OPTIMAL CONFIGURATION OF ELECTRICAL DISTRIBUTION NETWORKS USING HEURISTIC METHODS

R. Cherkaoui, A. Bart, A. J. Germond
Laboratoire de Réseaux d’Energie Electrique
Ecole Polytechnique Fédérale de Lausanne
1015 Lausanne, Suisse

Abstract - The search for an optimal configuration for a distribution network can be viewed by three different approaches based on the implicit enumeration of solutions. The approach called branch exchange seems to avoid the disadvantages of the two others. This approach has allowed us to imagine and develop, initially, two descent search strategies. Thereafter, to improve these strategies, they have been transformed to include the principles of an optimization method, called TABU. All strategies have finally been tested in the case of a real network to find a feasible configuration with minimal losses.

Keywords: Optimization, Descent search, Tabu search, Distribution systems.

INTRODUCTION

Distribution automation [1-3] permits not only to closely and continuously monitor the operation of a power distribution system, but also to extend its operating limits by offering a wide range of possible solutions for the problems that might arise. Also, it would be appropriate to take advantage of distribution automation to identify the solutions best suited to criteria and constraints specified by operators, in order to continually maintain a safe and, if possible, an economically optimal operation policy.

The paper analyses heuristic strategies to determine, among several possible arborescent configurations of a distribution network, the one that represents a practical optimum for a specified set of criteria and constraints.

After the mathematical formulation of the problem which will be given in the next section, we will review the different existing heuristic resolution approaches based on the implicit enumeration of solutions. For the particular case of the branch exchange approach, new search strategy methods, which constitute the original contribution of this paper, have been developed. Each of them will be described in details and illustrated by means of numerical examples, making it possible to compare not only the quality of results but also their performance. Indeed, to solve a reconfiguration problem by means of an heuristic process, it is not sufficient to assess the efficiency of one method only, but it is necessary to extend the assessment to more methods.

GENERAL MATHEMATICAL FORMULATION

Graphically, the structure of a distribution network is represented with a graph denoted \( G(X,B) \) that consists of a node set \( X=\{x_1,x_2,x_3,\ldots, x_N\} \) and a branch set \( B=\{b_1,b_2,b_3,\ldots,b_M\} \). Every node represents either a network supply point via a HV/MV transformer (source node) or a consumer point; every branch represents either an overhead line or an underground cable.

We define a set \( Y \) of state variables representing branch currents and node voltages. A set \( U \) of decision variables \((0/1)\) representing topological states of branches (off/on) is also defined. Our optimization task is to determine the components of \( Y \) and \( U \) in order to minimize \( f(Y,U) \) (objective function) under the constraints:

\[
g(Y,U) = 0 \quad \text{Kirchhoff equations}
\]

\[
h(Y,U) \leq 0 \quad \text{security constraints}
\]

\[
r(U) = 1 \quad \text{arborescence constraints}
\]

Security constraints indicate that no current of the branch must exceed an acceptable maximal limit and each voltage must be kept within the specified limits. Concerning arborescence constraints, the sum of the decision variables must be equal to the total number of nodes minus the number of source nodes. This condition is necessary but not sufficient to comply with the arborescence of the searched configuration.

The objective function of the problem might be composed of a unique criterion or of a linear combination of criteria. The choice of criteria depends on the operating mode of the network. For example, in the normal mode one can choose the loss criterion, while in the emergency mode it would be the load balancing one.

PRESENTATION OF THE SOLUTION APPROACHES

This problem may be apprehended using separation between decision and state variables. This separation indicates that a solution which is candidate for the feasible and optimal one is completely defined as soon as the assigned value for each decision variable is known. In that way we define a network configuration for which we can calculate the currents and the voltages, in a unique way and according to an adequate
algorithm. To do this, we must know the impedance of the branches as well as the loads at the network nodes. Regarding the load model, we will consider the constant current model that might be a compromise between the constant power and the constant impedance models. In this manner, with all the variables defined, the evaluation of the constraints and the objective function of the problem allows us to keep or reject the above-mentioned solution.

