Exploiting Linkage Information and Problem-Specific Knowledge in Evolutionary Distribution Network Expansion Planning

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

This paper tackles the Distribution Network Expansion Planning (DNEP) problem that has to be solved by distribution network operators to decide which, where, and/or when enhancements to electricity networks should be introduced to satisfy the future power demands. We compare two evolutionary algorithms (EAs) for optimizing expansion plans: the classic genetic algorithm (GA) with uniform crossover and the Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) that learns and exploits linkage information between problem variables. We study the impact of incorporating different levels of problem-specific knowledge in the variation operators as well as two constraint-handling techniques: constraint domination and repair mechanisms. Experiments show that the use of problem-specific variation operators is far more important for the classic GA to find high-quality solutions to the DNEP problem. GOMEA is found to have far more robust performance even when an out-of-box variant is used that doesn't exploit problem-specific knowledge. Based on experiments, we suggest that when selecting optimization algorithms for real-world applications like DNEP, EAs that have the ability to model and exploit problem structures, such as GOMEAs and estimation-of-distribution algorithms, should be given priority, especially when problem-specific knowledge is not straightforward to exploit, e.g. in the case of black-box optimization.

CCS Concepts

•Mathematics of computing → Evolutionary algorithms; •Hardware → Power networks;

Keywords

Power System, Capacity Planning, Linkage Learning, Variation Operators, Problem-Specific Knowledge

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

GECCO '15, July 11 - 15, 2015, Madrid, Spain
© 2015 ACM. ISBN 978-1-4503-3472-3/15/07...$15.00
DOI: http://dx.doi.org/10.1145/2739480.2754682

1. INTRODUCTION

A typical power system consists of power plants that generate the electricity, transmission networks that carry the high-voltage (HV) electricity from far-off generating sites to substations near residential and industrial areas, and distribution networks that feed the medium-voltage (MV) and low-voltage (LV) electricity from the substations to homes and businesses. In this paper, the focus is on distribution networks, but the methodologies can also be extended to transmission networks. In order for distribution networks to work properly, distribution network operators (DNOs) have to ensure that the capacities of network cables are enough to handle the magnitude of the power flows that are carried through the cables to satisfy customers' power demands. Otherwise, bottlenecks can cause overload, which heats up the cable wires. This is detrimental to the normal operation and safety of the networks, and may cause blackouts or earlier cable replacements. Therefore, DNOs need to perform Distribution Network Expansion Planning (DNEP) to determine *what* kinds of network enhancements should be made and *where* these enhancements should be made. The dynamic DNEP formulation also involves the question *when* those reinforcement activities should be started during the planning horizon while in the static DNEP formulation this time-dependent decision making issue is omitted. The goal of DNEP is to find the most economical expansion plan, in terms of investment and operation costs, for which the network satisfies the power demand over the planning horizon.

DNEP is sometimes simplified to be scalably solved by classical mathematical programming methods, compromising the true non-linear nature of DNEP that is due to its complicated constraints, and thus leading to unsatisfactory representations of the real problem [9]. On the other hand, evolutionary algorithms (EAs) have been widely applied and achieved satisfying results in DNEP with more realistic formulations, see e.g. [1, 3, 9]. This is mostly due to the straightforward implementation and broad applicability of EAs. However, most DNEP researches in literature overlook several important issues. First, experiments are usually conducted by using only one, arbitrarily chosen, EA with a customized problem-specific variation operator (VO), omitting both questions why that specific EA should be chosen over other available EAs and what the advantages that VO has compared to other alternatives. Second, the comparison of how effective various constraint-handling mechanisms help the solvers traverse the search space is often disregarded. In this paper, while aiming to solve the DNEP problem, we also address these issues. We employ 2 EA solvers: the classic genetic algorithm (GA) and a Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) [2, 10]. GA is arguably the most popular EA in DNEP literature, but it is rarely used out of the box in practice. Practitioners often have to customize GA's VOs (i.e. crossover and mutation) with expert and problem-specific knowledge (PSK) so that important problem structures are respected during variation, e.g. cables in the same feeder group in the network should be treated together when constructing new networks. Taking the perspective of black-box optimization, where such PSK is hardly available, GOMEA performs linkage learning (LL) to identify, during optimization, which variables are interdependent and considers them jointly when generating new solutions. Being a recently-developed LL EA, GOMEA has been shown to have superior performance and scalability on laboratory benchmarks and recently in power system optimization as well [6, 8]. The LL capability of GOMEA does not exclude the possibility of combining linkage knowledge with PSK if available. In this paper, we show how to combine the strength of LL with PSK exploitation.

Population sizes of EAs are often chosen arbitrarily or are customized to the specific problem instance at hand in DNEP literature [1, 3, 9]. This approach is difficult to generalize to applications elsewhere. Population size parameter settings have big impacts on how effective and how efficient problem instances are solved. Practitioners often need to manually try different population sizes to figure out a suitable population size for each problem instance, which is both time-consuming and difficult for comparing the performance of different EAs fairly. To get rid of this troublesome parameter, we adapt a population sizing-free scheme proposed in [7] and implement it into both GA and GOMEA.

The remainder of this paper is organized as follows. Section 2 formulates the DNEP problem. Section 3 outlines the two optimization algorithms GA and GOMEA together with the population sizing-free scheme. Section 4 introduces different variation operators and constraint-handling techniques for DNEP that can be incorporated into GA and GOMEA. Section 5 exhibits the experimental results. Section 6 concludes the paper.

