A Parallel Multi-Population Biased Random-Key Genetic Algorithm for Electric Distribution Network Reconfiguration

Haroldo de Faria Jr.
Federal University of ABC
P.O. Box 09210-170
Brazil
haroldo.faria@ufabc.edu.br

Willian Tessaro Lunardi
University of Luxembourg
P.O. Box 1855
Luxembourg
willian.tessarolunardi@uni.lu

Holger Voos
University of Luxembourg
P.O. Box 1855
Luxembourg
holger.voos@uni.lu

ABSTRACT

This work presents a multi-population biased random-key genetic algorithm (BRKGA) for the electric distribution network reconfiguration problem (DNR). DNR belongs to the class of network design problems which include transportation problems, computer network restoration and telecommunication network design and can be used for loss minimization and load balancing, being an important tool for distribution network operators. A BRKGA is a class of genetic algorithms in which solutions are encoded as vectors of random keys, i.e. randomly generated real numbers from a uniform distribution in the interval [0, 1). A vector of random keys is translated into a solution of the optimization problem by a decoder. The decoder used generates only feasible solutions by using an efficient codification based upon the fundamentals of graph theory, restricting the search space. The parallelization is based on the single program multiple data paradigm and is executed on the cores of a multi-core processor. Time to target plots, which characterize the running times of stochastic algorithms for combinatorial optimization, are used to compare the performance of the serial and parallel algorithms. The proposed method has been tested on two standard distribution systems and the results show the effectiveness and performance of the parallel algorithm.

CCS CONCEPTS

• **Computing methodologies** → Parallel computing methodologies; • **Applied computing** → Physical sciences and engineering; Operations research → Decision analysis

ACM Reference format:

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3321950

KEYWORDS

Distribution network reconfiguration, biased random-key genetic algorithms, parallel computation

1 INTRODUCTION

The electric distribution system is a part of an electric power system that is responsible for distributing the electric energy to the final customers of electricity. These customers include residential, commercial and industrial loads. The electric distribution network reconfiguration problem (DNR) is an important tool for the engineers who operate the distribution power system. It is used to modify the topology of the distribution system in order to reduce active power losses on the feeders, improve the voltage profiles of the consumers, raise the operational reliability of the system, eliminate and/or isolate electric faults and increase the capacity of the network to accommodate renewable distributed generation, among other objectives. These objectives are attainable by altering the open or closed status of normally closed (NC) sectionalizing switches and normally open (NO) tie-line switches, while maintaining the radiality of the network. The radial configuration is a tree of a distribution network graph, without any closed paths and isolated vertices (nodes). In this tree, there is one root node called the substation, which supplies with power all the other nodes.

The DNR is modeled mathematically as a nonlinear mixed integer optimization problem, due to the high number of switching elements in a distribution network, and to the nonlinear characteristics of the constraints used to model the electrical behavior of the system[1]. The most common version of the DNR problem is the one that minimizes electric losses in the network. Real distribution networks are unbalanced three-phase systems with loads of different nature and distributed energy sources connected throughout the system, but the classical DNR problem deals with a simplified network model. It assumes a balanced distribution network with loads modeled as constant power. This simplified DNR version is useful for comparison purposes because many different approaches have been applied to solve the same problem using the same standard test systems. It is one of the most studied combinatorial optimization problems in the power systems research field and large real-world problems are well suited for the application of metaheuristic techniques. Most methods applied to the DNR for loss reduction belong to the metaheuristic class of methods. A metaheuristic applies and coordinates more than one heuristic, such as local search, using the strengths of each one to efficiently explore the search space. They include genetic algorithms (GA), simulated annealing, tabu search, scatter search, ant colonies, variable neighborhood search, GRASP, and path-relinking [2].

