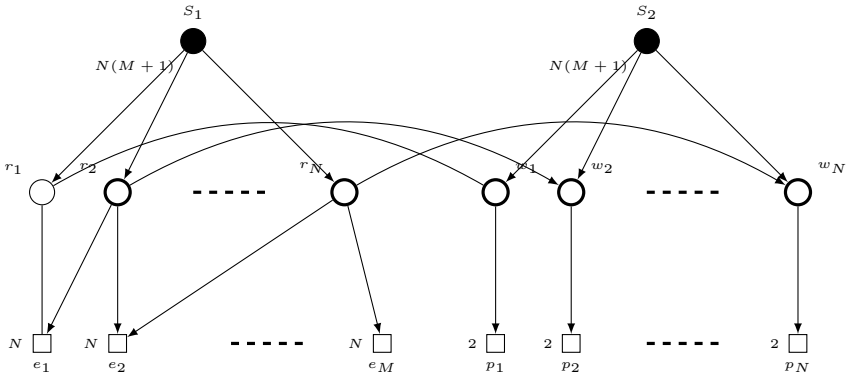


Figure 2: Nodes $r_3, .., r_{N-1}, w_3, .., w_{N-1}, e_3, .., e_{M-1}, p_3, ..p_{N-1}$ are missing on the figure. We suppose the switchs $r_2$ and $r_N$ activated (drawn thick) but $r_1$ not activated. This is the reason why edges $(r_1, e_1)$ and $(r_1, w_1)$ are not oriented.

Let us now consider a subset $C$ with cardinal $k$ of $R$ covering $\epsilon$. We obtain in polynomial time an activation and an orientation of $G$ as follows.

- Only vertices of $C$ are activated, then vertices in $R - C$ are disactivated.
- Edges $[r_i, e_j]$ are oriented from $r_i$ to $e_j$ for each $r_i \in C$.

- Edges $[r_i, w_i]$ are oriented from $r_i$ to $w_i$ for each $r_i \in C$; in this case $w_i$ has two predecessors $S2$ and $r_i$.

In the so obtained DAG the load of the two sources are

$$Charg(S1) = MN + Card(C) = MN + k.$$

$$Charg(S2) = \sum_{i:r_i \in C} \frac{F(w_i)}{2} + \sum_{i:r_i \in R-C} F(w_i) = k + 2(N-k) = 2N - k$$

Indeed, the $M$ consumers $e_i$ with called powers equal to $N$ are filled by $S_1$ and consumers in $C$ have a part of their flow equal to 1 coming from their corresponding $w_i$.

Finally, since each consumer is fully filled, the obtained DAG is valid and its load reserve is equal to

$$\frac{MN + k - 2N + k}{N(M+1)} = \frac{N(M-2) + 2k}{N(M+1)} = K$$

Consider now a valid DAG $G^{\mathcal{O}_\alpha}$ obtained from an activation $\alpha$ and an orientation $\mathcal{O}_\alpha$ of $G$ such that $Res(G^{\mathcal{O}_\alpha}) \leq K = \frac{MN-2(N-k)}{N(M+1)}$. For each consumer $e_j$, there exists at least one activated vertex $r_i \in R$ with an edge $[r_i, e_j]$ oriented from $r_i$ to $e_j$ since if not $e_j$ can not be filled. If edge $[r_i, w_i]$ is oriented from $w_i$ to $r_i$ the flow value $Trans(w_i)$ is then equal to $\frac{F(r_i)}{2} + Pow(p_i)$ which is greater than $Cap(w_i) = 2$. The DAG being valid, each edge $[r_i, w_i]$ is thus oriented from $r_i$ to $w_i$ if $r_i$ is activated. Thus, consumers $e_j \in \epsilon$ are filled only by $S1$. Let $C$ be the subset with cardinal $q$ containing all the activated vertices in $R$. Set $C$ covers $\epsilon$ since all the consumers $e_i$ are filled by functional sources. Morevoer,

$$Load(S1) = MN + q$$

and

$$Load(S2) = \sum_{i:r_i \in C} \frac{F(w_i)}{2} + \sum_{i:r_i \in R-C} F(w_i) = 2N - q$$

Since the load reserve is equal to $\frac{MN-2(N-k)}{N(M+1)}$ then $q = k$.

