Tree-Structure Based Deterministic Algorithms for Optimal Switch Placement in Radial Distribution Networks

Zbigniew Galias, *Senior Member, IEEE*

Abstract—**The problem of the optimal placement of sectionalizing switches in radial distribution networks is studied. Tree-structure based algorithms for the computation of performance indexes for the given positions of sectionalizing switches are presented. It is shown that the algorithms are very efficient and can be useful in search methods that require the evaluation of reliability indexes for a large number of test selections. Efficient deterministic algorithms are presented to find the optimal allocation of a given number of sectionalizing switches to minimize a selected performance/reliability index. The performance of the algorithms is assessed using the existing distribution networks of various sizes. It is shown that the proposed algorithms outperform the state-of-the-art approaches in terms of computational efficiency.**

Index Terms—**Power distribution reliability, protective device, sectionalizing switch, SAIDI, SAIFI, AENS.**

I. Introduction

REDUCING the frequency and duration of power interruptions to customers is one of the main objectives in the design of distribution networks [1]–[3]. Improving reliability and reduction of costs associated with power outages may be achieved by the introduction of automatic switching devices (sectionalizing switches).

In this work, we study the problem of optimal placement of sectionalizing switches in radial distribution networks to improve reliability and reduce power outage costs. We will consider the problems of minimization of three performance indexes: SAIDI (System Average Interruption Duration Index), SAIFI (System Average Interruption Frequency Index), and AENS (Average Energy Not Supplied). Several solutions to these problems have been proposed, including genetic algorithms [4], simulated annealing [5], immune algorithms [6], particle swarm optimization [7], and ant colony optimization-based methods [8]. These algorithms belong to the class of heuristic approaches and can be characterized by long computation times and no guarantee that the optimal solution has been found.

In [9], a sequential optimization algorithm using thinning techniques to reduce the search space is described. The authors claim that the proposed algorithm is capable of finding the optimal solution for real size networks. It will be shown that this algorithm may produce suboptimal results. A fuzzy dynamic programming approach is presented in [10]. Improving reliability in radial distribution systems with distributed generation is studied for example in [11], [12]. Sectionalizing strategy for parallel system restoration is discussed in [13], [14]. The problem of optimal switch placement considering switch malfunction is studied in [15].

The state-of-the-art methods to solve optimization problems related to optimal switch placement are based on integer programming. A binary programming based approach to optimize the SAIFI index is presented in [16]. In [17], mixed integer linear programming (MILP) is utilized for the minimization of customer outage costs in networks without and with alternative supply paths. Application of these techniques to systems with distributed generation is presented in [12]. Mixed integer nonlinear programming is used to solve the simultaneous optimal allocation of sectionalizing switches and protective devices in [18], [19]. In these approaches the switch placement problem is modeled as a MILP problem. Commercial solvers can solve the resulting optimization problems in a computationally efficient manner using for example the branch-and-bound algorithm. The number of test solutions which has to be considered in the branch and bound algorithm to ensure that the optimal solution is found grows exponentially with the complexity of the optimization problem. In consequence, optimization problems can be successfully solved only for networks of a limited size. Therefore, in many cases, additional constrains in the allowed positions of switches are introduced to improve the performance of MILP solvers. In practice, the number of possible switch locations and the maximum number of switches are usually less than 100 and 10, respectively. Additionally, economic constrains are often introduced to limit the search space.

In this work, we propose fast algorithms based on a tree structure of radial distribution networks. The algorithms use the dynamic programming approach. Partial solutions are found recursively starting from load nodes and proceeding towards the generator node. At each node all partial solutions which may

Manuscript received July 3, 2018; revised December 4, 2018 and January 27, 2019; accepted March 23, 2019. Date of publication May 2, 2019; date of current version October 24, 2019. This work was supported by the National Science Centre, Poland, under Grant 2014/15/B/ST8/02315. Paper no. TPWRS-01016-2018.

The author is with the Department of Electrical Engineering and Power Engineering, AGH University of Science and Technology, Kraków 30-059, Poland (e-mail: galias@agh.edu.pl).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TPWRS.2019.2909836

produce the optimal complete solution are remembered. The number of solutions, which are considered is reduced by introducing partial ordering of admissible solutions and eliminating partial solutions which cannot lead to the optimal solution. A version of this algorithm for the optimization of AENS is presented in [20]. Here, we present a more detailed description of the algorithm, extend it to optimize other performance indexes (SAIDI and SAIFI), and discuss its computational complexity. We also compare the performance of the algorithms with the state-of-the-art methods using existing distribution networks of various sizes as examples. We show that the proposed algorithms are orders of magnitudes faster than existing methods. In consequence, the optimization problem at hand can be efficiently solved for larger networks without the necessity to introduce additional constrains to the optimization problem. Another advantage of the proposed approach is that no commercial solvers are needed to find the optimal solution—the algorithms can be easily implemented in the C++ programming language or in the MATLAB environment.

The layout of the paper is as follows. In Section II, the problem is defined and proposed algorithms are presented in detail. In Section III, case studies are employed and high efficiency of the algorithms is confirmed. It is shown that the sequential optimization algorithm presented in [9] may produce suboptimal results. Using existing distribution networks of different sizes, it is shown that the proposed algorithms significantly outperform state-of-the-art methods.

II. Optimal Placement of Sectionalizing Switches

A. Problem Definition

We assume that the network has a *radial structure* and contains m connection lines. It follows that the number of nodes is $n = m + 1$. Let $V = \{v_1, v_2, \ldots, v_n\}$ denote the set of nodes. We assume that there is a single generator/supply node and m load and distribution nodes. To simplify the presentation, we assume that the index of the supply node is n. Each load node is connected to a single node. For each load node there exist a unique path connecting it to the supply node. It follows that the graph representation of the network has a tree structure, with the generator being the root of the tree and load nodes being leaves (nodes without children).

For $1 \le j \le m$ by c_j we denote the line segment between the node v_j and its parent node. Let λ_{v_j} and λ_{c_j} be the average failure rates (the average number of failures during the period of analysis; usually one year) of the node v_j and the line segment c_j, respectively. Let τ_{v_j} and τ_{c_j} be the average failure durations for the node v_j and the line segment c_j, respectively. The average total duration of failures of a given element can be computed as a product of the average failure rate λ and the average failure duration τ of this element, i.e. $t_{v_j} = \lambda_{v_j}\tau_{v_j}$ and $t_{c_j} = \lambda_{c_j}\tau_{c_j}$. Let us define $\lambda_j = \lambda_{v_j} + \lambda_{c_j}$ and $t_j = t_{v_j} + t_{c_j}$. Since the problems considered do not depend on failures of the supply node, we may assume that $\lambda_n = 0$ and $t_n = 0$.

Let $N_j \ge 0$ and $P_j \ge 0$ be the number of users and the average (active) power of the node v_j. In practice, $P_j > 0$ and $N_j > 0$ for load nodes and $P_j = N_j = 0$ for other nodes. The *total average power* is $\bar{P} = \sum_{i=1}^{n} P_i$, the *total failure rate* is $\bar{\lambda} = \sum_{i=1}^{n} \lambda_i$, and the *total duration of failures* is $\bar{t} = \sum_{i=1}^{n} t_i$.

The two most popular indexes used in reliability analysis of power networks are the System Average Interruption Frequency Index (SAIFI) and the System Average Interruption Duration Index (SAIDI) [1], [5].

SAIFI is the average number of interruptions during one year for a single user. It is defined as the total number of interruptions counted independently for each user divided by the *total number of users* $\bar{N} = \sum_{j=1}^{n} N_j$. SAIDI is the average outage duration. It is calculated as the sum of the durations of all interruptions counted independently for each user divided by $\bar{N}$. SAIFI and SAIDI are defined as

$$\mathrm{SAIFI} = \frac{\sum_{j=1}^{n} \mu_j N_j}{\sum_{j=1}^{n} N_j}, \quad \mathrm{SAIDI} = \frac{\sum_{j=1}^{n} U_j N_j}{\sum_{j=1}^{n} N_j}, \tag{1}$$

where $n = m + 1$ is the number of nodes, μ_j is the outage rate of the node v_j, i.e. the average number of interruptions involving the node v_j during one year, and U_j is the total duration of all interruptions involving the node v_j during one year.

