Power Distribution Network Reconfiguration by Evolutionary Integer Programming

Kaifeng Yang[1], Michael T.M. Emmerich[1], Rui Li[1],
Ji Wang[2], and Thomas Bäck[1]

[1] LIACS, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
[2] Central South University, 410083 Changsha, Hunan Province, China

Abstract. This paper presents and analyses new metaheuristics for solving the multiobjective (power) distribution network reconfiguration problem (DNRP). The purpose of DNRP is to minimize active power loss for single objective optimization, minimize active power loss and minimize voltage deviation for multi-objective optimization.

A non-redundant integer programming representation for the problem will be used to reduce the search space size as compared to a binary representation by several orders of magnitudes and represent exactly the feasible (cycle free, non-isolated node) networks. Two algorithmic schemes, a Hybrid Particle Swarm Optimization - Clonal Genetic Algorithm (HPCGA) and an Integer Programming Evolution Strategy (IES), will be developed for this representation and tested empirically.

Conventional algorithms for solving multi-objective DNRP are converting the multiple objective functions into a single objective function by adding weights. However, this method cannot capture the trade-offs and might fail in case of a concave Pareto front. Therefore, we extend the HPCGA and IES in order to compute Pareto fronts using selection procedures from NSGA-II and SMS-EMOA. The performance of the methods is assessed on large scale DNRPs.

Keywords: Power Distribution Network Reconfiguration, Integer Programming, Particle Swarm Optimization, Clonal Genetic Algorithm, Evolution Strategies, Multiobjective Optimization.

1 Introduction

With the sustainable development of economy, there is an increasingly high demand for the quality and reliability of the electricity supply in every industry. Power distribution network reconfiguration is an important method of optimizing the distribution system, which is significant to enhancing the security, the efficiency, and the reliability of the system. There are two types of switches in a power network system: normally closed switches and normally open switches. See Figure 1, for an example of a power distribution network configuration, the 119 bus system [1], where the black solid lines represent normally closed switches, and red dashed lines represent normally open switches. Network reconfiguration is the process of changing the topology of the power network by operating these switches for the purpose of minimization of the power loss. Since each switch has

T. Bartz-Beielstein et al. (Eds.): PPSN XIII 2014, LNCS 8672, pp. 11–23, 2014.

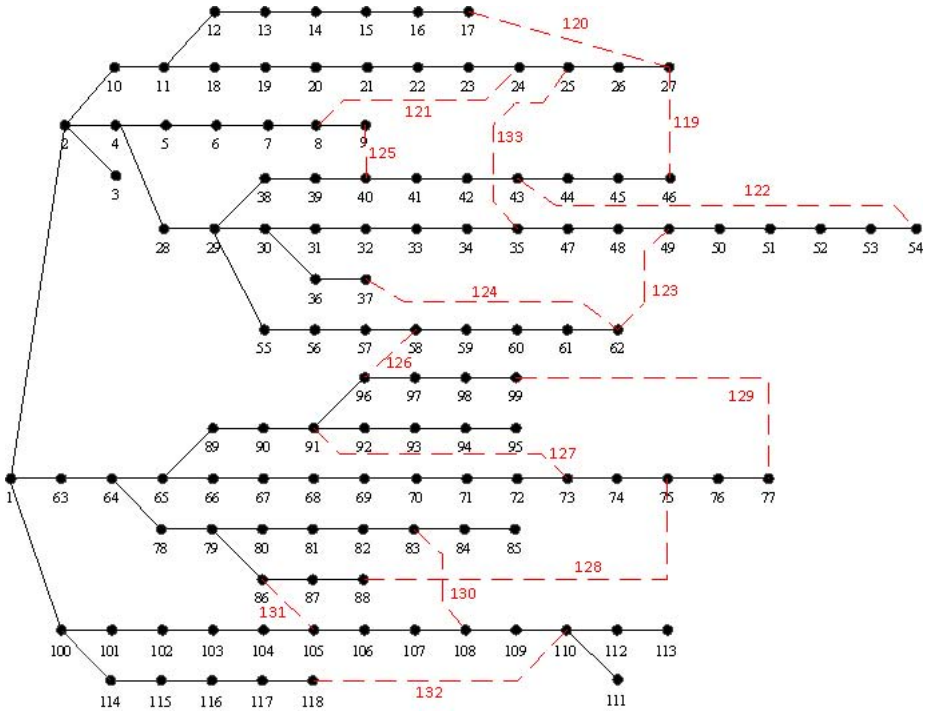

Fig. 1. Initial configuration of the 119-bus test system

two conditions, for a system which has N nodes, there should be 2^{N-1} potential switch configurations. To make sure that all the customers can get electricity and no short circuit exists in the system, there are, however, two constraints for network reconfiguration: no cycles (the radial structure of the network must be maintained in each new structure) and no islands (all the loads must be served).