2. DNEP FORMULATION

In this paper, we focus on optimizing expansion plans for medium voltage distribution (MV-D) networks. A typical MV-D network consists of cables branching out from HV/MV substations connecting MV nodes (MV/LV substations and MV customer substations) [6]. These cables form different feeder (cables) groups in ring-shaped or meshed structures. However, some cables are opened on one side, called normally open points (NOPs). Electricity cannot flow through those cables, so that every node is supplied its power demand through a single feed path. The whole network thus operates radially in normal situations. Figure 1 shows an example of a distribution network. We focus on expansion options for network cables as our main asset category. How the cables are connected and what types of cables are used, determine the capacity of the network. Increasing power demands in the future can create bottlenecks in some parts of the network, where the power flows have magnitudes that are greater than the nominal capacities of the cables. The goal of capacity planning is to find the best expansion op-

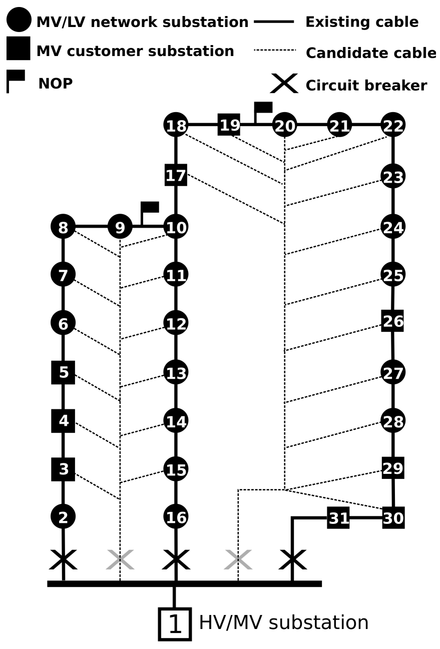

Figure 1: A distribution network example [6]

tions to solve these bottlenecks. Possible expansion options are: replacing existing cables with new cables of higher capacities or adding new cable connections and thus creating new feed paths to the network [6]. Adding new cables connections requires placements of new NOPs to satisfy the radiality constraint [6].

2.1 Decision Variables

We specify all the existing and potential cable connections (branches) that we want to consider for capacity planning. Let l denote the total number of branches. A distribution network can be represented as a vector of l integer elements:

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

where $\Omega(x_k)$ is the set of cable types that can be installed at the k^{th} branch ($\Omega(x_k) \subseteq \mathbb{N}$). The value of x_k indicates the status and the type of cable installed:

- $x_k = ID > 0$: A cable of type $ID \in \Omega(x_k)$ is installed.
- $x_k = 0$: No cable is installed at the k^{th} branch.
- $x_k = -ID < 0$: A cable of type $ID \in \Omega(x_k)$ is installed but is out of normal operation. This is an NOP.

2.2 Constraints

Given the forecast growth of the peak load at each node, the following constraints must be satisfied:

1. **Connectivity**: All nodes must be connected.
2. **Power flow constraint**: In normal operation, the voltage at each node stays within allowable limits and the magnitude of the power flow through each cable stays within the nominal capacity of that cable.
3. **Radiality constraint**: In normal operation, the power demand of each node is supplied by a single feed path.
4. **Reconfigurability constraint**: When an active cable ($x_k > 0$) fails, the part of the feeder group (from an HV/MV substation with circuit breaker to an NOP)

containing the failed cable is disconnected from the network. All customers connected to that feeder group are then out of service. The DNO has to bring the network back to operation by closing NOPs to temporarily re-route the power flow through other paths while the failed cable is being repaired. During this emergency situation, the radial operation constraint can be compromised and the network is allowed to endure a mild overload of 130% nominal capacity (as in [6]).

5. **Substation capacity constraint**: Each HV/MV substation has limited physical space to install new outgoing cables. We assume that maximum 3 new outgoing cables are allowed for each HV/MV substation (as [6]).

Alternating-current (AC) power flow calculations (PFCs [4]) are required to check constraints 2 and 4 for each solution plan. PFCs, which involve solving AC power flow models, are computationally expensive and dominate the computing time of the optimization process. Note that, due to the implementation, PFCs can only be performed for connected networks and the connectivity constraint is thus a crucial constraint that needs to be separately handled so that constraints 2 and 4 can be properly evaluated (see Section 4). Constraint evaluations are performed together with objective evaluations.

2.3 Objective Function

We employ the annuity method [11] to calculate the capital expenditures $CAPEX$ for new assets. The investment cost on a new asset is converted into a series of uniform annual payments, called annuities. We assume the length of this series to be equal to the (uniform) economic lifetime of the new asset t_{life}. The annuity AN_a of the asset a with a discount rate $i = 4.5\%$ (as in [6]) can be computed as:

$$AN_a = Price_a \cdot \frac{i}{1-(1+i)^{-t_{life}}} \tag{2}$$

$CAPEX$ for the asset a in a year t can be calculated as:

$$CAPEX_a(t) = \begin{cases} AN_a & \text{if } t_{inst_a} \le t < t_{inst_a} + t_{life} \\ 0 & \text{else} \end{cases} \tag{3}$$

with t_{inst_a} is the time of installing the asset a. Let A denote the set of all new assets a's that will be installed in the planning period $[t_0, t_{horizon}]$. The total $CAPEX$ on the whole network in a year t is defined as:

$$CAPEX(t) = \sum_{a \in A} CAPEX_a(t) \tag{4}$$

The operational expenditure $OPEX$ can be calculated by capitalizing the energy loss. Energy loss of the network in a year t can be taken as in [6, 11]:

$$E_{loss}(t) = P_{peak\ loss}(t) \cdot T_{loss}(t) \tag{5}$$

where $P_{peak\ loss}(t)$ is the peak loss which can be obtained from the PFC regarding the peak loads in year t. $T_{loss}(t)$ is the service time of peak loss for year t, defined by the area of the yearly loss profile [6, 11]. Given the forecast electricity price in a year t, we can capitalize the energy loss and regard it as the $OPEX$ in year t.