The Average Energy Not Supplied (AENS) is defined as the average value of energy not supplied to users during the period of analysis due to failures

$$\mathrm{AENS} = \sum_{j=1}^{n} U_j P_j. \tag{2}$$

For a given network, coefficients SAIFI, SAIDI, and AENS can be reduced by introducing sectionalizing switches at selected line segments. If a failure occurs behind the switch we may disconnect a part of the grid and energy supply to the remaining part of the grid may be continued in spite of the fault. If there is a switch at the line segment c_j we will say for short that there is a switch at the position j. A switch at this position should be placed close to the parent of v_j. This choice guarantees that this switch can be activated for all failures involving the line segment c_j.

Let $Q = \{i_1, i_2, \ldots, i_p\} \subset I_m = \{1, 2, \ldots, m\}$ be a set of positions of sectionalizing switches. Let us denote by $\mathrm{AENS}(Q)$ the average energy not supplied if there are switches at the positions in the set Q. If there are no sectionalizing switches in the network ($Q = \emptyset$) then a failure at any location in the grid causes energy supply interruption in the entire network. In consequence, $U_j = \mathrm{const} = \sum_{i=1}^{n} t_i$ and

$$\mathrm{AENS}(\emptyset) = \sum_{i=1}^{n} P_i \sum_{j=1}^{n} t_j = \bar{P} \cdot \bar{t}. \tag{3}$$

The optimization problem is to find for a given $p \in \{1, 2, \ldots, m\}$ the minimum value of AENS which can be obtained using p sectionalizing switches

$$\mathrm{AENS}_{\min}(p) = \min_{Q:\#Q=p} \mathrm{AENS}(Q), \tag{4}$$

and the corresponding positions of switches, where $\#Q$ denotes the cardinality of Q.

Other optimization problems are defined in a similar way. Let $\mathrm{SAIFI}(Q)$ and $\mathrm{SAIDI}(Q)$ be the SAIFI and SAIDI indexes for the case of sectionalizing switches located at the positions in the set Q. With no switches we have $\mu_j = \text{const} = \sum_{i=1}^{n} \lambda_i$, $U_j = \text{const} = \sum_{i=1}^{n} t_i$, and in consequence

$$\text{SAIFI}(\emptyset) = \sum_{j=1}^{n} \lambda_j = \bar{\lambda}, \quad \text{SAIDI}(\emptyset) = \sum_{j=1}^{n} t_j = \bar{t}. \tag{5}$$

The optimization problems involving SAIFI and SAIDI indexes are to find for a given $p \in \{1, 2, \ldots, m\}$

$$\text{SAIFI}_{\min}(p) = \min_{Q:\#Q=p} \text{SAIFI}(Q), \tag{6}$$

$$\text{SAIDI}_{\min}(p) = \min_{Q:\#Q=p} \text{SAIDI}(Q), \tag{7}$$

and the corresponding positions of sectionalizing switches.

B. *Efficient Computation of* SAIFI, SAIDI, *and* AENS

Let us introduce several notions which are useful for the evaluation of SAIFI, SAIDI, and AENS. For $j \in I_m$ let us denote by C_j the set of indexes of children of v_j. By D_j we denote the set containing the index j and indexes of descendants of v_j. Let $\bar{P}_j$ denote the sum of average powers of the node v_j and its descendants, i.e. $\bar{P}_j = \sum_{i \in D_j} P_i$. In a similar way, we define the sum of average failure times $\bar{t}_j$, the sum of failure rates $\bar{\lambda}_j$ and the total number of users for the node v_j and its descendants: $\bar{t}_j = \sum_{i \in D_j} t_i$, $\bar{\lambda}_j = \sum_{i \in D_j} \lambda_i$, and $\bar{N}_j = \sum_{i \in D_j} N_i$.

Let us consider an arbitrary set $Q \subset I_m$. First, we present the tree-structure based algorithm for the computation of $\text{AENS}(Q)$. The switch at the position $j \in Q$ is active only if a failure occurs beyond this switch but not beyond other switches in $Q_j = Q \cap D_j$. It follows that the activation time for this switch can be computed as $\bar{t}_j - \sum_{i \in R_j} \bar{t}_i$, where R_j is the set of switches in Q_j, which can be reached from v_j without passing through another switch. When the switch at the position j is active, the part of the grid containing the node v_j and its descendants is switched off. It follows that due to the presence of a sectionalizing switch at the position j, AENS is decreased by the product of the activation time of this switch and the total power $(\bar{P} - \bar{P}_j)$ of load nodes which are active when the switch at the position j is active. Hence

$$\text{AENS}(Q) = \bar{P} \cdot \bar{t} - \sum_{j \in Q} \left(\bar{P} - \bar{P}_j\right) \left(\bar{t}_j - \sum_{i \in R_j} \bar{t}_i\right). \tag{8}$$

A recursive procedure to compute $\text{AENS}(Q)$ is presented as the Algorithm 1. To explain how the algorithm works let us define the notion of a partial solution. The *partial solution s generated by Q at the position j* involves switches located in the set D_j. $Q_s = Q \cap D_j$ is the set of switches in the partial solution s. For the partial solution s we define the *gain g_s* to be the gain in the average energy not supplied obtained by using the partial solution s

$$g_s = \bar{P} \cdot \bar{t} - \text{AENS}(Q_s) = \sum_{j \in Q_s} \left(\bar{P} - \bar{P}_j\right) \left(\bar{t}_j - \sum_{i \in R_j} \bar{t}_i\right). \tag{9}$$

Algorithm 1: Tree Algorithm to Compute $\text{AENS}(Q)$.

```
Precondition: $Q$ is the set of switch positions
 1: function visitNode($Q, j$)
 2:     $(g_s, a_s) \leftarrow (0, 0)$
 3:     for $i \in C_j$ do                         ▷ process children
 4:         $(g_r, a_r) \leftarrow$visitNode$(Q, i)$
 5:         $(g_s, a_s) \leftarrow (g_s + g_r, a_s + a_r)$
 6:     end for
 7:     if $j \in Q$ then
 8:         $g_s \leftarrow g_s + (\bar{P} - \bar{P}_j)(\bar{t}_j - a_s)$
 9:         $a_s \leftarrow \bar{t}_j$
10:     end if
11:     return $(g_s, a_s)$
12: end function
13: function AENS($Q$)
14:     $(g_s, a_s) \leftarrow$visitNode$(Q, m + 1)$
15:     return $\bar{P} \cdot \bar{t} - g_s$
16: end function
```

The *total activation time* a_s of the partial solution s is the total time for which switches in Q_s are active

$$a_s = \begin{cases} \bar{t}_j & \text{if } j \in Q_s \\ \sum_{i \in R_j} \bar{t}_i & \text{if } j \notin Q_s. \end{cases} \tag{10}$$

The Algorithm 1 recursively computes gains g_s and total activation times a_s for partial solutions starting from load nodes and moving towards the root node. The nodes are visited using the depth-first search (DFS) algorithm in which a given node is processed after all its children have been processed [21]. If $j \notin Q$ then the gain and the total activation time are simply the sums of children gains and total activation times (line 5). In the opposite case, the gain has to be increased by $(\bar{P} - \bar{P}_j)(\bar{t}_j - a_s)$ (line 8), while the total activation is assigned the value $\bar{t}_j$ (line 9). At the final step the gain at the root node (generator) is subtracted from $\text{AENS}(\emptyset)$ (line 15). If the total average power $\bar{P}$ is known, the computations can be completed in a single pass of the tree structure, and in consequence the algorithm is very fast.

Similarly as for $\text{AENS}(Q)$ one may derive the following formulas:

$$\text{SAIDI}(Q) = \bar{t} - \frac{1}{\bar{N}} \sum_{j \in Q} \left(\bar{N} - \bar{N}_j\right) \left(\bar{t}_j - \sum_{i \in R_j} \bar{t}_i\right), \tag{11}$$

$$\text{SAIFI}(Q) = \bar{\lambda} - \frac{1}{\bar{N}} \sum_{j \in Q} \left(\bar{N} - \bar{N}_j\right) \left(\bar{\lambda}_j - \sum_{i \in R_j} \bar{\lambda}_i\right). \tag{12}$$

The algorithms to evaluate $\text{SAIDI}(Q)$ and $\text{SAIFI}(Q)$ are similar to the Algorithm 1. To compute $\text{SAIDI}(Q)$ we have to replace line 8 in the Algorithm 1 by $g_s \leftarrow g_s + (\bar{N} - \bar{N}_j)(\bar{t}_j - a_s)$ and replace $\bar{P} \cdot \bar{t} - g_s$ by $\bar{t} - g_s/\bar{N}$ in line 15. To compute $\text{SAIFI}(Q)$ we have to replace line 8 by $g_s \leftarrow g_s + (\bar{N} - \bar{N}_j)(\bar{\lambda}_j - a_s)$, replace line 9 by $a_s \leftarrow \bar{\lambda}_j$ and replace $\bar{P} \cdot \bar{t} - g_s$ by $\bar{\lambda} - g_s/\bar{N}$ in line 15.

