Efficiency Enhancements for Evolutionary Capacity Planning in Distribution Grids

Ngoc Hoang Luong
Centrum Wiskunde & Informatica (CWI)
Amsterdam, The Netherlands
N.H.Luong@cwi.nl

Marinus O.W. Grond
Eindhoven University of Technology (TU/e)
Eindhoven, The Netherlands
M.O.W.Grond@tue.nl

Han La Poutré
Centrum Wiskunde & Informatica (CWI)
Amsterdam, The Netherlands
Han.La.Poutre@cwi.nl

Peter A.N. Bosman
Centrum Wiskunde & Informatica (CWI)
Amsterdam, The Netherlands
Peter.Bosman@cwi.nl

ABSTRACT

In this paper, we tackle the distribution network expansion planning (DNEP) problem by employing two evolutionary algorithms (EAs): the classical Genetic Algorithm (GA) and a linkage-learning EA, specifically a Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA). We furthermore develop two efficiency-enhancement techniques for these two EAs for solving the DNEP problem: a restricted initialization mechanism to reduce the size of the explorable search space and a means to filter linkages (for GOMEA) to disregard linkage groups during genetic variation that are likely not useful. Experimental results on a benchmark network show that if we may assume that the optimal network will be very similar to the starting network, restricted initialization is generally useful for solving DNEP and moreover it becomes more beneficial to use the simple GA. However, in the more general setting where we cannot make the closeness assumption and the explorable search space becomes much larger, GOMEA outperforms the classical GA.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search

General Terms

Performance, Experimentation

Keywords

Electricity; Distribution Networks; Capacity Planning; Optimal Mixing; Linkage Learning

GECCO'14, July 12–16, Vancouver, BC, Canada.

http://dx.doi.org/10.1145/2598394.2605696.

1. INTRODUCTION

Electric distribution grids are parts of the total power system and carry the electricity from high-voltage (HV) transmission grids to the medium-voltage (MV) and low-voltage (LV) consumers. A distribution grid consists of various electric components, such as overhead power lines, underground cables, substations, transformers, meters, protection devices, etc., in which the branch configuration (i.e. lines/ cables) are of particular interest. How the cables are connected and which types of cables are chosen determine both the topology and the capacity of the grid. Distribution grids are typically constructed, maintained, and upgraded by distribution network operators (DNOs). The growth in the power demands of consumers requires DNOs to perform network reinforcements to ensure that the magnitude of the power flows that are required to satisfy consumers' demands are within the capacities of all network components. Traditionally, regarding one or multiple scenarios of forecast load growth, DNOs manually sketch a few likely network expansion plans based on engineers' expert knowledge and company-specific policies. Those plans are then evaluated for their investment cost, operational cost, and also technical feasibility, and usually the plan with the least total cost is chosen to be carried out. As distribution grids become increasingly large and complicated over time, including the introduction of new technologies, such as distributed generation or demand-side management, traditional manual generation of expansion plans is no longer thorough enough to guarantee that a good, let alone optimal, plan can be obtained from the vast search space of possible alternatives.

Optimization problems of computationally expensive evaluation functions and large search spaces often require the application of metaheuristics, such as genetic algorithms [3], simulated annealing [13], or particle swarm optimization [2]. The popular use of these classic metaheuristics in electrical engineer literature is due to, besides their general applicability and search capabilities, their straightforward implementations and wide availability in computation software (e.g. MATLAB). However, far more novel, robust developments in evolutionary computation exist, amongst which are EAs that perform linkage learning to be robust against a-priori unknown variable dependencies. These EAs are typi-

cally more involved, but offer better convergence properties. This is however more frequently tested on well-known benchmark functions rather than real-world applications, like the one we consider here. In previous, related work [8], we have applied a state-of-the-art linkage learning EA known as the Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) [1] to the DNEP problem and obtained promising results. In this paper, we tackle the problem under a different scenario, input settings, and evaluation function. We will also present simple linkage filtering techniques aiming to improve the efficiency of GOMEA.

In real-world optimization, expert knowledge and domain-specific heuristics are often employed to disregard some "impractical" alternatives, thereby reducing the search space. Especially in the field of electricity network expansion planning, another reason is that, regarding the current legacy of already-installed network facilities, DNOs prefer conventional solutions involving an incremental number of changes to any "upside-down" reconfigurations. Expansion plans involving many big changes, such as removals of cable connections and restructuring multiple parts of the grids, are often considered as unfavorable. However, many population-based optimization algorithms often start from randomly initialized populations. Considering this, we could also generate the initial population of candidate solutions by randomly changing only a few components in the existing (starting) network. In this paper, we will study if such initialization can help to obtain acceptable solutions faster. It should be noted that in related work [6], we proposed alternative methods of reducing the size of the search space from an electrical engineering point of view by introducing engineering rules as constraints. Here we will look at search space size reduction from a more algorithmic perspective.

To this end, we solve the distribution network expansion planning problem (DNEP) by using a classic genetic algorithm (GA) and the state-of-the-art Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) [1, 10] and study the difference in performance when using fully random population initializations as well as restricted initializations. The remainder of this paper is organized as follows. Section 2 presents the optimization model for the DNEP problem. Section 3 outlines our optimizers GA and GOMEA. Section 4 describes our efficiency enhancement techniques for solving DNEP problems. Section 5 presents the benchmark problem and exhibits experimental results. Finally, Section 6 concludes the paper.

