Project 5 IT3708:

Solving Multi-Objective Traveling Salesman Problem (MTSP) using Multi-Objective Evolutionary Algorithm

Purpose: Learn to implement and study the application of multi-objective evolutionary algorithm (MOEA) in combinatorial optimization problem. Another purpose of this project is to study the effects of various parameters of MOEA while solving any problem.

Groups allowed? For this project you may work alone or in groups of two.

1 Assignment

You will use Non-Dominated Sorting Genetic Algorithm-II (NSGA-II), a famous multi-objective evolutionary algorithm, to solve Traveling Salesman Problem (TSP). The TSP is a very simple problem to formulate but has been notoriously difficult to solve (the problem is NP-hard). The optimization problem that you will handle in this project is multi-objective TSP for which the aim is to obtain or to approximate the set of trade-off (Pareto-optimal) solutions by simultaneously optimizing multiple objectives. For experiment, a data set for 48 (forty eight) cities is attached with this project description on It's Learning.

You already know that the performance of EA largely depends on the proper selection of its parameters values; including population size, generation number, crossover and/or mutation rate, crossover and/or mutation mechanism, selection strategy and the like. In this project, based on NSGA-II, you will also investigate the effects of these tuning parameters on the performance of MOEA.

2 Multi-Objective Optimization Problem (MOOP)

In the world around us, very few problems are concerned with a single objective. Instead, many real-world problems involve multiple, often conflicting, objectives and constraints. The optimization of such problems requires reaching a compromise between multiple objectives while not violating any of the imposed constraints. A general MOOP is mathematically defined as:

$$\begin{array}{lll} \text{Optimize} & \mathbf{F}(x) = [f_1(x), f_2(x), \ldots, f_k(x)] & \\ \text{Subject to} & g_i(x) \leq 0, & j = 1, 2, \ldots, m \\ & h_l(x) = 0, & l = 1, 2, \ldots, e \\ & x_i^{(L)} \leq x_i \leq, x_i^{(U)}, & i = 1, 2, \ldots, n \end{array} \tag{1}$$

Where k is the number of objective functions, m is the number of inequality constraints, e is the number of equality constraints, $x = (x_1, x_2, \ldots, x_n)$ are the n optimization variables, L and U are the upper and lower bounds of optimization variables.

For MOOPs, the conflicts have to be met or optimized before any adequate solution is reached. While in single-objective optimization problems there is typically a single solution that gives the best objective value, in MOOPs there is no single optimal solution; there are, instead, a set of alternative solutions. This is because a solution that is optimal with respect to one objective requires a compromise in other objectives. Also, improvement of one objective may lead to deterioration of another. Thus, when optimizing multiple objectives, a single "absolute optimum" solution that can optimize all objectives simultaneously does not necessarily exist. In fact, in such real-world scenarios, several relevant trade-off solutions exist.

3 Pareto–Optimal Solutions

Multiple objectives are often aggregated into one overall objective function, and optimization is then performed on this single objective. In such method, the basic difficulty arises due to the strong dependency of the outcome on how the objectives are aggregated using weights. Also, the weight itself reflects the relative importance among the objective functions. Since the relative magnitudes of the weights may not be exactly known or pre-determined by the users in advance, the objective function that has the largest variance value may dominate.

The concept of Pareto-optimality is introduced to avoid these difficulties and in order to explore a broader set of trade-off solutions. Pareto-optimality is defined as the set in which every element is a solution for which no other solution is better in all design attributes. The term *domination* is used to define the optimality in MOOPs. The domination between two solutions is defined as follows:

Definition 1 A solution $\mathbf{x}^{(1)}$ is said to dominate another solution $\mathbf{x}^{(2)}$; if both the following conditions are true:

(i) The solution $\mathbf{x}^{(1)}$ is no worse than $\mathbf{x}^{(2)}$ in all objectives.

(ii) The solution $\mathbf{x}^{(1)}$ is strictly better than $\mathbf{x}^{(2)}$ in at least one objective.

For a given finite set of solutions (S), we need to perform pair-wise comparisons using the above definition to determine whether or not one solution dominates another. From these comparisons, we can find a subset of solutions (T) for which no member of T is dominated by any member of S, and all the other members of S are dominated by one or more members of T. This subset (T) is called the *Pareto-optimal set* or the *Pareto-front* for the given set of solutions. Pareto-optimal solutions are also called efficient, non-dominated, or non-inferior solutions. This can be illustrated by a set of nine solutions as presented in Fig. 1.