C. Fast Deterministic Algorithms to Optimize the Positions of Sectionalizing Switches

In this section, we present deterministic algorithms to minimize AENS, SAIFI, and SAIDI for a given number of sectionalizing switches. First, we present the algorithm to optimize AENS. Let $p_{\max}$ be the maximum number of sectionalizing switches to be considered. The general idea is to recursively construct partial solutions with $p \leq p_{\max}$ proceeding from load nodes up the tree structure. Partial solutions for a given node are constructed based on partial solutions found previously for its children. When the generator node is reached, we have solutions for the whole grid for all $p \leq p_{\max}$ and select the ones maximizing AENS.

Let us describe how to generate the set S_j of partial solutions at the position j. For each load node there are two partial solutions involving this node. The first one is the partial solution without a switch for which $Q_s = \emptyset$, $a_s = 0$, $g_s = 0$. The second one with $Q_s = \{j\}$, $a_s = \bar{t}_j$, $g_s = (\bar{P} - \bar{P}_j)\bar{t}_j$ is the partial solution with a switch at the position j.

Let us now consider an arbitrary node v_j. Let $C_j = \{i_1, i_2, \ldots, i_k\}$ be the set of indexes of children of v_j. Let us consider a selection $s_1 \in S_{i_1}$, $s_2 \in S_{i_2}$, ..., $s_k \in S_{i_k}$ of partial solutions for nodes in C_j. Using this selection, we can construct a partial solution at the position j in two ways. If we do not add a switch at the position j then $Q_s = \bigcup_{i=1}^k Q_{s_i}$, $a_s = \sum_{i=1}^k a_{s_k}$, and $g_s = \sum_{i=1}^k g_{s_k}$. If we add a switch at the position j then $Q_s = \left(\bigcup_{i=1}^k Q_{s_i}\right) \cup \{j\}$, $a_s = \bar{t}_j$, and $g_s = \sum_{i=1}^k g_{s_k} + (\bar{P} - \bar{P}_j)\left(\bar{t}_j - \sum_{i=1}^k a_{s_k}\right)$. The set of partial solutions S_j is obtained by considering all possible combinations $(s_1, s_2, \ldots, s_k)$ of partial solutions at positions in C_j. Note that this procedure can also be used to generate the set of partial solutions for user nodes (with $k = 0$). In this case we consider a single empty selection.

The idea described above is presented as the Algorithm 2. Data concerning a partial solutions s is stored as a triplet (g_s, a_s, Q_s). To find the best solution the procedure FINDPARTIALSOLUTIONS is called with $j = m + 1$ (line 25). On exit, the set S_{m+1} contains all solutions with the number of switches $p \leq p_{\max}$. The solution in the set $T_p = \{(g_s, a_s, Q_s) \in S_{m+1} : \#Q_s = p\}$ with the maximum value of g_s is the solution of the optimization problem (4) for the case of p switches.

The Algorithm 2 generates all partial solutions with the number of switches $p \leq p_{\max}$. This leads to an exponential growth of the number of partial solutions which has to be considered when we move up the tree structure. In consequence, the algorithm becomes unusable for larger networks. To reduce the computation time it is necessary to eliminate some partial solutions. One option is to store for each node v_j and for each $p \leq p_{\max}$ at most one partial solution at the position j with p switches. For the root node the maximum value of the gain g_s is equivalent to the minimum value of AENS. Hence, a natural choice is to select the partial solution maximizing g_s. This may however lead to elimination of the optimal solution because a partial solution with a lower gain g_s at a given node may have a larger contribution to the gain at the root node.

Algorithm 2: Find $\mathrm{AENS}_{\min}(p)$ for $p \leq p_{\max}$.

```
1:  procedure ADDPARTIALSOLUTION($g_s, a_s, Q_s, j, S_j$)
2:      if $\#Q_s \leq p_{\max}$ then
3:          $S_j \leftarrow S_j \cup \{(g_s, a_s, Q_s)\}$
4:      end if
5:  end procedure
6:  procedure FINDPARTIALSOLUTIONS($j, S_j$)
7:      $\{i_1, i_2, \ldots, i_k\} \leftarrow C_j$                 ▷ children of $j$
8:      for $\ell \leftarrow 1, k$ do                              ▷ process children
9:          FINDPARTIALSOLUTIONS($i_\ell, S_{i_\ell}$)
10:     end for
11:     $S_j \leftarrow \emptyset$
12:     repeat
13:         select $(g_{s_\ell}, a_{s_\ell}, Q_{s_\ell}) \in S_{i_\ell}$ for $\ell = 1, 2, \ldots, k$
14:         $g_s \leftarrow \sum_{\ell=1}^k g_{s_\ell}$             ▷ $g_s \leftarrow 0$ if $k = 0$
15:         $a_s \leftarrow \sum_{\ell=1}^k a_{s_\ell}$             ▷ $a_s \leftarrow 0$ if $k = 0$
16:         $Q_s \leftarrow \bigcup_{\ell=1}^k Q_{s_\ell}$          ▷ $Q_s \leftarrow \emptyset$ if $k = 0$
17:         ADDPARTIALSOLUTION($g_s, a_s, Q_s, j, S_j$)
18:         $g_s \leftarrow g_s + (\bar{P} - \bar{P}_j)(\bar{t}_j - a_s)$
19:         $a_s \leftarrow \bar{t}_j$
20:         $Q_s \leftarrow Q_s \cup \{j\}$
21:         ADDPARTIALSOLUTION($g_s, a_s, Q_s, j, S_j$)
22:     until no more selections
23: end procedure
24: procedure optimizeAENS($p_{\max}$)
25:     FINDPARTIALSOLUTIONS($m + 1, S_{m+1}$)
26:     for $p \leftarrow 1, p_{\max}$ do
27:         $T_p \leftarrow \{(g_s, a_s, Q_s) \in S_{m+1} : \#Q_s = p\}$
28:         select $(g_s, a_s, Q_s) \in T_p$ with the largest $g_s$
29:     end for
30: end procedure
```

Let us discuss how to identify partial solutions which cannot generate the optimal solution. Let us consider two partial solutions s' and s'' at the position j with the same number of switches $\#Q_{s'} = \#Q_{s''}$. We will show that if

$$g_{s'} \geq g_{s''}, \quad g_{s'} - g_{s''} \geq (\bar{P} - \bar{P}_j)(a_{s'} - a_{s''}) \tag{13}$$

then complete solutions constructed using s'' cannot be better than solutions constructed using s', and therefore the partial solution s'' can be eliminated.

Let r' and r'' be two complete solutions which are constructed using s' and s'', respectively. We assume that the solutions r' and r'' differ only within D_j. If for the solutions r' and r'' there is no switch between the node v_j and the root node then $g_{r'} - g_{r''} = g_{s'} - g_{s''} \geq 0$ and hence the solution r'' is not better. Now, let us assume that for the solutions r' and r'' there is a switch at the position i, v_j is a descendant of v_i and that there is no switch on the path from v_i to v_j. In this case the gain difference can be computed as $g_{r'} - g_{r''} = g_{s'} - g_{s''} - (\bar{P} - \bar{P}_i)(a_{s'} - a_{s''})$. It is clear that if $g_{s'} \geq g_{s''}$ and $a_{s'} \leq a_{s''}$ then $g_{r'} \geq g_{r''}$, and hence the solution r'' is not better. Let us now assume that $a_{s'} \geq a_{s''}$. Since $\bar{P}_i \geq \bar{P}_j$ we obtain $-(\bar{P} - \bar{P}_i)(a_{s'} - a_{s''}) \geq -(\bar{P} - \bar{P}_j)(a_{s'} - a_{s''})$ and $g_{r'} - g_{r''} = g_{s'} - g_{s''} - (\bar{P} - \bar{P}_i)(a_{s'} - a_{s''}) \geq g_{s'} - g_{s''} - (\bar{P} - \bar{P}_j)(a_{s'} - a_{s''}) \geq 0$. The last inequality follows

Algorithm 3: Add Partial Solution (An Improved Version).

```
 1: procedure ADDPARTIALSOLUTION($g_s, a_s, Q_s, j, S_j$)
 2:     if $\#Q_s > p_{\max}$ then                         ▷ too many switches
 3:         return
 4:     end if
 5:     $T \leftarrow \{(g_r, a_r, Q_r) \in S_j : \#Q_r = \#Q_s\}$
 6:     for $(g_r, a_r, Q_r) \in T$ do
 7:         if $g_s \geq g_r$ and $g_s - g_r \geq (\bar{P} - \bar{P}_j)(a_s - a_r)$ then
 8:             $S_j \leftarrow S_j \setminus \{(g_r, a_r, Q_r)\}$
 9:         end if
10:     end for
11:     $T \leftarrow \{(g_r, a_r, Q_r) \in S_j : \#Q_r = \#Q_s\}$
12:     for $(g_r, a_r, Q_r) \in T$ do
13:         if $g_s \leq g_r$ and $g_s - g_r \leq (\bar{P} - \bar{P}_j)(a_s - a_r)$ then
14:             return                                  ▷ skip $(g_s, a_s, Q_s)$
15:         end if
16:     end for
17:     $S_j \leftarrow S_j \cup \{(g_s, a_s, Q_s)\}$              ▷ add $(g_s, a_s, Q_s)$
18: end procedure
```

from (13). We have shown that if the conditions (13) are satisfied then s'' can be skipped.