□

## 4. Polynomial solutions for ring networks

We consider here a network topology where switches are connected in a ring logic. As it is illustrated in [19] for example, some real networks have a topology based on a ring. Indeed, they have been historically deployed on the outskirts of cities, with consumers located in the city and outside sources of energy.

Let us consider a ring instance $GC = (V = S \cup W \cup P, E, A)$ defined as follows

- the subgraph of $GC$ induced by $W$ is a cycle,
- each vertex of $S$ (resp. $P$) is connected to one vertex in $W$, and no vertex of $W$ is both connected to a vertex in $S$ and a vertex in $P$.

Our objective in this section is to prove that problems VALID and RES_CHG can both be solved in polynomial time. For this, we first prove that, to find a valid and optimal solution in $GC$, it is sufficient to search an optimal solution in instances where the graph is a caterpillar. A caterpillar instance $GP = (V = S \cup W \cup P, E, A, Prod, Cap, Pow)$ has the same definition as a ring instance except that the subgraph $P_W$ induced by $W$ is an undirected path instead of a cycle. Figure 3 gives examples of those equivalent caterpillar instances.

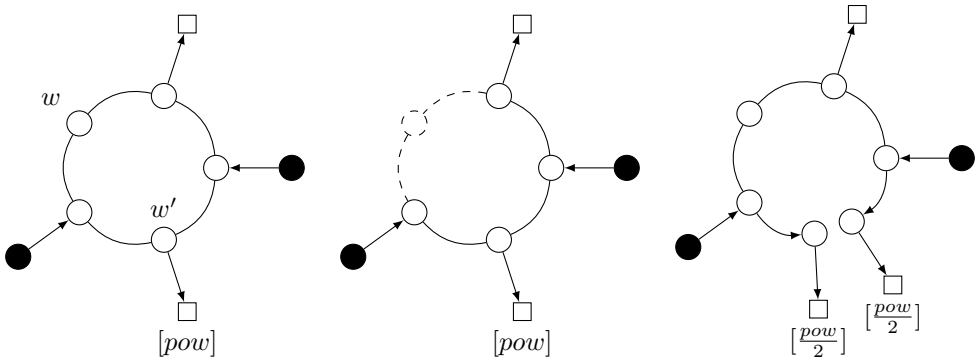


Figure 3: Example of equivalence between a ring instances (on the left) and caterpillar instances. The middle instance, $GP^A_w$, is equivalent to the case where, in an optimal solution, a switch $w$ is not activated. The right instance, $GP^B_{w'}$, is equivalent to the case where, in an optimal solution, the two edges incident to $w'$ are directed toward that switch.

**Lemma 4.** *There exists a family (of cardinality $2|W|$) of caterpillar instances $\mathcal{GP} = (GP^A_w, GP^B_w, w \in W)$ such that*

- *each instance of $\mathcal{GP}$ can be built in linear time*
- *if none of the instances of $\mathcal{GP}$ is valid, $GC$ is not valid*
- *if $GC$ is valid, there exists an instance $GP$ of $\mathcal{GP}$ with the same optimal value than $GC$. In addition, an optimal solution of $GC$ can be deduced from an optimal solution of $GP$ in linear time.*

*Proof.* For a solution to be valid, and even more optimal, it must be coherent. If in a feasible solution $(\alpha, \mathcal{O}_\alpha)$, all the nodes of $W$ are activated, there exists one node $w$ for which the two edges of $C_W$ incident to it are directed to $w$. Otherwise, $G^{\mathcal{O}_\alpha}$ would contain a directed cycle that is not possible for a coherent solution (let us recall that $\mathcal{O}_\alpha$ must be a Directed Acyclic Graph). Consequently, in any feasible solution, there exists a deactivated node of $W$ or a node of $W$ not having any successor in $W$.

