UNIVERSITÀ DELLA SVIZZERA ITALIANA

AI Cup Report

Francesco Berla

December 16, 2019

1 APPROACH

1.1 FAILED TRY WITH GENETIC ALGORITHMS

The first approach was using Genetic algorithms, I tried them because they fascinated me. I implemented the program quite fast but I discovered that the crossover operation wasn't very good, it would rarely produce better children and combined with the fact that it was slow I couldn't get to a nice result in the given time. I tried to implement many different crossover operations (Ordered Crossover, PMX crossover [Ücc02], ...) but the gain was not significant. I played a lot with all the parameters and with the positioning of the local search (optimizing each child, optimizing only the best of each generation, ...) but none of my efforts came to a good result. My algorithms was always too slow to converge and my solution was bad in the time frame that was given for solving the problem. The code of my implementation of a genetic algorithm is in the file *genetic.py*.

1.1.1 ITERATIVE LOCAL SEARCH

After the failure with the Genetic algorithms I had to go back to the *Iterative local search*. My proposed solution is based on the work of Umberto Junior Mele that he has given as foundation. It's simply an Iterative Local Search using the simulated annealing as base. The randomization for escaping the local optimum is created by a random swap of two nodes. The local search algorithm is 2opt. This has been chosen in order to do more iterations as possible. Using a Lin-Kernigan, or 3opt, approach would have produced better results but in a lot more time and this could have been negative considering the 3 minutes time frame.

1.2 Improved attempt with Cython

My Iterative Local Search still performed in a unsatisfactory way, therefore I tried to improve the running time using Cython. Cython has the advantage that it works directly with Python code without big modifications and it promises significant speedups. After the initial setup (see the *setup.py* and all the files extensions changed to *.pyx*) I ran all the algorithm noticing a some speed-up but nothing completely game-breaking. One of the possible reason is the unoptimized integration with numpy arrays, another one is that I am not using Cython correctly in order to leverage its power.

2 System

The system used is Arch Linux (kernel: Linux-hardened 5.3.13.a-1), python version 3.8.0.

3 Usage

Execute
`python runniamo.py ProblemName`.
If you want to modify something, or if you are running for the first time, for compiling you have to type
`python setup.py build_ext -inplace`.

4 literature

In the folder *papers* you will find some of the publications that I used to build my work. The majority are related to genetic algorithms. In addition to those scientific papers I visited many websites where people were presenting their ideas about possible optimizations and parameters.

References

[Ücc02] Göktürk Üccoluk. "Genetic Algorithm Solution of the TSP Avoiding Special Crossover and Mutation". In: *Intelligent Automation & Soft Computing* 8.3 (Jan. 2002), pp. 265–272. DOI: 10.1080/10798587.2000.10642829. URL: https://doi.org/10.1080/10798587.2000.10642829.