The improved procedure to add partial solutions with the elimination of partial solutions, which cannot produce the optimal global solution is presented as the Algorithm 3.

The Algorithms 2 and 3 can be modified to optimize SAIDI and SAIFI. Changes in the Algorithm 2 are as follows. To optimize SAIDI one should replace line 18 by $g_s \leftarrow g_s + (\bar{N} - \bar{N}_j)(\bar{t}_j - a_s)$. To optimize SAIFI one should replace line 18 by $g_s \leftarrow g_s + (\bar{N} - \bar{N}_j)(\bar{\lambda}_j - a_s)$ and replace line 19 by $a_s \leftarrow \bar{\lambda}_j$. In the Algorithm 3 one should replace $(\bar{P} - \bar{P}_j)$ by $(\bar{N} - \bar{N}_j)$ for the optimization of both SAIDI and SAIFI.

The Algorithms 2 and 3 are guaranteed to find the optimal solution. In the Algorithm 2 each node is visited only once. The total number of partial solutions depends on the number of selections which are considered at each node. In the following section, we show that the Algorithm 3 significantly reduces the number of partial solutions which have to be considered.

III. Simulation Results

In this section, we carry out a feasibility study of the algorithms presented in Section II and compare the performance of the proposed method with existing methods. Case studies involve existing radial distribution networks of different sizes.

First, we consider several existing networks located in the southern part of Poland. Data for these networks provided by an electricity company includes topology of the network, lengths and types of line segments, average power and the number of users for each load node, and history of faults. From the history of failures occurring during a two years period, failure rates and average failure durations for different types of elements are computed. The average number of faults is 3.1 in one year for every 100 km of a line segment. The average failure rate of a given line segment c_j can be computed as $\lambda_{c_j} = 3.1 \times 10^{-5} l_j$, where l_j is the line segment's length in meters. The observed average fault duration is $\tau_{c_j} = 0.983\,\text{h}$. The average total duration of failures of the line segment c_j is $t_{c_j} = \tau_{c_j}\lambda_{c_j} = 0.983 \cdot 3.1 \times 10^{-5} l_j$. The average number of faults in one year is $\lambda_{v_j} = 0.03$ for user nodes and $\lambda_{v_j} = 0.002$ for distribution nodes. The average duration of the fault is $\tau_{v_j} = 1\,\text{h}$ and $\tau_{v_j} = 0.5\,\text{h}$ for user and distribution nodes, respectively.

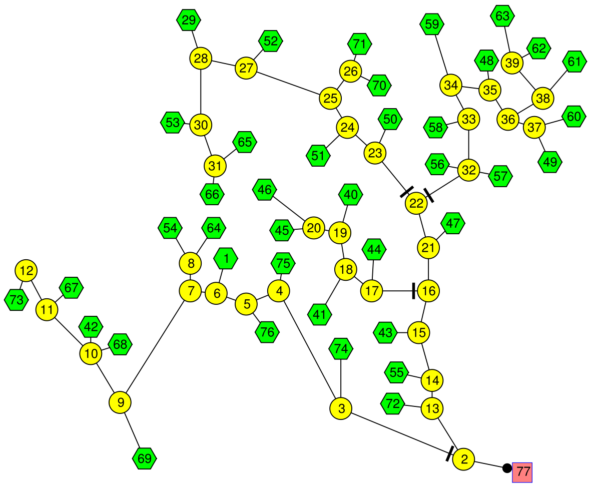

Fig. 1. An example distribution network with $m = 76$ line segments. The solution of the optimization problem $\text{AENS}_{\min}(4)$ is $Q = \{3, 17, 23, 32\}$.

We consider four example grids differing by network size m. The smallest network with $m = 76$ line segments is shown in Fig. 1. The supply node, distribution nodes, and user nodes are plotted as a red square, yellow circles, and green hexagons, respectively. Other networks contains 112, 199, and 262 line segments, respectively. All data defining these networks necessary to carry out the computations reported in this work can be found at http://www.zet.agh.edu.pl/power/pes2019. All computations are carried out using a single core 3.4 GHz processor.

First, let us test the speed of the Algorithm 1 for the evaluation of AENS, SAIDI, and SAIFI. We carry out the exhaustive search (ES) to find optimal positions of $p \leq p_{\max}$ sectionalizing switches. In the ES method all possible selections of p switches are considered to find the minimum value of a given performance index. The number $p_{\max}$ for a given network is selected in such a way that computations are completed in less than 10 hours. The results are presented in Table I. We report the number $N_{\text{ES}} = \sum_{p=0}^{p_{\max}} \binom{m}{p}$ of evaluations in the exhaustive search method, the computation time t_{ES} and the number $N_{\text{ES}}/t_{\text{ES}}$ of evaluations which are computed in one second. One can see that computation times for all performance indexes are similar. The algorithms are very fast. In one second, each performance index can be evaluated for 70000 to 240000 test selections depending on the network size. It follows that the Algorithm 1 may be useful in the ES method for a small number of switches or in heuristic methods where reliability indexes are evaluated for many test selections.

TABLE I
THE PERFORMANCE OF THE ALGORITHM TO COMPUTE AENS, SAIDI, AND SAIFI

m	$p_{\max}$	N_{ES}	index	t_{ES}	$N_{\mathrm{ES}}/t_{\mathrm{ES}}$
76	7	2424639382	AENS	10140.01	239116
			SAIDI	9942.83	243858
			SAIFI	10028.98	241763
112	6	2533006645	AENS	15464.03	163799
			SAIDI	15065.89	168128
			SAIFI	15185.49	166804
199	5	2536963640	AENS	27248.93	93103
			SAIDI	26342.63	96306
			SAIFI	26329.56	96354
262	4	194866169	AENS	2797.19	69664
			SAIDI	2742.01	71066
			SAIFI	2755.58	70716

Below, we assess the performance of the tree search method (TS) which uses Algorithms 2 and 3 for the optimization of AENS. Results obtained in the optimization of SAIFI and SAIDI are similar and are not reported for the sake of brevity. For comparison, we also present results obtained using the exhaustive search method (ES), the mixed integer linear programming (MILP) approach and the Celli-Pilo (CP) algorithm [9].

