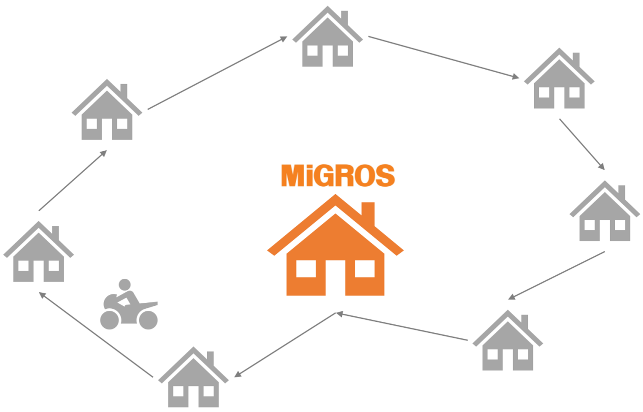
# Assignment 3
## Migros Delivery using Ant Colony Optimization

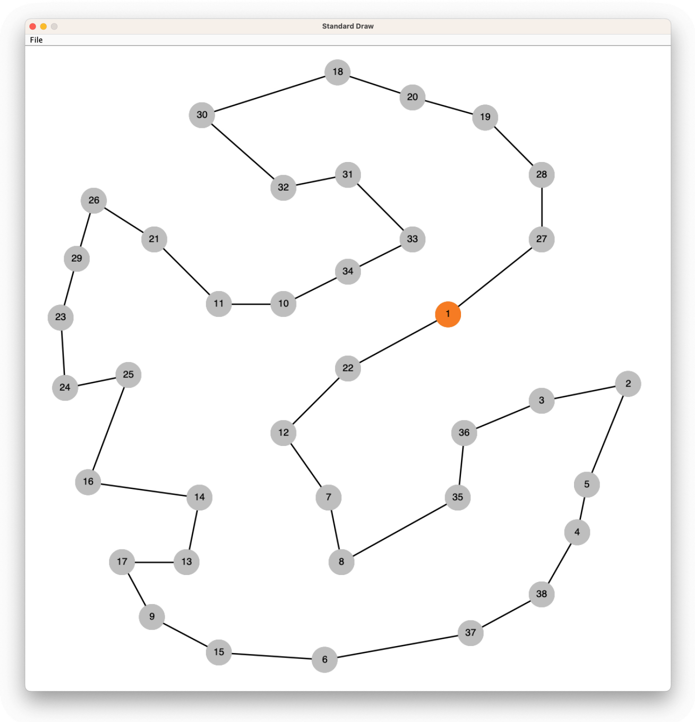
CMPE 160

Contact: Suzan Ece Ada (ece.ada@bogazici.edu.tr)

Due:
May 13th, 6 a.m. in the morning




In this assignment, you will write a Java program which finds the quickest delivery route of a Migros delivery car. In the map (See Figure 1a), there is a Migros store, denoted by the orange circle, and several houses, denoted by the gray circles. Each house places an order from Migros. The delivery car should visit each house only once and return to Migros. Coordinates of Migros and houses are given in an input text file as shown in Figure 1b where the first line denotes the Migros coordinate, and the rest denote house coordinates. During the delivery, each house should be visited at most once. Your program should find the shortest route and plot the final route as shown in Figure 1a.




```
0.6550,0.5838
0.9338,0.4762
0.8000,0.4500
0.8550,0.2462
0.8700,0.3200
0.4638,0.0487
0.4700,0.3000
0.4900,0.2000
// ...
// some house coordinates are omitted here.
// ...
0.8000,0.1500
```

(a) (b)

Figure 1. (a) Migros (orange) and houses (gray) and (b) sample format of the input file.

Your program should output the following details to the console. (Values are arbitrary and may not be correct)

```
Method: Brute-Force Method
Shortest Distance: 5.01263
Shortest Path: [1, 22, 12, 7, 8, 35, 36, 3, 2, 5, 4, 38, . . ., 1]
Time it takes to find the shortest path: 347.28 seconds.
```

You should try two solution methods and compare them:

1. Brute-Force Approach: Generate all possible permutations and try all of them. Pick the one which produces the shortest path. This method is very slow but finds the optimal solution.
2. Ant Colony Optimization Approach: This optimization algorithm may not find the best solution, but it is considerably faster than the brute-force approach. Usually, ant colony optimization heuristic finds a very close solution and may be preferred in some situations.

In your main file, define a variable *chosenMethod* which is set to 1 to run the brute-force method and 2 to run the ant colony optimization method so that we can be able to run the selected method by adjusting the *chosenMethod* variable.

Explanation of the Ant Colony Optimization Approach: In ant colony method, randomly walking ants are simulated. Each ant randomly traverses nodes and completes a cycle/path by visiting each node once. Each ant leaves a trail of pheromones. The amount of pheromone left from an ant is inversely proportional to the total cycle distance. For each ant traversal, pheromone intensity of an edge in the cycle is updated by the amount *delta=Q/totalCycleDistance* where Q is a constant value. totalCycleDistance is the sum of all distances in the traversed cycle for that ant. Update formula is: *newPheromoneIntensity* = *oldPheromoneIntensity* + *delta*. Initially, all edges have the same initial pheromone intensity.