$$OPEX(t) = E_{loss}(t) * Price_{electricity}(t) \tag{6}$$

2.3.1 *Total Cost Formulation*

We want to minimize the net present value (NPV) (i.e. at time t_0) of the total cost of both investment cost $CAPEX$ and operation cost $OPEX$ over a planning horizon of $t_{horizon}$ years with a discount rate i.

$$COST_{NPV} = \sum_{t=t_0}^{t_{horizon}} \frac{CAPEX(t) + OPEX(t)}{(1+i)^{t-t_0}} \tag{7}$$

2.3.2 *Static Planning*

The goal in static DNEP is to find the most economical solution plan that satisfies the peak load profile at the final year of the planning horizon. Static DNEP gives DNOs a general picture about what kinds of network reinforcements can be expected. However, the actual total cost objective function includes the important operation cost $OPEX$, which is the capitalized energy loss, that depends on the load profile and the specific network configuration at each year. We employ the following method, which has been proposed in [6], to estimate the $OPEX$ for a solution plan. Based on the (predicted) annual peak load growth rate R, we compute the peak load profile for each year from the beginning year t_0 until the final year $t_{horizon}$. We determine the year $t_{overload}$ when the first bottleneck happens in the network. We then assume that all expansion options in the solution plan are installed at the same time in the year $t_{overload}$, i.e. $t_{inst} = t_{overload}$ for all new assets. Therefore, to evaluate the total cost, from t_0 until $t_{overload}$, we use the current network topology, and from $t_{overload}$ until $t_{horizon}$ we switch to the new network topology at $t_{horizon}$.

2.3.3 *Dynamic Planning*

The question *when* each expansion options should be carried out is also an important issue that DNOs have to address. Dynamic DNEP formulations often contain time-dependent decision variables and complicated problem models that are much more difficult to solve than those of static DNEP. However, network cables (i.e. the main asset category in DNEP) generally have very long lifetimes (e.g. normally 30 years) while planning horizons of more than 30 years are regarded as impractical because it is difficult to properly predict power demand scenarios for a very distant future. These facts mean that $t_{inst}+t_{life} > t_{horizon}$, i.e. during a practical planning horizon, a network branch might require reinforcement only once (e.g. cable replacement or installation of a new cable connection). Situations such as installing a thin cable first and then upgrading with a different cable of higher capacity, or removing a previously-upgraded cable during the same planning horizon, are often regarded as impractical by DNOs. Thus, we propose a method to convert a static plan into a dynamic one, called the *decomposition heuristic*, as follows. Similarly to the static planning, we determine the year $t_{overload}$ when the first bottleneck occurs and assume that the whole solution plan is carried out in that year. We then loop through the list of all new assets in the solution plan in a random order and check if we can delay each expansion option by one year. If the postponement of an asset investment results in less total cost and all constraints are still satisfied, then we accept that postponement. Otherwise, that expansion option cannot be postponed further. We continue this postponement checking until no expansion option can be postponed any more. Finally, we obtain a solution plan with an investment time for each expansion option and the total cost objective value is evaluated accordingly. By employing this *decomposition heuristic*, we can bring the decision making about investment time into the static DNEP framework while still satisfying the requirements of DNOs in real-world practice.

3. OPTIMIZATION ALGORITHMS

Problem structures of DNEP are difficult for low-order local search (LS) to solve. For example an expansion option of adding a new cable connection requires the addition of a new NOP to make the network radial, which in turn requires relocations of other existing NOPs to obtain the topology with lowest energy loss. Such multivariate decision making cannot be found by a typical LS like bit-flip hillclimbing while higher-order LS would not be scalable. Global search meta-heuristics, like EAs, are more suitable for DNEP.

We will consider two EAs: the classic GA and a GOMEA variant, along with a population sizing-free scheme that can be integrated with any population-based optimization algorithm. Normally, EAs, like GA or GOMEA, start with randomly initialized populations. However, because DNEP is a highly-constrained engineering problem, randomly generated solutions are typically infeasible and violate many constraints. Therefore, we use a repair procedure that partially repairs infeasible solutions by comparing them with the current, i.e. starting, network situation. First, each decision variable x_k (i.e. a network branch) can only receive a random non-negative value (i.e. we do not place any NOPs yet) as long as it does not downgrade the currently existing cable. This also means that currently existing cables can only be left intact or replaced with cables of higher capacities. Existing connections are rarely removed because DNOs do not favor this. Solutions that are generated in this way satisfy the connectivity constraint because the current network is connected. Second, we go through HV/MV substations and check the number of cables branching out from each substation. If the number of outgoing cables is more than the allowable capacity of the substation (i.e. violating constraint 5), we randomly delete outgoing cables until constraint 5 is satisfied. Third, we go through all variables that have positive values (i.e. active cables) in a random order. For each positive-value decision variable, we try to place an NOP on that cable by negating its value. If the network is still connected, then the NOP can be placed; otherwise, undo the operation. This procedure returns a network of radial topology with random placements of NOPs (i.e. constraint 3 is satisfied). We do not repair the power flow and reconfigurability constraint (i.e. constraint 2 and 4) because they involve PFCs, which are computationally expensive.