The figure considers two objectives that are subject to minimization and maximization, respectively. After performing a pair-wise comparison using Def. 1, we find that the solutions 1, 2, 3, and 6 (dark colored) are non-dominated. The other solutions (red colored) are dominated by one or more members of this set of non-dominated solutions. For the set of nine solutions shown in the figure, the plot of these non-dominated solutions (1, 2, 3, 6) constitutes the Pareto-front.

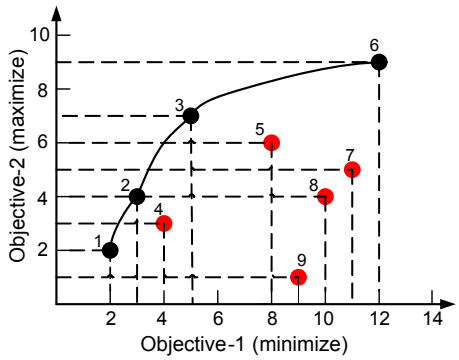

Figure 1: Pareto-front for a two objective optimization problem.

4 Multi-Objective Traveling Salesman Problem (MTSP)

The optimization problem for this project is multi-objective traveling salesman problem (MTSP). TSP is a challenging problem in combinatorial optimization. In the well-known single-objective version of this problem, a traveling salesman has to visit a set of cities without passing more than once through each city and return to the starting one. The goal is to find a route such that the total distance traveled is minimized. The problem is very simple for small numbers of cities, as one simply has to enumerate all the possible tours and choose the best one. However, as the number of cities (n) increases, the total number of possible tours is $(n-1)!$, which is very large. Fig. 2 presents a solution for a TSP consisting of 45 German cities. It is interesting mentioning that this the route shown in the figure is the one of the 2.658271574 $E+54$ possible ones !!!

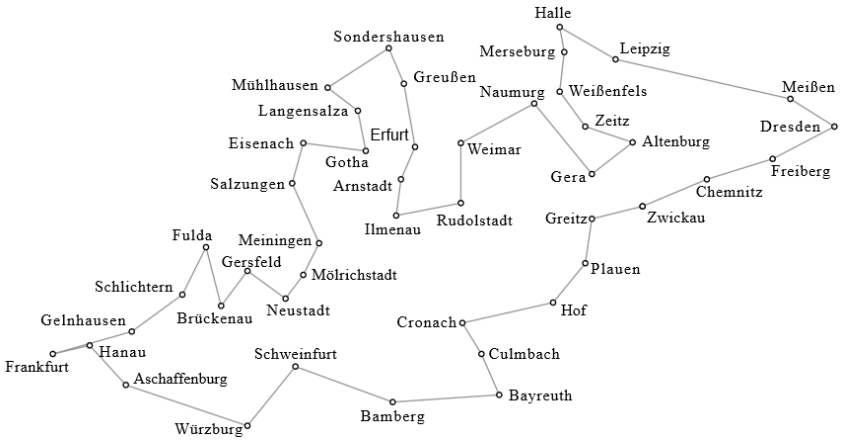

Figure 2: A solution to TSP for 45 German cities.

In MTSP, the traveling salesman not only has to minimize the total distance but also optimize other objectives, such as the overall traveling time, total cost, and so forth. Therefore, it is assumed that several quantities, such as distance, time and cost, are assigned to the connection between each pair of cities. Therefore, similar to MOOPs, for MTSP there exist no single "absolute optimum" tour, rather a set of trade-off routes optimizing multiple objectives.

5 Things To Do

For this project, you will solve the attached MTSP consisting of 48 cities to find a set of trade-off alternatives (routes) by optimizing the order of the cities so as to simultaneously minimize the two objectives of (i) traveling distance and (ii) traveling cost incurred by the traveling salesman.

For your experiment, two matrices are given on It's Learning. One is the geographic distances between every pair of cities. And, the other is the traveling costs between every pair of cities.

Note that, for implementing evolutionary operators (crossover, mutation, selection mechanism, elitism strategy and such others), you need to write the code from the scratch. **You are not allowed to use any existing library function**. You are free to design the genotype structure and the corresponding evolutionary operator types.

The tuning parameters such as population size, number of generations, mutation and crossover rates, etc. will also need to be determined experimentally: try combinations until you find one that gives good results. To investigate the influence of parameters, you need to report the best three combinations of the following parameters:

(population size, number of generations, crossover rate, mutation rate)

Also, you need to compare their performance and plot the objective values of every member of the final set of non-dominant (Pareto-optimal) solutions, i.e. the Pareto front, achieved by your MOEA using these best three combinations of parameters. The exact reporting structure is mentioned below.

