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[1]

Thomas Spanninger

Dr. Ping Huang

Dr. Alessio Trivella[1]

Matej Jusup[1]

Delay prediction

Rescheduling under uncertainty

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

Sponsor: FNSNF Swiss National Science Foundation

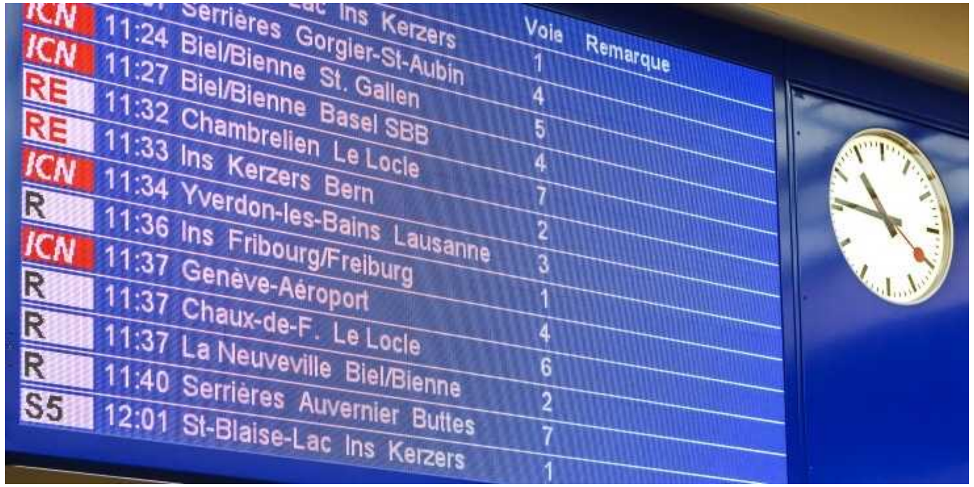
IVT

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



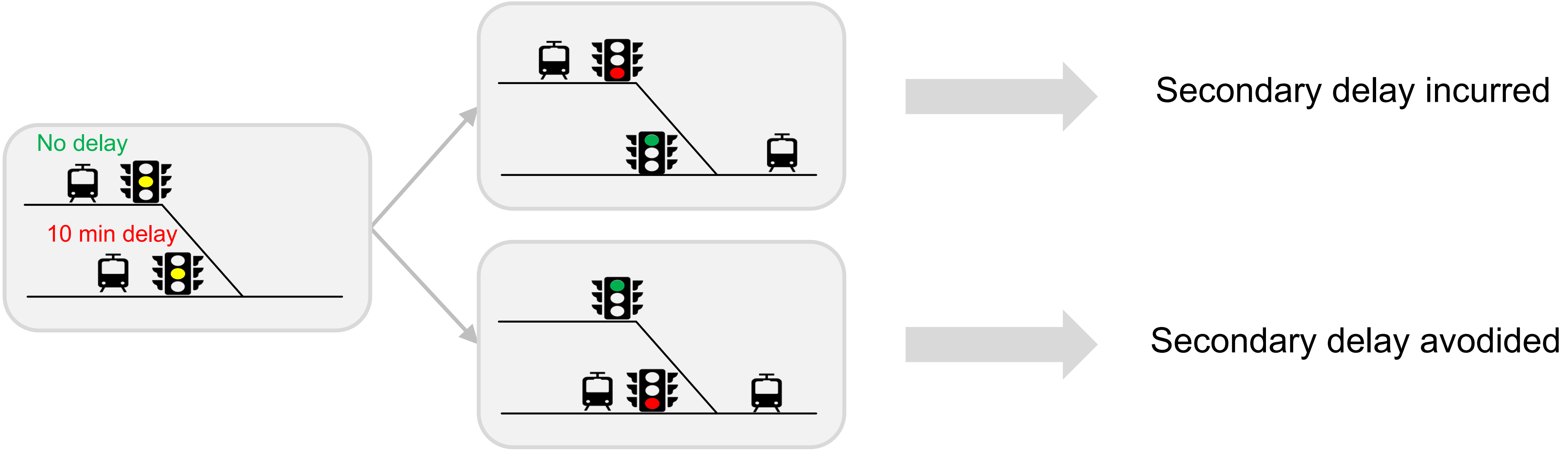
Relevance of the problem