Thus, to solve our problem, we propose heuristic strategies for the exploration of the space of possible configurations. The principle used is based on the following rule: at each stage, judiciously replace the current configuration $s$ with one of its neighbors. A configuration $s'$ is a neighbor of $s$ if it is obtained from the application of an elementary modification $m_s$, so that $s' = s \oplus m$ is of the same type as $s$. We denote $V(s)$ as a set of the neighbor configurations of $s$.

The proposed heuristic strategies might be classified into 3 possible exploration approaches: constructive, destructive and branch exchange approaches. Each approach is distinguished from the others by the type of the configuration examined during the exploration of the set of possible solutions.

**Constructive approach**

The constructive approach leads to strategies where we examine configurations without loops, even if they are not connected. In accordance to this approach, an elementary modification consists in closing one branch that was initially open. The simplest strategy is the greedy strategy. In this strategy the initial current configuration is defined so that all the nodes of the network are isolated, except the source node. This is performed by opening all the branches of the network. Subsequently, at each stage, we identify all the branches in the current configuration $s$ that are open and that have only one isolated node. Closing alternately and independently each one of these particular branches inside $s$, we obtain all the configurations that belong to the neighborhood $V(s)$. At the conclusion of the previous exploration, we move to the next stage, replacing the current configuration $s$ with the neighbor one from $V(s)$ that minimizes the objective function and respects the inequality constraints of the problem. In the best case and according to the above-mentioned, $V(s)$ will be empty at the last stage of the strategy. It means that the current configuration $s$ is the problem solution and we qualify it as a local optimum. In the opposite case, the current configuration $s$ will have a certain number of definitely isolated nodes and we say that the strategy failed.

We are not able to conclude that the final current configuration is globally optimal even if the strategy succeeds. It results from the fact that each branch closed for moving from one stage to the next corresponds to a locally (but not globally) optimal decision which is not questioned later.

We can improve the proposed strategy by accepting to question the decisions. A backtracking process would be the only way to extend the exhaustiveness of the search into the set of possible configurations in order to decrease the chance of the failure situation. But backtracking is difficult to manage in a general case.

As an example, if we ended up at a stage with the situation of failure, we should start to open all the branches that led to this stage, while successively marking each of them in the inverse order of their initial closing. This procedure should terminate as soon as we reach a configuration whose associated neighborhood will contain configurations that respect the inequality constraints and does not seek the closing of the marked branches. Afterwards, we could start a new search, different from the first one, on the possible configuration set. But the question at this point is: how do we treat the marked branches? If we leave them as they are, it is very probable that we will not have enough non-marked branches for the new search. In the opposite case, we will be faced with the multiple backtracking process in the sense that if we end up again in the failure situation, we will have to repeat the previous procedure again. In that case, cycling will occur, e.g. oscillations among different failure situations including, probably, the ones already met. In addition, a stopping condition should be defined, even if the strategy does not fail, in order to guarantee its convergence.

According to the literature studied, the constructive approach was used especially for the searching of a feasible configuration of a distribution network [4-6]. However, we note the publication of Glamocanin [7] where this approach permitted to establish a greedy strategy for the search of a minimal losses configuration.

**Destructive approach**

The destructive approach leads to strategies that examine connected configurations, even if they contain loops. According to this approach, an elementary modification consists in opening one branch that was initially closed. As an example, we can propose the dual constructive greedy strategy. For this new strategy, the initial current configuration $s$ is determined by closing the branches as well as the loads at the network nodes. We can improve the proposed strategy by accepting to question the decisions. A backtracking process would be the only way to extend the exhaustiveness of the search into the set of possible configurations in order to decrease the chance of the failure situation. But backtracking is difficult to manage in a general case.

As an example, if we ended up at a stage with the situation of failure, we should start to open all the branches that led to this stage, while successively marking each of them in the inverse order of their initial closing. This procedure should terminate as soon as we reach a configuration whose associated neighborhood will contain configurations that respect the inequality constraints and does not seek the closing of the marked branches. Afterwards, we could start a new search, different from the first one, on the possible configuration set. But the question at this point is: how do we treat the marked branches? If we leave them as they are, it is very probable that we will not have enough non-marked branches for the new search. In the opposite case, we will be faced with the multiple backtracking process in the sense that if we end up again in the failure situation, we will have to repeat the previous procedure again. In that case, cycling will occur, e.g. oscillations among different failure situations including, probably, the ones already met. In addition, a stopping condition should be defined, even if the strategy does not fail, in order to guarantee its convergence.