Ants move from one node to the other randomly. However, they choose the next node to visit probabilistically. Ants prefer to choose an edge with high pheromone intensity value and shorter length. Specifically, for an edge going from node I to node j, the following edge value is computed:

$$edgeValue_{ij} = \frac{P_{ij}^{alpha}}{D_{ij}^{beta}}$$

where $P_{ij}$ is the pheromone intensity of edge i to j and $D_{ij}$ is the edge distance. *alpha* and *beta* parameters give priorities to these quantities. Ants prefer to choose an edge (probabilistically) with a higher edgeValue. To compute the next edge to traverse, normalize the edge values for the node i by dividing them to the sum of all edge values going out from node i so that they represent probabilities. Figure 2 illustrates the procedure of converting edge values to probabilities. After computing the probabilities for all possible edges going out from node i, an ant probabilistically chooses the next edge to traverse. For example, in Figure 2, an ant may choose the edge i->m with a chance of 0.32 (out of 1). Ants cannot visit previously visited nodes.

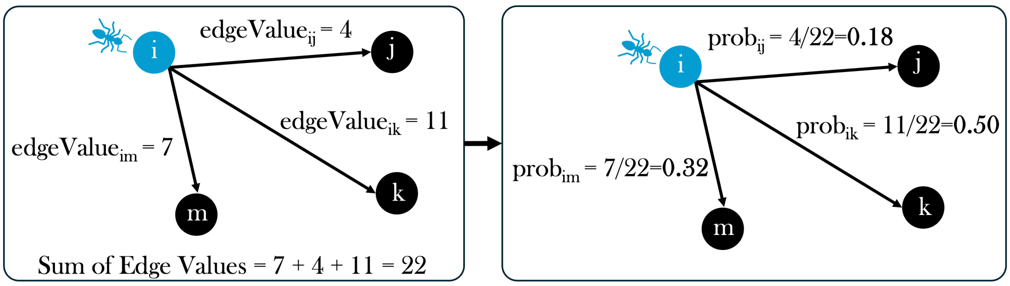


Figure 2. Illustration of computing probabilities using edge values.

Ant colony optimization method performs N iterations. At each iteration, M ants are generated. Each ant starts its traversal from a random starting node and completes its cycle by returning to its starting node. After a cycle is completed, each ant updates edge pheromone intensity values using delta values. Additionally, at the end of each iteration, pheromone levels are gradually decreased using the degradation constant, i.e., NewP$_{ij}$ = OldP$_{ij}$ x degradationConstant. Degradation procedure simulates the evaporation of past pheromones.

At each iteration, keep the best ant traversal. After N iterations, the overall best ant cycle can be used to determine the final answer. Re-order the visited nodes of the best cycle so that the starting node is always the first node given in the input file (Migros). For the ant colony method, you should plot the pheromone intensity graphs after executing N iterations, as shown in Figure 3 in addition to the best (shortest) path plot (see Figure 1a).

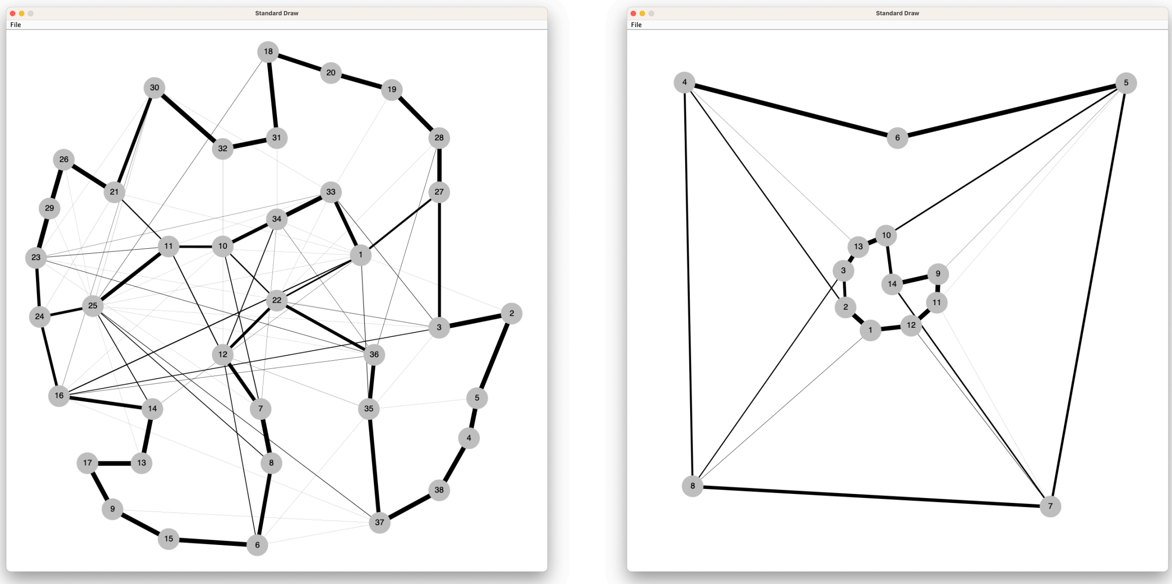



Figure 3. Pheromone intensities after 40 iterations for 50 random ants. Thicker lines denote higher pheromone levels.