The first application of a GA to solve the DNR for loss reduction was proposed in [3]. A modified GA presented by Chu-Beasley [4] was used in [5] to solve the problem. It proposes a codification where, instead of representing the switching devices, the entire network configuration resulting from the switching is used as the individual and considered as a tree graph, represented as a vector with the arcs in the tree (branches) sequentially organized, from top to bottom of the network, being the root node (generally the substation) the top. A combination of binary and discrete particle swarm optimization is proposed in [6] to solve the loss reduction problem. The method identifies groups of branches to represent the network and each group has unidimensional encoding. A Tabu Search algorithm with short term memory and an aspiration criterion based on the value of the objective function was implemented in [7]. It was able to obtain good-quality solutions for two real distribution systems of 136 nodes and 202 nodes. In [8], a node-depth encoding based on graph theory is proposed to solve very large scale DNR problems. A multi-objective evolutionary algorithm is used in conjunction with the node-depth encoding and two crossover operators: preserve ancestor operator (PAO) and change ancestor operator (CAO). These operators generate only feasible configurations, i.e., radial DNs that supply power to the entire network. The method obtains good quality solutions for very large distribution networks. An harmony search metaheuristic is proposed in [9] to solve the DNR for two test systems. The method is compared with a GA, an improved GA and an improved TS algorithm, showing better convergence characteristics. The work presented in [10] uses graph theory to represent the network and a GA to solve the reconfiguration problem. The objective function comprises electrical losses and switching mitigation. Two novel network representations that generate only radial topologies were proposed: subtractive sequential encoding and additive sequential encoding. Virtually every type of metaheuristic has already been applied to the DNR problem such as artificial bee colony algorithm, plant growth algorithm, mixed-integer hybrid differential evolution, among many others.

In all metaheuristic techniques, the encoding of the solution is fundamental for the efficiency of the method. The encoding ideally should be able to generate only feasible solutions, reducing the size of the search space and running times of the algorithm[11].

The rest of the paper is organized as follows: Section 2 presents the formulation for the problem of distribution network reconfiguration for loss reduction. Section 3 describes the codification for DNR problems used in this paper. Section 4 introduces biased random key genetic algorithms. Section 5 presents information about the test systems and BRKGA parameters. Section 6 presents the results of the study and Section 7 draws some conclusions.

2 PROBLEM FORMULATION

The electric distribution network reconfiguration problem for electric loss minimization can be formulated as follows:

$$\text{Min } f = \sum_{ij=1}^{|\Omega_l|} k_{ij} r_{ij} \left|I_{ij}\right|^2 \tag{1}$$

Subject to

$$P_{Si} - P_{Di} - \sum_{j \in \Omega_{bi}} k_{ij} P_{ij} = 0 \quad \forall i \in \Omega_b \tag{2}$$

$$Q_{Si} - Q_{Di} - \sum_{j \in \Omega_{bi}} k_{ij} Q_{ij} = 0 \quad \forall i \in \Omega_b \tag{3}$$

$$S_{ij} \leq \bar{S}_{ij} \quad \forall (ij) \in \Omega_l \tag{4}$$

$$\underline{V_i} \leq V_i \leq \bar{V}_i \quad \forall i \in \Omega_b \tag{5}$$

$$k_{ij} \epsilon \{0,1\} \quad \forall ij \in \Omega_l \tag{6}$$

$$g \epsilon G \tag{7}$$

In this formulation, i and j represent generic nodes of the electrical distribution system, where Ω_l is the set of all branches ij of the network connecting nodes i and j, and $\Omega_b$is the set of all nodes i. The symbol r_{ij} stands for the electrical resistance of branch ij. The symbols P_{ij} and Q_{ij} are the active and reactive power flows of branch ij.

P_{Di} is the active power demand at node i and P_{Si} is the active power supply at the same node. Q_{Di} is the reactive power demand at node i and Q_{Si} is the reactive power supply at the same node. The objective function (1) represents the power losses of the distribution system operation. Equations (2) and (3) represent the power flow balance equations, derived from Kirchhoff's current law. Equation (4) represents operational limits on branch capacity where $S_{ij}$is the apparent power flowing in branch ij and $\overline{S_{ij}}$ is the apparent power capacity of the branch. Equations (5) are the operational limits on the value of voltages at each node i of the network, where $\underline{V_i}$ and $\overline{V_i}$ are the minimum and maximum acceptable voltage magnitudes at node i, respectively. Equation (6) represents the binary nature of k_{ij}. The circuit between buses ij is connected if the corresponding value is equal to one and is not connected if it is equal to zero. Equation (7) represents the radiality constraint of the DNR problem. It states that the graph g of the solution must belong to a set G composed of all allowed network structures, i.e. the set excluding meshed and islanded networks. Many heuristic techniques used for solving the DNR problem consider constraint (7) implicitly, applying equation (8).

$$\text{M} = n_b - 1 \tag{8}$$

Where M is the number of branches of the solution and $n_b$is the number of nodes, where $n_b = |\Omega_b|$. However, this condition is necessary but not sufficient to guarantee the radiality constraint. Metaheuristic techniques normally ensure the radiality constraint in their solutions using graph theory or inside evolutionary operators. A feasible solution to the DNR for loss reduction is a distribution network that satisfies constraints (2) - (7) and whose electrical power losses can be calculated to obtain the value of the objective function f. The parallel BRKGA for DNR proposed in this work uses a set of rules derived from graph theory that ensure the feasibility of solutions generated by the metaheuristic.

