Comparing various approaches to the mTSP problem

Teymur Azayev

***Abstrakt*— We take a look at the mTSP problem and compare various approaches to finding approximate solutions, including local search, evolutionary algorithms, and an memetic search.**

I. Task

The Travelling Salesman Problem (TSP) is a task where given a number of cities we are looking for the shortest path that we can take while visiting all the cities only once and ending up at the city where we started at. This task can also be formalized as finding the shortest tour in a weighted non-oriented graph.
The mTSP task is a generalization to the TSP where we have multiple agents traversing the same graph. No agent is allowed to visit a city that has been visited by another agent.

II. Introduction

The mTSP, as the vanilla TSP problem are inherently NP-hard. We therefore have to use various approximate optimization techniques which help us find an 'acceptable' solution to the problem.

III. Goals

The point of this report is to compare and benchmark local approaches with evolutionary algorithms in the mTSP problem, and to discuss how various issues have been dealt with along the way. We also hope that by comparing the various approaches we will get an insight to the nature of the problem.

IV. Implementation

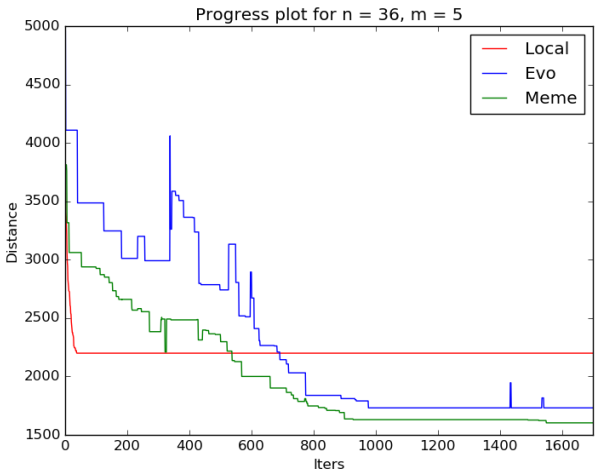
The problem is given by n cities and m agents. There is also an auxilliary city called the depot. Every agent starts from and finishes at the depot. The solution is represented by two parts: A sequence of cities and a pair of boundaries for each agent. Each agent starts at the depot, then proceeds sequentially through the solution, beginning his boundary start index and finishing at his boundary stop index. From there the agent goes back to the depot.
The quality of the solution (and the fitness function) is taken as the longest route of all agents. This not only implicitly takes into account the total length of all tours of the agents, but normalizes the tour lengths so that they all approximately cover the same distance. We generate a concrete problem by randomly generating n points within a reasonable distance from the origin. We then run the algorithm and evaluate the solution after a certain amount of iterations.
As mentioned in the abstract, we use three different approaches to finding solutions for the mTSP. The first approach is a simple local search algorithm. At every iteration we perform the following. We swap two of the cities in the sequence, evaluate the fitness and swap it back to how it was. This is done n times and the 'best swap' is recorded out of that iteration. The best swap is then performed and saved.
The second approach is a basic evolutionary algorithm on the same representation of the problem. A population of 200 is used at every iteration. The parents are selected using a standard roulette approach and two new offspring are created using a one-point crossover operator. In order to produce a valid solution every time we need to slightly modify the crossover operator. This is done by the offspring receiving part of the solution from one parent and sequentially recieving the rest of the sequence from the other parent such that the final sequence is valid. As a mutation, in each iteration we allow the boundaries of each agent to change with chance, meaning that each agent can have fewer or more cities in their path. The new offspring are accepted if they are better than their parents and accepted with a small chance if they are worse.
The last approach, the memetic algorithm simply extends the evolutionary algorithm by a local search at each iteration. This is done by using 2-opt on every new offspring after it is created. This is a method to let weak individuals locally improve and give them a chance to be selected in the next evaluation.

V. Experiments

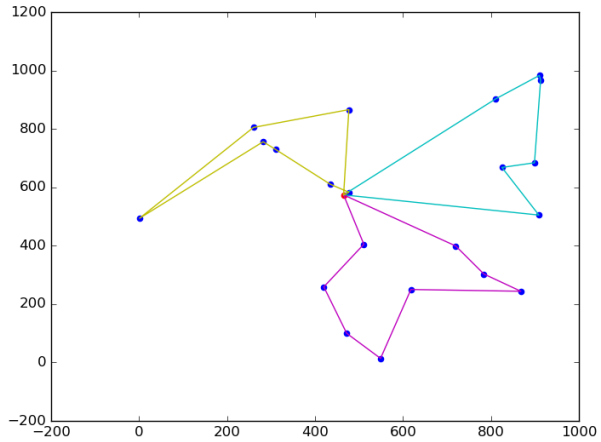
The initial experiments have been done on randomly generated graph to validate and compare the three algorithms. To make a fair comparison, however, we need to create a few benchmark instances and evaluate the algorithms on those instances. We use a total of 10 instances to evaluate each algorithm on and take the average score which we will then compare. In figure 1 we can see an iteration plot of all three algorithms to get a visual understanding and comparison of the progress of each algorithm. We can see that the local algorithm converges to a local solution almost immediately and does not continue to improve. Due to implementation reasons we do not consider execution times of each algorithm, but rather the iterations that they takes.

VI. Discussion

As expected, the local search converged immediately and did not improve further. The standard evolutionary algorithm took much longer to train but got decent results. The memetic



Obr. 1. Progress plot of each algorithm



Obr. 2. Visual solutions of n = 22, m = 3

algorithm got on average the best results. This was expected as the evolutionary algorithm sometimes has problems with small local changes. This is mediated by the 2-opt which is added in the memetic algorithm to let weaker individuals improve themselves. Results could most likely be further improved by tuning the evolutionary algorithm hyperparameters such as population size, mutation rate, etc.

VII. Conclusion

We have seen that a simple evolutionary algorithm can give us reasonable approximate solutions to an np-hard problem such as the mTSP. This approach is credited because it's relatively simple to implement. For a large amount of cities, however, this algorithm has trouble working out of the box and can be very slow so it requires an efficient implementation and more hyperparameter tuning.

TABULKA I

Average scores of the three algorithms

	Local	Evolution	Memetic
Distance of largest tour	2281	1825	1718