A Review of Real-time Railway and Metro Rescheduling Models using Learning Algorithms

Matej Jusup
Alessio Trivella
Francesco Corman
A Review of Real-time Railway and Metro Rescheduling Models using Learning Algorithms

- DADA project team
- Problem description
- Relevance
- RL components in rescheduling
- RL vs traditional rescheduling models
- Literature overview along the years
- Learning algorithms used for train rescheduling
- Flatland challenge
- Conclusions and future research
DADA project team

Prof. Francesco Corman

Thomas Spanninger
Dr. Ping Huang
Dr. Alessio Trivella
Matej Jusup

Delay prediction
Rescheduling under uncertainty

DADA project – Dynamic data driven approaches for stochastic delay propagation avoidance

Sponsor: FNS/NG

1 Authors: M. Jusup, A. Trivella, F. Corman | 13.09.2021 | Slide 3
Personal background

- BSc in Mathematics
- MSc in Mathematical Statistics
  - Master thesis: “Network Optimization in Railway Transport Planning”
- Various roles in an investment bank
  - 1 year in a technology team
  - 1.5 years in a quantitative research team
- Data scientist in a start-up
  - 1.5 years as a team leader
- Currently a PhD student in Transport Systems group
  - Supervisor: Prof. Dr. Francesco Corman
Problem description

- Railway and metro networks operate according to predefined schedules
- Real-life operations are subject to uncertainty in e.g. train running time and dwelling time and/or passenger demand causing conflicts in the schedule
- Goal of rescheduling is to compute an updated conflict-free schedule while minimizing deviations from the original schedule
Relevance of the problem

Delays
- Disturbances often occur during real-life operations
- Primary delays cannot be reduced
- Secondary delays result from delay propagation
  - We can reduce or prevent them by rescheduling actions

Good rescheduling actions can:
- Minimize secondary delays
- Improve user experience
- Increase infrastructure utilization
- Reduce energy consumption

![Diagram showing no delay, 10 min delay, secondary delay incurred, secondary delay avoided]
1. **Agent** – dispatcher observes the environment and executes actions

2. **Environment** – infrastructure and uncertainty

3. **State space** – environment’s representation available to the agent, e.g. train location and speed, number of in-vehicle passengers, passenger demand, section availability

4. **Action space** – includes e.g. adjusting train departure, running and/or dwelling time, modifying signal shown, changing train speed, rerouting trains

5. **Reward/cost function** – commonly a function of train delay, train running time, passenger delay, and/or energy utilization
**State space, actions and reward function modelling options**

**State space** might include:
- Train location
- Number of in-vehicle passengers
- Block section availability
- Disturbance time
- Disturbance duration
- Arrival time
- Dwelling time
- Train speed
- Train direction

**Actions** might be:
- **Station-level**
  - Varying dwelling time
  - Varying departure time
  - Adjusting running time
- **Block-section-level**
  - Modifying signalling
- **Train-level**
  - Adjusting speed

**Reward/cost** could be a function of:
- Train/passenger delay
- Train running time
- Passenger travelling time
- Energy utilization

---

**Block section signalling**

<table>
<thead>
<tr>
<th>Stage t</th>
<th>Stage t+1</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Block section signalling" /></td>
<td><img src="image2.png" alt="Block section signalling" /></td>
</tr>
</tbody>
</table>

**Train speed**

<table>
<thead>
<tr>
<th>Stage t</th>
<th>Stage t+1</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3.png" alt="Train speed" /></td>
<td><img src="image4.png" alt="Train speed" /></td>
</tr>
</tbody>
</table>
Reinforcement learning vs traditional rescheduling models

- Traditionally, rescheduling has been tackled using rolling horizon techniques, stochastic optimization or MILP-based models.

### Advantages of using learning models

<table>
<thead>
<tr>
<th>Advantages</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning decision policy offline</td>
<td>No guarantee of optimality bounds</td>
</tr>
<tr>
<td>Adaptiveness/online learning</td>
<td>Not easy to impose constraints</td>
</tr>
<tr>
<td>Instantaneous high-quality decisions</td>
<td>High computational resources for training</td>
</tr>
<tr>
<td>Potential of implementing transfer learning</td>
<td></td>
</tr>
</tbody>
</table>
Findings and takeaways:

- Increase of the research interest on the topic
- Recent experimentation of learning algorithms in rescheduling has shown promising results on simplified instances
Learning algorithms used for train rescheduling

<table>
<thead>
<tr>
<th>SARSA</th>
<th>Q-learning</th>
<th>Deep deterministic policy gradient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beneficial when we care about the agent’s performance during the</td>
<td>Preferable in situations where good training time performance is not</td>
<td>Works with continuous actions—e.g. we might control the speed very</td>
</tr>
<tr>
<td>training process—e.g. we don’t want to cause train accidents or deadlocks</td>
<td>necessary—e.g. we have weeks to train a model in the simulated environment</td>
<td>precisely</td>
</tr>
<tr>
<td></td>
<td>Better option for railway rescheduling</td>
<td>Hard to imagine real-life use-cases where we need such a precision</td>
</tr>
</tbody>
</table>
Flatland challenge

- Solving train rescheduling problem
  - One of NeurIPS 2020 challenges*

- Open-source Python package for easy environment construction**
  - Developed and maintained by SBB and Alcrowd

- Potential to become the community-wide benchmark

- Traditional operations research methods dominated the leaderboard
  - Focusing on RL approaches might change that dynamic

---

* https://www.aicrowd.com/challenges/flatland
** http://flatland-rl-docs.s3-us-west-1.amazonaws.com/
## Conclusions and future research

<table>
<thead>
<tr>
<th>Conclusions</th>
<th>Future research</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø Further improvements of the existing RL models are needed</td>
<td>Ø Expanding methodological scope by applying different classes of learning algorithms (e.g. deep Q-learning, graph neural networks)</td>
</tr>
<tr>
<td>Ø Hard to implement community-wide benchmark due problem’s representation high dependance on the infrastructure type</td>
<td>Ø Exploiting larger computational power</td>
</tr>
<tr>
<td>Ø <strong>Scaling up models from lines/junctions to networks is still an open challenge</strong></td>
<td>Ø Work on a community-wide benchmark might be beneficial (e.g. Flatland)</td>
</tr>
<tr>
<td></td>
<td>Ø Transfer learning might have a potential to tackle some of the challenges</td>
</tr>
</tbody>
</table>
Thank you!