Feel free to adjust/tune the hyperparameters of the ant colony optimization approach. You may start from the following suggestions and change them to find a better (faster and closer to optimal path distance) solution:
Iteration count (N) = 100, Ant count per iteration (M) = 50, Degradation factor = 0.9, Alpha = 0.8, Beta: 1.5, Initial pheromone intensity = 0.1, and Q value = 0.0001. Define all of these variable at the beginning of your main method.

In your report,

1. Provide outputs (graphical and console outputs) as shown in this document for each input file. For the ant colony method, provide both shortest route graphics and pheromone intensity graphics for each input file. All of these plots should be generated using the StdDraw graphics library. Provide your best ant colony hyperparameters (maximum iteration count, ant count per iteration, degradation factor, alpha, beta, initial pheromone intensity, and Q value etc.)
2. Compare the running times and path distances of the two methods in a table for each input file. See Table 1 as an example. In the table, provide the running times of each method in seconds, total shortest path distances, and the speed-up factor of using ant colony optimization method compared to the brute-force approach.

Table 1. Comparison of methods. Values are arbitrary and may not be correct.

| **Input File** | **Number of Houses + Migros** | **Brute-Force Time (seconds)** | **Ant Colony Time (seconds)** | **Speed Up Factor** |
|---|---|---|---|---|
| Input1 | 11 | 4053.59<br>(Distance: 4.901) | 65.56<br>(Distance:5.0133) | 62 times faster |
| Input2 | 13 | 706455.98<br>(Distance: 6.7889) | 103.55<br>(Distance: 6.904) | 6868 times faster |
| // … | // … | // … | // … | // … |

3. Plot the best distance vs. iteration graph as shown in Figure 4 for the input file with the largest node count (for the ant colony method). You can use Excel or Google Sheets to plot this graph.

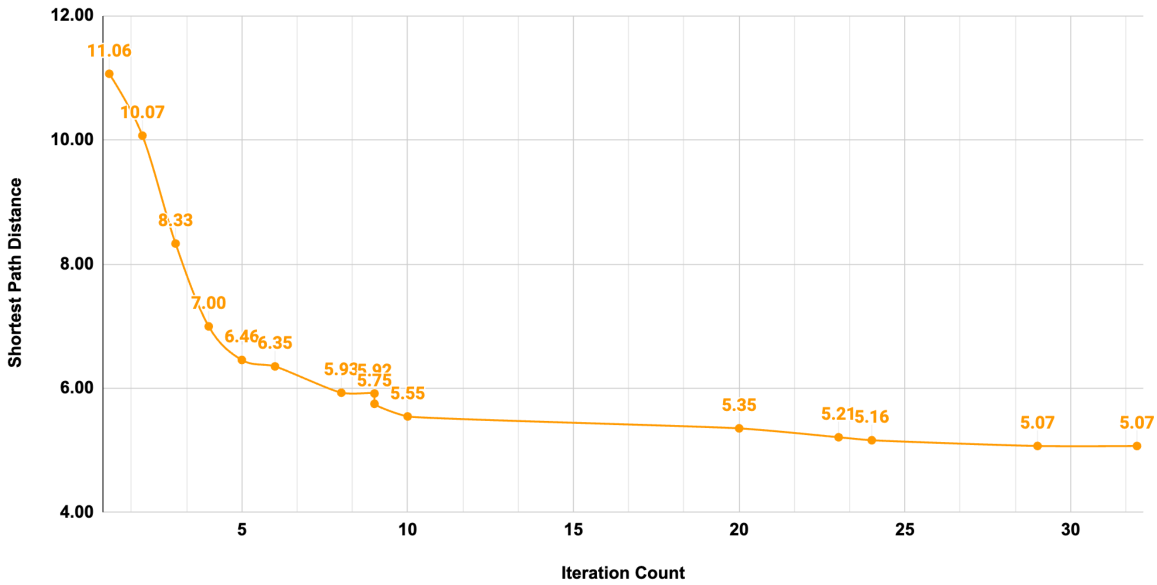



Figure 4. Shortest path distances found at ant colony method iterations.

4. Provide references to external resources if you use them to understand the ant colony optimization method.
5. Discuss (by providing a list of) advantages and disadvantages of the ant colony optimization method.

## Submission Guide

### Submission Files

Submit a single compressed (.zip) file, named as name_surname.zip, to the Moodle. It should contain all source code files (under the \code directory), report (in PDF format, under the \report directory) and all other files if needed (under \misc directory)

### File Naming

Name the main code as "NameSurname.java". Name your report as "NameSurname.pdf".

### Late Submission Policy

Maximum delay is two days. Late submission will be graded on a scale of 50% of the original grade.

### Mandatory Submission

Submission of assignments is mandatory. If you do not submit an assignment, you will fail the course.

### Plagiarism

Leads to grade F and YÖK regulations will be applied.