3 GRAPH THEORY BASED CODIFICATION

In evolutionary computation, codification is the process of representing an element which belongs to the search space of a problem being solved. The evaluation of the fitness of a feasible element is accomplished after it has been decoded and this information is used to guide the search process. The BRKGA framework allows the decodification of encoded solutions to result in exclusive feasible solutions, without reliance on GA operators due to the independence of GA and decoder.

Graph theory can be advantageously used to aid in the codification of solutions of DNR problems. A distribution network can be seen as a graph G composed of a set of nodes N and a set of edges $E - G(N,E)$. The codification used in this work is derived from graph theory and uses a set of rules to correct infeasible individuals and, thus, generate only feasible vectors during decodification. This set of rules was proposed in [12] to be used in conjunction with any metaheuristic technique. In a problem of DNR for loss reduction, feasibility of a solution means it is radial, without isolated nodes from the network. The radial configurations of a distribution network are called trees of its associated graph. Consider that every edge of the distribution network graph (DNG) contain a switch. A tree with N nodes contains $N-1$ graph edges or twigs. The edges that were removed to form the tree are called links. These links form a cotree, which is the complement of the tree. The number of links of a DNG is given by $l = E - (N-1)$, which is usually much less than the twigs. Thus, the links can be used in the codification of solutions of metaheuristic techniques, reducing the size of the solution vector. In the following, some terms are defined.

Principal node: the junction of three or more elements of the DNG.
Exterior node: the node located at the perimeter of the DNG.
Interior node: the node located inside the perimeter of the DNG.

Loop vector: the set of elements constituting a closed path in a DNG. This closed path cannot contain in its interior another closed path.
Common branch vector: the set of elements which are common between any two loop vectors of a DNG.
Prohibited group vector: the set of the common branch vectors. From each of them, if one element is opened, then one or more interior nodes of the DNG will be islanded. The size of a prohibited group vector cannot be greater than l.

With these definitions in mind, the switches that are actually links of a cotree, will be used to represent a solution of the DNR problem. The solution vector represents a cotree which must have a corresponding tree that is feasible. This is accomplished by forming the solutions vectors in accordance to the following set of rules:
Rule 1: each candidate switch must belong to its corresponding loop vector.
Rule 2: only one candidate switch can be selected from one common branch vector.
Rule 3: all the common branch vectors of a prohibited group vector cannot participate simultaneously to form an individual.
These set of rules guarantee the production of feasible individuals, meaning that only radial configurations without islanded nodes are built, avoiding the necessity of mesh checks on solutions. Rule 1 prevents the islanding of exterior node(s), Rule 2 prevents the islanding of interior node(s) and Rule 3 prevents the islanding of principle interior node(s) of the distribution network, respectively.
To illustrate the above definitions, Fig. 1 shows a 14-node graph where the principal nodes are 1,2,8, and 5. The exterior nodes are 1,2,3,4,5,6, and 7. The interior nodes are 8,9,10,11,12,13, and 14.

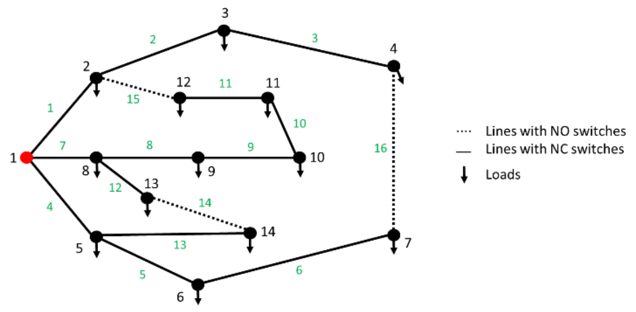

Figure 1: 14-node graph.

Table 1 gives the vectors used to form a feasible solution to the 14-node graph.

Table 1: Loop Vectors, Common Branch Vectors and Prohibited Group Vectors of the 14-Node System.

Loop vectors	Common branch vectors	Prohibited group vectors
L_1 = [1,7-11,15]	C_{12} = [7]	P_1 = [C_{12}, C_{13}, C_{23}]
L_2 = [4,12-14,7]	C_{13} = [8-11,15]	
L_3 = [8-11,15,2-3,16,5-6,12-14]	C_{23} = [12-14]	

More details about the codification can be found in [12].