If, in a feasible solution $(\alpha^*, \mathcal{O}^*)$, there exists a deactivated node $w$, we can remove $w$ from the graph, we then get a caterpillar instance $GP^A_w$. In that case, any feasible solution of $GP^A_w$ is also a feasible solution for $GC$ (more exactly, we copy that solution and complete it with $\alpha(w) = 0$) with same load reserve and conversely.

In the second case, $w$ ($w'$ on Figure 3) has no successor in $W$. Note that, necessarily, $w$ is linked to a consumer node $p$, otherwise there would not be any path going through $w$ and reaching a consumer node, and thus the solution would not be coherent. The called power of $p$ comes from the two switch neighbors of $w$, each sends half the power needed by $p$. Consequently, the instance is equivalent to a caterpillar instance in which $w$ and $p$ are duplicated. Each copy of $w$ is linked to one of the neighbors of $w$ in $W$ and to one copy of $p$. Each copy of $p$ has half the call power of $p$. We then get a caterpillar instance $GP^B_w$. As for the first case, any feasible solutions of $GP^B_w$ can be transformed in linear time into a feasible solution of $GC$ with same load reserve and conversely. □

We now prove that it is possible to find an optimal solution in any caterpillar instance in polynomial time. We are going to use a dynamic programming algorithm. The recursive function reduces the size of the instances using the same technique than the one we used to get a caterpillar instance from a ring instance in Lemma 4.

Given a feasible solution $(\alpha, \mathcal{O}_\alpha)$ of $GP$, $m(GP, \alpha, \mathcal{O}_\alpha)$ and $M(GP, \alpha, \mathcal{O}_\alpha)$ represent the minimum and maximum value of $\frac{Load(s)}{Prod(s)}$ over all sources of $S$. Using the function SOLVE in Algorithm 1, the following set is computed:

$$\mathcal{SOL}(GP) = \{(m(GP, \alpha, \mathcal{O}_\alpha), M(GP, \alpha, \mathcal{O}_\alpha)), \text{for all feasible solutions } (\alpha, \mathcal{O}_\alpha) \text{ of } GP\}$$

If no solution of $GP$ is valid, no values $m(GP, \alpha, \mathcal{O}_\alpha)$ and $M(GP, \alpha, \mathcal{O}_\alpha)$ are computed and no solution is added to $\mathcal{SOL}$. Algorithm 1 is used to compute $\mathcal{SOL}(GP)$. Note that $\mathcal{SOL}(GP)$ is a set. It contains only distinct values: a couple is added to $\mathcal{SOL}(GP)$ at Lines 10 and 13 of the function SOLVE and Line 11 of the procedure SOLVELEAF if and only if that couple is not already in the set.

Algorithm 1 computes $\mathcal{SOL}(GP)$ with the function SOLVE using the following property: in any feasible solution, there exists 3 possible cases, illustrated respectively with Figure 4, 5 and 6:

- a node $w$ of $W$ is deactivated. We can then remove $w$ from $GP$ and the problem can be decomposed into 2 independent subproblems ($GP1$ and $GP2$ in the algorithm)

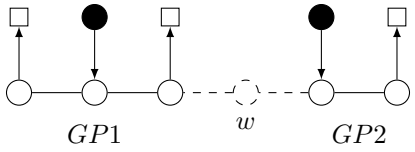


Figure 4: Reduction done when we assume a node of $W$ is deactivated.

- there exists a node $w$ of $W$ for which the two edges on $P_W$ are directed toward that node. As in Lemma 4, this node $w$ is linked to a consumer node $p$ and $w$ and $p$ can be duplicated with a called power of $\frac{pow(p)}{2}$ for $p_1$ and $p_2$. Once this is done, the problem can be decomposed into 2 independent subproblems ($GP3$ and $GP4$ in the algorithm).

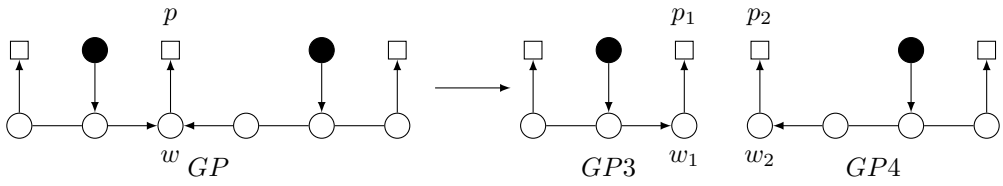


Figure 5: Reduction done when we assume two edges are directed toward a node of $W$.

