Practice 4 – Backtracking y Branch-and-Bound. A very efficient delivery man (Reduced version)

Data Structures and Algorithms II. University of Almeria

Version **curso.2022** - reduced

Goals

- Construct solutions to a problem using the ***backtracking*** **and** ***branch-and-bound*** **algorithmic methods**. Analyze and compare the implemented algorithms from diferent perspectives.
- Carry out an **analysis of the efficiency** of the solutions provided, and a comparison both from a theoretical and practical point of view.

Requirements

To pass this practice, the following must be done:

- Remember (from EDA I) the proper **data structures** for problem solving.
- Know how to implement the ***backtracking*** **and** ***branch-and-bound*** **programming schemes** to respond to a specific need (in this case, solving the Traveling Salesman Problem problem, TSP, and Hamiltonian cycles problem).
- **Evaluate the implemented** ***backtracking*** **and** ***branch-and-bound*** **programming algorithms**, relating both algoritmic methods.

Problem statement

As we know, in EDAland there are very varied characters, among them, we find ‘Braulio the delivery man’, a tireless and responsible worker from Almeria who has spent a lifetime distributing Hoy-Almeria newspapers throughout the country. Every morning, when the publisher of Hoy-Almeria closes the daily edition and the first copies come out of the printing press, Braulio is there, with his van, ready to deliver them to all the cities of EDAland. In addition to his efficiency in distributing, Braulio also stands out for being a person who takes his economy very much into account, and always tries to choose the circuit that involves him traveling the shortest possible distance or spending the least amount of fuel (and even more so now, when high price that is) possible. To be so effective, Braulio raised three important

questions. The first consisted of knowing all the possible paths, starting from Almeria, which go through each city where a batch of newspapers must be left, exactly once and later returning to Almeria. The second question was to determine a circuit that would cover each city exactly once, starting from and returning to Almeria, and having covered the shortest possible distance in total. The third and last question was to determine a circuit that runs through each city exactly once, leaving and returning to Almeria and whose fuel consumption is the minimum possible.

In order to respond to the questions raised by Braulio there is the **new reduced network of roads** of EDAland (Figure 1). In EDAland a new general road plan was carried out and some new roads (in blue) were added to the previous network (practice 2). As we can see, it connects the large cities of the country (21 cities with 41 roads), along with the distance in kilometers between two cities. For example, the distance that connects Almeria with Granada is 173 kilometers. This network will help us to verify the correct operation of the *backtracking* and *branch-and-bound* algorithms developed.

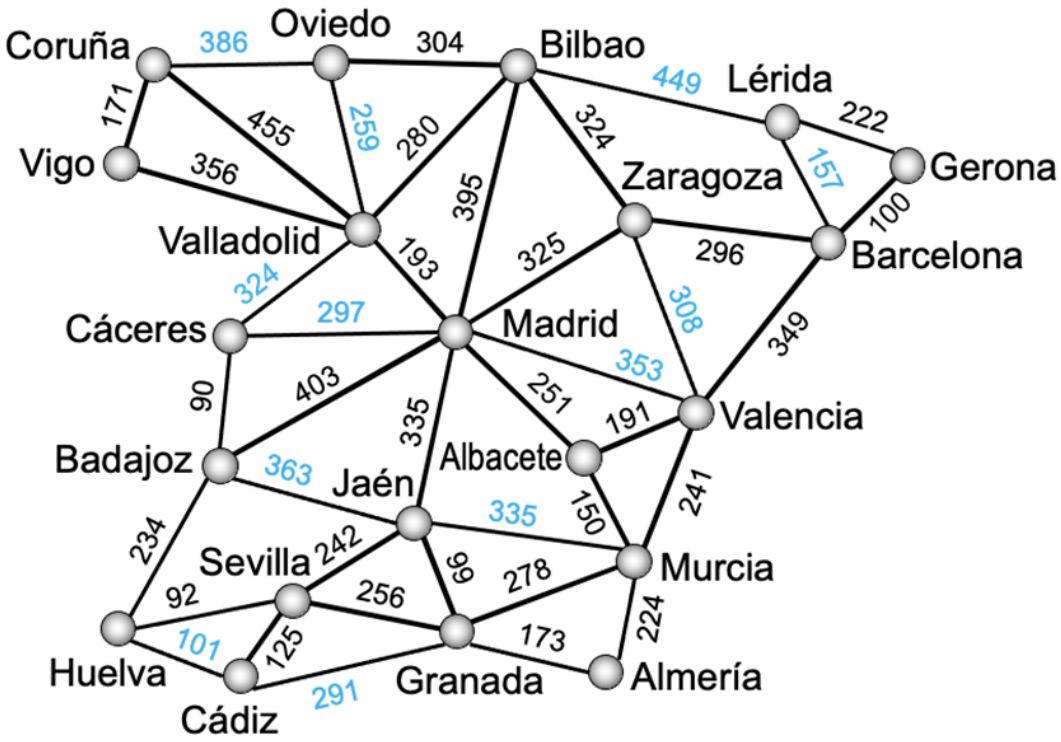

Figure 1. New reduced network of roads of EDAland

Once you check the correct operation of our algorithms in the new reduced network of roads, we will execute them in the **national network of roads** of EDALand (Figura 2, adapted from cities and roads graph published in [1]), that connects all population centers in the country (1053 population centers and 2017 roads with the distance between every two of them). That is the same network from practice 2.