3.1 Genetic Algorithm

We consider a typical implementation of the classic GA with uniform crossover and tournament selection. First, a population $\mathcal{P}$ of n candidate solutions is initialized following the initialization procedure mentioned above. Next, for every generation, we create an offspring population $\mathcal{O}$ of n new solutions from $\mathcal{P}$ by performing uniform crossover on every 2 randomly selected parent solutions, giving 2 offspring solutions. Then, we combine both the parent population $\mathcal{P}$ and the offspring population $\mathcal{O}$ into a selection pool $\mathcal{P} + \mathcal{O}$ of $2n$ solutions in total. We perform a tournament selection with tournament size 4 on this pool to select n survivor solutions, which form the new parent population $\mathcal{P}$ for the next generation, ensuring convergence by logistic growth of the optimal solutions [10]. Note that GAs in literature often have mutation operators, but we do not implement mutation here because mutation was not found to have positive impacts on the performance of GA solving our DNEP model.

3.2 Gene-pool Optimal Mixing EA

The performance of EAs depends on their ability to efficiently mix and preserve building blocks (BBs) (i.e. good partial solutions) in the population to create new solutions [10]. When just taken out of the box, classic EAs, like GA, are prone to disrupt these BBs due to their BB-blind variation operators (VOs). Practitioners then have to make them BB-aware by customizing them with problem-specific knowledge (PSK). If such valuable PSK is not available, BB information can be inferred from the working population of EAs by linkage learning (LL) procedures, in which problem variables having some degree of dependency are identified so that they can be jointly considered when generating new solutions. GOMEA [2, 10] is a recently developed LL EA that effectively exploits the learned linkage information.

We consider a popular variant of GOMEA that uses the Linkage Tree (LT) [2, 10] to model linkage. An LT is a set of linkage sets arranged in a hierarchical structure. Each linkage set contains variables having some degree of dependency. The lowest level (i.e. leaf nodes) contains singleton (univariate) sets of each variable separately. Other linkage sets at higher levels are multivariate sets that can be formed by merging pairs of lower sets until all sets are merged into a root node containing all variables. Thus, an LT can encode different levels of dependency, from the totally independent state (i.e. leaf nodes) to the all-dependent state (i.e. root node). The root node is removed from the LT because it assumes all variables should be jointly considered by VOs, which means no new solution is created. Note that while bit-flip hillclimbing is not generally useful for solving DNEP, the leaf nodes are still kept here because a solution plan might contain the expansion option of upgrading an existing cable, which is a univariate decision. An LT over a set of problem variables $\{1, 2, 3, 4, 5\}$ can be, for example, $\{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{1, 3\}, \{2, 5\}, \{1, 3, 4\}\}$. Given a set of l variables and a population of n solutions, an LT can be constructed efficiently by the Unweighted Pair Grouping Method with Arithmetic-mean (UPGMA) in $\mathcal{O}(nl^2)$ time [5]

We also start GOMEA with a population $\mathcal{P}$ of n candidate solutions generated by the initialization procedure mentioned above. Every generation, we use tournament selection with tournament size 2 to select a set $\mathcal{S}$ of n solutions out of $\mathcal{P}$. The linkage model building procedure is then performed on $\mathcal{S}$ to build the LT. Using the obtained LT, we transform each existing parent solution $\boldsymbol{p} \in \mathcal{P}$ into a new offspring solution $\boldsymbol{o} \in \mathcal{O}$ whose fitness value is equal to or better than the fitness value of $\boldsymbol{p}$. The offspring population $\mathcal{O}$ completely replaces $\mathcal{P}$ and becomes the new parent population $\mathcal{P}$ for the next generation.

Instead of fully creating new solutions and then evaluating them like in GA, the VO of GOMEA, called Gene-pool Optimal Mixing (GOM [10]), uses the learned LT to evolve each existing parent $\boldsymbol{p}$ into a new offspring $\boldsymbol{o}$ in an iterative manner. First, $\boldsymbol{o}$ and a backup $\boldsymbol{b}$ are cloned directly from $\boldsymbol{p}$. Then, each linkage set in the LT is traversed iteratively. For each linkage set, a donor $\boldsymbol{d}$ is randomly selected from the current population $\mathcal{P}$. If the donor $\boldsymbol{d}$'s values for the variables indicated the linkage set differ from those in $\boldsymbol{o}$ in at least one position, these values are copied from $\boldsymbol{d}$ into $\boldsymbol{o}$. This partially-altered solution $\boldsymbol{o}$ is evaluated and compared against its backup $\boldsymbol{b}$. If $\boldsymbol{o}$ is equally good or better than $\boldsymbol{b}$ (i.e. $fitness[\boldsymbol{o}] \geq fitness[\boldsymbol{b}]$), the changes are accepted (i.e. the values copied from $\boldsymbol{d}$) and updated into $\boldsymbol{b}$ as well. Otherwise, the changes are undone and $\boldsymbol{o}$ reverts to its backup state $\boldsymbol{b}$. Note that the acceptance of solutions having equal fitness can be beneficial to move across a fitness plateau [2]. It can be seen that each linkage set corresponds with a mixing event, in which the current solution is recombined with a random donor solution and the variables in the same linkage set are treated together, preserving the BB structure. When we traverse the whole LT, an offspring $\boldsymbol{o}$ is then fully constructed, replacing the original parent $\boldsymbol{p}$ in the next generation.

It can happen that GOM cannot improve the current parent solution $\boldsymbol{p}$ into a new offspring or that, because of a significant plateau, GOM keeps transforming back and forth solutions of different genotypes but with the same fitness value. To overcome this, if GOM cannot yield a new offspring or when the number of subsequent generations that the best-found-so-far solution $\boldsymbol{x}^{best}$ does not change, i.e. the no-improvement stretch (NIS), exceeds a certain threshold, we invoke the Forced Improvement (FI [2]) procedure. In essence, FI is similar to GOM but we always use $\boldsymbol{x}^{best}$ as the only donor solution. FI only accepts the mixing event that results in a strict improvement (i.e. $fitness[\boldsymbol{o}] > fitness[\boldsymbol{b}]$) and FI stops as soon as such mixing event is found. Previous research [2] on GOMEA suggest a threshold for NIS of $1 + \lfloor \log_{10}(n) \rfloor$. If FI does not succeed in evolving $\boldsymbol{p}$, $\boldsymbol{x}^{best}$ is returned as the new offspring.