- A third case can occur with no deactivated node and all the edges directed from a source $s$ (in such a case, there is no node in $W$ with the 2 edges on $P_W$ directed toward this node). In this case, all the nodes are activated and all the edges are oriented: $m(GP, \alpha, \mathcal{O}_\alpha)$ and $M(GP, \alpha, \mathcal{O}_\alpha)$ can be computed. In the following example, all the edges are directed from $S_2$. Another case should be considered with all edges directed from $S_1$.

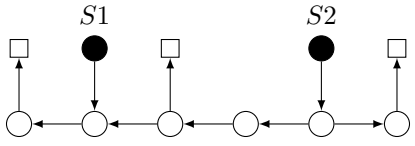


Figure 6: Case when none of the hypothesis of the cases of Figures 4 and 5 occur.

By these different and successive reductions of the problem into subproblems, all the possible solutions are covered.

**Lemma 5.** *Function* SOLVE *in Algorithm 1 is correct.*

*Proof.* Lines 8 and 11 of function SOLVE recursively explore those cases using functions SPLIT1 and SPLIT2 to split the instance and then merge the feasible solutions of the two generated subinstances at Lines 9 and

**Algorithm 1** Algorithms used to compute $\mathcal{SOL}(GP)$. The first function SolveDP is used to compute the recursive function Solve faster as every intermediate result is stored and not computed more than once.

$\mathcal{SOL}_{DP} \leftarrow$ an empty associative array (global variable)
**function** SolveDP($GP$)
    **if** $\mathcal{SOL}_{DP}[GP]$ does not exist **then**
        $\mathcal{SOL}_{DP}[GP] \leftarrow$ Solve($GP$)
    **return** $\mathcal{SOL}_{DP}[GP]$
**function** Solve($GP$)
    **Output:** $\mathcal{SOL}(GP)$
    $\mathcal{SOL} \leftarrow$ an empty set.
    **if** $GP$ has at least one source node **then**
        Remove the extremities of $P_W$ not linked to a source or a consumer.
        SolveLeaf($GP$, $\mathcal{SOL}$))
        **for** $w \in W$ **do**
            $GP1, GP2 \leftarrow$ SPLIT1(GP, w)
            **for** $(m_1, M_1) \in$ SolveDP($GP1$), $(m_2, M_2) \in$ SolveDP($GP2$) **do**
                Add $(\min(m_1, m_2), \max(M_1, M_2))$ to $\mathcal{SOL}$.
            $GP3, GP4 \leftarrow$ SPLIT2(GP, w)
            **for** $(m_3, M_3) \in$ SolveDP($GP3$), $(m_4, M_4) \in$ SolveDP($GP4$) **do**
                Add $(\min(m_3, m_4), \max(M_3, M_4))$ to $\mathcal{SOL}$.
    **return** $\mathcal{SOL}$

**function** Split1($GP$, $w$)
    **Output:** two caterpillar instances obtained when $w$ is deactivated.
    **Require:** $w \in W$ has two neighbors in $W$ and no neighbor in $P$.
    Remove $w$ from $GP$
    **return** the two connected components of $GP$.