4 BIASED RANDOM-KEY GENETIC ALGORITHMS

A biased random-key genetic algorithm is an evolutionary strategy that has been applied to solve search and optimization problems from many different research fields. They are nature inspired metaheuristics derived from the genetic algorithms with random keys (RKGA) introduced in [13] for solving combinatorial optimization problems involving sequencing. It uses a random-key alphabet comprised of random real numbers between 0 and 1. The search space comprised of vectors with entries between 0 and 1 is the search space of the BRKGA. Thus, these vectors must be decoded to find their representation in the search space of the problem being solved, and a problem-specific decoder must be devised to accomplish this task.

At the beginning of the evolutionary process, the BRKGA generates an initial population of p individuals. The initial population is then decoded so that the *fitness* of each individual can be computed. The population is then divided into a small set p_e of elite individuals with the best fitness, and another set of $p - p_e$ individuals. To avoid entrapment in local optima, the BRKGA introduces p_m mutants into the population. To form the population of the next generation, the set p_e is copied, unchanged, to the next generation. The algorithm completes the number of individuals by generating $p - p_e - p_m$ vectors of random keys using parametrized uniform crossover [14]. The crossover operation is the main difference between a BRKGA and a RKGA. In a RKGA crossover, both parents are chosen randomly from the entire population, whereas in a BRKGA, a parent is always chosen from an elite set, which introduces the elitism principle in the reproduction process. This modification is enough to make the biased version of the GA to outperform the unbiased version [15]. The BRKGA uses the parameterized uniform crossover where, after two parents are selected for mating, for each gene, we toss a biased coin to select which parent will contribute with the allele to the offspring chromosome. Fig. 2 shows the transitional process between consecutive generations of a BRKGA.

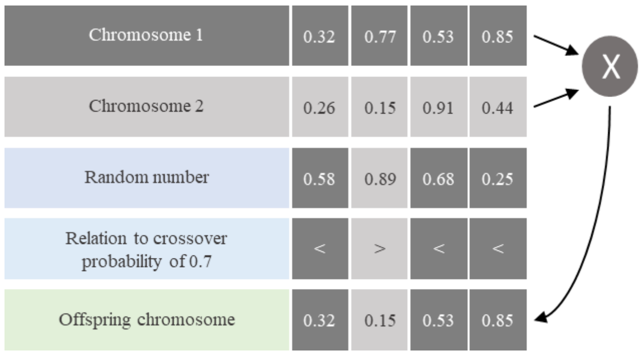

Figure 2: Parametrized uniform crossover used in BRKGAs.

The flowchart of the BRKGA is shown in Fig. 3. The BRKGA evolutionary process is problem-independent and this allows for reuse of software and permits the algorithm designer to concentrate on building the problem specific decoder. The fitness evaluation of individuals is done after the execution of a load flow software, which calculates the electrical state of the network using the chromosome produced by the evolutionary process.

4.1 Parallel Implementation

Biased random-key genetic algorithms are well suited for the application of parallelization techniques. Candidates for parallelization include the operations:

- Generate p vectors of random keys.
- Generate p mutants in next population.
- Combine elite parent with other parent to produce offspring.
- Decode each vector of random keys and compute its fitness.

Each of these four operations involve parallel computations, whereas the fourth is expected to contribute more significantly to overall computational speedup. Another type of parallel implementation involves the use of multiple populations that evolve independently and periodically exchange good quality solutions. This implementation represents the single program multiple data (SPMD) paradigm and was used in this work to speed up the convergence of the evolutionary process. Each core of a multiple-core processor runs a copy of the program and evolve a population of individuals. After a pre-determined number of generations, the two overall best chromosomes from all populations are inserted into all the other populations.

