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 network reconfiguration problem (DNR) is used to modify the topology of the distribution system in order to reduce active power losses on the feeders. This objective is 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.

2 METHODOLOGY AND BRKGAs

The electric distribution network reconfiguration problem for electric loss minimization is solved using a BRKGA. The mathematical formulation can be found in [1]. The codification used in this work is derived from graph theory and uses a set of three rules to correct infeasible individuals and, thus, generate only feasible vectors during decodification. This set of rules was proposed in [2] to be used in conjunction with any metaheuristic technique. The BRKGA framework allows the decodification of encoded solutions to result in exclusive feasible solutions due to the independence of GA and decoder.

This work uses a parallel BRKGA where multiple populations evolve independently and periodically exchange good quality solutions 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. The BRKGA parameter values are the number of genes in a chromosome, the population size, the size of the elite solution population, the size of the mutant solution population, and the elite allele inheritance probability, i.e. the probability that the gene of the offspring inherits the allele of the elite. These parameter settings follow the guidelines given in [3].

3 RESULTS AND DISCUSSION

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.

3.1 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 1 compares the performance of the serial and parallel versions using some performance indicators.

Table 1: 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. 1 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.

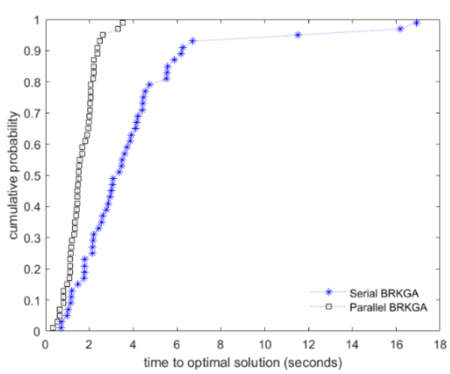

Figure 1: TTT plot for a population of 50 individuals – 33-Node.

3.2 69-Node Test System

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. Table 2 compares the performance of the serial and parallel versions

Table 2: 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

Table 3 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.

Table 3: 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

6 CONCLUSIONS

This paper proposes a parallel multi-population BRKGA to solve a classic power system combinatorial optimization problem called distribution network reconfiguration. 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.

ACKNOWLEDGMENTS

Haroldo de Faria Jr. acknowledges the University of Luxembourg for providing the necessary support for this research.

REFERENCES

[1] 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.

[2] 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.

[3] 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.