Università
della
Svizzera
italiana

Faculty
of Informatics

Student: Edoardo Riggio

AI Cup 2021

Final Report

Contents

1. Abstract

The purpose of this competition was to find the shortest possible tours given a number of Euclidean TSP problems. In order to do so, we had to propose, implement and test one or more heuristic algorithms learned during the semester.

2. Algorithms

I've decided to implement the *Best Nearest Neighbors* heuristic algorithm and the *2.5-opt* optimizer.

2.1. Best Nearest Neighbors

This algorithm is divided in two parts, the first is the loop on the n nodes of the graph, and the second is the actual computation of the path.

In the first part, every point of the graph is considered as the starting point of a new path. The path with the shortest length is the one chosen as the solution of the *Best Nearest Neighbors* algorithm. In the second part of the algorithm, the starting node is taken from the first part, and the path is actually computed. The computation starts from the starting node and continues by selecting the nodes that have the shortest Euclidean distance between one another.

Since the running time of this algorithm is relatively high for problems such as *fl1577*, I've decided to use the flag `-O3` when compiling the code. This made the code run much faster while still keeping the same results.

2.2. 2.5-opt

This algorithm is an extension of the 2-opt algorithm. In this case we consider the gain of a 2-opt swap, as well as two node shifts – which are 3-opt moves. By finding out which of these moves is the best – i.e. returns the shortest path – the swap is performed.

This algorithm is ran for every single pair of points in the graph and until no further improvement is possible.

3. Results

The combination of the two algorithms described above made me reach an average gap of 4.1% between the best possible solution and my solution.

As per requirements, the results are available in the `.xls` file given that was given to us. Alternatively, it is also available in the `README` file of this repository.

4. Technical Details

In order to write my submission for the AI cup I've used C++. In my solution I've implemented a class that reads and stores the problem, and one that stores the solvers for the problem. The actual solvers – i.e. the *Best Nearest Neighbors* algorithm and the *2.5-opt* algorithm – are implemented as functions that are passed to the solver class. A full documentation of both classes and functions can be found either as an `html` page or as a `pdf` file inside of the docs directory.

In order to run the solvers, just type:

```
make run
```

Inside of the root directory of the repository. This will compile and run the binary, making no further action necessary. The results of the solvers are displayed in a similar fashion as we saw in class during the tutorials.