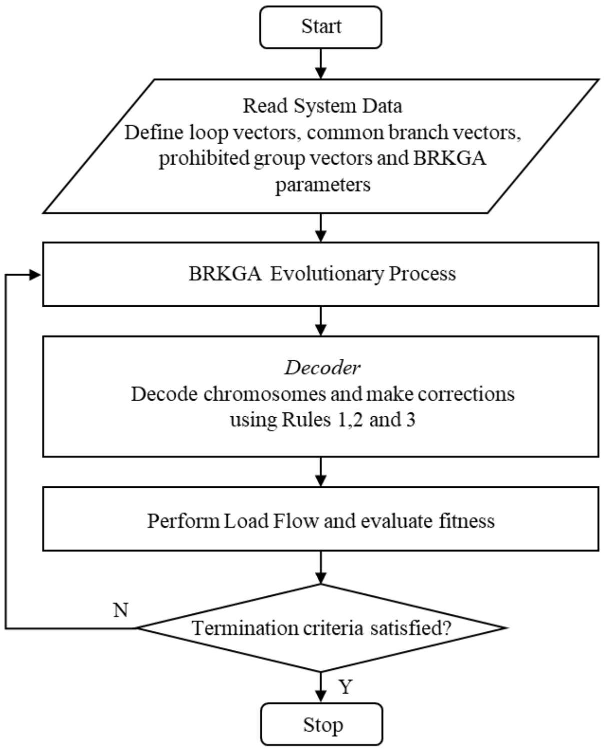
Start
Read System Data
Define loop vectors, common branch vectors, prohibited group vectors and BRKGA parameters
BRKGA Evolutionary Process
Decoder
Decode chromosomes and make corrections using Rules 1,2 and 3
Perform Load Flow and evaluate fitness
Termination criteria satisfied?
N
Y
Stop

Figure 3: Flowchart of the BRKGA.

5 IMPLEMENTATION DATA

This section presents information about the test systems and the BRKGA parameter setting.

5.1 BRKGA Parameters

A BRKGA has a problem-independent module, which executes the evolutionary process, and a problem-dependent module, which consists of the decoder. A few parameters need to be set in a BRKGA. They are the number of genes in a chromosome (n), the population size (p), the size of the elite solution population (p_e), the size of the mutant solution population (p_m), and the elite allele inheritance probability (ρ_e), i.e. the probability that the gene of the offspring inherits the allele of the elite. In this work, population sizes of 100, 80 and 50 individuals have been used. Table 2 gives the parameter values adopted for each population size. These parameter settings follow the guidelines given in [16].

Table 2: **BRKGA Parameter Values**

Population Size	p_e	p_m	ρ_e
100	15	10	0.7
80	12	8	0.7
50	7	5	0.7

5.2 Test Systems

The first test system is a 33-bus distribution system [1] with 37 branches and 5 NO switches. The second system is a 69-bus distribution system [17] with 73 branches and 5 NO switches. These two network topologies have five loop vectors, seven common branch vectors and six prohibited group vectors. Table 3 gives the initial configurations for the test systems used in this work.

Table 3: **Initial Configurations of Tested Systems**

Data	**Systems**	
	33-node	*69-node*
Number of Branches	37	73
Number of Open Switches	5	5
Active Load (MW)	3.7	3.8
Reactive Load (Mvar)	2.3	2.7
Nominal Voltage (kV)	12.66	12.66
Active Losses (MW)	0.208	0.239
Minimum Voltage (pu)	0.911	0.903

6 RESULTS AND DISCUSSION

This section presents the computational results obtained from the application of the parallel BRKGA to solve the test systems. The algorithm performance is also compared with other GA implementations. The algorithm was developed in Matlab© using the Matpower toolbox [18] and the parallel programming toolbox [19]. The simulations were executed on a personal computer with an Intel Core i7-6700HQ @2.6 GHz with 16GB of RAM. This processor has a total of 4 computing cores, which were used to run the parallel BRKGA.

6.1 33-Node Test System

To compare the performance of the serial and parallel version of the BRKGA, they were independently executed 50 times each and the CPU times needed to find the optimal solution were recorded. Table 4 gives the optimal solution to this case. The optimal solution was found in all runs of the algorithm.

Table 4: **Optimal Solution for 33-Node System**

Test system	Optimal configuration	Real power loss (MW)	Minimum node voltage (pu)
33-node	7,9,14,37,32	0.1389	0.9423

As with most stochastic search methods, the continuous random variable time to target solution of a BRKGA has an empirical distribution that approximates a shifted exponential distribution These graphs are used to characterize the running times of stochastic algorithms for combinatorial optimization. Time-to-target (TTT) plots display on the ordinate axis the probability that an algorithm will find a solution at least as good as a given target value within a given running time, shown on the abscissa axis [21]. Table 5 compares the performance of the serial and parallel versions using the number of generations, number of power flows and TTT in seconds as performance indicators.