In order to use the MILP approach we need to formulate the optimization problem at hand as a mixed integer linear programming problem. The mixed integer linear programming is a general technique to solve optimization problems where some or all variables are restricted to be integers. Let us recall that the set D_j contains the index j and indexes of descendants of v_j. For $i \in D_j, i \neq j$ let us denote by E_{ij} the set of indexes of nodes belonging to the interior of the unique path from v_i to v_j. If v_j is a parent of v_i then $E_{ij} = \emptyset$. The MILP formulation of the optimization problem (4) is the following: minimize

$$\mathrm{AENS}(x) = \bar{P} \cdot \bar{t} - \sum_{j=1}^{m} x_j \left(\bar{P} - \bar{P}_j\right) \left(\bar{t}_j - \sum_{i \in D_j, i \neq j} x_i \bar{t}_i \prod_{k \in E_{ij}} (1 - x_k)\right) \tag{14}$$

under the conditions

$$\sum_{j=1}^{m} x_j = p, \quad x_j \in \{0, 1\} \text{ for } j \in \{1, 2, \ldots, m\}, \tag{15}$$

where p is the number of switches and $x = (x_1, x_2, \ldots, x_m) \in \{0,1\}^m$ represents positions of switches; $x_j = 1$ if there is a switch at the position j and $x_j = 0$ otherwise. The problem (14), (15) is a nonlinear binary programming problem with m variables and a single equality constrain. This problem can be transformed into a linear binary programming problem by introducing auxiliary variables. A nonlinear term $a x_i x_j \prod_{k \in E_{ij}} (1 - x_k)$ can be converted to a linear binary term by introducing an auxiliary variable $z = x_i x_j \prod_{k \in E_{ij}} (1 - x_k)$ and two inequality constrains $(2 + s) z \leq x_i + x_j + \sum_{k \in E_{ij}} (1 - x_k) \leq z + (s + 2) - 1$, where $s = \#E_{ij}$. The inequalities ensure that $z = 1$ if and only if $x_i = x_j = 1$ and $x_k = 0$ for all $k \in E_{ij}$. For each pair (j, i) with $1 \leq j \leq m$ and $i \in D_j, i \neq j$ we introduce a single auxiliary variable. Hence, the total number of auxiliary variables m_{aux} is equal to the total number of descendants counted independently for each node $m_{\mathrm{aux}} = \sum_{j=1}^{m} (\#D_j - 1)$ and the total number of inequality constrains is twice as much. In this way, we arrive at a binary linear programming problem with $m + m_{\mathrm{aux}}$ variables, a single equality constrain and $2m_{\mathrm{aux}}$ inequality constrains.

To test the performance of the integer linear programming approach the linear binary programming problem described above is written using a CPLEX LP file format and solved using the MILP solver CPLEX 12.7 [22]. The branch-and-bound algorithm is applied to efficiently solve MILP problems. The branch-and-bound algorithm divides the search space and generates a sequence of subproblems. First, the relaxed solution (without integer constrains) is found and then a sequence of solutions is generated to replace non-integer variables by integer ones. A non-integer variable (a branching variable) defines active nodes which are subproblems with additional constrains forcing a given variable to be integer. At a given step of the branch-and-bound algorithm an active node is selected, and the corresponding subproblem is solved. Integer feasible solutions (with integers variables) are used to cut off other active nodes. Branching continues until there are no active nodes and the best integer solution found is the solution of MILP. The number of nodes generated in the branch-and-bound algorithm can grow exponentially with the problem size. Several techniques (various heuristics, cutting planes, or preprocessing) are used to improve problem solvability.

The proposed tree search based algorithm uses a completely different approach. It is not formulated in the form of MILP, auxiliary variables are not created, relaxed solution are not computed and the concept of branching variables is not used. The algorithm belongs to the class of dynamic programming methods. Partial solutions are constructed proceeding from load nodes to the supply node. Note that there is no correlation between subproblems in the branch-and-bound algorithm and partial solutions in the proposed algorithm. The former are complete solutions of the (partially) relaxed MILP problem, while the latter are solutions of the optimization problem for a part of the network. Elimination of partial solutions allows to limit the search space without even generating complete solutions. This is the main advantage of the method which results in its efficiency. In the MILP approach the large number of auxiliary variables which have to be defined for large networks is one of the reasons why this approach fails. No auxiliary variables are defined in the TS method. Moreover, partial solutions constructed for a given p can be used to construct partial solutions for $p + 1$. For all these reasons, the proposed algorithm is much faster than the MILP approach.

Let us now briefly recall the Celli-Pilo (CP) algorithm to minimize AENS. The algorithm starts by computing AENS obtained by installing a single switch at a given node. This is done for all nodes. In this way we obtain the set S_1 of possible solutions at the first decision level. In the pth step ($p > 1$) all solutions from the step $p - 1$ are considered. For each solution $Q \in S_{p-1}$ solutions $Q \cup \{j\}$ with $j \notin Q$ are tested. The solution which minimizes AENS is added to the set of solutions S_p at the decision level p. The algorithm is stopped when p reaches $p_{\max}$. The solution $Q \in S_p$ at the decision level p which minimizes AENS is returned as the solution of the minimization problem (4) with

TABLE II
OPTIMIZATION OF AENS FOR THE GRID WITH $m = 76$ LINE SEGMENTS. RESULTS ARE REPORTED RELATIVE TO $\mathrm{AENS}(\emptyset) = 7202\,\mathrm{kWh}$

		ES	MILP	TS	CP	
p	best	t[s]	t[s]	t[s]	best	t[s]
0	**1.0000**	0.00	0.00	0.00	**1.0000**	0.00
1	**0.6640**	0.00	0.00	0.00	**0.6640**	0.00
2	**0.5080**	0.01	0.05	0.00	**0.5080**	0.03
3	**0.4219**	0.29	0.29	0.01	**0.4219**	0.03
4	**0.3439**	5.17	0.38	0.00	0.3567	0.03
5	**0.2936**	75.58	0.66	0.01	**0.2936**	0.03
6	**0.2621**	905.22	2.47	0.00	**0.2621**	0.03
7	**0.2477**	9153.74	4.11	0.01	**0.2477**	0.02
8	**0.2364**		7.67	0.01	**0.2364**	0.03
9	**0.2266**		17.70	0.01	**0.2266**	0.04
10	**0.2169**		13.43	0.01	0.2182	0.03
11	**0.2086**		37.04	0.01	0.2100	0.03
12	**0.2009**		86.61	0.01	0.2021	0.03
13	**0.1935**		106.68	0.01	0.1947	0.03
14	**0.1861**		307.68	0.01	0.1873	0.03
15	**0.1800**		235.03	0.02	0.1809	0.03
total			819.89	0.12		0.41

p sectionalizing switches. The CP algorithm can be viewed as m runs of a greedy algorithm, which at each decision level selects the position of a single sectionalizing switch to minimize AENS. Runs differ by the selection of the node in the first step. By construction, the algorithm is equivalent to the exhaustive search method for $p \le 2$. It will be shown that for $p \ge 3$ the CP algorithm may produce suboptimal results.

Results of solving the problem of optimal switch allocation in the network with $m = 76$ line segments using different algorithms are presented in Table II. The results are reported relative to the case with no switches ($\mathrm{AENS}(\emptyset) = 7202\,\mathrm{kWh}$). Optimal values are written in bold. The first three methods (ES, MILP, TS) always find the optimal solution. The CP method produces suboptimal results for $p = 4$ and for $p \ge 10$. The largest relative difference of 3.7% is observed for $p = 4$. In this case the optimal solution is $Q = \{3, 17, 23, 32\}$ (see Fig. 1), while the CP method finds the solution $Q = \{3, 13, 23, 32\}$.

Let us now compare the results in terms of the computation time. For the TS and CP methods, we report the incremental computation time, i.e., the computation time needed to solve the problem for $p + 1$ using solutions for p. The exhaustive search method works for $p \le 7$. For $p = 7$ the number of test selections is above $2 \cdot 10^9$ and the computation time is approximately 2.5 hours. The MILP approach is much faster. For $p = 7$ the optimal objective function value is found after performing 24401 (dual simplex) iterations. The computation time is reduced to approximately 4 seconds, i.e. by the factor of 2000. The TS method is several orders of magnitude faster than the MILP approach. Computations for all $p \le 15$ are completed in 0.12 second. During the computations 12974 partial solutions are constructed. The heuristic CP method requires evaluation of the objective function for 68493 test selections and is slower than the TS algorithm.

