Assignment 1

COMP 9058 - Metaheuristic Optimisation

October 20, 2021

The assignment should be submitted on Canvas by **Sunday, November 7th, 22:00**. As per MTU regulations, submitting within 7 days of the deadline will result in a 10% penalty, between 7 and 14 days late will result in a 20% penalty, and later than 14 days will result in a 100% penalty applied.
The pdf MUST be named *Lastname_Firstname_StudentNumber_MH1.pdf*. Similarly the folder containing the code to be zipped MUST be named
Lastname_Firstname_StudentNumber_MH1_code.
You need to write a report with a comprehensive description of the experiments and an extensive evaluation (and analysis) of the results. Use tables and figures where appropriate. As always, the random number generator must be seeded to your student number for your experiments.

1 Assignment Description

1.1 NP-Completeness - [30 Marks]

This problem concerns the proofs of the NP-completeness of 3SAT and 3COL

1. Convert the formula F below into a 3SAT formula F', find a solution to F' and verify that this is a solution to F
   (a) If the last digit of your student id is less than 3 use
   $F = (z_1) \wedge (\neg z_1 \vee z_2 \vee z_3 \vee z_4 \vee \neg z_5)$
   (b) If the last digit of your student id is either 3, 4 or 5 use
   $F = (\neg w_1) \wedge (w_1 \vee \neg w_2 \vee w_3 \vee \neg w_4 \vee w_5)$
   (c) If the last digit of your student id is either 6 or 7 use
   $F = (p_1) \wedge (\neg p_1 \vee p_2 \vee \neg p_3 \vee \neg p_4 \vee p_5)$
   (d) If the last digit of your student id is greater than 7 use
   $F = (\neg q_1) \wedge (q_1 \vee q_2 \vee \neg q_3 \vee q_4 \vee \neg q_5)$
2. Convert the following subclauses in your F' to a 3Col graph:
   1st and 5th clauses if the first letter of your first name is in the range A-I.
   2nd and 5th clauses if the first letter of your first name is in the range J-R.
   3rd and 5th clauses if the first letter of your first name is in the range S-Z.

1.2 Genetic Algorithms - [70 Marks]

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. To date, no one has found a polynomial-time algorithm for the TSP.

Simple GA

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

1. Create an initial population of P chromosomes (generation 0)
2. Evaluate the fitness of each chromosome
3. Select P parents from the current population via fitness-biased selection (i.e., the selection probability is dependent on the fitness).
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)
5. Process each offspring by the mutation operation, and insert the resulting offspring in the new population
6. Repeat steps 4 and 5 until all parents are selected and mated (P offspring are created)
7. Replace the old population of chromosomes in the new population
8. Evaluate the fitness of each chromosome in the new population
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.

This assignment requires you to write a program that solves the TSP using Genetic Algorithms with the following parameter settings.

1.2.1 Population initialisation

Two forms to be tested, as implemented in Lab 1:

- Random tour generation
- Nearest neighbor insertion: Choose first city randomly, each city thereafter choose city closest to the last city added to the route and append to the route.

You may incorporate the lab solution that was provided or use your own implementation. Note the lab solution that was provided, and the skeleton GA for TSP code provided, is very basic so as to allow understanding. Efficiency of methods can be greatly improved.

1.2.2 Selection

Select parents for the mating pool using truncation selection, where the percentage truncation is an input parameter.

1.2.3 Crossover and Mutation

Crossover should be performed with crossover probability (P_C) (and if not performing crossover, the parents just become the children and get passed to mutation), and mutation should be performed with probability (P_M).
The crossover operator to be used is the (permutation) Order1 Crossover, the mutation operator is the Inversion mutation.

1.2.4 Replacement

Elitism with a percentage parameter, where 0 would mean full replacement.

I/O Specification

In this assignment you will use the same I/O format as defined in the first lab: The following problem instances will be used to evaluate the performance of your algorithms:

- if the first letter of your surname is in the range A-I :inst-0.tsp, inst-13.tsp, and inst-5.tsp
- if the first letter of your surname is in the range J-R: inst-4.tsp, inst-16.tsp, and inst-6.tsp
- if the first letter of your surname is in the range S-Z: inst-19.tsp, inst-20.tsp, and inst-7.tsp

2 Submission, marking and academic integrity

The deliverable of this project will consists of a python code file(s) and a report. You must follow the correct scheme with last digit of id and first letter of first and last names, **you must state what you used** to avoid confusion. Marks will be lost if this is not done correctly, similarly the **random seed must be set and must be your student id.**

2.1 Submission:

This assignment is due by 22:00 on Sunday, Nov 7th, 2021. You must submit the pdf separately via the Ouriginal assignment submission, and the pdf should include your work for Parts 1 and 2 of the assignment (i.e. everything except the assignment 2 implementation). You **must** submit the following files (in a single zip file) via the code submission:

- All source code.
- 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.