Table 5: **Performance Indicators for Serial and Parallel BRKGA – 33-Node**

Serial (S) And Parallel (P) Performance Indicators						
Indicators (Average)	Population Size					
	50		80		100	
	S	P	S	P	S	P
Generations	34.42	8.76	28.64	6.62	16	5.52
Power Flows	1721	488	1432	609.6	1600	652
Time To Target (sec.)	3.93	1.62	3.244	2.02	3.65	2.2

The parallel version of the algorithm using a population of 50 individuals shows the best performance in terms of running times and number of power flows executed. Fig. 4 shows the TTT plot for this case. The graph shows that the parallel algorithm always finds the optimal solution in less than 4 seconds and has a 70% probability of finding the optimal solution in around 2 seconds. The average speedup between the serial and parallel version is

2.42. The population of 100 individuals gives the best performance in terms of number of generations to reach the optimal solution.
The parallel version was set to exchange elite solutions between the processors once at every 10 generations. Table 6 gives the performance indicators for exchanges at every 5 and 7 generations for a population of 50 individuals.

Table 6: Performance Indicators for Parallel BRKGA – 33-Node

Parallel BRKGA Performance Indicators		
Indicators (Average)	Exchange of Elite Solutions	
	Every 5 Generations	Every 7 Generations
Generations	7.4	7.3
Power Flows	420	415
Time To Target (sec.)	1.39	1.39

As shown in Table 6, the parallel BRKGA with solution exchange at every 7 generations and 5 generations exhibit the same performance with respect to running times, which is better than the algorithm with exchanges at every 10 generations. The average speedup between the serial and parallel version is of 2.83.

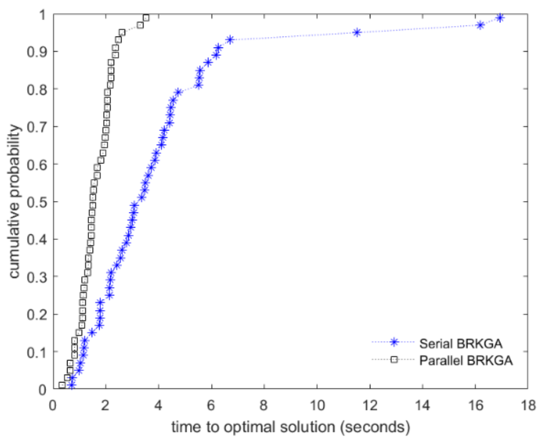

Figure 4: TTT plot for a population of 50 individuals – 33-Node.

These results can be compared with four different genetic algorithms presented in [10], namely, conventional GA, improved GA, SSE and ASE. In this study, ten algorithm runs were executed and the average number of generations to convergence were recorded. Running times cannot be compared due to the different hardware used in the simulations. Only the serial BRKGA and ASE reached the optimal solution in all 10 simulation runs and presented an average of 16 and 8.4 generations to converge, using a population of 100 individuals. The parallel BRKGA required an average of 5.52 generations for the population of 100 individuals.

6.1 69-Node Test System

Table 7 gives the optimal solution to this case. The optimal solution was found in all runs of the algorithm.

Table 7: Optimal Solution for 69-Node System

Test system	Optimal configuration	Real power loss (MW)	Minimum node voltage (pu)
69-node	14,56,61,69,70	0.0997	0.9423

Table 8 compares the performance of the serial and parallel versions for the 69-Node test system.

Table 8: Performance Indicators for Serial and Parallel BRKGA – 69-Node

Serial (S) And Parallel (P) Performance Indicators						
Indicators (Average)	Population Size					
	50		80		100	
	S	P	S	P	S	P
Generations	41.7	9.56	25.36	8.52	20.22	7.7
Power Flows	2085	528	2028.8	761.6	2022	870
Time To Target (sec.)	6.32	2.23	6.19	3.24	6.08	3.66

The parallel algorithm shows the best performance with a population of 50 individuals. The average speedup between the serial and parallel version is 2.83. Fig. 5 shows the TTT plot for this case. The parallel BRKGA takes at most 5 seconds approximately to find the optimal solution, while the serial BRKGA takes around 22 seconds in a worst-case execution.

