Habib University
CS 451 - Computational Intelligence
Spring' 2024
Assignment 1 – Evolutionary Algorithm

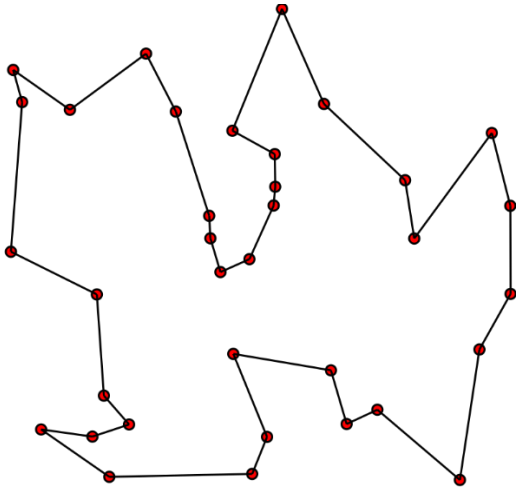
Objective:

The purpose of this assignment is to provide students an insight of stochastic optimization using Evolutionary Algorithms (EA). This exercise will enable them to address some popular computing problems that can be mapped to map to several real-world problems and are known to be computationally hard. The process will expose them to the overall process of EA, its underlying challenges and the impact of different parameters and selection schemes.

Q 1 – Evolutionary Algorithms

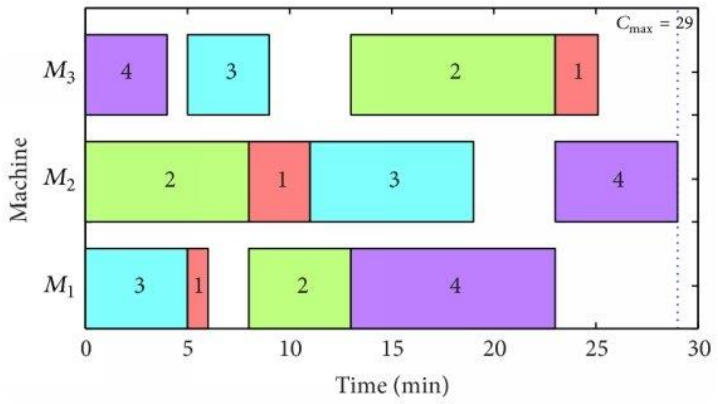
In this question, you will implement the standard process of EA to address the following benchmark problems. You will also plot graphs to show EA progress and its convergence with a concluding analysis. Specific details of the EA process are given in Appendix A.

- ***Travelling Salesman Problem (TSP):*** TSP asks the following question: "Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?"



There are multiple TSP problem instances available online. You will use the dataset of Qatar that contains 194 cities. The optimal tour reported so far for this dataset is of length ***9352*** (and the ***best we have achieved by any CI student is 9939***). The dataset and its related details can be found at: http://www.math.uwaterloo.ca/tsp/world/countries.html (look for Qatar dataset on this page).

- ***Job-shop Scheduling:*** The Job-Shop Scheduling Problem (JSSP) is a widely studied combinatorial, NP-hard optimization problem[1]. The aim of the problem is to find the optimum schedule for allocating shared resources over time to competing activities in order to reduce the overall time needed to complete all activities.



Within a JSSP you will have *n* number of jobs (J_1, J_2, ..., J_n), that will need to be completed using a *m* number shared resources, most commonly denoted as machines (M_1, M_2, ..., M_m). Each job will have operations (*O*) that will need to be completed in a specific order in order for the job to be completed. Operations must be completed at specific machines and require a specific amount of processing time (*p*) on that machine. Machines are assumed to only be able to process one operation at a time, driving the need to optimize the order in which these operations are completed. The goal of the JSSP is to minimize the overall time it takes to complete all *n* jobs within the problem.

There are multiple instances of JSSP available here. You will test and report your results on the first three instances (i.e. abz5-abz7)

You will ensure that your implementation is modular, generic and is structured in an objected oriented model with common classes being reused for both of the problems.

Grading:

The grading will be based on the following components:

Problem Formulation (for three problems)	15%
EA Implementation (EA Process, Selectin Schemes) (Correctness of code, Proper Structuring of code, Generic Implementation)	40%
Fine-tuning of parameters and final solution	10 %
Gathering statistics and plotting graph	15 %
Analysis and Findings	20 %

Q-2 Evolutionary Art [15 Points]

Modify the implementation of EA in Question-1 to evolve a human image using polygons. You can look at the example of evolving Mosa Lisa using polygons and its preliminary implementation. This implementation does not apply the conventional cycle of evolution and still takes quite long to evolve. Can you come up with an efficient yet effective implementation using EA? Try with pictures other than Mona Lisa to see if your implementation is generic enough to work with any human image. You are welcome to incorporate ideas that result in better/quick evolution.

Grading:

Problem Representation	25%
Fitness Computation	15%
EA Process - Modification/Optimization	35%
Image Rendering/Results	25%

Submission:

The assignment will be done in pairs. Please sign up for your Assignment 1 team on Canvas. You will submit your code and the pdf file describing (for all three problems):

- Chromosome representation
- Fitness function
- Optimal value achieved and the optimal solution
- Plots of Avg. BSF and Avg. ASF for various combinations of schemes (shown side by side)
- For Mona Lisa, the report should show a succession of images (say, after every few generations) generated during the evolution.

Finaly, you will mention your overall analysis of various settings and the configuration that worked best for you.

Appendix – A

EA Implementation:

Your EA will perform the following cycle:

- Step 0: Analyse the problem that you are addressing and decide your chromosome representation and fitness function.
- Step 1: Initialize the population randomly or with potentially good ***solutions***.
- Step 2: Compute the ***fitness*** of each individual in the population.
- Step 3: Select parents using a ***selection procedure***.
- Step 4: Create offspring by ***crossover*** and ***mutation*** operators.
- Step 5: Compute the ***fitness*** of the new offspring.
- Step 6: Select members of population to die using a ***selection procedure***.
- Step 7: Go to Step 2 until the termination criteria are met.

The following selection schemes will be implemented:

- Fitness Proportional Selection
- Rank based Selection
- Binary Tournament
- Truncation
- Random

Following parameters will be configurable in your program with some default values given below that you are free to change and observe the behavior of your algorithm:

- Population size: 30
- Number of offspring to be produced in each generation :10
- No. of generations: 50
- Mutation rate: 0.5
- No of Iterations: 10

EA Evaluation:

In each generation, you will note the average fitness (avg) of the generation and the best fitness so far (BSF). Suppose you run the process for 40 generations. You will have following observations:

Generation #	Best Fitness so far (BSF)	Average Fitness so far (ASF)
1		
2		

..		
..		
..		
50		

You need to repeat this exercise (for a single combination of parent and survival selection schemes) K times where K is your number of iterations. Each run will start with a new randomly initialized population.

You will have the following table after K iterations:

Generation #	Run # 1 BSF	Run # 2 BSF	..	..	Run # 10 BSF	Average BSF
1						
2						
..						
..						
..						
40						

A similar table will be obtained for average 'average fitness'.

- Once you have calculated (a) average best-so-far values and b) average average-fitness, you need to plot these values against generation # and analyze their pattern.
- You will repeat the process for different combinations of parent and survival selections such as (not limited to):
  - FPS and Random
  - Binary Tournament and Truncation
  - Truncation and Truncation
  - Random and Random
  - FPS and RBS

Finally, you will provide your analysis on the behavior of EA under different parameters and selection schemes. Which scheme did work best in your case and which one performed worst?