According to the literature studied, the constructive approach was used especially for the searching of a feasible configuration of a distribution network [4-6]. However, we note the publication of Glamocanin [7] where this approach permitted to establish a greedy strategy for the search of a minimal losses configuration.

**Destructive approach**

The destructive approach leads to strategies that examine connected configurations, even if they contain loops. According to this approach, an elementary modification consists in opening one branch that was initially closed. As an example, we can propose the dual constructive greedy strategy. For this new strategy, the initial current configuration $s$ is determined by closing the branches as well as the loads at the network nodes. We can improve the proposed strategy by accepting to question the decisions. A backtracking process would be the only way to extend the exhaustiveness of the search into the set of possible configurations in order to decrease the chance of the failure situation. But backtracking is difficult to manage in a general case.

As an example, if we ended up at a stage with the situation of failure, we should start to open all the branches that led to this stage, while successively marking each of them in the inverse order of their initial closing. This procedure should terminate as soon as we reach a configuration whose associated neighborhood will contain configurations that respect the inequality constraints and does not seek the closing of the marked branches. Afterwards, we could start a new search, different from the first one, on the possible configuration set. But the question at this point is: how do we treat the marked branches? If we leave them as they are, it is very probable that we will not have enough non-marked branches for the new search. In the opposite case, we will be faced with the multiple backtracking process in the sense that if we end up again in the failure situation, we will have to repeat the previous procedure again. In that case, cycling will occur, e.g. oscillations among different failure situations including, probably, the ones already met. In addition, a stopping condition should be defined, even if the strategy does not fail, in order to guarantee its convergence.

According to the literature studied, the constructive approach was used especially for the searching of a feasible configuration of a distribution network [4-6]. However, we note the publication of Glamocanin [7] where this approach permitted to establish a greedy strategy for the search of a minimal losses configuration.

**Destructive approach**

The destructive approach leads to strategies that examine connected configurations, even if they contain loops. According to this approach, an elementary modification consists in opening one branch that was initially closed. As an example, we can propose the dual constructive greedy strategy. For this new strategy, the initial current configuration $s$ is determined by closing the branches as well as the loads at the network nodes. We can improve the proposed strategy by accepting to question the decisions. A backtracking process would be the only way to extend the exhaustiveness of the search into the set of possible configurations in order to decrease the chance of the failure situation. But backtracking is difficult to manage in a general case.

As an example, if we ended up at a stage with the situation of failure, we should start to open all the branches that led to this stage, while successively marking each of them in the inverse order of their initial closing. This procedure should terminate as soon as we reach a configuration whose associated neighborhood will contain configurations that respect the inequality constraints and does not seek the closing of the marked branches. Afterwards, we could start a new search, different from the first one, on the possible configuration set. But the question at this point is: how do we treat the marked branches? If we leave them as they are, it is very probable that we will not have enough non-marked branches for the new search. In the opposite case, we will be faced with the multiple backtracking process in the sense that if we end up again in the failure situation, we will have to repeat the previous procedure again. In that case, cycling will occur, e.g. oscillations among different failure situations including, probably, the ones already met. In addition, a stopping condition should be defined, even if the strategy does not fail, in order to guarantee its convergence.

According to the literature studied, the constructive approach was used especially for the searching of a feasible configuration of a distribution network [4-6]. However, we note the publication of Glamocanin [7] where this approach permitted to establish a greedy strategy for the search of a minimal losses configuration.

**Destructive approach**

The destructive approach leads to strategies that examine connected configurations, even if they contain loops. According to this approach, an elementary modification consists in opening one branch that was initially closed. As an example, we can propose the dual constructive greedy strategy. For this new strategy, the initial current configuration $s$ is determined by closing the branches as well as the loads at the network nodes. We can improve the proposed strategy by accepting to question the decisions. A backtracking process would be the only way to extend the exhaustiveness of the search into the set of possible configurations in order to decrease the chance of the failure situation. But backtracking is difficult to manage in a general case.

