TP3: Ant Colony Optimization

Goals

The objective of this sheet is to:

- Develop the understanding of ant colony optimization heuristics through implementation.

Some other information:

- The coding language for this course is Python.
- Always write tests for each exercise to assure yourselves that your code is correct.
- You can discuss in groups, but the code and tests should be written individually.
- A class to represent a graph in Python is provided to you in the file `graph.py`, contained in the zip archive `bioalg-tp03-code.zip`.

1 The Shortest Path Problem

The Shortest Path Problem involves finding the minimum cost path between two vertices in a weighted graph. The input is a graph $G = (V, E)$ with edge weights $w\colon E \to \mathbb{R}_{\geq 0}$, as well as two vertices $s, t \in V$. The goal of the problem is to find a shortest path from s to t.

Exercise 1.1. Implement the ant system algorithm to solve the shortest path problem. You can use $\alpha = 1$, $\beta = 1$ for now. Test it using the graph files provided.

Note: You can test your algorithm using the `graph-metro.txt` file, by verifying that the shortest path from *Louise Michel* (171) to *Liberté* (167) has length 1340. You can also check the other tests available in the file `test_graphs.py`.

Exercise 1.2. Test the influence of the parameters α, β and of the natural interest function.

To get you started, you can compare the initial setting $(\alpha, \beta) = (1, 1)$ to $(\alpha, \beta) = (3, 1)$ and $(\alpha, \beta) = (1, 3)$.

Exercise 1.3. Compare the best version you found to Dijkstra's algorithm or the A* algorithm. If you have not implemented them for TDm01, a version of Dijkstra's algorithm is provided in the file `graphs.py`.

2 Bin Packing Problem

In the bin packing problem, we are given a set of n items with weights $w_1, \ldots, w_n$, as well as a value C, called capacity. Our goal is to pack all of the items into as few boxes as possible, while respecting their capacity. In other words, we assign each item to a bin, such that the total weight of items assigned to a bin does not exceed C, and we want to minimize the total number of bins used.

Exercise 2.1. Implement the ant colony optimization variants described in class. Test your algorithm using handmade examples.

Exercise 2.2. Implement a function that generates a bin packing instance, according to one of three methods.

The function takes as input an integer n and a method number m, and outputs a list of n weights and a capacity value, as follows:

- If $m = 1$, output an instance with uniform weights (every item has weight 1);
- If $m = 2$, output an instance with random weights between 1 and the capacity C;
- If $m = 3$, output an instance with large weights, close to the capacity value.

Exercise 2.3 (optional). Compare your algorithm against simple bin packing heuristics, such as First-Fit or Best-Fit.

Exercise 2.4 (optional). Create functions to visualize a solution to the problem, as well as to plot the quality of the solution over time.

3 Vehicle Routing Problem (optional)

The Vehicle Routing Problem (VRP) involves finding optimal routes for a fleet of vehicles to deliver goods to customers. In it, we are given a weighted graph $G = (V, E)$ with edge weights $w\colon E \to \mathbb{R}_{\geq 0}$, as well as a depot v_o, a set X of customers (vertices) to visit, and a capacity q. The goal of the problem is to visit every customer with a shortest-possible route, with the restriction that the route can visit at most q customers before having to return to the depot. In other words, the solution is a union of cycles all containing the depot, such that each customer is visited in some cycle and each cycle visits at most q customers, and we want to minimize the total weight of the solution.

We will assume that the graph is complete and that the edge weights respect the triangle inequalty, that is, $w_{uv} \leq w_{ux} + w_{xv}$, for any $u, x, v \in V$.

Exercise 3.1. Implement the ant colony optimization variants described in class. Test your algorithm using the graphs provided, by specifying some vertices as customers.

Exercise 3.2. Implement a mechanism that optimizes the routes before updating pheromones, by applying 2-opt local search to the route of each ant.

Exercise 3.3. Analyse the solutions obtained and implement mechanisms to further improve the solution, in particular if your routes contain cycles that are too short or too long.