3.3 Population Sizing-free Scheme

Setting the population size parameter in real-world applications of EAs is hard because a suitable population size setting depends on the structure of the problem instance at hand and also on the specific EA being used. DNEP practitioners often need to experiment with different population sizes. Here, we adapt a population sizing-free scheme, proposed and tested on academic benchmarks in [7]. In essence, we run multiple instances of the EA in parallel. Each instance has a different population size but larger populations have a slower generational cycle. We start with the first population P_1 of some small size n_1. Then, by doubling the population size, the next population P_i is twice as large as the previous one, i.e. $n_i = 2n_{i-1}$ for $i > 1$. All the populations are scheduled with the principle that for every 2 generations of population P_i, 1 generation of population P_{i+1} is run (or initialized if it does not exist yet). A population is terminated when it converges, i.e., all solutions in that population have the same fitness value. Having no maximum population size, the EA runs and grows its populations until the computing time budget is used up.

4. VARIATION OPERATORS

Being popularly applied in black-box optimization, EAs require little problem-specific knowledge (PSK) and their variation operators (VOs) (i.e. procedures to generate new offspring solutions) can operate on a wide range of problem landscapes. However, in real-world applications like DNEP, the problems are often highly constrained such that it is difficult for general-purpose VOs to traverse the search space of feasible solutions efficiently. For industrial optimization, efficiency is of high importance because the evaluations of candidate solutions are typically computationally expensive. Full constraint evaluations of solution plans for DNEP involve the simulation and solving PFCs [4], which dominate the computing time of the optimization process. Thus, it is beneficial to incorporate PSK into VOs of EAs as efficiency enhancement methods. Local search heuristics can normally be used for efficiency enhancement, but typical operators like bit-flip hillclimbing was not found to be helpful in improving the efficiency of EAs solving our DNEP model because such local search often fails to reach expansion options of interdependent activities (e.g. adding new cables and NOPs and relocating existing NOPs).

Among the constraints of DNEP, the connectivity constraint needs to be handled separately because the PFC cannot be performed on unconnected networks and the power flow constraint and reconfigurability constraint cannot be checked without PFCs. Thus, in order to compare solution plans (e.g. when performing tournament selection), we need to quantify their disconnectivity and use that as the comparison criterion or we have to repair the connectivity so that other constraints can be evaluated. Here, we introduce different VOs for GA and GOMEA along with their corresponding connectivity constraint-handling techniques.

4.1 Disconnectivity Quantification

The VO presented here can be considered as the out-of-the-box VO of GA or GOMEA because it does not assume any connectivity knowledge when generating offspring solutions. If the network encoded in a solution plan is unconnected, we do not evaluate other constraints but we quantify its disconnectivity by comparing it with the topology of the existing network. Loop through all the decision variables and count the number of positions where the existing network has a positive value (i.e. an active branch) but the solution has a negative value (i.e. an NOP) or where the existing network has a non-positive value (i.e. no cable connection or an NOP) and the solution has a positive value. This number is considered as the disconnectivity value. Note that connected networks do not need this disconnectivity quantification and are assigned the disconnectivity value 0. Then, we can compare candidate networks as follows. If both networks are unconnected, the one with a smaller disconnectivity value is selected as the better one. If only one network is unconnected, then the connected network is the better solution. If both networks are connected, then they can be compared by using the other evaluated constraint values and their objective values. We call the VO used together with this connectivity-constraint domination mechanism DQ1.

However, even if we recombine two connected parent networks, it is still difficult for the uniform crossover operator of GA to generate connected offspring networks, especially for big networks with many cable connections. Thus, we also propose a different VO, in which we allow each recombination of two parent networks to retry uniform crossover to generate connected offspring networks. After 100 times, if the offspring networks are still unconnected, they will be evaluated for the disconnectivity value as above. For GOMEA, during the process of constructing an offspring, for each mixing event, if the partially-altered solution is an unconnected network, we allow it to select randomly a different donor maximum 100 times. We call this VO DQ100.

4.2 Connectivity Repair

For GA, this VO is much like DQ100, but after 100 trials, if the networks are still unconnected they will be reverted back to the parent solutions. Because all the candidate solutions in the initial population are connected networks, the use of this VO implies that only connected offspring are allowed to be evaluated and enter tournament selection. For GOMEA, in each mixing event, if the partially-altered solution becomes unconnected, this is because there exist some variables whose positive values are replaced by some non-positive values from the donor. We can simply revert these decision variables to their backup values. We call the VO that uses this connectivity repair scheme CRP (i.e. Connectivity Repair by Parent).

We also propose a different connectivity repair scheme that compares an unconnected network $\boldsymbol{x}$ with the (overall) best-found-so-far solution $\boldsymbol{x}^{best}$. For each decision variable k, if $x_k^{best} > 0$ and $x_k < 0$, then $x_k \leftarrow -x_k$; if $x_k^{best} > 0$ and $x_k = 0$, then $x_k \leftarrow x_k^{best}$. On the other hand, if $x_k^{best} < 0$ and $x_k > 0$, then $x_k \leftarrow -x_k$; if $x_k^{best} = 0$ and $x_k > 0$, then $x_k \leftarrow 0$. In other words, we try to transform the topology of the unconnected network to the best solution's topology. We call the VO that uses this connectivity repair scheme CRB (i.e. Connectivty Repair by the Best solution).