The problem of finding an optimal distribution network reconfiguration is known to be NP hard, and larger instances with more than 100 nodes cannot yet be solved exactly, such that several metaheuristics were proposed: Zhang et al. [1] developed a tabu search algorithm for real power loss minimal reconfiguration in large-scale distributed systems. Aman et al. [2] proposed to use evolutionary programming (EP) to find the optimal topology of the distributed network for minimizing the real power loss. Rao et al. [3] presented artificial bee colony algorithms for determining the sectionalizing switch to be operated to solve the real power loss minimization problem. Niknam et al. [4] presented an interactive fuzzy satisfying method based on hybrid modified honey bee mating optimization for aggregation-based multiple objective optimization. Kasaei [5] used an ant colony algorithm to solve the optimal network reconfiguration and capacitor placement problem for power loss reduction and voltage profile enhancement in distribution networks.

The contribution of this paper is to discuss concise representations of the problem, discuss fast and reliable evolutionary integer programming solvers,

and extend them for Pareto-based (set-based) multiobjective optimization. The objective functions considered will be power loss and voltage profile enhancement, but as opposed to the study in [5], we are interested in visualizing the trade-off and the robustness by using multiobjective optimization techniques.

2 Problem Description

The network reconfiguration problem in a power distribution system is to find the best configuration of a radial network. The objective functions are the minimization of power loss and the maximization of the network's reliability. Besides, the network has to satisfy certain operating conditions [3].

2.1 Objective Function

The objective function for the **minimization of power loss** can be described as [6]:

$$\min \quad f_{loss} = \sum_{i=1}^{b} k_i R_i \frac{P_i^2 + Q_i^2}{V_i^2} = \sum_{i=1}^{b} k_i R_i \mid I_i \mid^2 \tag{1}$$

subject to:

$$V_i^{min} \leq V_i \leq V_i^{max} \tag{2}$$

$$I_i \leq I_i^{max},\ i = 1, ..., b \tag{3}$$

Here b is the number of branches and for each branch $i \in \{1, ..., b\}$, R_i is the branch resistance, P_i and Q_i are the active power and the inactive power of a branch terminal i, V_i is the terminal node voltage of branch i, V_i^{min} and V_i^{max} are the minimum and maximum bus voltage of branch i, respectively, k_i is the status variable of i-th switch. If k_i is 0, then switch i is open and if k_i is 1, then switch i is closed. I_i is the branch current and I_i^{max} is the maximum current in branch i.

The objective function for **minimization of voltage deviation** can be expressed as follows [7][8]:

$$\min \quad f_{VDI} = \max\{\mid 1 - U_{min} \mid, \mid 1 - U_{max} \mid\} \tag{4}$$

where U_{min} and U_{max} are respectively the minimum and maximum values of bus voltage divided by rated voltage to normalize them to value in $[0, 1]$.

2.2 Feasible Constraint on Network Topology

Recall that, in order to ensure the supply to all nodes and avoid short circuits, the network must be cycle free and not contain isolated nodes. The previous method for verifying whether an individual is feasible or not is based on the topology checks of the structure [9]. This method can be computed fast, but requires some manual parameter settings beforehand.

A design for an automatic **feasibility check** is proposed next. It rests upon the following idea of *cycle free network construction*: suppose there are N nodes in a system, and there should be at least $(N-1)$ line segments to connect all these nodes. Then the first line segment can connect 2 arbitrary nodes to the system. Any new line segment, which is not the first line segment, can only connect a new node to the existing system, because otherwise a cycle would be introduced.

The feasibility check algorithm is based on the relationship between the number of the nodes and line segments. Suppose there are N_bus buses and N_line branches in the system. Based on the aforementioned, we can easily conclude that: if $N\_line > N\_bus$, there must be at least one cycle in the structure. Clearly, if $N\_line < N\_bus - 1$, there must be at least two separated components in the structure. Therefore, N_line must equal $N\_bus - 1$, if the structure is feasible. Given $N\_line = N\_bus - 1$, feasibility is implied by the non-existence of isolated nodes, due to the following: if there is a cycle and only $N\_bus - 1$ edges can be used, at least one of the nodes cannot be connected and it will be an isolated node. As detecting isolated nodes is simple, we can now state the following fast and correct algorithm for checking whether a network is cycle free and does not contain isolated nodes:

Algorithm 1. Topology Feasibility Check
Step 1 Verify N_line equals $N\_bus - 1$; if not equal, the individual is not feasible, otherwise go to step 2;
Step 2 Verify whether there is a separated component or not. If there exists a separated component, the individual is not feasible, otherwise it is feasible.