The results for networks with $m = 112, 199, 262$ line segments are reported in Table III. Conclusions are similar. The ES method works only for small p. The MILP method fails to find solutions for $m = 199, 262$ and $p > 10$ in a reasonable time. The TS algorithm finds optimal solution for the largest network for all $p \le 15$ is less than 3 minutes. For $m = 76, 112$ this method

TABLE III
OPTIMIZATION OF AENS FOR THE GRID WITH $m = 112, 199, 262$ LINE SEGMENTS. RESULTS ARE REPORTED RELATIVE TO $\mathrm{AENS}(\emptyset)$

		ES	MILP	TS	CP	
p	best	t[s]	t[s]	t[s]	best	t[s]
m = 112, AENS(∅) = 15958 kWh						
0	**1.0000**	0.00	0.00	0.00	**1.0000**	0.00
1	**0.7422**	0.00	0.03	0.00	**0.7422**	0.00
2	**0.6030**	0.04	0.11	0.00	**0.6030**	0.09
3	**0.4677**	1.36	0.47	0.01	**0.4677**	0.09
4	**0.3837**	37.22	0.63	0.00	**0.3837**	0.07
5	**0.3307**	809.74	2.00	0.01	**0.3307**	0.08
6	**0.3080**	14615.67	17.28	0.01	**0.3080**	0.08
7	**0.2877**		74.27	0.02	**0.2877**	0.08
8	**0.2697**		397.00	0.01	0.2729	0.08
9	**0.2555**		840.83	0.03	0.2587	0.08
10	**0.2442**		1528.23	0.01	0.2462	0.08
11	**0.2342**		1186.68	0.02	0.2350	0.08
12	**0.2250**		1537.79	0.02	**0.2250**	0.09
13	**0.2172**		2822.91	0.02	**0.2172**	0.08
14	**0.2100**		1343.70	0.03	**0.2100**	0.09
15	**0.2034**		7659.61	0.03	**0.2034**	0.09
total			17411.63	0.22		1.16
m = 199, AENS(∅) = 58226 kWh						
0	**1.0000**	0.00	0.00	0.00	**1.0000**	0.00
1	**0.6973**	0.01	0.05	0.00	**0.6973**	0.01
2	**0.4579**	0.21	0.17	0.01	**0.4579**	0.45
3	**0.3649**	13.70	0.33	0.01	**0.3649**	0.41
4	**0.3039**	686.28	4.51	0.02	**0.3039**	0.41
5	**0.2695**	26548.74	51.83	0.04	**0.2695**	0.42
6	**0.2427**		243.96	0.06	**0.2427**	0.42
7	**0.2169**		766.16	0.11	**0.2169**	0.42
8	**0.2006**		3610.68	0.16	**0.2006**	0.43
9	**0.1867**		15152.67	0.27	**0.1867**	0.42
10	**0.1728**		57723.77	0.41	0.1732	0.43
11	**0.1594**		104197.92	0.70	**0.1594**	0.43
12	**0.1509**			0.92	**0.1509**	0.44
13	**0.1447**			1.15	**0.1447**	0.43
14	**0.1391**			1.50	**0.1391**	0.44
15	**0.1340**			1.93	**0.1340**	0.44
total				7.29		6.00
m = 262, AENS(∅) = 105636 kWh						
0	**1.0000**	0.00	0.00	0.00	**1.0000**	0.00
1	**0.7461**	0.01	0.07	0.01	**0.7461**	0.01
2	**0.5469**	0.49	0.20	0.01	**0.5469**	0.97
3	**0.3524**	42.39	0.21	0.02	**0.3524**	0.93
4	**0.2810**	2754.31	0.22	0.04	**0.2810**	0.94
5	**0.2396**		16.07	0.13	**0.2396**	0.94
6	**0.2201**		120.86	0.36	**0.2201**	0.96
7	**0.2012**		1050.88	0.86	**0.2012**	0.96
8	**0.1864**		4168.30	1.88	**0.1864**	0.96
9	**0.1730**		17657.33	3.68	**0.1730**	0.96
10	**0.1636**		88098.86	6.65	**0.1636**	0.97
11	**0.1547**			11.11	**0.1547**	0.97
12	**0.1470**			17.61	**0.1470**	0.98
13	**0.1393**			26.73	**0.1393**	0.98
14	**0.1319**			38.75	0.1322	0.98
15	**0.1259**			54.98	0.1262	0.98
total				162.82		13.49

is faster than the CP method, while for $m = 199, 262$ the CP method is faster.

In several cases the CP method finds a suboptimal solution, for example when $m = 112$ and $8 \le p \le 11$. However, even in cases when the algorithm fails to find the global minimum, the quality of the solution is usually high. We conclude that the CP algorithm is a fast heuristic method usually finding high quality solutions of the optimization problems considered.

Let us now discuss the computational complexity of the algorithms. The number of test selection, which should be considered to solve the optimization problem with m line segments and p switches using the ES method is $N_{\mathrm{ES}} = \binom{m}{p}$. N_{ES} grows very fast with p for large m. In consequence, the ES method is useful for small p only.

TABLE IV
THE NUMBER OF PARTIAL SOLUTIONS N_{PS} FOR $p_{\max} = 15$

m	76	112	199	262
N_{PS}	12974	24720	1291746	26223891
$n_{\max}$	8	13	13	13

For the CP algorithm, at each decision level the number of solutions is limited by m. In the first step m solutions are considered. In subsequent steps, for each solution with p switches no more than $m-p$ test solutions are compared. Hence, the number of evaluations is limited by $N_{\mathrm{CP}} \leq m + \sum_{p=1}^{p_{\max}-1} m(m-p) < m^2 p_{\max}$ and the computational complexity is of the order $m^3 p_{\max}$ (compare Tables II and III).

Let us now consider the TS method. In the best possible scenario for each node we store only a single partial solution with p switches. In this case, the number of partial solutions, which are stored at the node v_i is not larger than $\max(p_{\max}, \#D_i)$. Hence, the total number of partial solutions N_{PS} is bounded by $N_{\mathrm{PS}} \leq \sum_{j=1}^{m} \prod_{i \in C_j} (\max(p_{\max}, \#D) + 1) \leq \sum_{j=1}^{m} (p_{\max} + 1)^{\#C_j}$, where $\#C_j$ is the number of children of the node v_j. In case of a binary tree ($\#C_j \leq 2$) we obtain the bound $N_{\mathrm{PS}} \leq m(p_{\max} + 1)^2$.

Let us now assume that the network has a structure of a complete binary tree, i.e. the number of nodes $n = m + 1$ is equal to $2^{h+1} - 1$, where h is the height of the tree (the length of the longest path from the root node to a leaf). Let us assume that leaf nodes are at the height 0, and the root node is at the height h. At a given height $k \in \{0, 1, \ldots, h\}$ there are 2^{h-k} nodes. At each of them we need to consider $2 \cdot 2^k \cdot 2^k$ selections. Hence, the total number of selections to be considered is $\sum_{k=0}^{h} 2^{1+2k+h-k} = 2^{h+1} \sum_{k=0}^{h} 2^k \leq 2^{2(h+1)} = (m+2)^2$. It follows that the complexity of the algorithm for $p_{\max} = m$ is quadratic in m. This bound is valid for networks having a structure of a complete binary tree under the assumption that the maximum number of partial solutions with a fixed number of switches which are stored at a given node is $n_{\max} = 1$.

The number of partial solutions N_{PS} generated during the optimization of AENS and the corresponding value of $n_{\max}$ is reported in Table IV. The rate of growth of N_{PS} is faster than m^2. This is caused by the fact that the grid considered is not a binary tree and is also related to the necessity of using $n_{\max} > 1$. However, the number of partial solutions is still very low when compared with N_{ES}. Recall that without elimination of partial solutions in the Algorithm 3 the number of complete solutions in the TS method is equal to N_{ES}. Obtained results confirm that the Algorithm 3 is very successful in limiting the search space.

To test the performance of the methods for larger networks let us consider the IEEE 8500 Node Test Feeder [23], [24]. The feeder data is available at http://sites.ieee.org/pes-testfeeders/resources/. This test feeder is a large radial system with 170 km of connection lines obtained from a real US distribution network. It is published in two versions, with balanced and with unbalanced secondaries. We consider the former version, in which the network contains 3637 nodes including 1177 user nodes. Relative coordinates of buses (nodes) are included in the feeder data. The network is shown in Fig. 2.

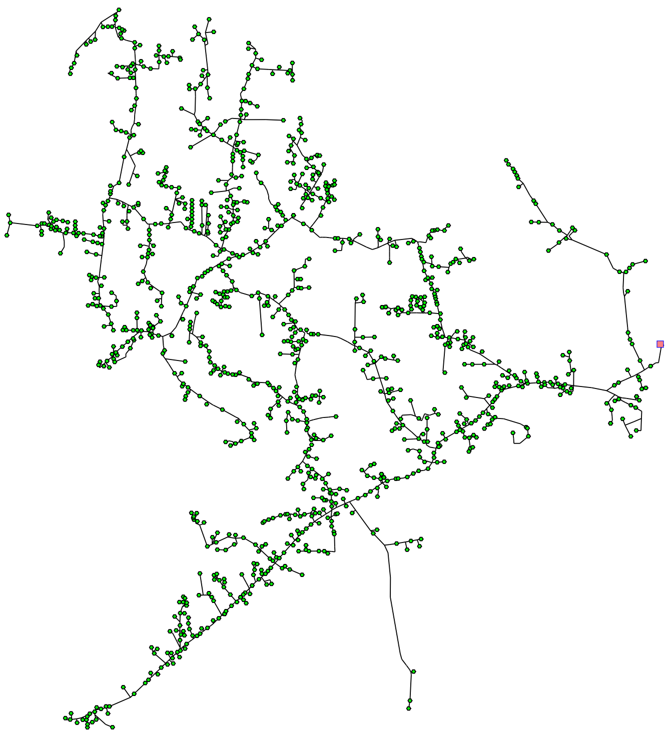
Fig. 2. The test distribution network with $m = 3637$ nodes.