**function** Split2($GP$, $w$)
    **Output:** two caterpillar instances obtained when $w$ is activated and when the two edges linking $w$ to its two neighbors of $W$ are directed to $w$.
    **Require:** $w \in W$ has two neighbors in $W$, one in $P$ and zero in $S$.
    $v_1, v_2 \leftarrow$ the two neighbors of $w$ in $W$.
    $p \leftarrow$ the neighbor of $w$ in $P$.
    Remove $w$ from $GP$ and add two nodes $w_1$ and $w_2$ to $W$
    Remove $p$ from $GP$ and add two nodes $p_1$ and $p_2$ to $P$ with $pow(p_1) = pow(p_2) = pow(p)/2$
    Add the edges $(w_1, v_1)$, $(w_2, v_2)$, $(w_1, p_1)$ and $(w_2, p_2)$.
    **return** the two connected components of $GP$.

```
1: procedure SolveLeaf(GP, SOL)
2: |   Description: Add to SOL the feasible solutions where no node of W is deactivated and where all
   the edges are directed from a source
3: |   α(w) ← 1 for all w ∈ W
4: |   for all Source s ∈ S(GP) do
5: |   |   w_s ← neighbor of s in GP.
6: |   |   O_α ← Direct every edge of GP from w_s
7: |   |   Compute the load of each node.
8: |   |   if (α, O_α) is not valid then
9: |   |   |   Skip to the next iteration
10: |  |   else
11: |  |   |   Add (m(GP, α, O_α), M(GP, α, O_α)) to SOL
```

Lines 12. The line 6 handles the third case with the procedure SolveLeaf. By exploring the three cases, we get the set $\mathcal{SOL}(GP)$ at the end of the function Solve.

As previously said, another possibility is that we can detect that there is no feasible solution with the load of a source exceeding the production in which case no new set is added to $\mathcal{SOL}(GP)$ (line 9 of the procedure SolveLeaf). □

We use the dynamic programming function SolveDP to compute Solve faster. Every recursive call of Solve is instead a call to SolveDP (at Lines 9 and 12). That function starts by checking if $\mathcal{SOL}(GP)$ is already stored in the associative array $\mathcal{SOL}_{DP}$. In that case it returns the value otherwise it computes it, stores it and returns it. That array is a global variable, shared by all the calls of SolveDP. This way, the function Solve is called at most once per instance. We thus need to prove that SolveDP is called a polynomial number of times and that the complexity of Solve (excluding the recursive calls) is polynomial.

**Lemma 6.** *The number of possible instances called as an argument of* SolveDP *is* $O(|W|^2)$.

*Proof.* Indeed, such an instance contains a subpath of the undirected path induced by $W$ in the main instance and all the sources and consumers connected to that path. Consequently, there is at most one instance per subpath, thus at most $O(|W|^2)$ instances. □

Note that a consequence of Lemma 6 is that there is, at most, a polynomial number of pairs stored in the array $\mathcal{SOL}_{DP}$. The complexity of SolveDP, excluding the calls to Solve, is then polynomial.

**Lemma 7.** *The size of* $\mathcal{SOL}(GP)$ *is* $O(|S|^2|P|^4)$.

*Proof.* We begin by proving that the load of a given source $s$ can have a polynomial number of distinct values among all the feasible solutions. That load depends on which consumers $s$ is filling and which other sources fill this same subset of consumers. Let $w_s$ be the neighbor of $s$ in $W$. There are two possible cases: either the two edges of $P_W$ incident to $w$ are directed in the same direction (one from $w$ and the other to $w$) or the two edges are directed from $w$.

In the first case, $s$ can only fill the consumers in that direction. Let $p$ be the farthest consumer node filled by $s$. Then, necessarily all the consumers between $s$ and $p$ are filled by $s$ and all the sources between $s$ and $p$ fill $p$. Thus, in all the feasible solutions where $p$ is the farthest consumer filled by $s$, that source has the same load. Consequently $s$ has, in that case, at most $O(|P|)$ loads.

If, on the other hand, the two edges incident to $w$ are directed from $w$, then we can use the same argument except that we have to consider two consumers instead of one, one for each direction. There are, in that case, at most $O(|P|^2)$ distinct loads for $s$.

Consequently, there exists at most $O(|S||P|^2)$ possible distinct values for $m(GP, \alpha, \mathcal{O}_\alpha)$ and $M(GP, \alpha, \mathcal{O}_\alpha)$, if we consider all the feasible solutions $(\alpha, \mathcal{O}_\alpha)$ for $GP$. As $\mathcal{SOL}(GP)$ is a set, it contains only distinct couples of values, and thus, no more than $O(|S|^2|P|^4)$. □

As $|\mathcal{SOL}(GP)| = O(|S|^2|P|^4)$, the For loops at Lines 9 and 12) do at most $O(|S|^2|P|^4) \cdot O(|S|^2|P|^4)$ iterations.