3 Power Loss Minimization Algorithms

Two algorithms for minimizing power loss in DNRP will be discussed next, and extended in section 6 to bi-objective optimization methods. All algorithms are based on a concise sequence encoding of the feasible search space.

3.1 Encoding Strategy

The encoding strategy is a main part which influences the efficiency of the algorithm in distribution network reconfiguration. For a genetic algorithm, it is common to use **binary encoding**, and it is also straightforward in DNRP as each switch can be associated with a binary variable. However, considering the topology constraint, many solutions of the search space induced by binary encoding would be infeasible. Instead, we therefore propose to use **sequence encoding** [10], which has the following advantages: 1. It is easy to realize; 2. The probability of generating a feasible solution is high.

The *sequence encoding system* regards each loop (potential cycle) as a gene. In each loop exactly one switch has to be open, as otherwise either there would

be isolated components (two open switches) or there would be short circuits (no open switch). Each loop of the power network is encoded by a natural number, and all the switches are coded sequentially by natural numbers starting from 1. For each gene, the value of this gene is the position of the switch that is open in the loop represented by the gene. Sequence coding strategy can eliminate most of the infeasible individuals from the search space.

Example 1. Figure 2, is an example of the IEEE-16 distribution system [6][11]. Here a connection of two bus nodes is also a loop. There are three loops in this system. Loop 1 is composed by branches 5, 6, 8 and 9; loop 2 is composed by branches 4, 7, 13, 14 and 15; loop 3 is composed by branches 10, 11 and 12. To make sure the structure satisfying the constraints, one branch in each loop should be opened. Therefore, the search space size is $4 \times 5 \times 3 = 60$ using the sequence encoding system. Compared to a search space size of $2^{12} = 4096$ using binary encoding system the search space can be dramatically reduced.

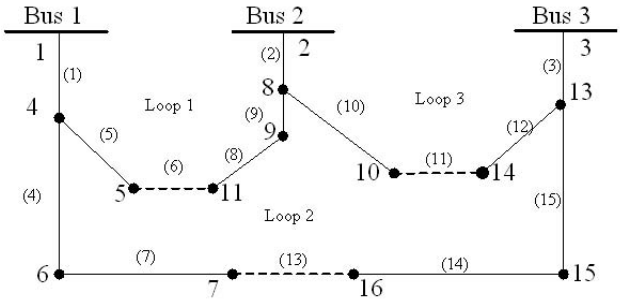

Fig. 2. IEEE-16 distribution system after coding

However, for a real world problem, there could be hundreds or thousands of transformers in one city, and the number of solutions in the search space is very large, even using this sequence coding. For example, in a IEEE-69 system, the binary encoding generates a search space of size $2^{68} \approx 2.95 \times 10^{20}$ and sequence encoding approximately generates a search space size of 1.78×10^6. For the 119 system (see Figure 1), the search space size is $2^{119} \approx 6.65 \times 10^{35}$ and 1.44×10^{18}, respectively, using binary coding and sequence encoding strategy.

3.2 Hybrid Particle Swarm/Clonal Genetic Algorithm

The Hybrid Particle Swarm Optimization/Clonal Genetic Algorithm (PSO-CGA) is based on two generational transitions (variation and selection steps) that are applied in an alternating manner. The PSO is fast for local optimization, and the CGA is mainly integrated to the PSO in order to prevent premature convergence and increase diversity, e.g., by a special mutation-shift operator. The flowchart of PSO-CGA can be seen in Figure 3.

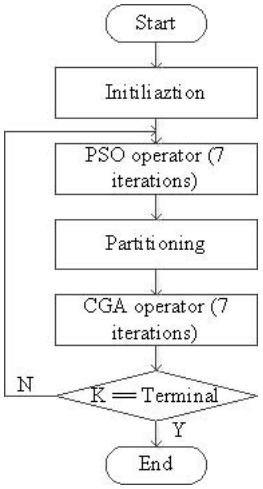

Fig. 3. Flowchart of hybrid PSO-CGA

In PSO it is assumed that the solution space has dimension D, and the population is composed by N particles $X = \{x_1, ..., x_i, ..., x_n\}$, the position of the i-th particle is $x_i = (x_{i1}, x_{i2}, ..., x_{iD})^T$, $P_i^{best} = (P_{i1}^{best}, P_{i2}^{best}, ..., P_{iD}^{best})^T$ stands for the best known position of particle i, and $g^{best} = (g_1^{best}, g_2^{best}, ..., g_D^{best})^T$ stands for the best known position of the entire swarm, the velocity of particle i is $V_i = (v_{i1}, v_{i2}, ..., v_{id})^T$. Particle x_{iD} updates its velocity and position information by [12]:

$$V_{id}^{k+1} = \omega \times V_{id}^k + c_1 \times r_1 \times (P_{id}^{best} - X_{id}^k) + c_2 \times r_2 \times (g_d^{best} - X_{id}^k) \quad (5)$$

$$X_{id}^{k+1} = X_{id}^k + V_{id}^{k+1} \qquad i = 1, 2, ..., N; d = 1, 2, ..., D \quad (6)$$

where ω is the inertia factor, c_1 and c_2 are the learning factors, r_1 and r_2 are random real numbers within $[0, 1]$.

Traditionally, clonal genetic algorithms (CGA) [13] use roulette wheel selection. This paper uses another selection mechanism, and this selection mechanism in CGA partitions the parent population P in three subpopulations, say P_B, P_M, P_W, the size of which is fixed by $|P_B| = u_1|P|$, $|P_M| = u_2|P|$, and $|P_M| = u_3|P|$ with $u_1 + u_2 + u_3 = 1$. The new generation is generated in three steps:

Step 1: Select $u_1 \times 100\%$ best offspring for mutation, if there is no improvement after mutation, choose the offspring before the mutation; the general idea is to search for improvements nearby current best solutions.

Step 2: Select $u_3 \times 100\%$ worst offspring for initialization; the general idea behind this mechanism is to maintain the population's diversity.

Step 3: Apply mutation operator for the remaining offspring, the difference between this step 3 and step 1, is that after step 3, the mutated individuals will always be kept, no matter whether there is an improvement or not. This allows slow diffusion away from a local optimum.

3.3 CGA Shift and Mutate Operator

The CGA variation operator is composed by shift and mutation operators [13] [14]. The multi-shift operator adds (or subtracts) the same random offset *Number_Shift* to all genes in a random subset of genes $gene\_shift$. A random direction flag *Direction_Flag* is used to decide whether to add or subtract. If the interval boundary gets exceeded the value of the gene is set to the interval boundary.

Example 2. An example for the multi-shift operator is shown in the table below. Where $gene\_shift = [2, 4, 5]$, and the current individual is [× × 7 11 21].

Parent	$Number\_Shift$	$Direction\_Flag$	Offspring
[× 4 × 11 21]	5	0	[× 9 × 16 26]
[× 4 × 11 21]	5	1	[× 1 × 6 16]

The mutation operator is another main part in CGA. It determines first the genes to be mutated and then sets them to a new random value. A *single-shift (single-mutation)* operator shifts (mutates) only a single gene.

4 Integer Programming Evolutionary Strategy

We studied evolution strategies for integer programming (IES) by Rudolph [15], as an alternative search algorithm approach. It features a $(\mu + \lambda)$-selection scheme, that is λ offspring Q are generated based on μ parents P and the best μ individuals out of $P \cup Q$ form the next parent population. Each individual is created by selecting randomly two parents (sexual case) or more than two parents (panmictic case), applying discrete recombination (choose each gene randomly from one of the parents) or intermediate recombination (averaging) to create a single offspring, which then is mutated. The mutation operator perturbs each gene by the difference of two geometrically distributed pseudo-random numbers. For each gene a step-size ς_i is maintained and undergoes a mutation, too, which makes it possible for the step-size to adapt. The mutation maps an individual $(\mathbf{X}, \varsigma) \in (\mathbb{Z}^n \times (\mathbb{R}^+)^n)$ to its mutant $(\mathbf{X}', \varsigma') \in (\mathbb{Z}^n \times (\mathbb{R}^+)^n)$ as follows:

$$\begin{aligned}
&\varsigma_i' = \max\{1, \varsigma_i \exp(\tau N_c + \tau' N_i)\},\ N_i \sim \text{Normal}(0,1),\ N_c \sim \text{Normal}(0,1) \\
&z_{ij} = \left\lfloor \frac{ln(1-u_{ij})}{ln(1-\varphi_i)} \right\rfloor, \varphi_i = 1 - \zeta_i(1 + \sqrt{1+\zeta_i^2})^{-1}, u_{ij} \sim \text{Uniform}(0,1), j = 1,2 \\
&X_i' = X_i + z_{i1} - z_{i2},\ i = 1, \ldots, n
\end{aligned} \tag{7}$$

Since the original geometric distribution is single tailed, Rudolph proposed the use of the difference $z_{i1} - z_{i2}$ and could show that the resulting multivariate distribution has ℓ_1 symmetry, maximal entropy, and infinite support. It features (multiple) self-adaptive mutation step sizes and, for a minimal stepsize greater than zero, global convergence for $t \to \infty$. In case interval boundaries are exceeded reflection at the interval boundary is used [16]. To prevent stagnation the stepsize is bounded from below by 1. Learning rates τ and τ' determine the speed of step-size adaptation. See also [16] for a detailed description, default parameters and analysis.

