ALMA MATER STUDIORUM · UNIVERSITY OF BOLOGNA

Flatland Challenge

FLATLAND

Deep learning course final project

Leonardo Calbi (leonardo.calbi@studio.unibo.it)
Alessio Falai (alessio.falai@studio.unibo.it)

December 5, 2020

Contents

List of Figures

Foreword

The Flatland challenge is a competition organized by AIcrowd [1] with the help of SBB (Swiss Federal Railways) to foster innovation with what regards the scheduling of trains trajectories in a railway environment.

As reported on the official challenge website, SBB operates the densest mixed railway traffic in the world. It maintains and operates the biggest railway infrastructure in Switzerland. Today, there are more than 10000 trains running each day, being routed over 13000 switches and controlled by more than 32000 signals.

The Flatland challenge aims to address the vehicle rescheduling problem by providing a simplistic grid world environment and allowing for diverse solution approaches. In particular, the first edition of the challenge was hosted during 2019 and the submitted solutions were mainly based on OR (Operation Research) methodologies, while the second edition of the competition, i.e. the NeurIPS 2020 edition, had the goal of favoring the implementation of RL (Reinforcement Learning) based solutions.

Introduction

At the core of this challenge lies the general vehicle re-scheduling problem (VRSP) proposed Li, Mirchandani, and Borenstein in 2007 [2]:

> *The vehicle rescheduling problem (VRSP) arises when a previously assigned trip is disrupted. A traffic accident, a medical emergency, or a breakdown of a vehicle are examples of possible disruptions that demand the rescheduling of vehicle trips. The VRSP can be approached as a dynamic version of the classical vehicle scheduling problem (VSP) where assignments are generated dynamically.*

The problem is formulated as a 2D grid environment with restricted transitions between neighboring cells to represent railway networks. On the 2D grid, multiple agents with different objectives must collaborate to maximize the global reward.

The overall goal is to make all agents (trains) arrive at their target destination with a minimal travel time. In other words, we want to minimize the time steps (or wait time) that it takes for each agent in the group to reach its destination.

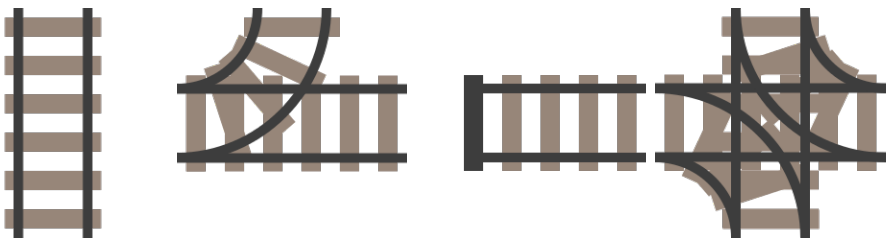
Background

As already pointed out, the Flatland environment is represented as a 2D grid and each cell can be one of many different types. The different types of cells can belong to the following categories: `rail`, `target` and `empty`.

About the `target` cells, they represent the destination of one or more agents (different agents could have the same target) and the number of possible targets present in the environment is clearly limited by the number of agents.

The `rail` cells are more intricated, in that there exists different types of them. In particular, figure 2.1 shows examples of possible `rail` cells that can be used to build up a railway environment in Flatland. Other than the ones shown in figure 2.1 there are also diamond crossings (i.e. two orthogonal straight rails crossing each other), single slip switches (i.e. the same as double slip switches but with a single choice) and symmetrical switches (which are special kinds of switches that bifurcate to a left and right branch). Moreover, every `rail` cell can be rotated by $90°$ and mirrored along both axis.

An important fact about the different types of `rail` cells is that only switches require an agent to make a choice. In Flatland (like in reality) a maximum of two options is available. There does not exist a switch with three or more options.



(a) Straight (b) Simple switch (c) Deadend (d) Double slip switch

Figure 2.1: Different `rail` cell types

Finally, every cell that is not `rail` or `target` is `empty`. As shown in figure 2.2, it is interesting to notice that Flatland is a very sparse environment, meaning that there are a lot more `empty` cells than non-`empty` ones: because of this, representing the environment as a simple dense matrix could lead to overheads and efficiency issues, especially when dealing with relatively big environments.

An agent in the Flatland environment is a train that starts from a random

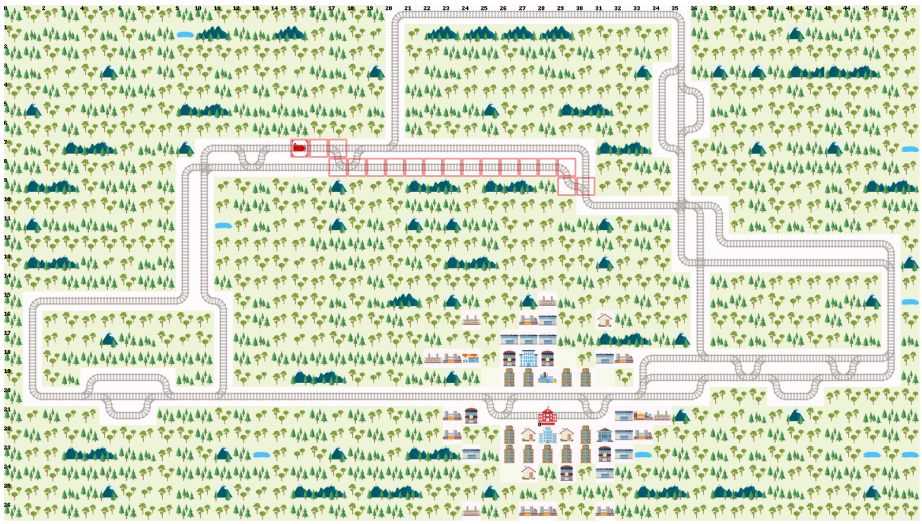


Figure 2.2: An example of a railway environment

`rail` cell in the map and has to arrive to its assigned `target` cell. To do so, the agent can only occupy `rail` and `target` cells. To move from a cell to another one the agent has to make a choice and, depending on the cell type that they are on

Each `rail` cell

Flatland has a discrete action space, meaning that only 5 possibilities have to be considered at each transition. In particular, Flatland considers the following actions:

1. `MOVE_FORWARD`
2. `MOVE_LEFT`
3. `MOVE_RIGHT`
4. `STOP_MOVING`
5. `DO_NOTHING`

Environment

3.1 Railway encoding

3.2 Observations

3.2.1 Tree

3.2.2 Binary tree

3.3 Predictions

3.3.1 Shortest path

3.4 Choices

3.5 Rewards shaping

Policy

4.1 Action masking

4.2 Action selection

4.2.1 ϵ-greedy

4.2.2 Boltzmann

4.3 Replay buffers

4.3.1 Uniform

4.3.2 Prioritized

DQN

5.1 Architectures

5.1.1 Vanilla

5.1.2 Double

5.1.3 Dueling

5.2 Bellman equation

5.2.1 Max

5.2.2 Softmax

GNN

Results

Conclusions

Bibliography

[1] AIcrowd. *AIcrowd*. URL: https://www.aicrowd.com/.

[2] Jing-Quan Li, Pitu B. Mirchandani, and Denis Borenstein. "The vehicle rescheduling problem: Model and algorithms". In: *Networks* 50.3 (2007), pp. 211–229. DOI: https://doi.org/10.1002/net.20199.