As an example, if we ended up at a stage with the situation of failure, we should start to open all the branches that led to this stage, while successively marking each of them in the inverse order of their initial closing. This procedure should terminate as soon as we reach a configuration whose associated neighborhood will contain configurations that respect the inequality constraints and does not seek the closing of the marked branches. Afterwards, we could start a new search, different from the first one, on the possible configuration set. But the question at this point is: how do we treat the marked branches? If we leave them as they are, it is very probable that we will not have enough non-marked branches for the new search. In the opposite case, we will be faced with the multiple backtracking process in the sense that if we end up again in the failure situation, we will have to repeat the previous procedure again. In that case, cycling will occur, e.g. oscillations among different failure situations including, probably, the ones already met. In addition, a stopping condition should be defined, even if the strategy does not fail, in order to guarantee its convergence.
Owing to the existing duality between this strategy and its constructive homologue, the problems remain the same if one wants to improve it with the backtracking process.

According to the literature, the destructive approach was used in [8,9] for the search of a minimal losses configuration. The developed strategies have also a greedy nature. They propose to simply open the least loaded branch when passing from one stage to another.

Branch exchange approach

The branch exchange method leads to strategies where exclusively arborescent configurations are examined. In this case, an elementary modification consists in the exchange of the topological states of a closed branch and an open one, while observing the arborescence constraints. According to this definition, the neighborhood \( V(s) \) associated with a configuration \( s \) can be defined by first locating the list \( L_b \) of the branches in \( s \) that are open. Afterwards, we find in \( s \) the unique path \( L_i \cup L_j \) for each branch \( b_h \in L_b \) that links its two end nodes. Then we exchange consecutively and independently the topological state of \( b_h \) with that of each branch \( b_e \in L_i \cup L_j \) in order to obtain, each time, a different neighbor configuration \( s' = s \oplus (b_h, b_e) \) (fig. 1).

In this approach, when a configuration presents inequality constraints violations, it is possible to adopt a strategy that aims at improving these constraints from stage to stage until we eventually obtain a configuration that respects these constraints. This suggests the replacement of every inequality constraint with a weighted penalty that is added to the objective function. As an example, if the branch current \( I \) is greater than the admissible maximal current \( I_{\text{max}} \), then the penalty will be \( I/I_{\text{max}} \), and in the opposite case it will be 0. Consequently, we will consider that the penalties are included in the objective function in the same way as the optimization criteria.

The branch exchange approach differs from the two others by the fact that it leads to strategies where every decision taken for moving to a given stage may be naturally reversed for moving to an ulterior stage without the need for explicit backtracking procedure. An example is illustrated in figure 2: the accepted elementary modification for moving from stage \( n \) to stage \( n+1 \) indicates the closing of branch \( b_2 \) according to a given criterion. Later on, the branch \( b_2 \) had to be opened according to the same criterion for moving to stage \( n+2 \).

According to the literature, different strategies using the branch exchange approach have been elaborated in order to solve not only the operation problems [10-12], but also the problem of electrical distribution networks planning [13-15].

STRATEGIES USING THE BRANCH EXCHANGE APPROACH

Descent strategies

The basic principle of a descent strategy consists in generating at each stage a set \( \tilde{V}(s) \supseteq V(s) \) of the configurations that are neighbors of a current configuration \( s \) and defining in this set a subset \( V^*(s) \) so that each element \( s' \) of \( V^*(s) \) has a cost (objective function value) smaller than the cost of \( s \), i.e., \( f(s') < f(s) \). Consequently, we choose one configuration from \( V^*(s) \) that will be substituted to \( s \) in order to move to the next stage. If it is impossible to identify the subset \( V^*(s) \) at a given stage, i.e., \( V^*(s) = \emptyset \), the strategy is interrupted and the current configuration \( s \) is declared an optimal local solution of the problem.