4.3 Branch Exchanging

This VO aims to directly generate connected offspring networks by following the principle that during the recombination of two connected networks $\boldsymbol{x}$ and $\boldsymbol{y}$, if we bring a cable connection from $\boldsymbol{x}$ to $\boldsymbol{y}$ (i.e. $x_i \leftrightarrow y_i, x_i > 0, y_i \leq 0$), we need to return a different cable connection from $\boldsymbol{y}$ to $\boldsymbol{x}$ (i.e. $x_j \leftrightarrow y_j, x_j \leq 0, y_j > 0$) and vice versa. For GA, when we recombine 2 parent solutions $\boldsymbol{x}$ and $\boldsymbol{y}$, for variables whose values are both positive (i.e. both are active cables) or both non-positive (i.e. no cable connections or NOPs), we can perform uniform crossover as normal since these positions have the same structure in both networks. For each variable i where $x_i > 0$ and $y_i \leq 0$ (or $x_i \leq 0$ and $y_i > 0$), if we perform crossover, we need to find a different variable j where $x_j \leq 0$ and $y_j > 0$ (or $x_j > 0$ and $y_j \leq 0$) such that exchanging values at variables i and j between 2 solutions $\boldsymbol{x}$ and $\boldsymbol{y}$ still maintains the connectivity of both networks. If we cannot find such a variable j then we do not perform crossover at the variable i. For GOMEA, in each mixing event, this procedure works similarly for the current solution $\boldsymbol{o}$ and a donor $\boldsymbol{d}$, but we only need to search for the variable j when $o_i > 0$ and $d_i \leq 0$ and the scope of searching for j is limited among the variables in the current linkage set. Also, we only need to maintain the connectivity of the current solution. This VO is problem-specific because it employs the connectivity knowledge of DNEP when generating new offspring. We call this VO BX (i.e. branch exchanging).

Originally, GOMEAs do not have mutation operators. We here experiment with DNEP-specific mutation. After every mixing event with a linkage set, but before the evaluation of the partially-altered solution, we go through every variable in the linkage set and perform a mutation with probability $1/l$ (l is the length of the solution). The mutated values must still maintain the connectivity of the network; otherwise the mutation is undone. In preliminary research, we also tried this mutation operator with GA but found that it did not have positive impact on GA. We call the branch exchanging VO with this mutation operator BX-M.

Table 1: Benchmark Network Size

ID	# Branches (Variables)	# Nodes	# HV/MV Substations	# Cable types
1	17	10	1	3
2	59	31	1	3
3	190	51	4	12

5. EXPERIMENTS

5.1 Benchmark Problems

We perform experiments on 3 benchmark networks of different sizes that are adapted from real distribution networks of a DNO. The sizes of benchmark networks are shown in Table 1. Details of network 1 and network 2 can be found in [8] and [6]. The maximum number of evaluations is 50000 for network 1, 100000 for network 2, and 200000 for network 3. The planning horizon is 30 years, and for each problem instance and each variation operator, GA and GOMEA are run 30 times. For computational reasons, for dynamic DNEP of network 3, the planning horizon is 10 years and each solver is run 10 times. The averaged convergence graphs showing the qualities of the obtained solutions during the optimization process from the beginning until termination are shown in Figures 2 and 3. Some graphs (especially in network 2) show a lot of fluctuation from the beginning until a certain point because we only consider feasible solutions, and it takes a different number of evaluations in each run before the first feasible solution is obtained.

5.2 Results

5.2.1 *Static DNEP*

Figure 2 shows the performance of GA and GOMEA integrated with different VOs solving the static DNEP. For networks 1 and 2, GAs and GOMEAs exhibit the same effectiveness in obtaining (near-)optimal solutions. After spending a certain number of evaluations (i.e. computing budget), all solvers have similar convergence until termination. For network 1, all GOMEA instances have a little better performance than their corresponding GA instances, reaching the same final network with slightly less evaluations. For network 2, GA-DQ100 and GA-CRP work slightly better than GOMEA-DQ100 and GOMEA-CRP. However, on small networks, these differences between GAs and GOMEAs, or between different VOs, can be seen as insignificant.

In contrast, when solving DNEP for a large network, the performance gap between GAs and GOMEAs and the impacts of different VOs are considerable. For network 3, the large DNEP instance, using the out-of-the-box VO, GA-DQ1 is unable to solve DNEP and hardly obtains any good solutions. Moreover, while also assuming no connectivity knowledge, given the same computing budget (i.e. number of evaluations), GOMEA-DQ1 clearly outperforms both GA-DQ1 and GA-DQ100, and obtains solution plans of much better quality. DQ100 gives GA multiple trials of recombination to generate new connected networks, and that indeed helps GA improve its performance a lot (but still worse than GOMEA-DQ1 in terms of the quality of obtained solutions at termination); but DQ100 can only slightly accelerate GOMEA. Similarly, while the connectivity repair VOs CRP and CRB clearly enhance the effectiveness and efficiency