2. DNEP OPTIMIZATION MODEL

A distribution network can be represented as a graph $G = (V, E)$, in which V is the set of m nodes (vertices) and E is the set of l branches (edges). A node can be a substation, which is usually installed with a HV/MV transformer feeding electricity from a (sub-)transmission system into the distribution grid, or a consuming unit (i.e. a customer station demanding directly MV electricity or a network station associated with an MV/LV transformer supplying LV electricity for an area of households). A branch is a connection between two nodes, which can be a power line running overhead or a cable running underground. In conventional distribution grids, the branch configuration creates feed paths for power flows from supplying substations *downward* to consuming units. In a scenario with distributed generation, such as photovoltaics at households, a consuming unit can at times become a supplying unit creating an *upward* flow feeding electricity into the grid. Nevertheless, regardless of the directions, all electric currents have to flow through cable connections, which have specific nominal capacities. Cable overload is defined as when a cable connection has to carry a power flow whose magnitude is larger than its capacity. Overload for an extended amount of time can damage the facility and causes system failures. DNOs are thus required to perform network capacity planning to assure that growths in power demand are satisfied while no overload should occur. In this regard, distribution grids may consist of various kinds of electric facilities, but for the problem of capacity planning the configurations of the cables are usually of particular interest. In this paper, we assume that DNOs perform DNEP by making decisions on how the cables are connected and which type of cable is used for each connection.

2.1 Decision Variable Codification

To solve a DNEP problem, all cable connections on which a DNO can make reinforcement decisions need to be specified. These consist of all the currently existing branches that can be upgraded and the potential branches that can be newly installed. The set of potential branches is limited by expert knowledge to dismiss impractical connections, e.g. it is uneconomical to connect two far-away nodes.

Let l be the total number of branches (i.e. both existing and potential branches), a distribution network can be represented as a vector $\boldsymbol{x}$ of l elements:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_l), \quad |x_i| \in \Omega(x_i), \quad i \in \{1, 2, \ldots, l\} \quad (1)$$

where each x_i is a categorical variable indicating which type of cable is installed at the i-th branch of the network. The set of possible alternatives of cable types $\Omega(x_i)$ that can be installed at the branch i-th is DNO-specific, which usually consists of standardized cable sizes. The value of x_i determines the status of the i-th branch:

- $x_i = id > 0$: A cable of type $id \in \Omega(x_i)$ should be installed at the i-th branch.
- $x_i = 0$: No cable should be installed at the i-th branch.
- $x_i = -id < 0$: A cable of type $id \in \Omega(x_i)$ should be installed and put into reserve status at the i-th branch. This cable connection is put into inactive state, carrying no power flow during normal operation, by opening one of the two switches at its two endings. Such cable connection is usually termed as a Normally Open Point (NOP), which is used to reconfigure the network in emergency situations.

The original distribution network is thus represented as a vector whose elements corresponding with currently existing branches receive non-zero values while elements corresponding with potential branches are set to 0.

2.2 Constraints

For a given scenario of forecast growth in power demand and, in case of distributed generation, the power generating capability at every node, the following constraints must be satisfied:

1. **Connectivity constraint**: All node should be connected, i.e. there is a feed path from a substation to a consuming unit.

2. **Voltage constraints**:

$$|V_i|^{min} \leq |V_i| \leq |V_i|^{max}, \quad i \in \{1, 2, \ldots, m\} \tag{2}$$

where $|V_i|$ is the voltage magnitude at node i. $|V_i|^{min}$ and $|V_i|^{max}$ are minimal and maximal allowed voltage magnitude at node i, respectively. All nodal voltage should stay within the allowable range.

3. **Power flow constraints**:

$$|S_i| \leq |S_i|^{max}, \quad i \in \{1, 2, \ldots, l\} \tag{3}$$

where $|S_i|$ is the magnitude of the power flow through the cable connection at the i-th branch, and $|S_i|^{max}$ is the nominal capacity of that cable. There should be no overload in normal operation.

4. **Radiality constraint**: A consuming unit should be supplied its power demand through only one single feed path in normal operation. This constraint is often employed due to security and protection policies of DNOs.

5. **Reconfigurability constraint**: When an active cable fails, the resulting short-circuit power will be disconnected by a circuit breaker to prevent damaging other components. Some parts of the network will be out of service. The DNO then reconfigures the network by closing the switches of reserve branches (i.e. NOPs) so that all power demands are supplied again. In such temporary emergency situations, the network can tolerate a mild overload for a short time due to thermal dynamics. Here, we assume the emergency capacity of a cable is 130% of its nominal capacity.

Constraints (1) and (4) can be easily verified by checking the topology of the network. To evaluate constraints (2) and (3), it is required to perform a power flow calculation (PLC) [4], which involves solving a system of non-linear equations, called the AC power flow model. A full evaluation of the constraint (5) for a network of k active cables requires k PLCs because every active cable is taken out of service at a time (i.e. to simulate a single cable fault) and it is then checked to see if closing NOPs can bring back the normal service. It is normally acceptable to simulate single cable fault only for cable connections that branching out directly a substation because these cables usually carry the heaviest loads. Nevertheless, due to performing multiple PLCs, constraint evaluation for a DNEP solution is a computationally expensive operation.

2.3 Objective Function

Solving a DNEP problem aims to minimize the net present value (NPV) of the total cost of investment $CAPEX$ and operation $OPEX$ over a planning horizon of T years. In this paper, we consider $OPEX$ as solely the cost of energy loss on cables, which depends on the load and the types of cables being used. An accurate objective evaluation requires the DNEP is solved in a dynamic planning manner, in which the optimal year t_i ($t_0 \leq t_i \leq t_T$) to perform reinforcement at a branch x_i need to be determined. Dynamic planning is typically very expensive in terms of computation effort, involving costly constraint checking for each year during the planning horizon for every candidate solution. In this paper, assuming a limited computation budget, we consider a cheaper pseudo-dynamic approach as follows.