In the case of the two descent strategies that will be described below, the choice of the substitution configuration for moving from one stage to another is the one that will minimize the objective function on \( V^*(s) \). However, we note that this choice is not necessarily the best one, given the heuristic nature of the strategies in question. The substitution configuration could also be chosen randomly in \( V^*(s) \), or furthermore could simply correspond to the first configuration identified inside this set.

Descent strategy examples

The first descent strategy proposed suggests we examine at each stage all the neighbor configurations of the current configuration \( s \), i.e., \( \tilde{V}(s) = V(s) \). We will call it Maximal Neighborhood Descent Strategy (MNDS). To do this, the exploration of \( V(s) \) is performed, at each stage, by going through the list \( L_b \) of branches from \( s \)
that are open. For each branch $b_h \in L_{bh}$ we exchange
alternately and independently its topological state with this
state of each branch $b_g$ from the path $L_1 \cup L_2$ that
link the terminal nodes of $b_h$. In that way we obtain,
each time, a new neighbor configuration $s' = s \oplus (b_h, b_g)$
whose cost can be evaluated, i.e., $f(s')$. As soon as the
exploration progresses, we constantly retain the
exchange that led to the best configuration already
examined as well as the cost associated with it. The
exchange or elementary modification in question will be
denoted $m^* = (b_h^*, b_g^*)$ and the cost of the resulting
configuration $f(s^*)$. At the end of the exploration, if
$f(s^*) < f(s)$, then we apply $m^*$ to $s$ as $s = s \oplus m^*$
for moving to the next stage and we introduce $b_h^*$ into $L_{bh}$
instead of $b_g^*$. In the opposite case, the strategy
terminates and the solution sought after is the current
configuration $s$.

The second descent strategy proposed suggests we
examine at each stage only one neighborhood subset of
$V(s)$, i.e., $\tilde{V}(s) \subset V(s)$, whose size is dynamically
determined. Its general principle is similar to that of the
previous strategy except that the end of a stage occurs
as soon as we reach the first branch $b_h \in L_{bh}$ that
permits to generate one or several configurations with
costs strictly inferior to the cost of the current
configuration $s$. The best of these particular
configurations is automatically substituted for $s$ in order
to start the next stage, and the list $L_{bh}$ is readjusted in
the same manner as for the previous strategy. Following
the example of the first strategy, we could go through
the list $L_{bh}$ at each stage starting from its first position
($k=1..k_{max}$). However, we propose to start from the
position $k=k^*+1$ (except for the first stage), where $k^*$
corresponds to the position whose branch has led to the
substitution configuration at the end of the previous
stage. If the end of the list $L_{bh}$ is reached during a given
stage, i.e., position $k=k_{max}$, then we automatically
return to its first position, i.e., $k=1$; in that way we go
through $L_{bh}$ in circular manner (we note: if $k=k_{max}$,
then $k+1$ becomes $1$; if $k=1$, then $k$ becomes $k_{max}$).
The last stage of this strategy will correspond to going
through $L_{bh}$, starting from the position $k=k^*+1$ to $k=k^*$.
In other words, we will examine the whole
neighborhood $V(s)$ without the possibility to detect any
substitution configuration. Due to the dynamic size of
explored neighborhoods, this second strategy will be
called Dynamic Neighborhood Descent Strategy
(DNDS).

In order to illustrate this strategy, which seems more
complex than the first one, we will show the first three
stages for a 13-nodes 16-branches network (fig. 3). Its
initial configuration $s_1$ is given and its source node is
$x_5$.

In the particular case of the DNDS, the achieved
final configuration generally depends on the initial
layout of the branches in the list $L_{bh}$. So we can
consider several initial layouts of these branches and, as
one goes on, apply each time the strategy in question. In
that way we will obtain a set of final configurations
from which we will choose the best one as the problem
solution.

Stage 1:

Stage 2:

Stage 3:

Revised descent strategies

The revision of a descent strategy consists in
introducing rules into the strategy to aim at reducing, if
possible, the size of the examined neighborhood in each
stage without degrading the quality of initially obtained
results. These rules, whether exact or heuristic, should
contribute to quickly achieving the solutions. The rules
that are explained below were established from the
experiences derived from MNDS and DNDS.

