DeepSimplex for Travelling Salesman Problem

Numerical Linear Algebra

RoboRangers team

Introduction

- Linear Programs (LPs) - fundamental class of optimization problems.
- A popular method to solve LPs is the Simplex method.
- Pivoting rules play an important role.
- Implementation of reinforcement learning techniques can be useful.

LP general formulation

Find a vector x that minimizes $c^T x$, subject to $Ax = b$ and $x \geq 0$, where:

- $c \in R^n$;
- $b \ \in R^m$
- $A \ \in R^{m \times n}$
- $n > m$

LP and simplex algorithm

The main idea of the simplex algorithm is to find an extreme point and implicitly check its adjacent extreme points.

- Form a basis matrix $B \in R^{m\times m}$;
- Compute reduced costs $\bar{c}_j = c_j - c_B B^{-1} A_j$ for all nonbasic indices $j \in \{1, \dots, n\}$;
- Compute $u = B^{-1} A_j$;
- Form a new basis by replacing $A_{B(l)}$ with A_j.

LP and travelling salesman problem (TSP)

TSP considers a list of cities on a connected graph and finds the shortest route that visits each city exactly once and returns to the origin city.

- Set of cities $N = \{1, \ldots, n\}$;
- Length of an arc $i, j \in N$ is c_{ij};
- Decision variables $x_{ij} = 1$, iff $i, j \in N$.

LP and travelling salesman problem (TSP)

In connection to LP it is needed to:

- Minimize sum:
  $\sum_{i,j \in N: i \neq j} c_{ij} x_{ij}$;

- subject to:
  $\sum_{j \in N: j \neq i} x_{ij} = 1, \forall i \in N$;
  $\sum_{i \in N: i \neq j} x_{ij} = 1, \ \forall j \in N$;
  $x_{ij} \in \{0,1\}, \ \forall i, j \in N: i \neq j$

Learning approach

How to reduce the solution time of the LP relaxation for the TSP?

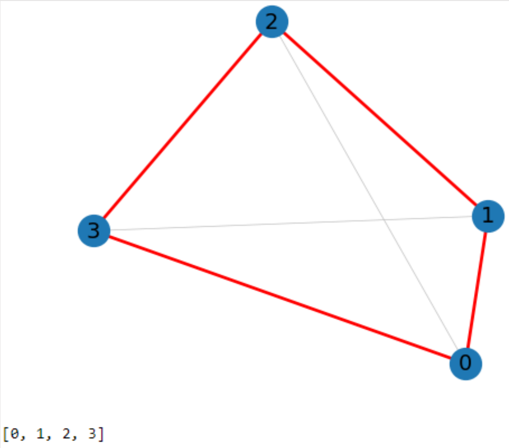
Main steps in an iteration:

- Formulate the problem;
- Use the phase one implementation of a linear programming solver to find a basic feasible solution;
- Pass a reduced cost vector and the objective value to a ReLU NN to estimate the Q-value;
- Based on the Q-value choose a pivoting rule.

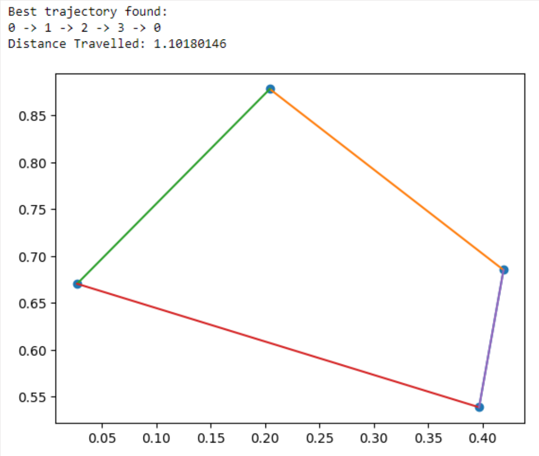
Experiment design

1. Generate coordinates and distances between them;
2. Picked two metrics (euclidean and cityblock) to check the difference;
3. Define a reward function, where the Dantzig’s rule is cheaper, than the steepest edge rule ;
4. Define a Q-value function as the total of expected discounted future rewards;
5. Choose a neural network architecture as 4 fully connected hidden layers.