2.3.1 OPEX Estimation

In this section, we present how to estimate the $OPEX$, i.e. the cost of energy loss, of an expansion plan $\boldsymbol{x}$ for an existing grid $\boldsymbol{x}^{origin}$. The peak loss $P_{loss}(\boldsymbol{x}, t)$ on an electricity grid $\boldsymbol{x}$ in a year t can be calculated from the result of a computationally expensive PLC for that grid regarding the peak load in that year. A less accurate, but for planning purposes commonly accepted, approach is to assume the peak loss also has a growth rate related to the load growth R as follows

$$P_{loss}^{estimated}(t) = P_{loss}(t-1) * (1+R)^2 \tag{4}$$

Given R, we can estimate the peak load at every node in each year. Regarding the currently existing configuration of the network $\boldsymbol{x}^{origin}$, we determine the year t_X that the first overload happens. The whole expansion plan $\boldsymbol{x}$, which would eventually satisfy the energy demand in the final year t_T, is then assumed to be carried out all together in the year t_X. In this regard, the year t_X would be the same for all candidate solutions $\boldsymbol{x}$'s because all expansion plans start from the same original network $\boldsymbol{x}^{origin}$. Therefore, the losses from the beginning until t_X would also be the same for all candidate solutions. The losses from the year t_X until and including the year T are different per candidate solution, depending on what types of cables are used. Therefore, we choose to estimate the peak loss $P_{loss}(\boldsymbol{x}, t)$ of an expansion plan $\boldsymbol{x}$ in year t as follows

$$P_{loss}(\boldsymbol{x}, t) = \begin{cases} P_{loss}(\boldsymbol{x}^{origin}, t) & \text{if} \quad t < t_X \\ P_{loss}^{estimated}(\boldsymbol{x}, t) & \text{if} \quad t_X \leq t < T \\ P_{loss}(\boldsymbol{x}, T) & \text{if} \quad t = t_T \end{cases} \tag{5}$$

where $P_{loss}(\boldsymbol{x}^{origin}, t)$ can be calculated in an accurate way by performing t_X PLCs once for all candidate solutions $\boldsymbol{x}$'s. $P_{loss}^{estimated}(t)$ can be estimated by performing one PLC to calculate $P_{loss}(\boldsymbol{x}, T)$ first and then using the Equation 4 to deduct backward. The yearly energy loss $E_{loss}(\boldsymbol{x}, t)$ of a grid $\boldsymbol{x}$ in year t is then computed as

$$E_{loss}(\boldsymbol{x}, t) = P_{loss}(\boldsymbol{x}, t) * T_{loss}(t) \tag{6}$$

where $T_{loss}(t)$ is the service time of peak loss, defined by the area of the yearly energy loss profile, and is input data resulting from scenarios. The $OPEX(\boldsymbol{x}, t)$ of a network $\boldsymbol{x}$ in year t is computed as

$$OPEX(\boldsymbol{x}, t) = Price * E_{loss}(\boldsymbol{x}, t) \tag{7}$$

where $Price$ is the electricity price used in analysis.

2.3.2 CAPEX Calculation

The total sum of investment cost of an expansion plan $\boldsymbol{x}$ can be easily computed by comparing $\boldsymbol{x}$ with the currently existing network $\boldsymbol{x}^{origin}$ as follows

$$Investment = \sum_{i=1}^{l} cost(x_i^{origin}, x_i) \tag{8}$$

where

$$cost(x_i^{origin}, x_i) = \begin{cases} 0 & \text{if } x_i^{origin} = x_i \\ \text{cost of } x_i & \text{if } x_i^{origin} \neq x_i \end{cases} \tag{9}$$

However, in scenario-based planning studies, the $CAPEX$ is usually computed by using the annuities method [9, 12], in which the payment for investment is converted into a series of uniform annual payments called annuities. The length of

this series of payments is the economic lifetime $T^{lifetime}$ of the devices. Here, we assume that all devices have the same $T^{lifetime} = 30$ years. The annuity, i.e. the yearly payment, with a discount rate $i = 4{\cdot}5\%$ is computed as

$$Annuity = Investment * \frac{i}{1-(1+i)^{-T^{lifetime}}} \quad (10)$$

In a true dynamic expansion planning, the annuity should be computed for each invested device separately regarding its specific time of installation. However, in this paper, because we assume that all investments be carried out at the same year t_X when the first overload occurs and that all devices have the same lifetime $T^{lifetime}$, we can compute the annuity for the total investment of all devices. The $CAPEX(\boldsymbol{x}, t)$ in year t is then the annuity

$$CAPEX(\boldsymbol{x}, t) = \begin{cases} Annuity & \text{if} \quad t_X \leq t \leq t_X + T^{lifetime} \\ 0 & \text{else} \end{cases} \quad (11)$$

Note that due to the long lifetime, it can happen that $t_X + T^{lifetime} > t_T$. Since we only consider the cost until the end of the planning horizon t_T, the annuities of the years after the planning horizon will be disregarded. This annuities method is used to compare fairly the total costs between different scenarios [9].

2.3.3 Total Cost Objective Function

The objective of the DNEP problem is to minimize the NPV of the total cost of expansion, which consists of the operation cost $OPEX$ and the investment cost $CAPEX$, over T years with discount rate i. The objective value of a candidate solution $\boldsymbol{x}$ is thus computed as

$$f(\boldsymbol{x}) = \sum_{t=t_0}^{t_T} \frac{OPEX(\boldsymbol{x}, t) + CAPEX(\boldsymbol{x}, t)}{(1+i)^{t-t_0}} \quad (12)$$

3. OPTIMIZATION ALGORITHMS

3.1 Classical Genetic Algorithm (GA)

In this paper, we consider a popular implementation of a classical, widely-applied, GA with uniform crossover and tournament selection. First, GA starts with the random initialization of a population $\mathcal{P}$ of n candidate solutions. For a candidate solution, each decision variable x_i (i.e. a cable connection) can receive any possible value (i.e. $0, id$, or $-id$ with $id \in \Omega(x_i)$) as long as it does not downgrade the currently existing facility. All solutions are evaluated against the required constraints and objective function (Section 2.2 and 2.3). In every iteration, an offspring population $\mathcal{O}$ of n new candidate solutions are created from $\mathcal{P}$. To do this, every 2 parent solutions randomly chosen from $\mathcal{P}$ are recombined through uniform crossover to generate 2 offspring solutions. These fully constructed solutions are evaluated for their constraints and objective values. Next, the parent population $\mathcal{P}$ and the offspring population $\mathcal{O}$ are combined into a selection pool $\mathcal{P} + \mathcal{O}$ of $2n$ solutions, where a tournament selection with tournament size 4 is then performed to select n survivors. These n survivors form the new parent population $\mathcal{P}$ for the next iteration of GA.