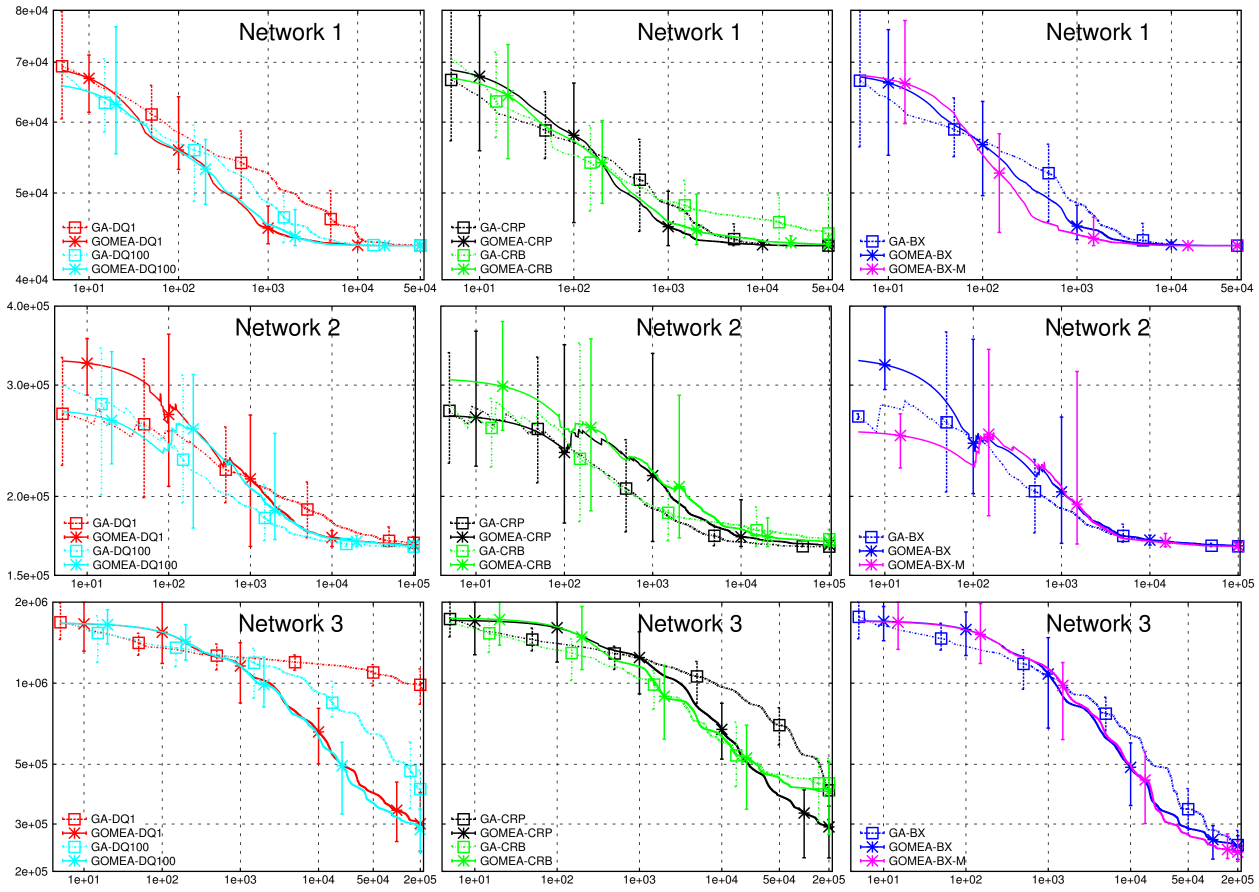

Figure 2: Static DNEP. Horizontal axis: number of evaluations. Vertical axis: NPV of total cost (EUR).

of GA, they does not show any significant improvement in speed over GOMEA-DQ1. Interestingly, repairing by parent solutions (CRP) appears to be better than repairing by the best solutions (CRB) for DNEP. This can be because it is difficult for general-purpose VOs to generate connected networks and keeping matching unconnected offspring networks with the slowly-changing best topology would reduce the beneficial diversity in the population, making the algorithm prone to premature convergence.

Solving the issue of maintaining connectivity, the customized VO branch exchanging (BX) brings out superior efficiency for EAs solving DNEP. GA-BX has excellent performance, similar to its corresponding GOMEA-BX in obtaining (near-)optimal solutions. BX also has positive impacts on GOMEA but the size of the effect is much less significant than on GA. The results here suggest 2 conclusions. First, problem-specific VOs are crucial for classic EAs, like GA, to efficiently solve (larger) real-world optimization problems. Second, EAs that can learn and exploit linkage structures, like GOMEA or estimation-of-distribution algorithms (EDAs), are indeed far more robust solvers, which can be used out-of-the-box and still obtain solutions of good quality close to those found by a customized EA with problem-dedicated VOs (i.e. comparing GOMEA-DQ1 in leftmost graphs with GA-BX and GOMEA-BX in rightmost graphs in Fig. 2). Besides, the mutation operator which we design for GOMEA here has a slight improvement over GOMEA-BX, making GOMEA-BX-M the best solver in this test case (i.e. the fastest solver given the upper bound on number of evaluations used in our experiments). While it might not be necessary for GOMEA solving many laboratory benchmarks, mutation can still be beneficial when tackling real-world problems.

5.2.2 *Dynamic DNEP*

Figure 3 exhibits the convergence performance of GAs and GOMEAs solving dynamic DNEP. The performances of GAs and GOMEAs are similar to those observed in static DNEP, suggesting that the use of the decomposition heuristic does not introduce addtional complexity to the optimization problem. GOMEAs are slightly better than their corresponding GAs in solving network 1, and GAs slightly outperform GOMEAs in solving network 2, but the these differences are insignificant. For network 3, GOMEAs have much better convergence than GAs even when the out-of-the-box variant GOMEA-DQ1 is used. The DNEP-specific branch exchanging (BX) VO is again shown to be able to greatly improve the efficiency of the classic GA while having a minor acceleration on GOMEA. GOMEA-BX-M is still the overall best solver.