Rule 1

When examining the neighborhood at a given stage, we
can find different configurations that are neighbors of the
current configuration $s$ as well as among themselves
in the sense that we can pass from one configuration to another with only one branch exchange. This results from the fact that each of these configurations is obtained from \( s \) by an elementary modification that implied closing or opening the same branch. If one of these configurations is chosen as a substitution configuration, then all the others will automatically belong to the neighborhood to be explored in the next stage, including the current configuration. Therefore, it would be worthwhile to avoid treating these configurations during the new stage, given that the cost associated with each of them will be anyway greater than the new current configuration cost. Considering that, it will be sufficient if we have at the beginning of each stage the couple of branches whose topological exchange led to the current configuration, and afterwards, during the neighborhood examination, we will ignore every elementary modification that implies one of these two branches (not exclusively).

**Rule 2**

Referring to figure 1, the path that links the extremity nodes \( x_1 \) and \( x_J \) of an open branch \( bh \) is defined as the union of paths \( L_i \) and \( L_j \) that are supplied from the node \( x_c \). When we exchange alternately and independently the topological states of \( bh \) and those of the branches from \( L_i \cup L_j \) starting from \( x_c \) to \( x_1 \) and afterwards from \( x_1 \) to \( x_c \), the experiments showed that the values of the objective function related to the resulting configurations generated in that order generally constitute a convex profile. However, this result can be guaranteed only if there is a decrease of the profile of the currents along the paths \( L_i \) and \( L_j \), starting from \( x_c \) to \( x_1 \) and from \( x_1 \) to \( x_c \) respectively. Considering this result, the rule will be the following: for an open branch \( bh \) and its associated path \( L_i \cup L_j \), we make topological exchanges of \( bh \) and each branch from \( L_i \) (\( L_j \)) starting from \( x_1 \) to \( x_c \) (from \( x_j \) to \( x_c \) respectively) as long as every resulting neighbor configuration generated in this order has a cost that is smaller than the previous one (the current configuration \( s \) being the initial neighbor configuration). If we start with the path \( L_i \) and observe the objective function decreasing, then it is useless to deal with the path \( L_j \), and vice versa.

**Rule 3**

When we make a topological exchange of an open branch \( bh \) and each branch from the associated path \( L_i \cup L_j \) during a given stage of the MNDS, we can observe that all obtained neighbor configurations have a greater cost than the cost of the current configuration \( s \). If the path \( L_i \cup L_j \) associated with that branch \( bh \) does not change in the next stage, the same branch exchanges can appear again. Therefore, we can avoid considering them again if the current of each branch of this path was not perturbed due to the optimal branch exchange that permitted us to start this new stage. To do this, suppose that at the beginning of a given stage we know all the open branches, subject to the above-mentioned case, during the previous stage. Each of these branches is marked. In addition, suppose that we know all the branches whose currents were perturbed in order to obtain the current configuration \( s \). The previous stage number is associated to each one of these branches except the one that was opened (its current is 0). This particular branch will anyway be subject to Rule 1 and is marked in the same manner as the open branches mentioned before.

During the exploration of the neighborhood \( V(s) \), we observe the next cases according to the considered open branch \( bh \):

- If \( bh \) is not marked, we directly apply to \( s \) the branch exchanges implying \( bh \). If no exchanged branch leads to a neighbor configuration with a cost smaller than the cost of \( s \), then \( bh \) is marked in view of the next stage.

- If \( bh \) is marked, we check if there is at least one branch from the associated path \( L_i \cup L_j \) with the number equal to the number of the previous stage. If it is the case, then we remove the mark from \( bh \) and we proceed as in the previous point; in the opposite case, we keep \( bh \) marked and we move automatically to the next branch from the list \( Lbh \).

Once the exploration of the neighborhood \( V(s) \) is finished, we know the resulting optimal branch exchange, i.e., \( m^* = (bh^*,be^*) \). We mark \( be^* \) and we associate the number of the stage to \( bh^* \) and all the corresponding branches from the path \( L_i \cup L_j \), except for \( be^* \). Consequently, we apply \( m^* \) to \( s \) in order to determine its substitution configuration.