3.2 Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA)

Although GAs can be very efficient, the conditions under which they are efficient are highly dependent on the match between the way new solutions are generated (i.e. the operators of variation), the level of dependencies between problem variables and the way the problem is encoded. If this match causes important dependencies to be disrupted during variation, GAs will exhibit poor scale-up behavior, typically requiring exponentially many evaluations to find the optimum as the problem size increases [11]. Linkage learning specifically addresses the task of figuring out, during optimization, which variables are dependent so as to better respect these dependencies during variation. If linkage learning is successful, the scale-up behavior can become extremely favorable, typically requiring only a low-polynomial number of evaluations (sometimes even sublinear) instead [7, 10].

In this paper, we also consider a linkage-learning EA to solve the DNEP problem. This EA is an instance of the GOMEA family [10]. Similar to the classical GA, GOMEA also starts with a randomly initialized population $\mathcal{P}$ of n candidate solutions, which are then evaluated against the constraints and objective function. In every iteration, a set $\mathcal{S}$ of n solutions is selected out of $\mathcal{P}$ by tournament selection with tournament size 2. A linkage model, describing dependencies among decision variables, is then learned from this set $\mathcal{S}$. Exploiting linkage structures captured by the linkage model, GOMEA performs a genetic local search-like variation operation to transform each existing solution $\boldsymbol{x}$ into a new offspring solution $\boldsymbol{o}$ in a step-wise manner. Finally, the population $\mathcal{P}$ is completely replaced by the offspring population $\mathcal{O}$ of n newly constructed solutions.

The linkage model structure of GOMEA follows the Family Of Subset (FOS) concept. Let $L = \{1, 2, \ldots, l\}$ be the set containing all decision variable indices. A FOS $\mathcal{F}$ over L is a subset of the powerset of L : $\mathcal{F} \subseteq \mathcal{P}(S)$. $\mathcal{F}$ can be written as $\mathcal{F} = \{\boldsymbol{F}^1, \boldsymbol{F}^2, \ldots, \boldsymbol{F}^{|\mathcal{F}|}\}$ where $\boldsymbol{F}^i \subseteq \{1, 2, \ldots, l\}$, $i \in \{1, 2, \ldots, |\mathcal{F}|\}$. A FOS can thus be seen as a set of linkage groups, in which each linkage group is in turn a set of decision variables indices, indicating that those variables are (inter-)dependent to some degree. In this paper, we use the Linkage Tree (LT) as the FOS model, which gives a popular instance of GOMEA, also called the Linkage Tree Genetic Algorithm (LTGA). An LT over the set L represents its linkage groups in a hierarchical manner. All l leaf nodes are singleton linkage groups, i.e. $\boldsymbol{F}^i = \{i\}, i \in \{1, 2, \ldots, l\}$, describing the univariate structure. Then, for each branch node $\boldsymbol{F}^i$ containing more than one decision variable indices, describing multivariate dependencies, there exist subsets $\boldsymbol{F}^j$ and $\boldsymbol{F}^k$ such that $\boldsymbol{F}^j \cap \boldsymbol{F}^k = \emptyset, |\boldsymbol{F}^j| < |\boldsymbol{F}^i|, |\boldsymbol{F}^k| < |\boldsymbol{F}^i|$ and $\boldsymbol{F}^j \cup \boldsymbol{F}^k = \boldsymbol{F}^i$. The root node, which contains all variable indices and is thus the set L itself, is removed from the LT because it results in copying the entire solutions when performing linkage-based recombination, which is not useful. An LT over the set L of l decision variables can be constructed by the Unweighted Pair Grouping Method with Arithmetic-mean (UPGMA) procedure in $\mathcal{O}(nl^2)$ time [5], resulting in exactly $2l - 2$ linkage groups. For more details on building the LT, please refer to the literature [1, 10].

GOMEA transforms each existing solution $\boldsymbol{x}$ into a new offspring solution $\boldsymbol{o}$ by a variation operator called Gene-pool Optimal Mixing (GOM). First, $\boldsymbol{o}$ is a direct clone from $\boldsymbol{x}$, and a backup version $\boldsymbol{b}$ of $\boldsymbol{x}$ is created. Traversing the linkage groups in the LT FOS $\mathcal{F}$, for every $\boldsymbol{F}^i \in \mathcal{F}$, a donor solution $\boldsymbol{p}$ is randomly selected from the population $\mathcal{P}$. The

values of decision variables whose indices are indicated by $\boldsymbol{F}^i$ are copied from $\boldsymbol{p}$ to $\boldsymbol{o}$, if those values are different between $\boldsymbol{p}$ and $\boldsymbol{o}$ in at least one position. The constraints and objective values of this partially-altered $\boldsymbol{o}$ are evaluated and compared against the backup $\boldsymbol{b}$. If such mixing results in an improvement or an equally good solution (i.e. $fitness[\boldsymbol{o}] \geq fitness[\boldsymbol{b}]$), then the changes are accepted and also updated into the backup. Otherwise, discarding the changes, $\boldsymbol{o}$ is reverted to its backup state $\boldsymbol{b}$. Note that the acceptance criterion of equal fitness can help the search to move across a fitness plateau. When all $\boldsymbol{F}^i \in \mathcal{F}$ are traversed, an offspring solution $\boldsymbol{o}$ is fully constructed and will replace its parent solution $\boldsymbol{x}$ in the next generation.