5 Single-objective Optimization Result

The simulation results[1] are based on distribution system 119 [1]. The test system is a 11 kV distribution system with 118 sectionalizing switches and 15 tie switches, and the total power loads are 22709.7 kW and 17041.1 kVAr. The topology of the distribution system 119 is shown in figure 1. All simulations were performed in MATLAB 8.2, CPU: Intel 2 Core 3.16GHz, 2.0 GB DDR RAM (800 MHz). The power flow calculation (to calculate power loss and voltage deviation) is using Newton Method based on MATPOWER [17], and maximum number of iterations is 20, termination tolerance on per unit is 1e-8.

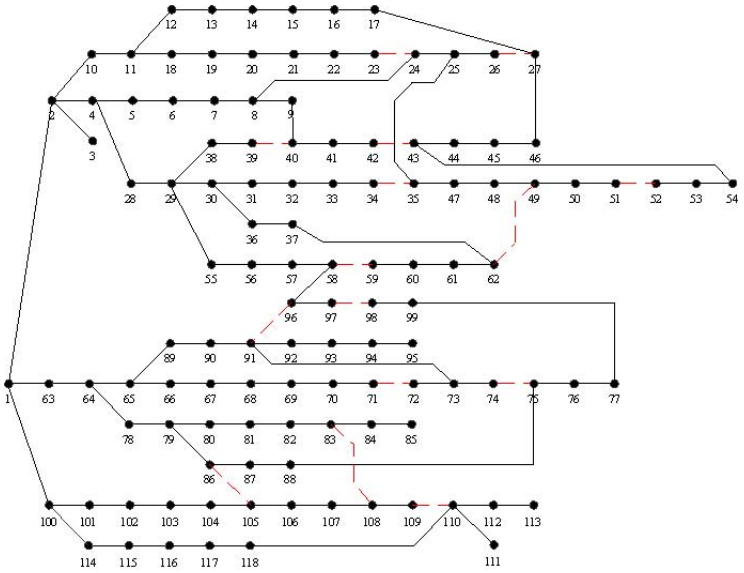

Fig. 4. The best structure

PSO-CGA Hybrid: The PSO-CGA hybrid was run with a population size of 30, and the PSO parameters were $\omega = 0.8$, $c_1 = 2$, $c_2 = 2$, $v\_max = 4$ (Max speed factor), $v\_min = -4$ (Min speed factor). For each run, 50 PSO steps (7 generations per step) and 50 CGA steps (7 generations per step) were conducted in alternation. One run takes 15-30 minutes. In the CGA one of the operators, single-shift, single-mutation, multi-shift, or multi-mutation is chosen randomly. In case of multi-shift and multi-mutation the number of genes to be mutated is chosen randomly, too, between 1 and the number of genes. An experiment was designed to find optimal settings for parameters u_1, u_2, and u_3 within their bounds $[0, 1]$ and respecting the constraint $u_1 + u_2 + u_3 = 1$. A **design of experiments for mixtures** was applied following parameter setting in [18].

[1] The MATLAB source code of the numerical experiments is available on request by the authors and on `http://natcomp.liacs.nl/index.php?page=code`.

The results (shown in Table 1) of the experimental design are visualized as ternary diagrams (Figure 5). The best results are achieved with a low rate of reinitialization (u_3) and a relatively high rate for u_1.

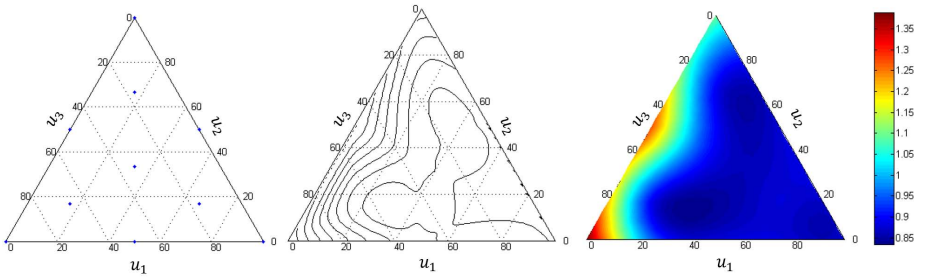

Fig. 5. Mixture design (left) and interpolated results (middle, right) of u_1 (selection), u_2 (diffusion), and u_3 (reinitialization) for CGA-PSO hybrid on Test System 119

