CORK INSTITUTE OF TECHNOLOGY
INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

Metaheuristic Optimization

Assignment 1

Due date:

Assignment should be submitted to Canvas before 11PM **Tuesday Oct 23rd**

Part 1: NP-completeness

1) This problem concerns of the proof of the NP-completeness of 3COL
    a) Convert the formula F into a 3COL graph
    b) Find a solution for the 3COL instance of F and verify that it is a solution for F

If the first letter of your first name ranges between [A,F] use (e.g, Alejandro)
$F = (z_1 \vee z_2) \wedge (z_1 \vee z_2 \vee z_3 \vee z_4)$

If the first letter of your first name ranges between [G, M] use (e.g., Kian)
$F = (z_1 \vee -z_2) \wedge (z_1 \vee z_2 \vee -z_3 \vee z_4)$

If the first letter of your first name ranges between [N, R] use (e.g., Omar)
$F = (-z_1 \vee z_2) \wedge (z_1 \vee z_2 \vee z_3 \vee -z_4)$

If the first letter of your first name ranges between [S, Z] use (e.g., Walter)
$F = (-z_1 \vee -z_2) \wedge (z_1 \vee z_2 \vee -z_3 \vee -z_4)$

Part 2: Genetic Algorithms

The Traveling Salesman Problem (TSP) is one which has commanded much attention in Artificial Intelligence because it is so easy to describe and so difficult to solve. The problem can simply be stated as: if a traveling salesman wishes to visit exactly once each of a list m cities (where the cost of traveling from city i to city j is c_{ij}) and then return to the home city, what is the least costly route the traveling salesman can take.

The importance of the TSP is that it is representative of a larger class of problems known as combinatorial optimization problems. The TSP problem belongs in the class of combinatorial optimization problems known as NP-hard. Today, no one has found a polynomial-time algorithm for the TSP.

A Simple Genetic Algorithm

A simple genetic algorithm can be defined in the following 9 steps:

Step 1: create an initial population of P chromosomes (generation 0)
Step 2: evaluate the fitness of each chromosome
Step 3: Select P parents from the current population via proportional selection (i.e., the selection probability is proportional to the fitness).
Step 4: choose at random a pair of parents for mating. Exchange bit strings with a crossover operation to create two offspring (e.g., one-point crossover)
Step 5: process each offprint by the mutation operation, and insert the resulting offspring in the new population
Step 6: repeat steps 4 and 5 until all parents are selected and mated (P offspring are created)
Step 7: replace the old population of chromosomes in the new population
Step 8: evaluate the fitness of each chromosome in the new population
Step 9: go back to step 3 if the number of generations is less than some upper bound. Otherwise, the final result is the best chromosome created during the search.

Selection Probability:

The parent chromosomes are selected for mating via proportional selection, also known as “roulette wheel selection”. It is defined as follows:

1) Sum up the fitness values of all chromosomes in the population
2) Generate a random number between 0 and the sum of the fitness values
3) Select the first chromosome whose fitness value added to the sum of the fitness values of the previous chromosomes is greater than or equal to the random number

Initial population

In addition to the crossover and mutation operators you will also evaluate the impact of the GAs with the following initial populations:

- Randomly generated population
- Nearest neighbor insertion

Crossover and mutation operators

Write a program that solves the TSP using Genetic Algorithms. Explain and implement the following crossover & mutation operators:

- Uniform order-based crossover
- Cycle Crossover (CX)
- Reciprocal exchange mutation
- Scramble Mutation

I/O Specification

In this assignment you will use the same I/O format as defined in the first lab (week 2).

Problem instances

~~In this project, you will use the following problem instances to evaluate the performance of your algorithms: inst-0.tsp, inst-1.tsp, inst-2.tsp, inst-3.tsp, inst-4.tsp, inst-5.tsp, inst-6.tsp, inst-7.tsp, , inst-8.tsp, and inst-9.tsp.~~

Update: Problem instances
In this project, you will use the following problem instances to evaluate the performance of your algorithms: inst-0.tsp, inst-13.tsp, and inst-16.tsp

Assignment of Marks:

The deliverable of this project will consists of a python code file(s) and a report. The following is the allocation of marks.

- Solution - Part 1 (20%)
- Implementation of Genetic algorithms with the above mentioned crossover and mutation operations (35%)
- A basic evaluation (10%): the basic evaluation should describe the following two basic configurations:

Configuration	Initial Solution	Crossover	Mutation	Selection
1	Random	Uniform Crossover	Reciprocal Exchange	Random Selection
2	Random	Cycle Crossover	Scramble Mutation	Random Selection

Population size = 100
Mutation rate = 0.1

- Extensive evaluation of your GA, i.e., initial population, crossover & mutation operators (20%). You are expected to extensively evaluate the following combination of operations:

Configuration	Initial Solution	Crossover	Mutation	Selection
3	Random	Uniform Crossover	Reciprocal Exchange	Roulette Wheel

4	Random	Cycle Crossover	Reciprocal Exchange	Roulette Wheel
5	Random	Cycle Crossover	Scramble Mutation	Roulette Wheel
6	Random	Uniform Crossover	Scramble Mutation	Best and Second best candidates

Additionally, you are expected to investigate the impact of varying the population size, explore different mutation rates, and evaluate the impact of your GA with and without elite survival.

- Your report should contain a comprehensive description of the algorithms and an extensive evaluation of your results. (15%) And it should describe the experimental design, what experiments are and what they are intended to show. Use tables and figures where appropriate.

~~You should assess the overall performance of your optimization algorithms for solving the travelling salesman problem. Typically, to evaluate the performance of your algorithm for a single configuration you would run your algorithm multiple times (at least 5 times in this project with at least 1000 iterations per execution) and record the best fitness for all runs. You can then report the mean and median fitness across all runs. Your results should show that you have considered the suggested operators (e.g., crossover, mutation and selection).~~

Update:

You should assess the overall performance of your optimization algorithms for solving the travelling salesman problem. Typically, to evaluate the performance of your algorithm for a single configuration you would run your algorithm multiple times (at least 3 times in this project with at least 300 iterations per execution) and record the best fitness for all runs. You can then report the mean and median fitness across all runs. Your results should show that you have considered the suggested operators (e.g., crossover, mutation and selection).

Submission:

This assignment is due on Tuesday, Oct 23rd, 2018. You must submit the following files (in a single zip file):

- All source codes.
- A Readme file, which briefly describes all submitted files. In the Readme file, you should also provide information about compiling environment, compiling steps, execution instructions, etc.
- A Sample file, which describes the test you have run on your program. Also describes any cases for which your program is known not to work correctly.
- Report document (pdf) - including your solutions for Part 1 & Part 2