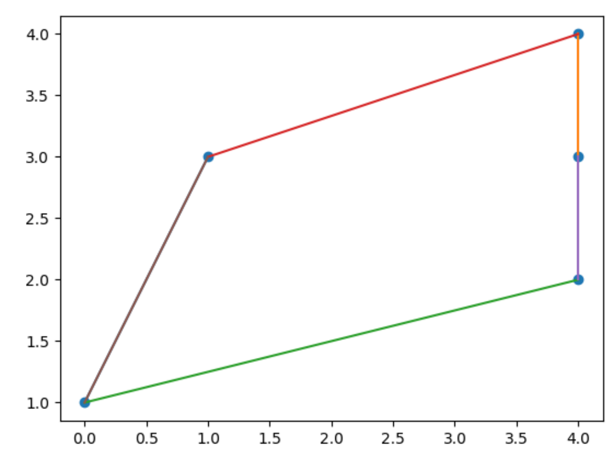
Results of the Q-function



Common approach



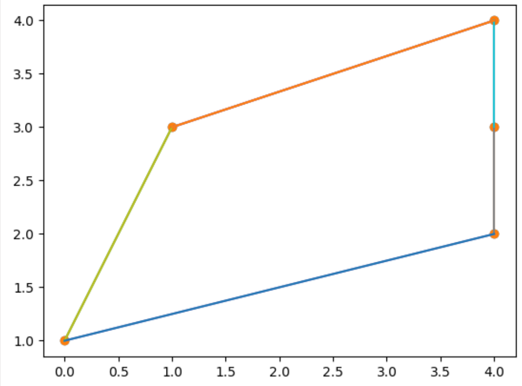
Manual realisation of the Q-function



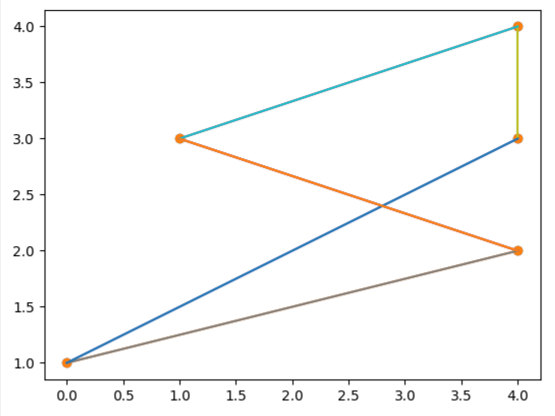
Manual realisation of the Q-function (5 nodes)

https://github.com/GrikTad/NLA_Final_Project

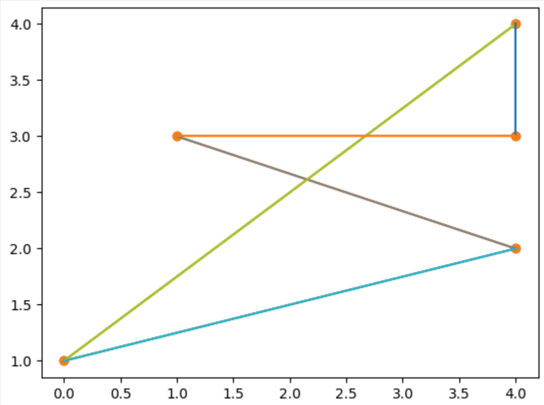
Results of the NN for 5 vertices



100 epochs



1000 epochs



8000 epochs

https://github.com/GrikTad/NLA_Final_Project

Applications and future work

Applications:
- Delivery, traveling, industrial drones.

Future work:
- Increase a number of epochs;
- Try out another approach with graph embeddings, encoders and decoders;
- Conduct more experiments.

Our team

Grik
Tadevosyan

General algorithm principles

Ivan
Razvorotnev

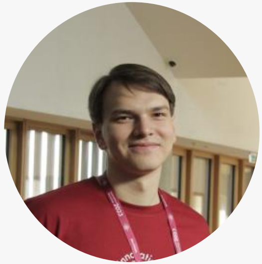
Experiment design and conduction

Nikita
Khoroshavtsev

Code refactoring, graphics

Maksim
Osipenko

Examples generation, experiment conduction, presentation



Elvir
Karimov

Common approach realization

Thank you for your attention!