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

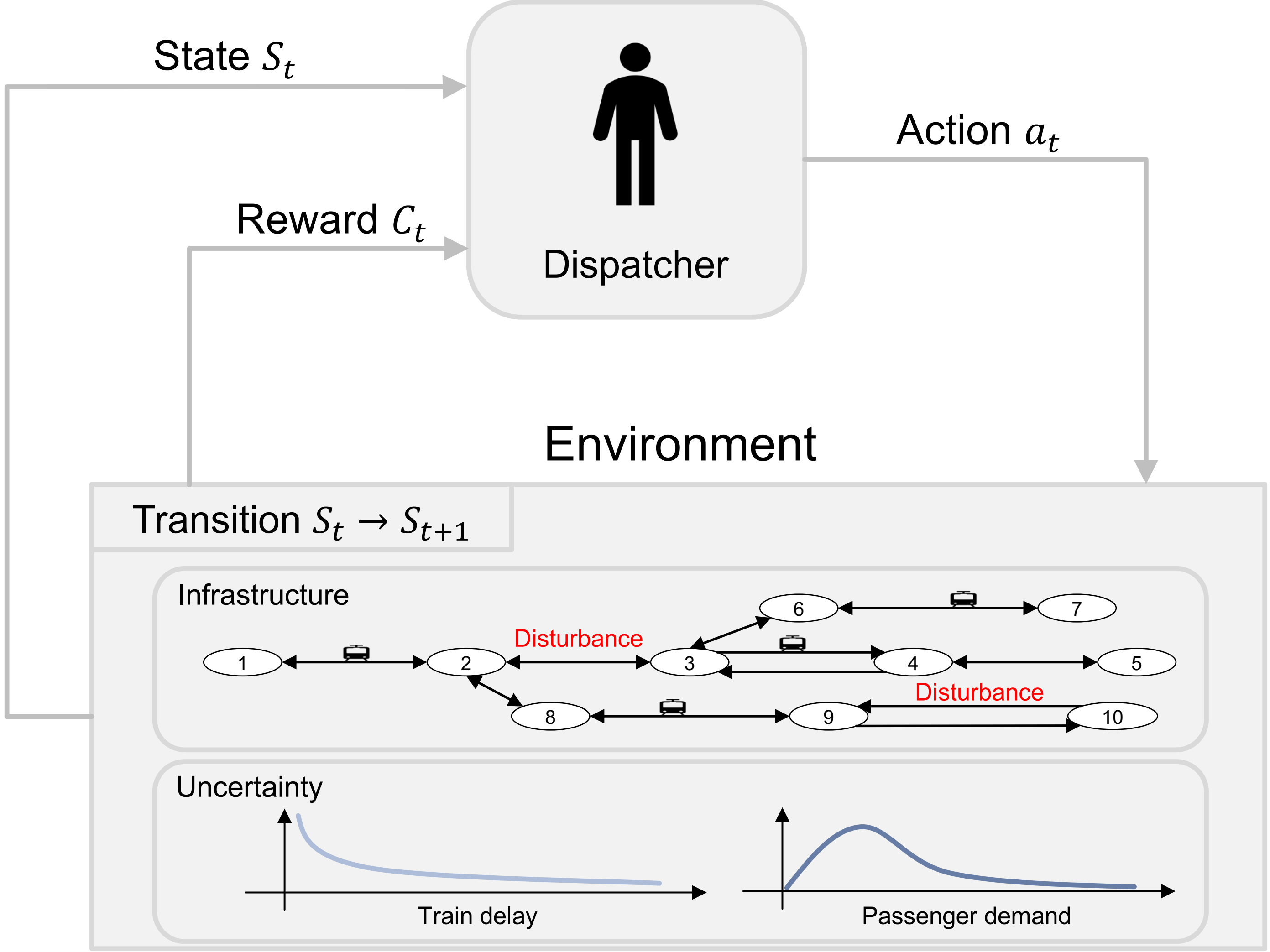
Good rescheduling actions can:

- Minimize secondary delays
- Improve user experience
- Increase infrastructure utilization
- Reduce energy consumption



Reinforcement learning components in rescheduling

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



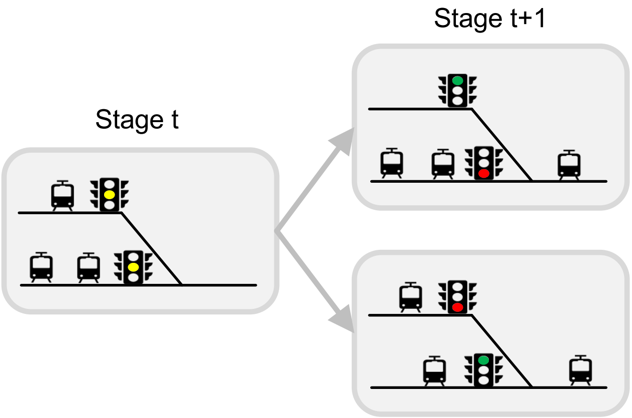
State space, actions and reward function modelling options