**Theorem 3.** *Problems VALID and RESCHG can be solved in time $O(n^{15})$ for caterpillar instances.*

*Proof.* By Lemma 5, Algorithm 1 is correct. Let $n$ be the size of $W$, note that $|S|, |P| \leq n$. The time complexity of the algorithm is the following: for an instance $GP$, the function SOLVE either does nothing or calls SOLVELEAF once, and, for each $w \in W$ performs SPLIT1, SPLIT2 and two for loops. SOLVELEAF requires $O(n^2)$ operations. SPLIT1 and SPLIT2 are at most linear. The For loops does at most $O(n^{12})$ iterations (by Lemma 7) and four recursive calls to SOLVEDP. Calling SOLVE does then at most $O(n^{13})$ operations and $O(n)$ calls to SOLVEDP. Such a call is either a recursive call to SOLVE or an access to $\mathcal{SOL}_{DP}$ in time $O(\log(k))$ where $k$ is the number of keys in $\mathcal{SOL}_{DP}$. In addition, SOLVE may be called at most once per subinstance. By Lemma 6, there are at most $O(n^2)$ such instances: $k = O(n^2)$ and SOLVE is called at most $O(n^2)$ times. The total number of operations of all the calls of the functions SOLVE and SOLVEDP are then respectively at most $O(n^{15})$ and at most $O(n^3 \log(k)) = O(n^3 \log(n^2))$. Consequently, the complexity of the algorithm is $O(n^{15} + n^3 \log(n^2)) = O(n^{15})$.

□

**Theorem 4.** *VALID and RESCHG can be solved in time $O(n^{16})$ for ring instances.*

*Proof.* By Theorem 3 and Lemma 4. □

Let us remark that our aim was to prove that the two problems are becoming polynomial in the case of a ring topology and not to find the best complexity. This is the reason why the question of obtaining in this case a complexity lower than $O(n^{16})$ remains open. In addition, this theoretical worst case upper bound is mainly based on the bound of Lemma 7 which is not reached on average.

**Remark.** *Algorithm 1 works because of two properties: removing a switch or dupplicating a switch disconnect the instance as in Figures 4 and 5, and the number of couples in $\mathcal{SOL}(GP)$ is polynomial. Consequently, extending this result to other topologies like instances where the subgraph induced by $W$ is a tree or two cycles is not obvious as those two properties are not necessarily true.*

## 5. Conclusion

In this paper, we have defined two graph algorithmic problems to obtain a good configuration of a switched electrical network. The first problem consists in finding a valid configuration through the activation of switches and the orientation of edges and the second problem consists in maximizing the balance of the rates of use of the different sources in order to guarantee a fault tolerance in the network. We have proved that these two problems are difficult in the general case. However, we have showed that these problems become polynomial in the case where the switches constitute a ring topology. The objective is now to study topologies extending the ring and to propose polynomial exact algorithms or practical heuristics with guarantees of performances, if such algorithms exist.

## References


[1] ATKINS, K., CHEN, J., KUMAR, V. A., AND MARATHE, A. The structure of electrical networks: a graph theory based analysis. *International Journal of Critical Infrastructures 5*, 3 (2009), 265.

[2] BABKIN, D., AND MILOVANOVA, K. Short-Term planning and operating conditions optimization of power systems under market conditions. In *IEEE Russia Power Tech* (jun 2005), IEEE, pp. 1–5.

[3] CADINI, F., ZIO, E., AND PETRESCU, C. Optimal expansion of an existing electrical power transmission network by multi-objective genetic algorithms. *Reliability Engineering & System Safety 95*, 3 (mar 2010), 173–181.

[4] Carvalho, P., Ferreira, L., and Barruncho, L. Optimization approach to dynamic restoration of distribution systems. *International Journal of Electrical Power & Energy Systems 29*, 3 (mar 2007), 222–229.

