CS 451 : Computational Intelligence

Assignment 1

Evolutionary Algorithms

Mustafa Sohail | ms06860@st.habib.edu.pk
Muhammad Azeem Haider | mh06858@st.habib.edu.pk

February 11, 2024

Dhanani School of Science and Engineering
Habib University
Fall 2023

Copyright @ 2023 Habib University

Contents

1 Introduction

The first assignment of the course Computational Intelligence (CS-451) required us to implement evolutionary algorithms for three problems which are as followed:

1. **Travelling Salesman Problem:** The Travelling Salesman Problem (TSP) is a classic problem in combinatorial optimization. It is the problem of finding a tour of a set of n cities that is of minimum cost. A tour is a permutation of the cities, and the cost of a tour is the sum of the distances between adjacent cities in the tour.
2. **Job-Shop Scheduling Problem:** The job-shop scheduling problem is a classic problem in combinatorial optimization. It is the problem of scheduling a set of n jobs on a set of m machines. Each job consists of a sequence of operations, each of which must be processed on a specific machine for a specific amount of time.
3. **Evolutionary Art (Mona Lisa):** The evolutionary art problem is a classic problem in evolutionary computation. It is the problem of evolving a population of images to match a target image. Each image in the population is represented as a string of genes, and the fitness of an image is the similarity between the image and the target image.

2 Travelling Salesman Problem

2.1 Problem Formulation

The travelling salesman problem requires finding of the minimum cost (distance) to cover all the cities. The **chromosome generation** is being carried out by randomly shuffling the cities of the Qatar dataset.

Furthermore, the **mutate function** assigns each chromosome a random number from 0 to 1, and if the number is less than the mutation rate, mutation occurs.

Finally, the **crossover function** once again, carries out crossover between two parents by assigning the crossover point, and all the repeated cities from the second parent, that are already added in the first part, are added linearly in the end.

2.2 Analysis

While carrying out initial combination for selection schemes for parent selection and survival selection, it was clear that a combination of an explorative scheme as a parent selection and an exploitative scheme as a survival selection would be the best. As a result, two main combinations that were tested were the following:

1. **Parent Selection:** Random **Survival Selection:** truncation
2. **Parent Selection:** tournament selection **Survival Selection:** truncation

The result reported in *Figure 1*, was obtained by the following parameters:

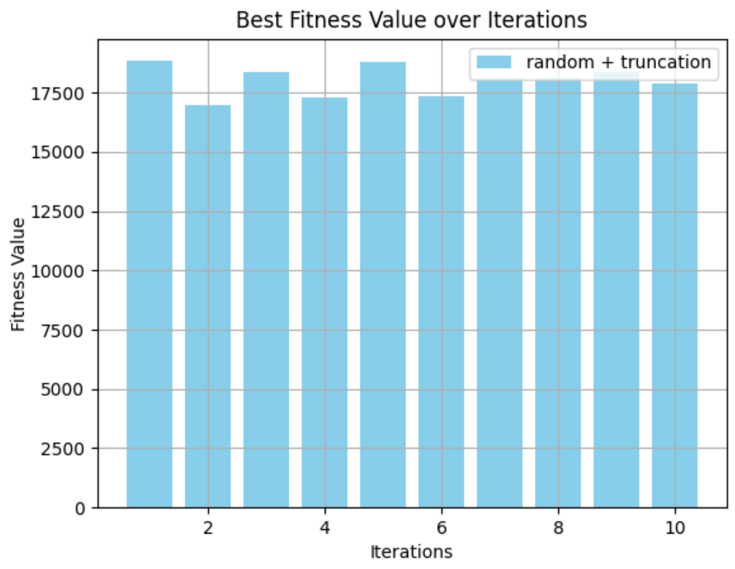

Figure 1: Travelling Salesman Problem: Combination 1

1. **Population Size:** 1000
2. **Offspring Size:** 200
3. **Number of Generations:** 5000
4. **Mutation Rate:** 0.45
5. **Iterations:** 10

3 Job-Shop Scheduling Problem

3.1 Problem Formulation

The job-shop scheduling problem requires finding the minimum time to complete all the jobs on all the machines.