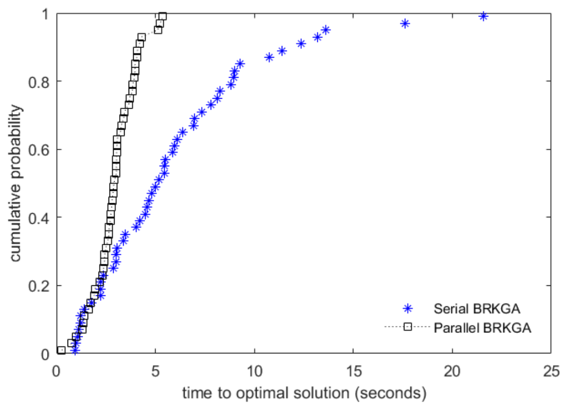

Figure 5: TTT plot for a population of 50 individuals – 69-Node.

Table 9 gives the performance indicators for exchanges at every 5 and 7 generations for a population of 50 individuals. As it can be seen, the performance is almost identical, but worse than the algorithm with exchanges at every 10 generations. In this case, the average speedup between the serial and parallel version is 2.54.

This is because the 69-Node system requires more generations to find the optimal solution, and too frequent exchanges of solutions contributes more to the running times due to communication overhead between the cores.

The performance of the BRKGA for the 69-Node system can also be compared with respect to average convergence generation with an artificial immune algorithm [20], a hybrid genetic particle swarm optimization algorithm [21], and a hybrid intelligent algorithm [22].

Table 9: Performance Indicators for Parallel BRKGA – 69-Node

Parallel BRKGA Performance Indicators		
Indicators (Average)	Exchange of Elite Solutions	
	Every 5 Generations	Every 7 Generations
Generations	9.98	10.04
Power Flows	549	552
Time To Target (sec.)	2.49	2.49

These algorithms converge, on average, in 34, 43 and 30.20 generations for 100 independent runs, respectively, whereas the serial BRKGA converges in 21.28 generations. The parallel BRKGA required an average of 7.7 generations for 50 simulation runs.

7 CONCLUSIONS

This paper proposes a parallel multi-population biased random key genetic algorithm to solve a classic power system combinatorial optimization problem called distribution network reconfiguration. The BRKGA produces solutions that are called random keys inside the real-valued interval [0,1) and a decoder is used to map these solutions into the problem's search space. The decoder is built applying a set of rules derived from graph theory that guarantees the generation of only feasible solutions to the problem. This procedure reduces the size of the search space and avoids time consuming feasibility checks on solutions. Elitism is used efficiently by always using an elite parent on the crossover phase and by copying the entire elite set of one generation onto the next generation. Time to target plots for the BRKGA using different population sizes were drawn to assess the performance of the algorithm. Performance comparisons were made between the serial and parallel implementations of the BRKGA. Average speedups of 2.83 times were obtained with the parallel version for the most efficient implementations. The bigger sized populations were able to converge to the optimum solution in fewer generations in comparison to the smaller populations for both the serial and parallel implementations.

The two test systems used are considered medium sized problems and future investigations will be done on larger systems. The use of multiple populations permits the exchange of information regarding good individuals found in each of these populations and reduces the computation time to find the optimal solution for the test systems.

ACKNOWLEDGMENTS

Haroldo de Faria Jr. acknowledges the University of Luxembourg for enabling his research stay and for providing the necessary conditions to conduct this research.

REFERENCES

[1] Savier, J.S. and Das, D. 2007. Impact of network reconfiguration on loss allocation of radial distribution systems. IEEE Transactions on Power Delivery. 22, 4 (Oct. 2007), 2473–2480.

[2] Festa, P., Pardalos, P., Pitsoulis, L. and Resende, M. 2007. GRASP with path relinking for the weighted MAXSAT problem. Journal of Experimental Algorithmics. 11, (Feb. 2007), 2.4. DOI:https://doi.org/10.1145/1187436.1216581.

[3] Nara, K. A., Kitagawa, M. and Ishihara, T. 1992. Implementation of genetic algorithm for distribution systems loss minimum re-configuration. IEEE Transactions on Power Systems. 7, 3 (1992), 1044–1051. DOI:https://doi.org/10.1109/59.207317.

[4] Chu, P.C. and Beasley, J.E. 1997. A genetic algorithm for the generalized assignment problem. Computers & Operations Research. 24, 1 (Jan. 1997), 17–23. DOI:https://doi.org/10.1016/S0305-0548(96)00032-9.

[5] Carreno, E.M., Romero, R. and Padilha-Feltrin, A. 2008. An efficient codification to solve distribution network reconfiguration for loss reduction problem. IEEE Transactions on Power Systems. 23, 4 (Nov. 2008), 1542–1551. DOI:https://doi.org/10.1109/TPWRS.2008.2002178.