**TABU strategies**

Using a descent strategy there is the risk of obtaining a locally optimal solution whose cost could be far from the globally optimal solution cost. To decrease this risk, we propose to transform such a strategy into another one with a more exhaustive search in order to visit better local optima and perhaps even the global optimum. This transformation would consist in providing a strategy with the principles of a general combinatorial optimization method called TABU [16,17].

According to this method, the first principle would authorize, if necessary, the choice of a substitution configuration \( s^* \) from \( \hat{V}(s) \) that is worse than the current configuration \( s \) \((f(s^*)>f(s))\). In this way, we can avoid staying at a local optimum. However, when this situation occurs the risks of cycling become important. As the topological states of the branches can constantly be questioned, it is possible that we return to an already visited configuration and consequently to describing an endless trajectory into the set of possible configurations.
To alleviate this cycling, the second principle of this method suggests we define as tabu each branch implied in the branch exchange that permitted to achieve the n stage current configuration from the n-1 stage current one. It means that during a certain fixed number of stages, starting from stage n, each branch exchange that takes into consideration one of these two branches will simply be ignored during the neighborhoods' exploration. Therefore, two cyclic lists called tabu are established: a list denoted Th of length th for the opened branches, and a list denoted Te of length te for the closed branches. Each new branch introduced in Th (Te) takes the place of the branch which is in Th (Te) from th (te) stages. We check at each stage that each branch exchange, candidate to being examined, does not imply branches belonging to Th or Te.

However, this cyclic fixation of the degrees of liberty of the problem could compromise the opportunity to converge towards a good solution since several interesting configurations could not be examined. In order to overcome this difficulty, the third principle of the method suggests to add an aspiration rule. It means that we can examine even a branch exchange whose one or the other element is tabu (tabu elementary modification) as any other branch exchange, and we can keep it as a candidate to establish the substitution configuration at the end of a given stage only if the resulting configuration cost is lower than a certain value, qualified as threshold. As an example, this threshold could be the cost of the best configuration visited until now, i.e., s+. According to this aspiration rule, all branch exchanges in a given stage will be examined and the first part of the search into the set of possible solutions will be descending until we reach the first local optimum of the objective function, or the stage where we will choose for the first time a substitution configuration that is worse than the current one.

According to the first principle, it is necessary to indicate a criterion that permits to interrupt the search into the set of possible solutions. As an example, we can suggest that the number of stages between two successive improvements of s+ must be below a specified limit.

Finally, TABU method principles were adapted to MNDS and DNDS in order to obtain TABU strategies that we will call MNTS and DNTS, respectively.

**NUMERICAL RESULTS**

To illustrate numerically the MNDS, DNDS, MNTS, DNTS strategies, we propose to apply them to a real urban network (fig. 4). The aim of the proposed optimisation is to find a new feasible configuration with minimum losses. The urban network used is constituted of 230 nodes and 306 branches. The nominal voltage is 6.4 KV and the total load is 69 MVA. The network is supplied by four sources, but in order to apply the strategies presented, each source is connected to a unique imaginary source through a branch with the capacity of the supplier transformer. The currently used topology of the network will be used as the initial topology for the four strategies. Although this topology is the current one, it contains three overload lines. For the strategies we have chosen a penalty weight of 60 KW for each overload. The total losses of this initial topology is 2079.96KW. The load factor of the heavily loaded branch is 115 % (max. I/I_max) and the greatest voltage drop is 8 % (max. ΔU/U).