If GOM cannot transform a solution $\boldsymbol{x}$ into a new offspring $\boldsymbol{o}$, a procedure called Forced Improvement (FI) is invoked. FI can be seen as a second round of GOM in which the donor solution is always the best-found-so-far solution $\boldsymbol{x}^{best}$. A mixing step in FI is only accepted when it results in a true improvement (i.e. $fitness[\boldsymbol{o}] > fitness[\boldsymbol{b}]$), and FI is stop as soon as such improvement is found. However, FI may never be called if there exist significant plateaus and the acceptance criterion of equal fitness in GOM simply transforms back and forth solutions of different genotypes but the same fitness value. To overcome this, if the number of subsequent generations that the best solution $\boldsymbol{x}^{best}$ does not change, termed as the no-improvement stretch (NIS), exceeds a threshold, FI is triggered. [1] suggests a threshold of $1 + \lfloor \log_{10}(n) \rfloor$. Lastly, if FI cannot transform an existing solution $\boldsymbol{x}$, the current best solution $\boldsymbol{x}^{best}$ is returned as the new offspring $\boldsymbol{o}$. Figure 1 shows pseudo-code.

```
GOMEA //population size n
  for i ∈ {1, 2, ..., n} do
    P_i ← CreateRandomSolution()
    EvaluateFitness(P_i)
  x^best ← argmax_{x∈P}{fitness[x]}
  t ← 0; t^NIS ← 0
  while ¬TerminationConditionsSatisfied do
    S ← TournamentSelection(P, n, 2)
    LearnLinkageModel(S)
    for i ∈ {1, 2, ..., n} do
      O_i ← FI-GOM(P_i)
    P ← O
    x^best ← argmax_{x∈P}{fitness[x]}
    if fitness[x^best(t)] > fitness[x^best] then
      t^NIS ← 0; x^best ← x^best(t)
    else
      t^NIS ← t^NIS + 1
    t ← t + 1
```

```
FI-GOM(x)
  b ← o ← x; fitness[b] ← fitness[o] ← fitness[x];
  changed ← false
  for i ∈ {1, 2, ..., |F|} do
    p ← Random({P_1, P_2, ..., P_n})
    o_{F^i} ← p_{F^i}
    if o_{F^i} ≠ b_{F^i} then
      EvaluateFitness(o)
      if fitness[o] ≥ fitness[b] then
        b_{F^i} ← o_{F^i}; fitness[b] ← fitness[o]; changed ← true
      else
        o_{F^i} ← b_{F^i}; fitness[o] ← fitness[b]
  if ¬changed or t^NIS > 1 + ⌊log10(n)⌋ then
    changed ← false
    for i ∈ {1, 2, ..., |F|} do
      o_{F^i} ← x^best_{F^i}
      if o_{F^i} ≠ b_{F^i} then
        EvaluateFitness(o)
        if fitness[o] > fitness[b] then
          b_{F^i} ← o_{F^i}; fitness[b] ← fitness[o]; changed ← true
        else
          o_{F^i} ← b_{F^i}; fitness[o] ← fitness[b]
      if changed then breakfor
  if ¬changed then
    o ← x^best; fitness[o] ← fitness[x^best]
```

Figure 1: Pseudo-code for GOMEA

4. EFFICIENCY ENHANCEMENTS

4.1 Restricted Initialization

Electricity grids are probably among the most complex technological networks with numerous interconnected facilities, which have been built over time. The legacy of currently existing electric components should be taken into account when conducting any real-world network expansion planning. Regarding the huge sums of investment that have been made into the current grid, network operators typically do not favor expansion plans involving radical changes, such as re-designing the grid from scratch, because this is economically unaffordable, resulting in high tariffs for consumers. Favorable solutions for DNEP problem instances in practice rather consists of incremental changes which differs from the original network in a few cable connections. Engineering heuristics and guidelines are employed to disregard impractical solutions, e.g. the addition of new cable connections should be limited, or the removal of existing connections are often discouraged. These engineering "rules of thumb" help to reduce the search space while also limit the number of changes that can be made to the current grid.

Evolutionary optimization algorithms, such as GA and GOMEA, usually start from a population of randomly generated solutions, providing an initial population of good diversity to begin with. For DNEP, this means that a complicated expansion plan of radical changes can still be obtained if doing so benefits the objective function. However, initialization can also be customized to reflect electrical-engineering expert knowledge and the assumption that only limited changes are required/desired. Initial solutions can then be direct clones of the current grid with only a limited number k of positions having different randomly generated values. We aim to investigate the balance between the computation budget and the quality of obtained solutions by experiment with different values of k that governs the size of the explorable search space of possible alternatives.

4.2 Linkage Filtering

An LT consists of $2l - 2$ linkage groups, in which not every linkage relation is necessarily beneficial in GOM if such linkages are of weak (inter-)dependencies. Filtering spurious linkage groups out of a linkage model is thus an important research topic for GOMEA. However, determining optimal filtering thresholds as in [1] often requires intensive analysis and experiments. Here, we consider a simpler filtering mechanism directly based on the sizes of linkage groups. Given that the level of changes to an existing grid that a DNO would favor is often limited, we hypothesize that large linkage groups of many variables do not contribute much to the search. As the favorable expansion plans are usually of incremental changes, it might be more computationally economical if GOMEA only attempts mixing events of small linkage groups. Therefore, larger linkage groups, which can introduce radical network restructuring, should be filtered out of the LT. We generalize this filtering mechanism by introducing a threshold L_{FOS}^{max} for the maximum size of link-