[5] Christiano, P., Kelner, J. A., Madry, A., Spielman, D. A., and Teng, S.-H. Electrical flows, laplacian systems, and faster approximation of maximum flow in undirected graphs. In *Proceedings of the 43rd annual ACM symposium on Theory of computing* (New York, New York, USA, 2011), ACM Press, p. 273.

[6] Dulau, L. I., Bica, D., and Ronay, K. Algorithm for optimal configuration of electric networks. In *International Conference on Energy and Environment* (oct 2017), IEEE, pp. 41–44.

[7] Gu, X., and Zhong, H. Optimisation of network reconfiguration based on a two-layer unit-restarting framework for power system restoration. *IET Generation, Transmission & Distribution 6*, 7 (2012), 693.

[8] Guha, S., Moss, A., Naor, J. S., and Schieber, B. Efficient recovery from power outage (extended abstract). In *Proceedings of the thirty-first annual ACM symposium on Theory of computing* (New York, New York, USA, 1999), ACM Press, pp. 574–582.

[9] Guo, J., Hug, G., and Tonguz, O. K. Intelligent Partitioning in Distributed Optimization of Electric Power Systems. *IEEE Transactions on Smart Grid 7*, 3 (may 2016), 1249–1258.

[10] Han, X., Liu, J., Liu, D., Liao, Q., Hu, J., and Yang, Y. Distribution network planning study with distributed generation based on Steiner tree model. In *IEEE /PES Asia-Pacific Power and Energy Engineering Conference* (dec 2014), IEEE, pp. 1–5.

[11] Karp, R. Reducibility among combinatorial problems. In *Complexity of Computer Computations*. Springer, 1972, pp. 85–103.

[12] Li, J. *Reconfiguration of power networks based on graph-theoretic algorithms*. PhD thesis, Iowa State University, Digital Repository, Ames, 2010.

[13] Li, J., Ma, X.-Y., Liu, C.-C., and Schneider, K. P. Distribution System Restoration With Microgrids Using Spanning Tree Search. *IEEE Transactions on Power Systems 29*, 6 (nov 2014), 3021–3029.

[14] Li, Y.-Q., Wang, L., Xie, H.-L., and Xie, Q. Distribution Network Optimal Planning Based on Clouding Adaptive Ant Colony Algorithm. In *Asia-Pacific Power and Energy Engineering Conference* (mar 2009), IEEE, pp. 1–4.

[15] Miu, K. N., Chiang, H.-D., and McNulty, R. J. Multi-tier service restoration through network reconfiguration and capacitor control for large-scale radial distribution networks. *IEEE Transactions on Power Systems 15*, 3 (2000), 1001–1007.

[16] Moslehi, K., and Kumar, R. A Reliability Perspective of the Smart Grid. *IEEE Transactions on Smart Grid 1*, 1 (jun 2010), 57–64.

[17] NERC. State of reliability. Tech. rep., NERC, Atlanta, 2015.

[18] Quiros-Tortos, J., and Terzija, V. A graph theory based new approach for power system restoration. In *2013 IEEE Grenoble Conference* (jun 2013), IEEE, pp. 1–6.

[19] RTE. Réso - Edition spéciale "Grand Paris", 2014.

[20] Shen, T., Li, Y., Xiang, J., Shen, T., Li, Y., and Xiang, J. A Graph-Based Power Flow Method for Balanced Distribution Systems. *Energies 11*, 3 (feb 2018), 511.

[21] Tang, L., Yang, F., and Ma, J. A survey on distribution system feeder reconfiguration: Objectives and solutions. In *IEEE Innovative Smart Grid Technologies - Asia* (may 2014), IEEE, pp. 62–67.

[22] Wang, W., and Yu, N. Phase Balancing in Power Distribution Network with Data Center. *ACM SIGMETRICS Performance Evaluation Review 45*, 2 (oct 2017), 64–69.

[23] Watel, D., Weisser, M. A., Bentz, C., and Barth, D. Directed Steiner trees with diffusion costs. *Journal of Combinatorial Optimization 32*, 4 (2016), 1089–1106.