5.1 The MOEA

To implement your MOEA, the first thing you must do is to design the genotype. Every single genotype will represent a complete tour for the given MTSP. Therefore, the population pool will consist of as many different alternative tours as the number of genotypes in the population. Another key issue is to design the corresponding evolutionary operators (crossover and mutation). Several alternatives for designing genotypes, crossover and mutation in the case of TSP are available on the Web, so you can easily find some inspiration there (but you need to implement everything yourself!). **Note**: you need to be careful when designing the genotype structure and evolutionary operators, because it will influence the performance of your MOEA.

Since you will implement the TSP using NSGA-II, all of your evolutionary runs should include **elitism** in the form of *crowding tournament selection operator* and *crowding distance*. More information can be found on It's Learning, in the slides from lecture 9 ("Multi-Objective Evolutionary Algorithms"). The slides related to NSGA-II begin on page 62.

Your genotypes, evolutionary operators, adult selection mechanisms and parent selection mechanisms may all need to be changed from the previous EA projects in this course, in order to implement TSP using NSGA-II. You might therefore find it easier to start working on this project with a new code base instead of reusing your previous EA code.

5.2 Fitness Functions

As mentioned earlier, you will simultaneously optimize two objectives for the MTSP:

- **Fitness function 1:** traveling distance for the complete tour of each genotype.
- **Fitness function 2:** total traveling cost incurred by the traveling salesman for the complete tour of each genotype.

Both of these objectives are subject to minimization.

5.3 Visualization

At the end of a complete evolutionary run (i.e. after the final generation, when using a given combination of tuning parameters), your system must be able to generate two types of plots as follows:

(i) The values of objectives for each member of the final population, where fitness function-1 will be on the x-axis and fitness function-2 will be on the y-axis. For each fitness function/axis, there should be clear markers for the best and the worst values in the final population.

(ii) The values of objectives for each of the final non-dominated (trade-off) solutions only, where fitness function-1 will be on the x-axis and fitness function-2 will be on the y-axis. This is the **Pareto-front**. Again, there should be clear markers for the best and the worst values of each fitness function in the final Pareto-front.

These two visualizations should be produced as graphical 2D plots by your system, not as e.g. command-line printouts of an array. See Fig. 1 for an indication of what type of plots are expected (although without the dashed lines and the line connecting the Pareto front).

6 Report

You should write a report answering the points below. The report can give a maximum score of 13 points. Your report must not exceed 4 pages in total. Over-length reports will result in points being deducted from your final score. Print on both sides of the sheets, preferably. Bring a hard copy of your report to the demo session

a) Document your implementation (3p)

- A detailed description of the complete MOEA design. This includes the genotype representation, implementation of crossover and mutation operators, and your selection strategy.
- Describe whether or not your chosen crossover and mutation operators might produce infeasible off-spring(s) after executing (e.g. invalid TSP solutions). If yes, how did you handle that? If not, why?

b) Document your EA choices. (3p)

- A table summarizing values of the best three combinations of the following parameters:

  (population size, number of generations, crossover rate, mutation rate)

- For each combination, the table should also contain the best and worst values found for each objectives.
- Also, you need to report the total number of non-dominated solutions found in the Pareto-front for each of the three parameter combinations.

c) Plotting of final solutions (7p)

You must include the following 2D plots in your report (described more in detail in Section 5.3):

- For *each* of your three chosen parameter combinations, a plot showing values of both objectives for every member of the final population, as described under point (i) in Section 5.3. There should be clear markers for the best and the worst values of each fitness function.
- For *each* of your three chosen parameter combinations, a Pareto-front showing values of both objectives for each of the final non-dominated solutions only, as described under point (ii) in Section 5.3. Again, there should be clear markers for the best and the worst values of each fitness function.
- One plot which will compare all the three Pareto-fronts mentioned just above. You must use different colors and types of marker for each Pareto-front.

In accordance with this list you should end up with 7 different plots in total.

7 Demo

There will be a demo session where you will show us the running code and we will verify that it works. This demonstration can give a maximum of 7 points.

For this demo, you will use your MOEA to evolve the given MTSP with our supplied set of parameters. Your MOEA should be able to produce near-optimal solutions in reasonable time during the demo.

8 Delivery

You should deliver your report + a zip file of your code on It's Learning. The deadline is given on the assignment on It's Learning. The 20 points total for this project is 20 of the 100 points available for this class.

For this project you can work alone or in groups of two. If you work in a group, you only need to deliver once on It's Learning (but both group members must the registered as part of the submission on It's Learning!). Both group members need to attend the demo session.