The chromosome is generated by providing each job with an index, and repeating that number of index for the number of operations on that job. For example for abz5, if there are 10 jobs with 10 operations each, each job will be given an index from 0-9 and each number from 0-9 will repeat exactly 10 times. In this manner, we have flattened our chromosome.

The mutate and crossover function work in similar fashion to the travelling salesman problem. The mutate function checks probability assigned and if it is less than mutation rate, mutation occurs, and crossover occurs between two parents, and repeated jobs from parent two are added in the end of the new offsprings.

3.2 Analysis

Unlike, the travelling salesman problem, the job-shop scheduling problem had three input files which were abz(5-7). The analysis for each file will be divided into subsections.

3.2.1 First input file "abz5"

The strategy to solve the job-shop scheduling problem was to use the same combination of parent selection and survival selection as the travelling salesman problem. Since keeping the parent selection scheme entirely explorative through *random* function or on the higher exploration through rank-based selection or tournament selection. The survival selection was to be kept entirely exploitative through truncation or on the higher exploitation through fitness proportional selection. The best score achieved **1242** was through **random** and **truncation**, with the following parameters.

1. **Population Size:** 200
2. **Offspring Size:** 60
3. **Number of Generations:** 2000
4. **Mutation Rate:** 0.5
5. **Iterations:** 10

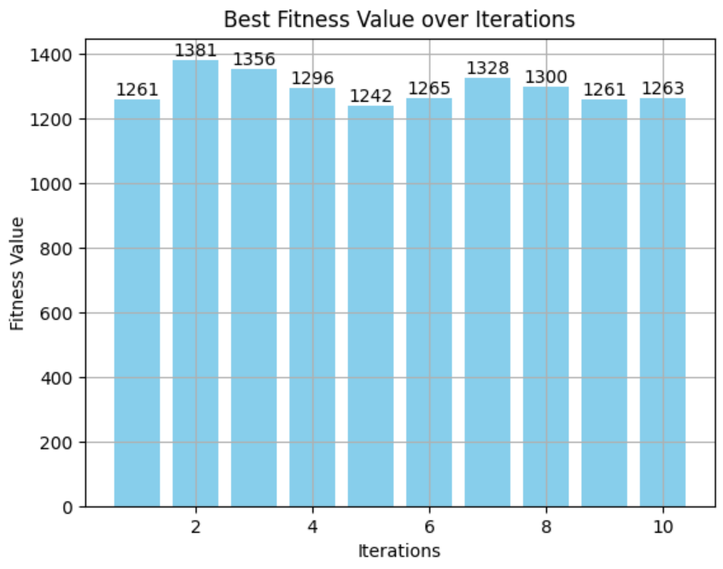

Figure 2: Job-shop Scheduling Problem: Combination 1

The ratio between population size and offsprings was tinkered with as well as the value of mutation rate. Other combinations of schemes such as rank-based selection as parent