Reliability data is not provided in the feeder data. However, the data contains information on line segment lengths. We assume that the failure rate is 5 failures per yer per 100 km of a line segment. Under this assumption the average failure rate for a line segment c_j can be computed as $\lambda_{c_j} = 5 \times 10^{-5} l_j$ where l_j is the length in meters of the line segment c_j. We assume that the average failure duration is $1\,\mathrm{h}$.

The whole network contains $m = 3636$ line segments, which is the size of the search space. Apart from the supply node the network contains 1387 single child nodes. Most of these nodes are transformers connected directly to user nodes. Note that for a single feeder network it does not make sense to put a sectionalizing switch in a line segment connecting a single child node to its only child. It is always more efficient to put a sectionalizing switch in the line segment connecting a single child node to its parent. This property permits reducing the search space. An equivalent network with a smaller number of nodes is obtained by skipping all single child nodes and merging reliability data of single child nodes with reliability data of their children. More specifically, if v_j is a single child node and v_i is its only child we set $\lambda_i = \lambda_i + \lambda_j$ and $t_i = t_i + t_j$. Applying this procedure to the network considered leads to a network with 2249 line segments.

For test purposes we also generate three smaller networks. We select a subnetwork containing 200 user nodes and paths connecting them to the supply node. After removing single child nodes, we obtain a network with 383 line segments. In a similar way starting from 300 and 600 user nodes, we obtain test networks with 567, and 1148 line segments, respectively.

TABLE V
OPTIMIZATION OF AENS FOR THE GRIDS WITH $m = 383, 567, 1148$ LINE SEGMENTS. RESULTS ARE REPORTED RELATIVE TO AENS($\emptyset$)

		ES	MILP	TS	CP	
p	best	t[s]	t[s]	t[s]	best	t[s]
			m = 383			
0	**1.0000**	0.00	0.04	0.00	**1.0000**	0.00
1	**0.7571**	0.01	0.49	0.01	**0.7571**	0.01
2	**0.6281**	1.34	6.56	0.02	**0.6281**	2.63
3	**0.5219**	178.74	12.67	0.04	**0.5219**	2.62
4	**0.4610**	17968.37	132.56	0.08	**0.4610**	2.65
5	**0.4101**		1030.56	0.12	**0.4101**	2.73
6	**0.3798**		7693.87	0.12	**0.3798**	2.81
7	**0.3627**			0.13	**0.3627**	2.84
8	**0.3471**			0.15	**0.3471**	2.76
9	**0.3324**			0.18	**0.3324**	2.90
10	**0.3178**			0.22	**0.3178**	2.92
11	**0.3046**			0.31	**0.3046**	2.91
12	**0.2926**			0.25	**0.2926**	3.12
13	**0.2818**			0.26	**0.2818**	2.93
14	**0.2717**			0.30	**0.2717**	2.95
15	**0.2622**			0.32	**0.2622**	2.96
total				2.51		39.74
			m = 567			
0	**1.0000**	0.00	0.09	0.00	**1.0000**	0.00
1	**0.7414**	0.02	0.31	0.01	**0.7414**	0.02
2	**0.5224**	4.46	30.80	0.03	**0.5224**	9.17
3	**0.4564**	899.92	38.79	0.09	**0.4564**	9.79
4	**0.4097**		7950.53	0.16	**0.4097**	9.26
5	**0.3694**		88377.09	0.17	0.3712	9.73
6	**0.3339**			0.20	0.3356	9.54
7	**0.3039**			0.26	0.3057	9.38
8	**0.2820**			0.28	0.2837	9.92
9	**0.2691**			0.35	0.2709	9.45
10	**0.2587**			0.40	0.2604	9.40
11	**0.2484**			0.41	0.2501	9.53
12	**0.2392**			0.48	0.2409	9.51
13	**0.2303**			0.50	0.2321	9.89
14	**0.2214**			0.56	0.2232	10.12
15	**0.2129**			0.68	0.2147	9.99
total				4.58		134.70
			m = 1148			
0	**1.0000**	0.00	0.38	0.00	**1.0000**	0.00
1	**0.7315**	0.07	1.24	0.02	**0.7315**	0.07
2	**0.5116**	39.30	319.05	0.07	**0.5116**	79.89
3	**0.4473**	16413.57	395.63	0.14	**0.4473**	79.90
4	**0.3907**		32477.53	0.25	**0.3907**	80.24
5	**0.3519**			0.44	**0.3519**	80.72
6	**0.3204**			0.53	**0.3204**	80.79
7	**0.2978**			0.48	0.2993	81.27
8	**0.2766**			0.61	0.2818	81.43
9	**0.2592**			0.78	0.2689	81.14
10	**0.2445**			0.89	0.2527	81.22
11	**0.2317**			1.07	0.2410	114.37
12	**0.2197**			1.22	0.2298	94.86
13	**0.2102**			1.33	0.2209	81.33
14	**0.2028**			1.47	0.2127	81.48
15	**0.1959**			1.50	0.2045	102.40
total				10.80		1201.11

First, let us compare the performance of optimization methods for networks with $m = 383, 567, 1148$ line segments. Optimization results obtained for $p \leq 15$ using different optimization methods are presented in Table V. Calculations are stopped when the computation time exceeds 24 hours. The first three methods always produce the optimal result provided that the computations are completed. The CP method produces optimal results for all $p \leq 15$ for the smallest network. For $m = 567$ and $m = 1148$ the CP method fails to find the optimal result for $p > 4$ and for $p > 6$, respectively. The relative difference between the value of the cost function for the CP solution and the optimal value is below 1% for $m = 567$ and belongs to $[0.5\%, 5.1\%]$ for $m = 1148$ and $p \geq 7$. The largest relative difference is obtained for $p = 13$.

TABLE VI
OPTIMIZATION OF AENS FOR THE GRIDS WITH $m = 2249, 3636$ LINE SEGMENTS. RESULTS ARE REPORTED RELATIVE TO AENS($\emptyset$)

		ES	MILP	TS	CP	
p	best	t[s]	t[s]	t[s]	best	t[s]
			m = 2249			
0	**1.0000**	0.00	1.74	0.00	**1.0000**	0.00
1	**0.7452**	0.30	5.31	0.06	**0.7452**	0.30
2	**0.5223**	472.29	7011.63	0.17	**0.5223**	677.05
3	**0.4313**		7265.61	0.31	**0.4313**	668.12
4	**0.3740**			0.57	**0.3740**	668.45
5	**0.3376**			0.76	**0.3376**	668.70
6	**0.3086**			0.99	0.3122	671.35
7	**0.2832**			1.17	0.2931	670.17
8	**0.2641**			1.40	0.2764	732.09
9	**0.2479**			1.73	0.2577	698.32
10	**0.2332**			1.96	0.2415	697.37
11	**0.2209**			2.33	0.2292	695.41
12	**0.2089**			2.62	0.2172	692.92
13	**0.2018**			3.04	0.2054	693.14
14	**0.1948**			3.53	0.1985	701.97
15	**0.1881**			3.92	0.1922	857.81
total				24.56		9793.17
			m = 3636			
0	**1.0000**	0.00	11.55	0.01	**1.0000**	0.00
1	**0.7452**	1.16	33.59	0.09	**0.7452**	1.15
2	**0.5223**	1754.65		0.19	**0.5223**	3391.57
3	**0.4313**			0.36	**0.4313**	4461.89
4	**0.3740**			0.64	**0.3740**	4907.68
5	**0.3376**			0.94	**0.3376**	3945.53
6	**0.3086**			1.30	0.3122	3519.35
7	**0.2832**			1.65	0.2931	3464.46
8	**0.2641**			2.06	0.2764	3509.38
9	**0.2479**			2.69	0.2577	3533.27
10	**0.2332**			3.28	0.2415	3262.35
11	**0.2209**			3.92	0.2292	3128.82
12	**0.2089**			4.60	0.2172	3147.16
13	**0.2018**			5.37	0.2054	3157.73
14	**0.1948**			6.10	0.1985	3575.71
15	**0.1881**			7.08	0.1922	4143.55
total				40.28		51149.60

It all cases the TS method is the fastest. The CP method is slower and the difference grows with m. For example, the CP method is 15 times slower than the TS method for $m = 383$ and 110 times slower for $m = 1148$. The ES method solves the problem for $p \leq 4$ in the first case and for $p \leq 3$ in the remaining cases. The MILP method find the optimal solutions with $p \leq 6$, $p \leq 5$, and $p \leq 4$ sectionalizing switches for $m = 383, 567, 1148$, respectively. Let us note that for $p \leq 2$ the computation time for the MILP method is longer than for the ES method. This is related to the large number of auxiliary variables $m_{\text{aux}} = 11306, 22104, 63202$ which are used in the MILP method for networks with $m = 383, 567, 1148$ line segments, respectively. The number of auxiliary variables is independent on p. In consequence, the computations are slowed down even for small p.