Table 1. Experiments on Hybrid PSO-CGA

u_1	1	0	0	0.5	0.5	0	0.333	0.167	0.167	0.666	0.333
u_2	0	1	0	0.5	0	0.5	0.333	0.167	0.666	0.167	0.333
u_3	0	0	1	0	0.5	0.5	0.333	0.666	0.167	0.167	0.333
Exp.1	875.155	1037.9	1329.96	883.828	875.155	1325.33	891.033	891.864	875.155	890.918	874.86
Exp.2	875.155	1085.33	1478.89	875.155	875.155	1120.87	875.155	990.197	875.155	874.86	874.86
Exp.3	875.539	1110.29	1487.06	874.86	890.918	1057.02	875.155	875.155	888.966	875.155	874.86
Exp.4	870.856	1209.17	1138.57	874.86	874.86	1412.98	883.858	887.39	875.155	875.155	875.155
Exp.5	875.155	1068.33	1408.56	875.155	886.437	1366.81	903.49	876.178	876.729	883.641	874.86
Exp.6	875.155	1130.18	1492.56	875.155	875.155	1405.2	875.155	874.86	874.86	874.86	875.155
Exp.7	875.155	1195.5	1492.28	883.858	875.155	1363.57	875.155	869.727	874.86	891.57	875.155
Exp.8	887.505	1081.05	1390.6	877.34	878.96	1186.35	874.86	890.918	874.86	878.209	889.16
Exp.9	874.86	1014.13	1281.5	874.86	874.86	1277.85	878.334	890.918	887.39	874.86	875.155
Exp.10	875.155	1041.54	1384.4	874.86	874.86	1177.81	875.155	875.155	1013.09	869.727	874.86
Min	870.856	1014.13	1138.57	874.86	874.86	1057.02	874.86	869.727	874.86	869.727	874.86
Max	887.505	1209.17	1492.56	883.858	890.918	1412.98	903.49	990.197	1013.09	891.57	889.16
Mean	875.969	1097.34	1388.438	876.993	878.152	1269.38	880.735	892.236	891.622	878.896	876.408
Deviation	4.27671	65.1574	114.1382	3.68630	5.78028	126.08	9.60066	35.3921	43.0215	7.37168	4.482993

Integer Programming Evolution Strategy: The parameters for the IES are as follows: $\mu = 15$, $\lambda = 100$, $\sigma_{initial} = (ub - lb)/8$, $ub_{sigma} = (ub - lb)/3$, $lb_{sigma} = lb$, $\tau = \frac{1}{\sqrt{2n_z}}$ and $\tau' = \frac{1}{\sqrt{2\sqrt{n_z}}}$, where ub and lb is the upper bound and lower bound vector of the variables, $N = 15$ in case of the 119 system. Each run had 100 iterations and took about 5-10 minutes. For the IES preliminar experimentation showed that the recombination type was a crucial choice. Table 2 shows results of different recombination methods based on the Integer ES. We tested four recombination types: none (1), discrete (2), panmictic discrete (3), and intermediate (4); see [19]. The best result that was found was with a discrete recombination on the object variables, and a panmictic discrete recombination on the step size.

Table 2. Experimental results for the 119 system based on ES

		s._X = 1	s._X = 2	s._X = 3	s._X = 4
s._ς = 1	Min	939.6829	893.6304	935.478	930.895
	Max	1302.274	1687.879	1556.437	1904.853
s._ς = 2	Min	881.8714	874.8604	869.7271	878.3646
	Max	1083.146	978.1069	974.6982	1048.195
s._ς = 3	Min	875.155	894.2617	878.3646	876.0975
	Max	1009.264	1088.795	1120.487	1090.32
s._ς = 4	Min	912.3288	1103.351	950.3428	1059.776
	Max	3666.716	7029.378	3071.888	10456

Both Hybrid PSO-CGA and IES can find the same optimal result, see Table 3. The open switches in best result, whose structure is shown in Figure 4, are 42-43, 26-27, 23-24, 51-52, 62-49, 58-59, 39-40, 91-96, 71-72, 74-75, 97-98, 108-83, 105-86, 109-110 and 34-35. Comparing PSO-CGA and IES, it is found that the former was more robust (better average values) while the latter found better results and converged faster (IES 5-10 min, PSO-CGA 15-30 min). All strategies perform significantly better than PSO-CGA with settings $u_1 = 0, u_2 = 0, u_3 = 1$, which is essentially a trial and error strategy (within the sequence representation). The improvement is by a factor of 1298.0861/869.7271 , i.e. the power loss found by PSO-CGA or IES metaheuristics is only ca. 67% of the power loss of a solution found by trial and error for the same number of evaluations.

