Evolutionary Algorithms

Practical Assignment 1

Solving a Power Distribution Network Reconfiguration Problem (PDNR) Using Genetic Algorithms

Power Distribution Network Reconfiguration (PDNR) is an important method of optimizing the distribution system, which is significant to enhancing the security, the efficiency, and the reliability of the system. It is also a well-known NP-hard problem in industry. The purpose of PDNR is to **minimize** the active power loss in this practical assignment.

Two types of switches exist in a power network system, and they are: normally closed switches and normally open switches. Network reconfiguration is the process of adjusting the topology of the power network, by changing the status of these switches in a power network. The constraints of the power network are: no cycles and no islands in the topology, ensuring no short circuits and supplying power to all the customers in a power network.

Assignment

Implement a Genetic Algorithm (GA) in MATLAB according to the generational GA model.

Test the performance on the 119 test problem provided in **PA1.zip** and compare the performance of your GA implementation to Monte Carlo (MC) search of which an implementation.

Use a maximum of 10,000 function evaluations for each run, and report on your results averaged over 20 runs per approach.

Deadline: Friday 9 November 2018, 23:59.

Your submission should consist of three files (no more, no less): a report in PDF format (≥4 pages), your MC implementation and your GA implementation in MATLAB. These are to be

1) submit through Blackboard;
2) delivery the hardcopy of your report (only report, not including codes) to Furong Ye (room #163).

Read carefully the *MATLAB Implementation Details* and the *Report Structure Guidelines* provided below.

PA1.zip containing all files mentioned can be downloaded from Blackborad.

MATLAB Implementation Details

Your MC implementation should consist of **one** .m file named ***studentnumber1_studentnumber2*_mc.m** (replacing lastname1 and lastname2 by your own names) and should be structured as follows:

```
function [aopt, fopt] =
studentnumber1_studentnumber2_mc(eval_budget)

…

end
```

Your GA implementation should consist of **one** .m file named ***studentnumber1_studentnumber2*_ga.m** (replacing lastname1 and lastname2 by your own names) and should be structured as follows:

```
function [aopt, fopt] =
studentnumber1_studentnumber2_ga(eval_budget)

…

end
```

Here, *eval_budget* is the number of function evaluations that the GA is allowed to use per run (i.e., the allowed number of calls to **calculation_119.m**, see below, not the number of generations!), *aopt* is the best solution (**decimal format**) found by the algorithm, and *fopt* the accompanying best fitness.

For your convenience, we provide **valid_119.m** and **calculation_119.m**: the former file is the feasible check function for 119 system and return 1 when an evaluated offspring is feasible; the latter returns an active power loss value of a **feasible** offspring; **para119.mat** is the parameters of 119 system, where *para.n*, *para.lb* and *para.ub* represent how many loops, the lower bound of each loop, and the upper bound of each loop respectively. Load this file in your code during the parameter setting part.

Feasible check of an offspring *a* (represented as a **column** vector of **decimal** numbers) in MATLAB is thus done as follows:

```
flag = valid_119(a);
```

`flag = 1` when an offspring *a* is feasible, and `flag = 0` when *a* is infeasible;

Evaluating a **feasible** offspring *a* (represented as a **column** vector of **decimal** numbers) is done as follows:

```
fitness = calculation_119(a);
```

This evaluation function (**calculation_119.m**) is based on the matpower4.1 toolbox, which is an open source platform for power flow calculation.

Some important things:

- Add the path of matpower4.1 folder to MATLAB environment firstly, for calling the evaluation function. You can add by GUI or using *addpath* command in MATLAB.
- The input parameter (offspring *a*) for **calculation_119.m** and **valid_119.m** should be a decimal vector (actually an integer vector), you can decide what kind of encoding system is in your GA code. If you use binary system, you need to write your own function to convert a binary bitstring into a decimal vector, for the feasible check and evaluation functions. OR, you can use integer encoding in your GA.
- You need to write Monte-Carlo algorithm by yourself.

We will compare all submissions using an automated script, this requires all plotting, printing to the command line etc. to be disabled in the submitted version of your work!

Some References on Power Distribution Network Reconfiguration

- Electric power distribution
- Electric power transmission
- Power distribution network reconfiguration by evolutionary integer programming
- Power loss minimization in distribution system using network reconfiguration in the presence of distributed generation

Report Structure Guidelines (≥4 pages)

Title + Authors (names, email addresses, and student numbers)

Introduction

Introduction text here.

Problem Description

Brief description of the optimization problem here.

Implementation

Outline of your algorithm, algorithm parameters, and settings used for those parameters. Make sure that the algorithm and the results are reproducible from your description.

Experiments

Description of the experiments and the results. Use the following tables and figures to report on the results:

	Monte-Carlo Search		Genetic Algorithm	
119 System	Avg	Std dev	Avg	Std dev

Table 1: Final solution quality after 10,000 function evaluations, averaged over 20 runs

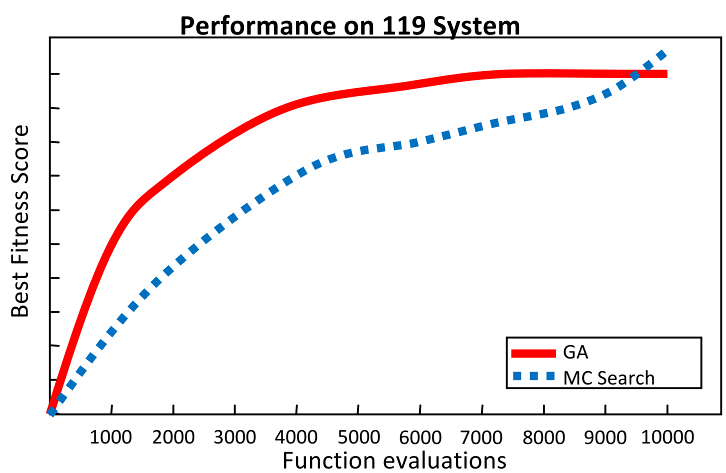

Figure 1: Final solution quality after 10,000 function evaluations, averaged over 20 runs

Make sure to present your results in a way that is convenient to the reader, do not blindly include plots of all your experiments, try to combine data in figures!

Discussion and Conclusion

Summarize the results and conclude your report.