Genetic Algorithms and Evolutionary Computing Project

2018-2019

Introduction

The aim of the project is to evaluate and to improve a Genetic Algorithm to solve the Traveling Salesman Problem using the Genetic Algorithms Matlab Toolbox, to experiment with various representations, genetic operators and parameters, and to report on the results.

You can use a template Matlab program for the TSP, that uses the Genetic Algorithms Matlab Toolbox and some extra procedures (Matlab .m files) for:

- input and visualization,
- representation (or coding) of a 'tour' using the adjacency representation,
- initialization of a population,
- recombination (or cross-over) using the alternating edge strategy,
- swap mutation operator
- conversion between adjacency representation and path representation.

Relevant procedures in the GA Toolbox are: `ranking.m, select.m, recombin.m, reins.m`. You can find more information about these functions in the GA Toolbox tutorial (available on Toledo). Relevant procedures in the template program are:

- `tspgui.m` is the main user interface script (use this function to start the program). Use this function as a quick demonstration of the available functions.
- `run_ga` is the main template program. You can use this file as basis for the project.
- `tspfun.m` is the fitness function for adjacency representation. If you decide to use path representation, you have to adapt this function
- `xalt_edges.m` implements the alternating edge cross-over operator. It calls the low-level function `cross_alternate_edges.m`
- `mutateTSP.m` implements mutation. It calls the low-level functions `reciprocal_exchange.m` or `inversion.m`
- `visualizeTSP.m` performs visualization of a TSP tour
- `adj2path.m, path2adj.m` perform conversions between representations

Tasks

1. On Toledo, you can find the GA Toolbox, the template program and tutorials about Matlab. Test the Matlab program to solve a TSP.

2. Perform *a limited set* of experiments by varying the parameters of the existing genetic algorithm (population size, probabilities, ...) and evaluate the performance.

3. Implement a stopping criterion that avoids that rather useless iterations (generations) are computed.

4. Implement and use another representation and appropriate crossover and mutation operators. Perform some parameter tuning to identify proper combinations of the parameters.

5. Test to which extent a local optimisation heuristic can improve the result.

6. Test the performance of your algorithm using *some* benchmark problems (available on Toledo) and critically evaluate the achieved performance.
   Keep in mind that for a large number of cities the search space is extremely large! If your algorithm doesn't perform well for a rather small number of cities, it doesn't make sense to use it for a benchmark problem with a large number of cities ...
   Note: For most of the benchmark problems the length of the optimal tour is known. However, the Matlab template program scales the data. Therefore this scaling must be switched off to be able to compare your result with the optimal tour length.

7. You should select *at least one* task from the list below:

   (a) Implement and use two other parent selection methods, i.e. fitness proportional selection and tournament selection. Compare the results with those obtained using the default rank-based selection.

   (b) Implement one survivor selection strategy (besides the already implemented elitism). Perform experiments and evaluate the results.

   (c) Implement and use one of the techniques aimed at preserving population diversity (e.g. subpopulations/islands, crowding, ...). Perform experiments and evaluate the results.

   (d) Incorporate an adaptive or self-adaptive parameter control strategy (e.g. parameters that depend on the state of the population, parameters that co-evolve with the population, ...). Perform experiments and evaluate the results.

8. Write a short report ($\approx$ 10 pages, appendices and code not included), discussing your implementation and explaining your results. Include your code in the appendix.
   Note: A carefully written report of $\approx$ 10 pages can contain a lot of information, if well written. Be precise but concise! Do not repeat information from the handbook or slides.

Alternatives

- If you don't want to use the Matlab code, you can use another implementation of the Genetic Algorithm (written by yourself or by someone else). However the effort to implement (or to install) and to use another implementation will NOT be taken into account in the evaluation. Only the results and the analysis for the tasks described above will be taken into account.

- If you would like to implement and use a Genetic Algorithm for another problem than the TSP, you should submit a 1-page proposal to Dirk.Roose@cs.kuleuven.be. If the feasibility of this proposal is evaluated positively, you will be allowed to work out this alternative project.

Practical arrangements

- This project should be made in groups of 2 students. If you have a valid reason to make the project alone, send an email to Dirk.Roose@cs.kuleuven.be to ask exception. The estimated workload per student is 35 hours.
- The report (as a PDF file) and the source code (as a ZIP archive) have to be uploaded on Toledo: "Toledo/Assignments/Upload of the project report" Please name both the report and the ZIP-file according to the pattern: *LastName1stAuthor-LastName2ndAuthor*. The hard copy of the report must be put in the postbox near the secretariat of the Department of Computer Science. The report will be briefly discussed during the examination.
  Deadline: Monday January 7 2019, 10 am.

Installing and using the Toolbox

The template Matlab programs for the TSP problem (`tsp.zip`) and the Genetic Algorithm Toolbox (`Toolbox.zip`) can be downloaded from Toledo. It is a slightly modified version of the original toolbox, which was developed at the University of Sheffield, UK.

(on Toledo).

Extract both files (tsp and toolbox) in a directory `../ga` and start Matlab. In Matlab click File... Set Path... Add with Subfolders. And then choose the directory `../ga`. Now Matlab can find both the toolbox and the template.

To change Matlab working directory to the location where you extracted the `tsp.zip` archive, double tap on that directory in the navigation panel. Start the algorithm by typing `tspgui` on the command line. The program initializes with 16 random cities. You can vary different parameters of the genetic algorithm using the sliders. Click 'input' to input a new set of cities. Click 'RUN' to run the genetic algorithm.

The user interface shows three different figures:

- Figure 1 (top left) shows the tour-length of the best individual at each generation.
- Figure 2 (top right) shows the evolution of the tour lengths (maximum, mean and minimum length at each generation).
- Figure 3 (bottom left) shows the histogram of tour length in the population. All the individuals in the population are put in bins with respect to their tour length: (x-axis: distance (tour length), y-axis: number of individuals). Figures 2 and 3 change at each generation step.

For your experiments you may find it useful to use the same placing for your cities several times. You can do this by changing the upper part of `tspgui` (where x and y are created) by e.g. (for NVAR=8) x=[0.1 0.2 0.3 0.4 0.5 0.2 0.3 0.4]'; y=[0.1 0.1 0.1 0.2 0.5 0.2 0.3 0.4]';
or by reading the data from a file.
If you are unfamiliar with Matlab, there is an extensive Matlab Getting Started Tutorial available (on-line and printer-ready):

- `http://www.mathworks.com/access/helpdesk/help/techdoc/learn_matlab/bqr_2pl.html`
- `http://www.mathworks.com/access/helpdesk/help/pdf_doc/matlab/getstart.pdf`

More documentation about Matlab can be found on Mathworks Documentation Website:
`http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.shtml`.