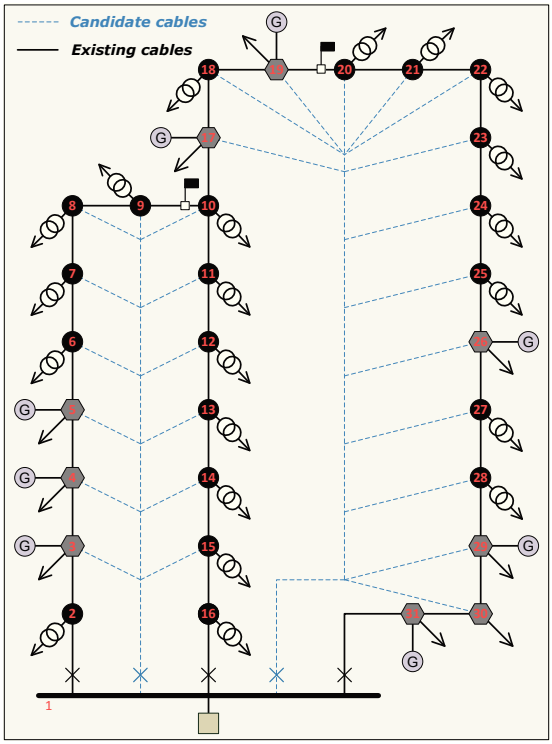

Figure 2: Topology of the benchmark network.

age groups. If a linkage group contains more than L_{FOS}^{max} variable indices, it will be ignored when GOM constructs offspring solutions. Assuming that restricted initialization will initialize the search near the optimal solution, we will experiment to see if this linkage filtering mechanism with different values of L_{FOS}^{max} can further accelerate GOMEA.

5. EXPERIMENTS

5.1 Benchmark Problem

Figure 2 shows the topology of the network that we used for benchmarking. The network is based on a real MV distribution grid containing 31 nodes and 59 branches: 30 currently existing active cable connections, 2 existing reserve cable connections (NOPs) on the 9-10 and 19-20 branches, and 27 potential connections branching out from the substation at node 1. The DNO has to make reinforcement decisions on these 59 cable connections. Note that network stations at nodes 3, 4, 5, 17, 19, 20, 29 and 31 are connected with distributed generation facilities. To evaluate an expansion plan for such an electricity grid with DGs, we need to verify the feasibility of that plan against the 2 situations that can cause the heaviest power flows on the grid as follows.

- Dominance of load: 100% peak load occurs at all nodes and all DG facilities are turned off.
- Dominance of DG: All DG facilities are turned on and operate at their full 100% generation capacity while the demand at all nodes are only 25% of the peak load.

Details about the peak loads, generation capacities of DGs, and electric cables can be found in the Appendix. Here, we consider a planning horizon of $T = 30$ years.

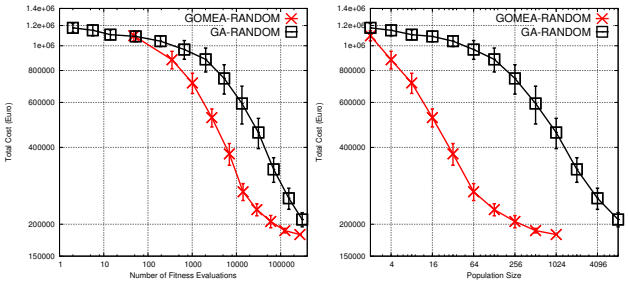

Figure 3: Performance of GA and GOMEA on solving DNEP, averaged over 30 independent runs. Vertical axis: Total cost (in EUR). Horizontal axis: Number of fitness evaluations (left) and Population size (right).

5.2 Results & Discussion

5.2.1 *Random Initialization*

Figure 3 compares the performance of the classical GA and GOMEA on solving a DNEP problem instance with random initialization. GOMEA clearly outperforms GA in terms of obtaining solutions of better quality using fewer network evaluations. It should be noted that GOMEA reaches near-optimal results and that although the GA seems to overtake GOMEA for even larger population sizes, this will not be the case. This will also be corroborated by results further-on in this paper. The classical GA requires much larger population sizes and longer computation time until convergence toward solutions whose fitness values are close to those found by GOMEA. The fact that the standard deviations of the results found by GOMEA are smaller than those of GA indicate the better reliability of GOMEA compared to GA. We thus propose that it can be beneficial to employ state-of-the-art EAs, which involve linkage learning, to solve industrial optimization problems rather than just relying on traditional EAs available in computation software.

5.2.2 *Restricted Initialization*

Figure 4 shows the performance of the classical GA and GOMEA on solving a DNEP problem instance with the restricted initialization mentioned in Section 4.1. We performed experiments for $k = 2, 4, 8, 16$ (the number of positions that a candidate solution can different in from the currently existing network). First, it can be seen that restricted initialization greatly reduces the diversity of the population, causing both optimizers GA and GOMEA to quickly converge. With a large enough population size (i.e. > 16), optimizers with restricted initialization can obtain solutions of better quality and using fewer fitness evaluations than when the same optimizers of similar population sizes are initialized randomly. Note that using too small population sizes (i.e. ≤ 16) may lead the optimizers to infeasible solutions, which can not be seen from the results in the graph. The bottom left graph in Figure 4 indicates that the best found solution is still the one obtained by GOMEA with random initialization that required longer computation time. Lastly, we concede that under the effect of restricted initialization GOMEA does not have any advantages over the classical GA for the tested network. This is likely because the number of meaningful mixing events is significantly limited with restricted initialization. While the optimum consisting of only a few incremental changes is not hard to