Finally, let us consider the whole network. We study two cases: the simplified network with $m = 2249$ line segments obtained by removing single child nodes and the original network with $m = 3636$. The results are presented in Table VI. Let us note that the results obtained for both networks are the same. This confirms that the simplification procedure does not alter the optimization problem.

The TS method is significantly faster than other methods. The total computation time is 24.56 seconds and 40.28 seconds for $m = 2249$ and $m = 3636$, respectively. In this case reducing the search space decreases the computation time by only 40%. For other methods removing single child nodes significantly reduces the computation time. The CP method is 400 times slower than the TS method for $m = 2249$ and 1250 times slower for $m = 3636$. The ES method and the MILP method solve the problem only for very small values of p. Note that in most cases the MILP method is slower than the ES method. The reason is the huge number of auxiliary variables $m_{\mathrm{aux}} = 182167$ for $m = 2249$ and $m_{\mathrm{aux}} = 587807$ for $m = 3636$.

The CP method finds optimal solutions for $p \leq 5$. The relative difference between the value of the cost function for the solutions found by the CP method and the optimal values belong to $[1.2\%, 4.7\%]$ for $p \geq 6$.

Similar computations have been carried out for the network with randomized values of line segments' lengths and user nodes' active powers. The results are similar which confirms that the worse performance of the MILP approach is not related to the symmetry of the network.

From the results presented above it follows that the proposed method can successfully solve the optimization problem even for very large networks with several thousands of nodes. The MILP approach cannot handle such networks unless the number of switches is very small. The CP method does not always find the optimal solution, contrary to what is stated in [9]. For the networks considered the difference between the value of the cost function for the solution found by the CP algorithm and for the optimal solution is up to 5%. The computation time for the CP method is significantly larger than for the proposed approach.

IV. Conclusion

Efficient algorithms for the computation of reliability indexes for radial distribution networks with sectionalizing switches have been presented. The algorithms are based on the tree structure of the graph representing the network. The algorithms are very efficient, which makes them useful for solving switch allocation problems via the exhaustive search method for small number of switches and heuristic methods which require handling a large number of test selections.

Fast tree search algorithms to solve the switch allocation problem in radial distribution networks with a single supply node have been presented. Algorithms have been tested using existing distribution networks of various sizes. It has been shown that the proposed algorithms outperform state-of-the-art techniques and can be successfully used to find optimal placement of switches for relatively large networks.

Future work includes extending algorithms to networks with alternative supplies, distributed generation and multiobjective optimization.

References

[1] H. L. Willis, *Power Distribution Planning Reference Book*. Boca Raton, FL, USA: CRC Press, 2004.

[2] J. Savier and D. Das, "Impact of network reconfiguration on loss allocation of radial distribution systems," *IEEE Trans. Power Del.*, vol. 22, no. 4, pp. 2473–2480, Oct. 2007.

[3] A. Zidan *et al.*, "Fault detection, isolation, and service restoration in distribution systems: State-of-the-art and future trends," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2170–2185, Sep. 2017.

[4] G. Levitin and S. Mazal-Tov, "Optimal sectionalizer allocation in electric distribution systems by genetic algorithm," *Elect. Power Syst. Res.*, vol. 35, no. 3, pp. 149–155, 1995.

[5] R. Billinton and S. Jonnavithula, "Optimal switching device placement in radial distribution systems," *IEEE Trans. Power Del.*, vol. 11, no. 3, pp. 1646–1651, Jul. 1996.

[6] C.-S. Chen, C.-H. Lin, H.-J. Chuang, C.-S. Li, M.-Y. Huang, and C.-W. Huang, "Optimal placement of line switches for distribution automation systems using immune algorithm," *IEEE Trans. Power Syst.*, vol. 21, no. 3, pp. 1209–1217, Aug. 2006.

[7] A. Moradi and M. Fotuhi-Firuzabad, "Optimal switch placement in distribution systems using trinary particle swarm optimization algorithm," *IEEE Trans. Power Del.*, vol. 23, no. 1, pp. 271–279, Jan. 2008.

[8] H. Falaghi, M.-R. Haghifam, and C. Singh, "Ant colony optimization-based method for placement of sectionalizing switches in distribution networks using a fuzzy multi objective approach," *IEEE Trans. Power Del.*, vol. 24, no. 1, pp. 268–276, Jan. 2009.

[9] G. Celli and F. Pilo, "Optimal sectionalizing switches allocation in distribution networks," *IEEE Trans. Power Del.*, vol. 14, no. 3, pp. 1167–1172, Jul. 1999.

[10] A. Esteban and A. Alberto, "Optimal selection and allocation of sectionalizers in distribution systems using fuzzy dynamic programming," *Energy Power Eng.*, vol. 2, pp. 283–290, 2010.

[11] Y. Mao and K. N. Miu, "Switch placement to improve system reliability for radial distribution systems with distributed generation," *IEEE Trans. Power Syst.*, vol. 18, no. 4, pp. 1346–1352, Nov. 2003.

[12] A. Heidari, V. G. Agelidis, and M. Kia, "Considerations of sectionalizing switches in distribution networks with distributed generation," *IEEE Trans. Power Del.*, vol. 30, no. 3, pp. 1401–1409, Jun. 2015.

[13] J. Quirs-Torts, M. Panteli, P. Wall, and V. Terzija, "Sectionalising methodology for parallel system restoration based on graph theory," *IET Gener. Transmiss. Distrib.*, vol. 9, no. 11, pp. 1216–1225, 2015.

[14] L. Sun *et al.*, "Network partitioning strategy for parallel power system restoration," *IET Gener. Transmiss. Distrib.*, vol. 10, no. 8, pp. 1883–1892, 2016.

[15] M. Farajollahi, M. Fotuhi-Firuzabad, and A. Safdarian, "Optimal placement of sectionalizing switch considering switch malfunction probability," *IEEE Trans. Smart Grid*, vol. 10, no. 1, pp. 403–413, Jan. 2019.

[16] F. Soudi and K. Tomsovic, "Optimized distribution protection using binary programming," *IEEE Trans. Power Del.*, vol. 13, no. 1, pp. 218–224, Jan. 1998.

[17] A. Abiri-Jahromi, M. Fotuhi-Firuzabad, M. Parvania, and M. Mosleh, "Optimized sectionalizing switch placement strategy in distribution systems," *IEEE Trans. Power Del.*, vol. 27, no. 1, pp. 362–370, Jan. 2012.

[18] Z. Popovic, B. Brbaklic, and S. Knezevic, "A mixed integer linear programming based approach for optimal placement of different types of automation devices in distribution networks," *Elect. Power Syst. Res.*, vol. 148, pp. 136–146, 2017.

[19] A. Heidari, Z. Y. Dong, D. Zhang, P. Siano, and J. Aghaei, "Mixed-integer nonlinear programming formulation for distribution networks reliability optimization," *IEEE Trans. Ind. Inform.*, vol. 14, no. 5, pp. 1952–1961, May 2018.

[20] Z. Galias, "On optimum placement of sectionalizing switches in radial distribution networks," in *Proc. IEEE Int. Symp. Circuits Syst.*, Baltimore, MD, USA, May 2017, pp. 1030–1033.

[21] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA, USA, MIT Press, 2001.

[22] *IBM ILOG CPLEX Optimization Studio CPLEX User's Manual, Version 12 Release 7*, IBM Corporation, 2017.

[23] R. F. Arritt and R. C. Dugan, "The IEEE 8500-node test feeder," in *Proc. IEEE Transm. Distrib. Conf. Expo.*, New Orleans, LA, USA, 2010, pp. 1–6.

[24] K. P. Schneider *et al.*, "Analytic considerations and design basis for the IEEE distribution test feeders," *IEEE Trans. Power Syst.*, vol. 33, no. 3, pp. 3181–3188, May 2018.