Table 3. Best results for 119 system

	Before reconfiguration			After reconfiguration based on MATPOWER		
Refference	[1]	[3]	Matpower	[1]	[3]	Hybrid PSO-CAG & IES
Unit: kW	1294.3	1298.09	1298.0861	887.5055	869.7271	869.7271

6 Multiobjective Optimization

We extended IES to a multiobjective optimization algorithm by replacing the selection scheme by that of a multiobjective algorithm, namely the $(\mu + \mu)$ selection of NSGA-II [20] and the $(\mu + 1)$ selection of SMS-EMOA [21] (which has earlier been used also in Pareto archivers [22]). As a second objective voltage deviation was minimized (see equation 4).

As an adaptation, we introduced a variant of SMS-EMOA and NSGA-II with a self-adaptive single step size. Whenever more than five mutations per individual were unsuccessful, the step size was multiplied by a constant factor of 1/1.2 (following the 1/5th success rule). Success of a generation was registered if a new non-dominated solution entered the archive of non-dominated solutions among all solutions encountered so far.

The results (attainment curves) for a population size of $\mu = 30$ and 11 runs per algorithm are shown in Figure 6, where $f1$ and $f2$ represent voltage deviation and

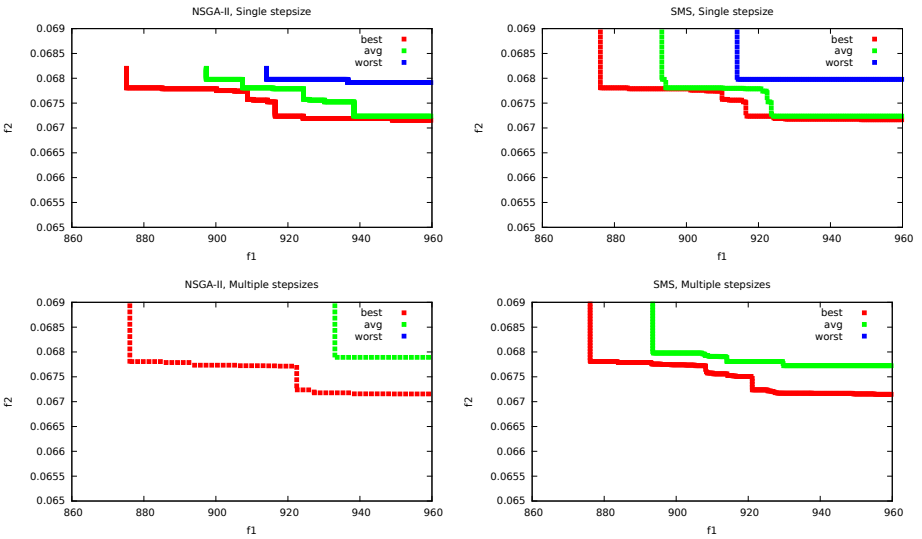

Fig. 6. Best, worst, and average attainment curves for the multiobjective optimization of 119 DNRP

power loss. In the following discussion by Pareto front, we mean the archive of all non-dominated solutions encountered in a single run. SMS-EMOA with a single step size provides the best Pareto fronts in the best case and also in the average case. Interestingly, all strategies find a Pareto front with a concave part. The interpretation of this is that locally there is a strong conflict between power loss minimization and voltage deviation minimization for this problem. However, the range of voltage deviation is relatively small, so that 'from a distance' the Pareto front has a clear knee point region. Solutions in this region can be recommended as good compromise solutions, whereas points located on the flanks of the Pareto front are not recommended as small improvements in one objective will cause large deterioration of the other objective.

7 Conclusions

In this paper two well performing optimization strategies for solving the power distribution network reconfiguration problem have been described and tested on a challenging problem with more than 100 switches. A concise integer representation was chosen and it was demonstrated that it reduces the search space size by many orders of magnitude as opposed to the binary representation used in genetic algorithms so far. In the experiments we focused on finding a good ratio between exploration and exploitation in the PSO-CGA and on choosing a good recombination operator in the IES with self-adaptive mutation. This turned out to be discrete recombination on object and step-size variables. The results clearly show that it is much better to use metaheuristics instead of trial and

error strategies when minimizing power loss. Also multiobjective optimization algorithms, NSGA-II and SMS-EMOA, were applied to compute a Pareto front between voltage deviation and power loss objectives. These objectives turned out to be conflicting in the knee point region but globally, when zooming out, they appear to be complementary. For the future work, it is recommended to test the strategies on a broader set of benchmarks and further investigate multiobjective problem formulations, for instance including objective functions on reliability.