Figure 2. EDAland National Road Network

Work to develop

You must propose and implement algorithms solutions using ***backtracking*** and ***branch-and-bound***, as required, to the following problems:

- (***Backtracking***) Determine all the possible paths, if there were more than one, starting from Almeria, that go through each city of the new reduced road network of EDAland, where Braulio must leave a batch of newspapers, exactly once and return to Almeria. If there is more than one path, determine the path with the shortest distance.

  For this first case (***Backtracking***), implement the algorithm *The traveling salesman problem* of the theory class transparencies, recomposing it so that it works (since the transparencies one may have some error), defining the variables in such a way that accommodate recursion. Also make a more or less adjusted calculation of the stack needs that you will need. In case of problems with the EDA I graph, use an NxN matrix.

- (***Branch-and-Bound***) Determine a circuit that, starting from Almería, visits each city exactly once, returning to Almería and having covered the shortest possible distance in total. Solve this problem for the new reduced road network of EDAland.
- (*Optative*) Using the algorithms (***Backtracking*** and ***Branch-and-Bound***) implemented in previous sections, you must try to obtain the path in the complete national road network, starting from any urban nucleus, visiting each town exactly once, returning to the starting nucleus and having the shortest possible distance. To verify that the algorithms work on said on the complete national road network, add an *execution trace* on them in showing its

status depending of the iteration or every certain time. What conclusions do you get from the attempt? Is there any way to solve the problem that has been raised? Indicate everything in a reasonable way.

To do this, you must complete the following sections:

- **Study of the implementation**: Explain the most important details of the implementation, both of the data structures used to solve the specific problem, and of the implemented algorithms. The code must be reasonably well documented (`JavaDoc`).
- **Experimental study**: Validation of the ***backtracking*** and ***branch-and-bound*** algorithms implemented on the EDAland networks provided. To do this, the execution times of the implemented algorithms must be obtained and compared.

Submissions

A private GitHub repository (same repository for all EDA II practices) with all the documentation and source code required in the practice must be submitted on date:

- In that repository, a new folder `practica_4`, with at least two subfolders, one for the documentation, `docs`, and one for the source code, `src`, including the JavaDoc and class diagrams generated automatically from the source code, folowing the same project structure explained in previous practice.
- A simplified memory document that explains everything you have done in practice. The memory must have the format indicated below.
- Source code of the application, developed in JAVA, which solves everything raised in practice. Remember that you will have to measure execution times of your solutions, so you must include the necessary commands for this in the source code. Likewise, if you run your algorithms on the large EDAland network, don't forget to show the trace of their execution in which you show their status depending on the iteration or every so often.

The **memory** of this practice to deliver must be brief, clear and well written. This should include the following sections:

- A section for each of the **proposed sections** to be developed in this practice (implementation study, and experimental study).
- An **annex** with the design of the implemented code will also be included, with a class diagrams automatically generated.

Assessment

Each section will be evaluated independently, although it is a necessary condition to pass the internship that the implemented programs work correctly.

- The implementation together with the documentation of the code will be valued out of ~~40%~~ 60%

- The study of the implementation will be valued out of ~~10%~~ 20%
- ~~The theoretical study will be valued out of 15%~~
- The experimental study will be valued out of ~~35%~~ 20%

The defense of the code and memory by the teacher may be required.

Deadline

Deadline: **June 5th 2022**

Results (*New*)

Running the algorithms on the **new reduced network of roads** of EDAland (Figure 1) should give **34 paths** (circuits). The following example shows 3 of the 34 resulting circuits. The shortest one must be highlithed at the end:

```
Almeria - Granada - Cadiz - Huelva - Sevilla - Jaen - Badajoz - Caceres - Madrid -
Valladolid - Vigo - Corunya - Oviedo - Bilbao - Lerida - Gerona - Barcelona - Zaragoza
- Valencia - Albacete - Murcia - Almeria  ==> 4999.0
Almeria - Granada - Cadiz - Huelva - Sevilla - Jaen - Badajoz - Caceres - Valladolid -
Vigo - Corunya - Oviedo - Bilbao - Lerida - Gerona - Barcelona - Valencia - Zaragoza -
Madrid - Albacete - Murcia - Almeria  ==> 5271.0
Almeria - Granada - Cadiz - Huelva - Sevilla - Jaen - Badajoz - Caceres - Valladolid -
Vigo - Corunya - Oviedo - Bilbao - Lerida - Gerona - Barcelona - Zaragoza - Madrid -
Albacete - Valencia - Murcia - Almeria  ==> 5192.0
...
...
Number of circuits: 34

Shortest circuit:
Almeria - .... ==> ...
```

References

- [1] Gines García Mateos. The Traveler's Challenge. Available online on http://dis.um.es/~ginesgm/retoviajante.html [Date of consult: 2022/03/19]