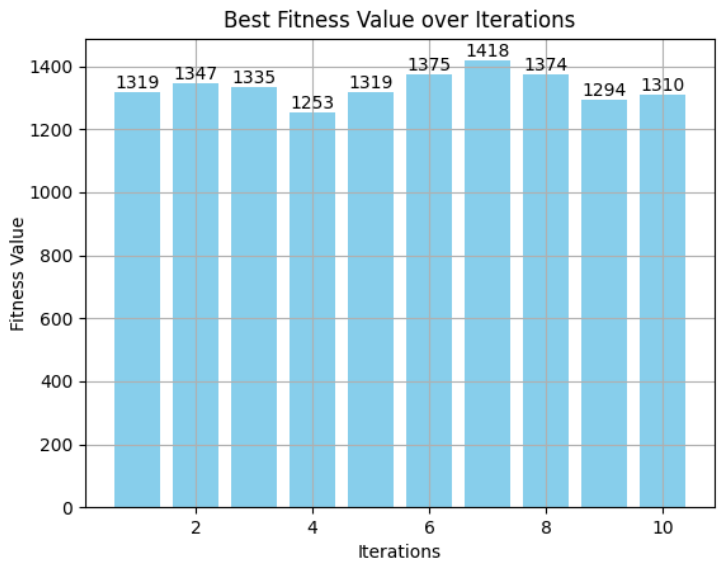

Figure 3: Job-shop Scheduling Problem: Combination 2

selection and truncation as survival selection yielded the following result, **1253** which came very close to our best score.

The following parameters were used to achieve the result in *Figure 3*:

1. **Population Size:** 150
2. **Offspring Size:** 40
3. **Number of Generations:** 2000
4. **Mutation Rate:** 0.5
5. **Iterations:** 10

3.2.2 Input files “abz6” and “abz7”

The combination of schemes for both the input files are random and truncation with the following parameters:

1. **Population Size:** 1000
2. **Offspring Size:** 200
3. **Number of Generations:** 500
4. **Mutation Rate:** 0.45
5. **Iterations:** 10

The best score for the input file “abz6” was **972**. Whereas, the best score for the input file “abz7” was **776**. The reason for the scores of input file “abz6” and “abd7” being lower than the input “abz5” is due to the fact that the time needed by each process on each machine is significantly low.

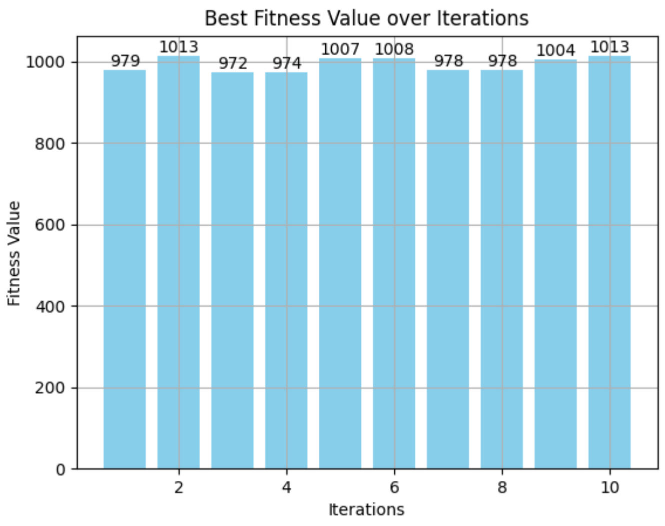

Figure 4: Job-shop Scheduling Problem: Input file abz6

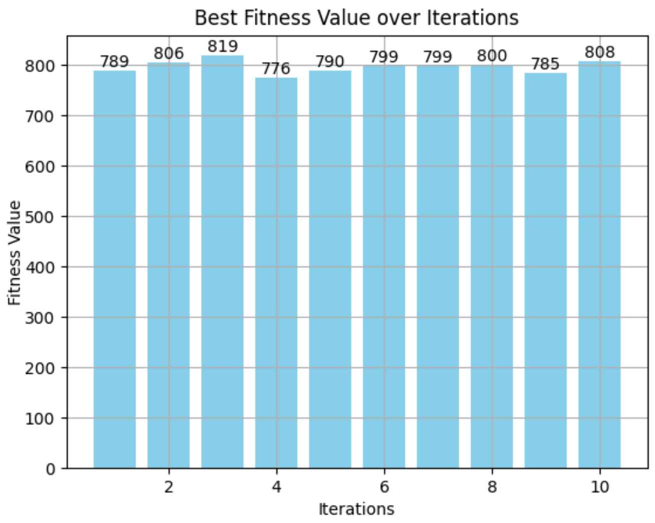

Figure 5: Job-shop Scheduling Problem: Input file abz7

4 Evolutionary Art: Mona Lisa

4.1 Problem Formulation

4.2 Analysis