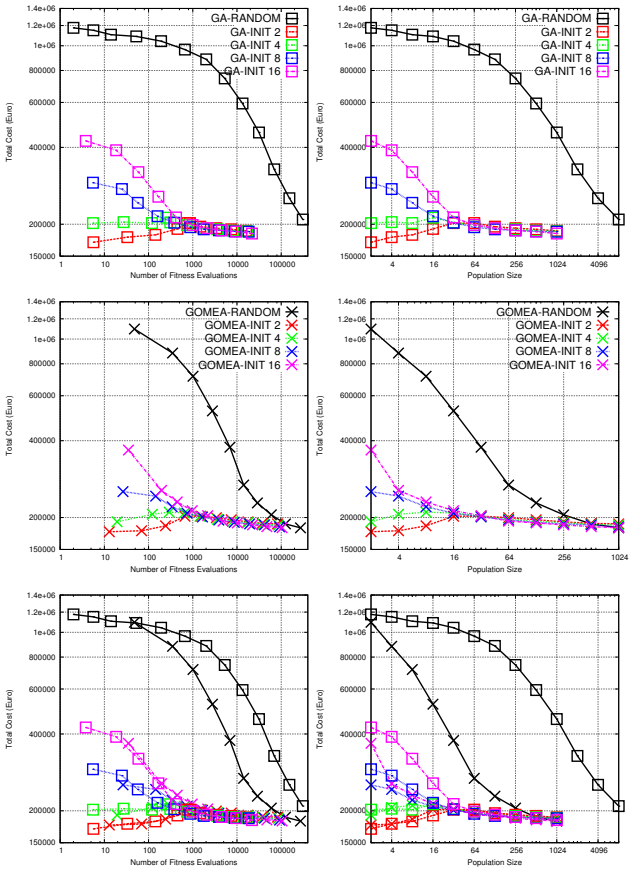

Figure 4: Performance of GA and GOMEA on solving DNEP with restricted initialization. Vertical axis: Total cost (in EUR). Horizontal axis: Number of fitness evaluations (left) and Population size(right).

find from the initial population, many more different mixing events, which are now largely superfluous, are attempted by GOMEA by traversing the whole linkage tree. Furthermore, solutions found by GOMEA and GA are of similar quality but GOMEA takes more fitness evaluations until the whole population converges. Note that the difficulty of a DNEP problem instance may be strongly related to not only the network size but also the required level of network changes needed to obtain the optimum. In this paper, our benchmark network contains 59 decision variables (i.e. cable connections) but only a small number of reinforcements (in this case, 5) are needed to transform the existing grid into the optimum. In related work [6], we reduced the search space by employing electricity expert knowledge, and obtained similar results, indicating that the initial population might be quite close to the optimum for this benchmark network.

5.2.3 *Restricted Initialization & Linkage Filtering*

Figure 5 shows the performance of GOMEA when combined with restricted initialization and linkage filtering as mentioned in Section 4. Table 1 shows an example result of GOMEA with restricted initialization of 16 different components, when coupled with linkage filtering for $L_{FOS}^{max} = 16$, i.e. removing linkage groups containing more than 16 variables out of the LT. Our simple linkage filtering slightly improves the quality of the obtained solutions. Different values

Table 1: Performance of GOMEA when coupled the restricted initialization of 16-different components and filtering of linkage groups with $L_{FOS}^{max} = 16$

	INIT 16		INIT 16-MAX 16	
Pop.Size	Objective	#Evaluations	Objective	#Evaluations
2	367624.54	35.13	355677.71	38.4
4	255586.89	196.5	253368.71	253.06
8	230316.22	445.6	223857.87	738.9
16	212774.28	971.13	200677.04	1646.86
32	203419.42	2172.5	194684.77	3543.93
64	193616.43	4820.43	189295.7	7536.06
128	190555.51	10202.93	184798.1	15573.33
256	187123.88	22471.76	183858.28	32145.13
512	184270.65	47435.8	182483.36	63931.13
1024	182506.32	104579.1	182097.35	116233.86

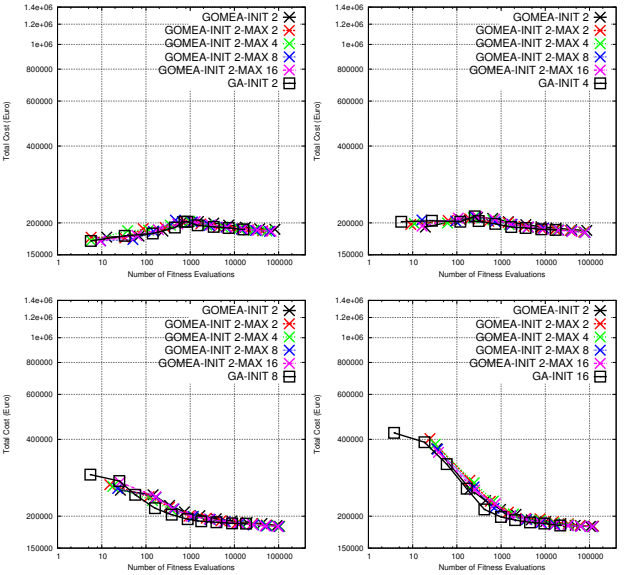

Figure 5: Performance of GOMEA in solving DNEP with restricted initialization and linkage filtering. Vert. axis: Total cost (in EUR). Horiz. axis: Number of fitness evaluations.

(2, 4, 8 or 16) of the linkage filtering threshold L_{FOS}^{max} do not result in significant differences. However, from closer observation it appears that the larger L_{FOS}^{max} is, the better the obtained solutions are. Figure 5 shows that although linkage filtering with threshold L_{FOS}^{max} improves GOMEA with restricted initialization, it is still not enough to accelerate GOMEA to outperform the restrictedly initialized classical GA. We emphasize that the better performance of GA here might be very specific to this benchmark network, where the required level of changes is very small compared to the total number of decision variables, and restricted initialization simply generates the initial population very close to the optimal solution. Therefore, future works are required to consider more complicated benchmarks with higher level of necessary changes to give a more accurate answer on the scalability of GA and GOMEA on the DNEP problem.