Your pdf **MUST** be named Lastname_Firstname_StudentNumber_MH1.pdf (e.g. Grimes_Diarmuid_R001234567_MH1.pdf). Similarly the folder containing the code to be zipped MUST be named *Lastname_Firstname_StudentNumber_MH1_code*, as must the main *.py* file.

2.2 Rubrics:

Part 1: NP-Completeness

Rubric (Solution and verification aspect only refers to SAT to 3SAT):

Reduction is logically well designed, documented and explained. Solution is correct.	Reduction is logically well designed. Solution is correct	Reduction has slight logic errors that do no significantly affect the results. Solution has slight errors	Reduction has significant logic errors. Solution has errors.	Reduction is completely incorrect. Solution completely incorrect.
(22-30 Marks)	(13-21 Marks)	(5-12 Marks)	(0-5Marks)	

Part 2: GA

Implementation

Pay careful attention to the instructions, your implementation must be based on the exact specification in this document, including naming conventions, and command line format of arguments.
The main file must follow the same naming convention (Lastname_Firstname_StudentNumber_MH1.py) and be created such that it can be run from the command line with parameters passed in for each parameter as follows:

```
python  Grimes_Diarmuid_R001234567_MH1.py inst_file  nRuns nIterations popSize
initalisation Pc  Pm truncationPercentage elitismPercentatge
```

For the *intiialisation* parameter, 0 should be passed in for using random tour, 1 for using nearest neighbor.

The algorithm is logically well designed and efficient without inappropriate design choices (e.g., unnecessary computations per iteration). Code is well commented.	The algorithm always works properly and meets the specification. Code is clean, understandable and well organized.	The algorithm works properly in limited cases	The algorithm is incorrectly implemented
(22-30 Marks)	(13-21 Marks)	(5-12 Marks)	(0-5 Marks)

Advice: *Use small toy instance as discussed in the first lab, for testing and debugging.*

Evaluation

Your evaluation should involve the following experiments:

- Impact of initialization method on performance (random versus insertion).
- Impact of crossover probability (test 2/3 different values including the default of 0.8).
- Impact of mutation probability (test 2/3 different values including the default of 0.05).
- Impact of elitism (with 10% elitism versus without).
- Impact of population size (test 2/3 different values including the default of 100).

The default settings are given as follows (using the command line arguments listed above in that order):

```
python  Grimes_Diarmuid_R001234567_MH1.py inst_file  10 500 100  1 0.8  0.05 0.25 0.0
```

You should evaluate in terms of both solution quality and runtime, including the best and average for solution quality across runs.
You may also want to analyse the evolution of the solution quality across iterations for some sample runs.

Excellent presentation, depth and insight analysis of the empirical results.	Good presentation of the results (e.g., describing the results with well structured tables).	Incomplete and/or unclear presentation of the results.	The results are poorly presented and interpreted, does not demonstrate understand-ing.
(31-40 Marks)	(21-30 Marks)	(11-20 Marks)	(0-10 Marks)

2.3 Academic Integrity

This is an **individual assignment**. The work you submit must be your own. In no way, shape or form should you submit work as if it were your own when some or all of it is not. Any online source that is used must be cited, and a full citation given, e.g. do NOT give "stackoverflow", but the full citation, e.g. https://meta.stackoverflow.com/questions/339152/plagiarism-and-using-copying-code-from-stack-overflow-and-submitting-it-in-an-as. If you are unsure on whether something should be cited, general rule of thumb is to err on the side of caution and include the citation. You can also ask me via email
Collusion: Given how much freedom there is in the assignment, everybody's work will be different. It will be obvious if there is collusion. ***All parties to collusion will be penalized.***
Deliberate plagiarism: You must not plagiarise the programs, results, writings or other efforts of another student or any other third-party. Plagiarism will meet with severe penalties, which can include exclusion from the Institute.
Your report and code will be checked for signs of collusion, plagiarism, falsification and fabrication. You may be called to discuss your submission and implementation with me and this will inform the grading, any penalties and any disciplinary actions.