Acknowledgment. Kaifeng Yang acknowledges financial support from China Scholarship Council (CSC), CSC No.201306370037.

References

[1] Zhang, D., Fu, Z.C., Zhang, L.C.: An improved ts algorithm for loss-minimum reconfiguration in large-scale distribution systems. Electric Power Systems Research 77(5), 685–694 (2007)

[2] Aman, M.M., Jasmon, G.B., Naidu, K., Bakar, A.H.A., Mokhlis, H.: Discrete evolutionary programming to solve network reconfiguration problem. In: TENCON Spring 2013 Conference, pp. 505–509. IEEE, Sydney (2013)

[3] Rao, R.S., Narasimham, S.V.L., Ramalingaraju, M.: Optimization of distribution network configuration for loss reduction using artificial bee colony algorithm. International Journal of Electrical Power and Energy Systems Engineering 1(2), 116–122 (2008)

[4] Niknam, T., Meymand, H.Z., Mojarrad, H.D.: An efficient algorithm for multi-objective optimal operation management of distribution network considering fuel cell power plants. Energy 36(1), 119–132 (2011)

[5] Kasaei, M.J., Gandomkar, M.: Loss reduction in distribution network using simultaneous capacitor placement and reconfiguration with ant colony algorithm. In: Asia-Pacific Power and Energy Engineering Conference, APPEEC 2010, pp. 1–4. IEEE, Chengdu (2010)

[6] Qin, Y.M., Wang, J.: Distribution network reconfiguration based on particle clonal genetic algorithm. Journal of Computers 4(9), 813–820 (2009)

[7] Chiou, J.P., Chang, C.F., Su, C.T.: Variable scaling hybrid differential evolution for solving network reconfiguration of distribution systems. IEEE Transactions on Power Systems 20(2), 668–674 (2005)

[8] Niknam, T.: A new hybrid algorithm for multi-objective distribution feeder reconfiguration. Cybernetics and Systems: An International Journal 40(6), 508–527 (2009)

[9] Wang, S.X., Wang, C.S.: A novel network reconfiguration algorithm implicitly including parallel searching for large-scale unbalanced distribution systems. Automation of Electric Power Systems 24(19), 34–38 (2000)

[10] Ma, X.F., Zhang, L.Z.: Distribution network reconfiguration based on genetic algorithm using decimal encoding. Transactions of China Electrotechnical Society 19(10), 65–69 (2005)

[11] Bi, P.X., Liu, J., Liu, C.X., Zhang, W.Y.: A refined genetic algorithm for power distribution network reconfiguration. Automation of Electric Power Systems 26(2), 57–61 (2002)

[12] Poli, R., Kennedy, J., Blackwell, T.: Particle swarm optimization. Swarm Intelligence 1(1), 33–57 (2007)
[13] Li, M.J., Tong, T.S.: A partheno genetic algorithm and analysis on its global convergence. Automatization 25(1), 68–72 (1999)
[14] Wang, J., Luo, A., Qi, M.J., Li, M.J.: The improved clonal genetic algorithm & its application in reconfiguration of distribution networks. In: Power Systems Conference and Exposition, PSCE 2004, pp. 1423–1428. IEEE, New York (2004)
[15] Rudolph, G.: An evolutionary algorithm for integer programming. In: Davidor, Y., Männer, R., Schwefel, H.-P. (eds.) PPSN III. LNCS, vol. 866, pp. 139–148. Springer, Heidelberg (1994)
[16] Li, R., Emmerich, M., Eggermont, J., Bäck, T., Schütz, M., Dijkstra, J., Reiber, J.H.: Mixed integer evolution strategies for parameter optimization. Evolutionary computation 21(1), 29–64 (2013)
[17] Zimmerman, R.D., Murillo-Sánchez, C.E., Thomas, R.J.: Matpower: Steady-state operations, planning, and analysis tools for power systems research and education. IEEE Transactions on Power Systems 26(1), 12–19 (2011)
[18] Anderson, M., Whitcomb, P.: Find the optimal formulation for mixtures (2002), http://www.statease.com/pubs/chem-2.pdf
[19] Bäck, T.: Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms. Oxford University Press, New York (1996)
[20] Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.A.M.T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2), 182–197 (2002)
[21] Beume, N., Naujoks, B., Emmerich, M.: SMS-EMOA: Multiobjective selection based on dominated hypervolume. European Journal of Operational Research 181(3), 1653–1669 (2007)
[22] Knowles, J., Corne, D.: Properties of an adaptive archiving algorithm for storing nondominated vectors. IEEE Transactions on Evolutionary Computation 7(2), 100–116 (2003)