6. CONCLUSIONS

We measured the performance of the two population-based optimization algorithms: a classical GA, and a variant of a state-of-the-art linkage-learning EA, GOMEA, on solving the DNEP problem. Without problem-specific assumptions, starting from randomly generated populations, GOMEA was

found to outperform GA. Considering DNEP common practice, we then used a restricted-initialization mechanism to reduce the size of the explorable search space and to likely have the initial population close to the optimum. With restricted initialization, GA was found to outperform GOMEA. Aiming to accelerate GOMEA, we then introduced a simple linkage filter that disregards overly large linkage groups. Although accelerations are indeed observed, this simple linkage filtering is still not enough for GOMEA outperform the simple GA in terms of number of evaluations until convergence when solving a specific DNEP problem instance. We conclude that for easy problem instances, where the optimum is close to the initial population, it is more beneficial to use the simple GA. However, in general cases, as shown in experiments with random initialization, GOMEA is found to be the better optimizer. To study the scalability of GA and GOMEA on the DNEP problem, future works also need to consider realistic, tunable and scalable distribution-network benchmarks that are more meshed and more complex.

7. REFERENCES

[1] P. A. N. Bosman and D. Thierens. More concise and robust linkage learning by filtering and combining linkage hierarchies. In *Proc. of the Genetic and Evolutionary Comp. Conf. - GECCO-2013*, pages 359–366. ACM, 2013.

[2] Y. Del Valle, G. Venayagamoorthy, S. Mohagheghi, J.-C. Hernandez, and R. Harley. Particle swarm optimization: Basic concepts, variants and applications in power systems. *IEEE Trans. on Evol. Comp.*, 12(2):171–195, April 2008.

[3] E. Diaz-Dorado, J. Cidras, and E. Miguez. Application of evolutionary algorithms for the planning of urban distribution networks of medium voltage. *Power Systems, IEEE Transactions on*, 17(3):879–884, Aug 2002.

[4] J. J. Grainger and W. D. Stevenson. *Power System Analysis*. McGraw-Hill Education, 2003.

[5] I. Gronau and S. Moran. Optimal implementations of UPGMA and other common clustering algorithms. *Information Processing Letters*, 104(6):205 – 210, 2007.

[6] M. O. W. Grond, N. H. Luong, J. Morren, P. A. N. Bosman, J. G. Slootweg, and H. La Poutré. Practice-oriented optimization of distribution network planning using metaheuristic algorithms. In *18th Power Systems Computation Conference (PSCC'14), Wroclaw, Poland*, 2014 (submitted).

[7] G. Harik, E. Cantú-Paz, D. E. Goldberg, and B. L. Miller. The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Evol. Comp.*, 7:231–253, 1999.

[8] N. H. Luong, M. O. W. Grond, P. A. N. Bosman, and H. La Poutré. Medium-voltage distribution network expansion planning with gene-pool optimal mixing evolutionary algorithms. In *Biennial Int. Conf. on Artificial Evolution (EA-2013)*, pages 96–107, 2013.

[9] J. Schlabbach and K.-H. Rofalski. *Power System Engineering (chapter 4)*, pages 37–44. Wiley-VCH Verlag GmbH & Co. KGaA, 2008.

[10] D. Thierens and P. A. N. Bosman. Optimal mixing evolutionary algorithms. In *Proc. of the Genetic and Evol. Comp. Conf. - GECCO-2011*, pages 617–624. ACM, 2011.

[11] D. Thierens and D. E. Goldberg. Mixing in genetic algorithms. In *Proceedings of the 5th International Conference on Genetic Algorithms*, pages 38–47, San Francisco, CA, 1993. Morgan Kaufmann Publishers Inc.

[12] R. A. Verzijlbergh, M. O. W. Grond, Z. Lukszo, J. G. Slootweg, and M. D. Ilic. Network impacts and cost savings of controlled EV charging. *Smart Grid, IEEE Transactions on*, 3(3):1203–1212, Sept 2012.

[13] K. P. Wong. Solving power system optimization problems using simulated annealing. *Engineering Applications of Artificial Intelligence*, 8(6):665 – 670, 1995.

APPENDIX

Cable Parameters			
ID	R $[\Omega/km]$	X $[\Omega/km]$	C $[\mu F/km]$
1	0.257	0.085	0.38
2	0.20858	0.09592	0.3833
3	0.13517	0.10823	0.43443
4	0.08077	0.09972	0.5344
5	0.0511	0.09272	0.64103

Node Information				
Node	Load		DG	
[#]	P [kW]	Q [kVAR]	P [kW]	Q [kVAR]
1	0	0		
2	35	17		
3	1113	539	-1960	-398
4	348	216	-980	-199
5	871	286	-1960	398
6	332	109		
7	132	82		
8	170	82		
9	22	14		
10	202	98		
11	120	0		
12	88	55		
13	284	137		
14	219	136		
15	314	152		
16	185	90		
17	127	79	-980	-199
18	17	8		
19	896	434	-1960	-398
20	314	152		
21	125	77		
22	248	120		
23	85	41		
24	123	76		
25	209	130		
26	566	274	-1960	-398
27	266	129		
28	126	61		
29	360	174	-980	-199
30	273	169		
31	263	163	-980	-199

Cable Distances			
Existing		Estimated	
Branch [#]	Distance [m]	Branch [#]	Distance [m]
1-2	481	1-3	676
1-16	246	1-4	621
1-31	761	1-5	1199
2-3	96	1-6	1306
3-4	48	1-7	980
4-5	498	1-8	1465
5-6	86	1-9	1551
6-7	288	1-10	2135
7-8	935	1-11	2121
8-9	200	1-12	2121
9-10	470	1-13	1635
10-11	851	1-14	1386
11-12	220	1-15	1144
12-13	300	1-17	2218
13-14	284	1-18	2163
14-15	479	1-19	1988
15-16	846	1-20	2003
10-17	736	1-21	1751
17-18	101	1-22	1622
18-19	154	1-23	1720
19-20	283	1-24	1648
20-21	308	1-25	1535
21-22	133	1-26	1451
22-23	132	1-27	1348
23-24	138	1-28	1315
24-25	140	1-29	1057
25-26	103	1-30	1124
26-27	215		
27-28	129		
28-29	218		
29-30	136		
30-13	160		