6. CONCLUSIONS

This paper contributes guidelines and methodologies for the application of EAs in tackling the real-world optimization DNEP. First, we proposed a decomposition heuristic that can help available static planning solvers be able to solve the dynamic DNEP in a practical manner. Second, we suggested practitioners of DNEP to employ population sizing-free schemes to get rid of the notoriously-difficult-to-

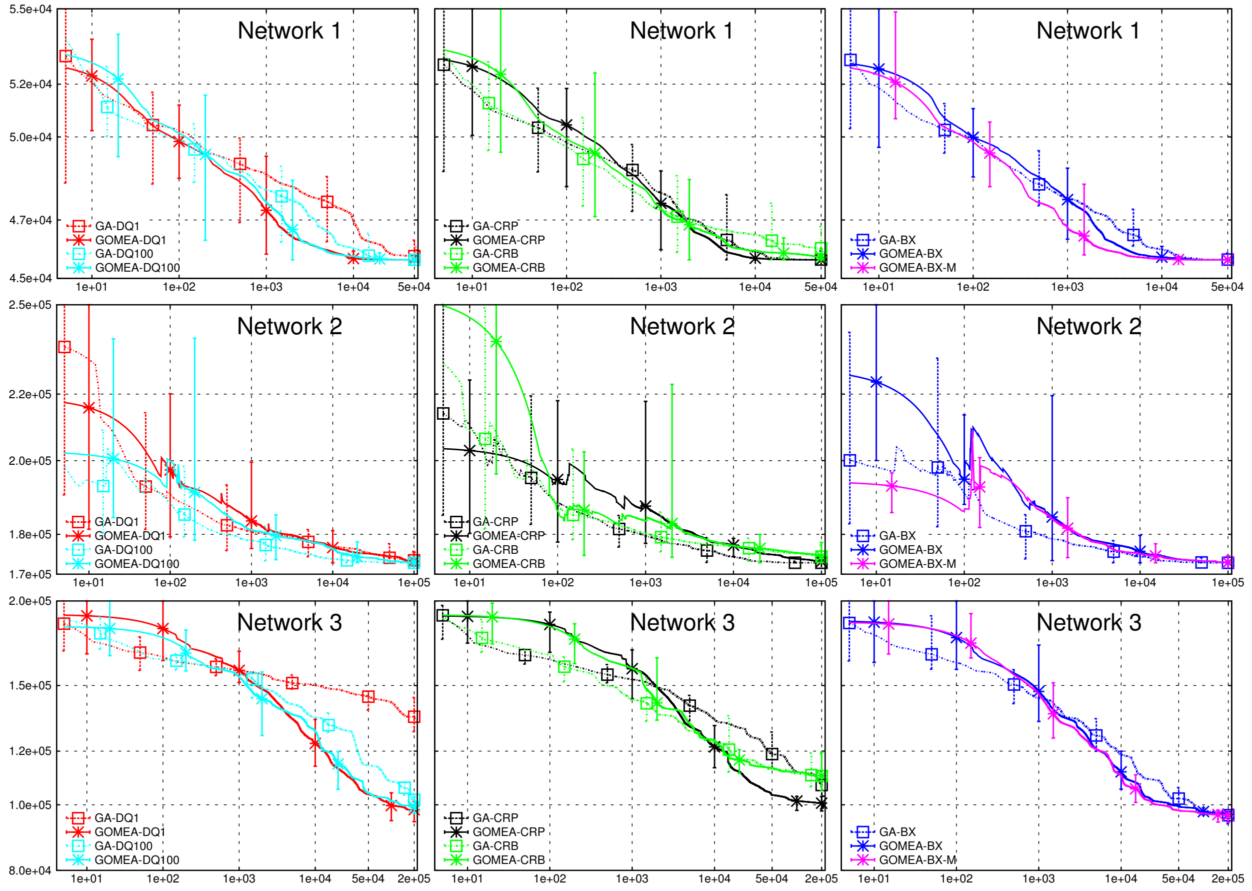

Figure 3: Dynamic DNEP. Horizontal axis: number of evaluations. Vertical axis: NPV of total cost (EUR).

set population size parameter. Third, we introduced multiple variation operators that can be employed by EAs solving DNEPs and showed their impacts on the performance of 2 typical EAs: the classic GA and the GOMEA, which is capable of linkage learning (LL) and exploiting linkage information. Due to LL, GOMEA is shown to be a far more robust solver for solving DNEP on our benchmark networks. Using the same number of solution evaluations like GA, GOMEA obtains better results even when assuming a minimal amount of problem-specific knowledge (PSK). Adding PSK to GOMEA improves performance results but the improvement gap, for a fixed budget of evaluations, is much less significant than that for classic GA, which again confirms the usefulness of LL in detecting problem structures. As problem size increases, LL is of great importance to the scalability of EAs. Lastly, based on the experiment results, we suggest that LL EAs, like GOMEA, should be given priority when selecting EAs for solving DNEP.

7. REFERENCES

[1] C. L. T. Borges and V. F. Martins. Multistage expansion planning for active distribution networks under demand and distributed generation uncertainties. *International Journal of Electrical Power & Energy Systems*, 36(1):107 – 116, 2012.

[2] 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.

[3] E. Diaz-Dorado, J. Cidras, and E. Miguez. Application of evolutionary algorithms for the planning of urban distribution networks of medium voltage. *IEEE Transactions on Power Systems*, 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*, pages 1–8, 2014.

[7] G. R. Harik and F. G. Lobo. A parameter-less genetic algorithm. In *Proc. of the Genetic and Evolutionary Comp. Conf. (GECCO 1999)*, pages 258–265, 1999.

[8] H. N. 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 *Artificial Evolution - 11th International Conference, EA 2013*, pages 93–105, 2013.

[9] I. Ramirez-Rosado and J. Bernal-Agustin. Genetic algorithms applied to the design of large power distribution systems. *IEEE Transactions on Power Systems*, 13(2):696–703, May 1998.

[10] D. Thierens and P. A. N. Bosman. Optimal mixing evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO-2011*, pages 617–624. ACM, 2011.

[11] 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. *IEEE Transactions on Smart Grid*, 3(3):1203–1212, Sept 2012.