[6] Zhenkun, Li, Chen, X., Yu, K., Sun, Y. and Liu, H. 2008. A hybrid particle swarm optimization approach for distribution network reconfiguration problem. 2008 IEEE Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century (Jul. 2008), 1–7.

[7] Guimarães, M.A.N. and Castro, C.A. 2005. Reconfiguration of distribution systems for loss reduction using tabu search. 15th PSCC (Liège, 2005).

[8] Santos, A.C., Delbem, A.C.B., London, J.B.A. Jr. and Bretas, N.G. 2010. Node-depth encoding and multiobjective evolutionary algorithm applied to large-scale distribution system reconfiguration. IEEE Transactions on Power Systems. 25, 3 (Aug. 2010), 1254–1265. DOI:https://doi.org/10.1109/TPWRS.2010.2041475.

[9] Srinivasa Rao, R., Narasimham, S.V.L., Raju, M.R. and Srinivasa Rao, A. 2011. Optimal network reconfiguration of large-scale distribution system using harmony search algorithm. IEEE Transactions on Power Systems. 26, 3 (Aug. 2011), 1080–1088. DOI:https://doi.org/10.1109/TPWRS.2010.2076839.

[10] de Macedo Braz, H.D. and de Souza, B.A. 2011. Distribution network reconfiguration using genetic algorithms with sequential encoding: Subtractive and additive approaches. IEEE Transactions on Power Systems. 26, 2 (May 2011), 582–593. DOI:https://doi.org/10.1109/TPWRS.2010.2059051.

[11] de Faria Jr., H., Resende, M.G.C. and Ernst, D. 2017. A biased random key genetic algorithm applied to the electric distribution network reconfiguration problem. Journal of Heuristics. (Aug. 2017), 1–18. DOI:https://doi.org/10.1007/s10732-017-9355-8.

[12] Swarnkar, A., Gupta, N. and Niazi, K.R. 2011. A novel codification for meta-heuristic techniques used in distribution network reconfiguration. Electric Power Systems Research. 81, 7 (Jul. 2011), 1619–1626. DOI:https://doi.org/10.1016/j.epsr.2011.03.020.

[13] Bean, J.C. 1994. Genetic algorithms and random keys for sequencing and optimization. ORSA Journal on Computing. 6, 2 (May 1994), 154–160. DOI:https://doi.org/10.1287/ijoc.6.2.154.

[14] Spears, V.M. and DeJong, K.A. 1991. On the virtues of parameterized uniform crossover. *In Proceedings of the fourth international conference on genetic algorithms*. (1991), 230--236.

[15] Gonçalves, J.F., Resende, M. and Toso, R.F. 2014. An experimental comparison of biased and unbiased random-key genetic algorithms. Pesquisa Operacional. 34, 2 (Aug. 2014), 143–164. DOI:https://doi.org/10.1590/0101-7438.2014.034.02.0143.

[16] Gonçalves, J.F. and Resende, M.G.C. 2011. Biased random-key genetic algorithms for combinatorial optimization. Journal of Heuristics. 17, 5 (Oct. 2011), 487–525. DOI:https://doi.org/10.1007/s10732-010-9143-1.

[17] Baran, M. E. and Wu, F. F. 1989. Network reconfiguration in distribution systems for loss reduction and load balancing. IEEE Transactions on Power Delivery. 4, 2 (1989), 1401–1407.

[18] Zimmerman, R.D., Murillo-Sánchez, C.E. and Thomas, R.J. 2011. Matpower: Steady-state operations, planning and analysis tools for power systems research and education. IEEE Transactions on Power Systems. 26, 1 (2011), 12–19.

[18] Parallel Computing User's Guide. 2004-2017. The MathWorks, Inc.

[20] Wenchuan, M. and Jiaju, Q. 2006. An artificial immune algorithm to distribution network reconfiguration. *In Proceedings of the CSEE.* 26, 17 (2006), 25-29.

[21] Zhang, C. Q., Zhang, J. J. and Gu X. H. 2007. The application of hybrid genetic particle swarm optimization algorithm in the distribution network reconfigurations multi-objective optimization. *In Proceedings of the Third International Conference on Natural Computation*. 2, (2007), 455-459.

[22] Zifa, L., Shaoyun, G. and Yixin, Y. 2005. A hybrid intelligent algorithm for loss minimum reconfiguration in distribution networks. In Proceedings of the CSEE 5, 15 (2005), 73-78.