![fig. 4 Urban distribution network test](image)

The results are summarized in the following table:

<table>
<thead>
<tr>
<th>Strategy</th>
<th>MNDS</th>
<th>DNDS</th>
<th>MNTS</th>
<th>DNTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Losses [KW]</td>
<td>1630.02</td>
<td>1632.34</td>
<td>1619.38</td>
<td>1620.03</td>
</tr>
<tr>
<td>(−21%)</td>
<td>(−21%)</td>
<td>(−22%)</td>
<td>(−22%)</td>
<td></td>
</tr>
<tr>
<td>Max I/I_max [%]</td>
<td>96</td>
<td>98</td>
<td>95</td>
<td>96</td>
</tr>
<tr>
<td>Max ΔU/U [%]</td>
<td>5.5</td>
<td>5.5</td>
<td>5.5</td>
<td>5.5</td>
</tr>
<tr>
<td>Nb of examined configurations</td>
<td>3556 (12671)</td>
<td>684 (1304)</td>
<td>179991</td>
<td>41753</td>
</tr>
<tr>
<td>CPU time [s]</td>
<td>1.31 (3.04)</td>
<td>0.28 (0.38)</td>
<td>49.04</td>
<td>13.71</td>
</tr>
</tbody>
</table>

For a same quality of results, we can see that each strategy with a dynamic neighborhood is more efficient than the corresponding strategy with maximal neighborhood. The performance can be measured with the total number of examined configurations (neighbors and currents). In the particular case of the descent
strategies, the values in parenthesis correspond to the case where the rules of neighborhoods' reduction are not used. So we may conclude that these rules are very useful and that they do not modify the quality of the results. Although the TABU strategies give better results than the descent strategies, they require a more important search. However, we have repeated DNDS and DNTS strategies 100 times, each time with a same initial configuration but with a different initial layout of the open branches in the list Lbh. Among the 100 solutions obtained in the case of the DNDS, 32 were feasible (no overload) whereas among the 100 solutions obtained in the case of DNTS, 98 were feasible. Moreover the losses of these 98 configurations were practically similar and globally better than those of the 32 (Tab I). The same type of results has also been observed with the MNDS and MNTS strategies when repeated 100 times, starting each time with a different initial configuration (Tab II).

Tab. I Results of experiments with 100 different initial layouts of Lbh

<table>
<thead>
<tr>
<th>Strategy</th>
<th>DNDS</th>
<th>DNTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Losses [kW] :</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best value</td>
<td>1618.41</td>
<td>1618.29</td>
</tr>
<tr>
<td>Worst value</td>
<td>1688.60</td>
<td>1624.40</td>
</tr>
<tr>
<td>Mean value</td>
<td>1640.12</td>
<td>1619.66</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>26.89</td>
<td>1.45</td>
</tr>
<tr>
<td>Nb of feasible solutions found</td>
<td>32</td>
<td>98</td>
</tr>
<tr>
<td>Total CPU time [s]</td>
<td>27.53</td>
<td>1515.17</td>
</tr>
</tbody>
</table>

Tab. II Results of experiments with 100 different initial configurations

<table>
<thead>
<tr>
<th>Strategy</th>
<th>MNDS</th>
<th>MNTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Losses [kW] :</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best value</td>
<td>1618.29</td>
<td>1618.29</td>
</tr>
<tr>
<td>Worst value</td>
<td>1680.34</td>
<td>1621.64</td>
</tr>
<tr>
<td>Mean value</td>
<td>1631.73</td>
<td>1618.83</td>
</tr>
<tr>
<td>Standard deviation</td>
<td>18.3</td>
<td>0.64</td>
</tr>
<tr>
<td>Nb of feasible solutions found</td>
<td>19</td>
<td>71</td>
</tr>
<tr>
<td>Total CPU time [s]</td>
<td>322.06</td>
<td>5157.29</td>
</tr>
</tbody>
</table>

From these results we conclude that the TABU strategies are more regular than the descent strategies and that the performance of the TABU strategies will particularly benefit from a parallel implementation.

CONCLUSION

In this paper, we dealt with different enumerative approaches to solve the problem of the optimal reconfiguration of distribution systems. The approach chosen allowed us to develop four strategies where the decisions can be questioned at each time if necessary. In order to validate these strategies in a general way, we intend to implement them in a real environment. This allows us to take into account new optimization criteria and constraints. Further, we expect that in a real environment new rules have to be constructed for better guidance of the search algorithm. Finally all developed strategies were implemented on HP Apollo 400 (μp 68040) in PASCAL.

ACKNOWLEDGEMENTS

This paper is based upon work supported by the "Fonds National pour la recherche énergétique" (project NEFF 405)

REFERENCES