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

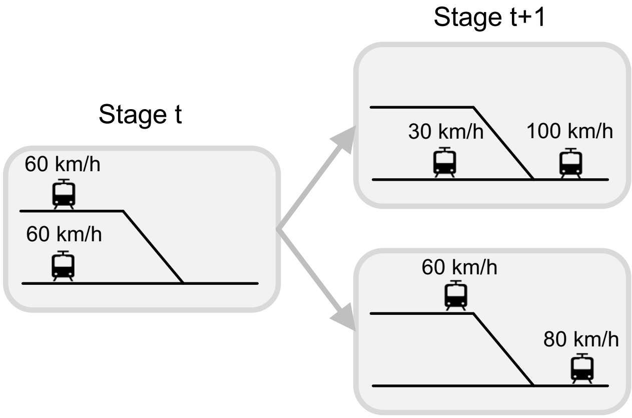
Actions might be:
- Station-level
  - Varying dwelling time
  - Varying departure time
  - Adjusting running time
- Block-section-level
  - Modifying signalling
- Train-level
  - Adjusting speed

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

Block section signalling



Train speed



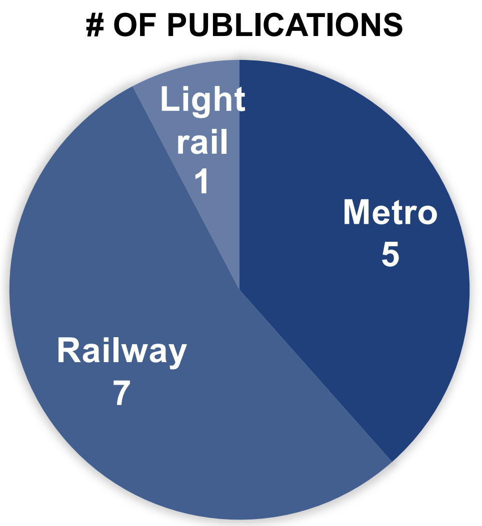
Reinforcement learning vs traditional rescheduling models

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

Advantages of using learning models

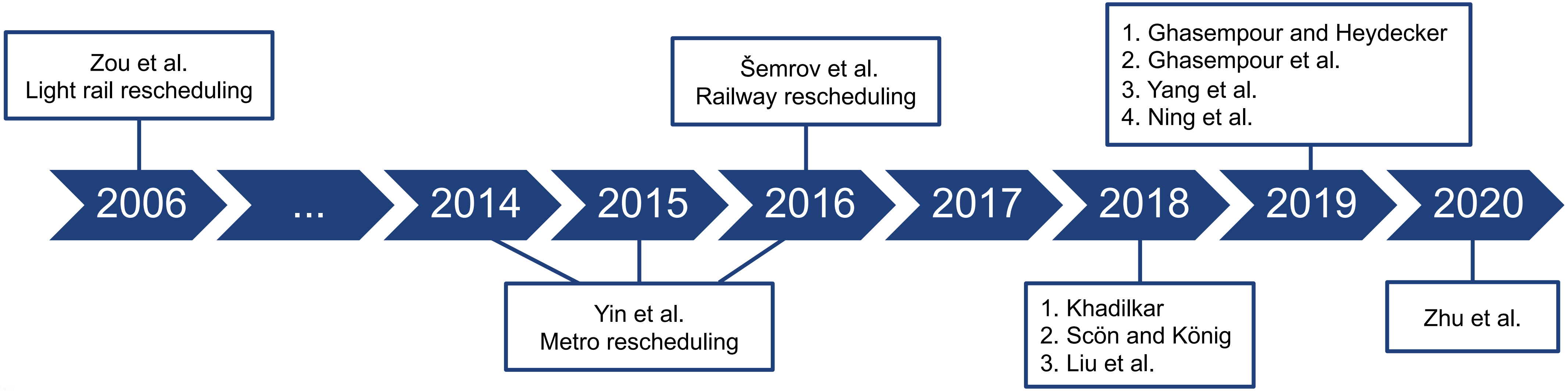
Advantages	Limitations
Learning decision policy offline	No guarantee of optimality bounds
Adaptiveness/online learning	Not easy to impose constraints
Instantaneous high-quality decisions	High computational resources for training
Potential of implementing transfer learning	

Literature overview along the years



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

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

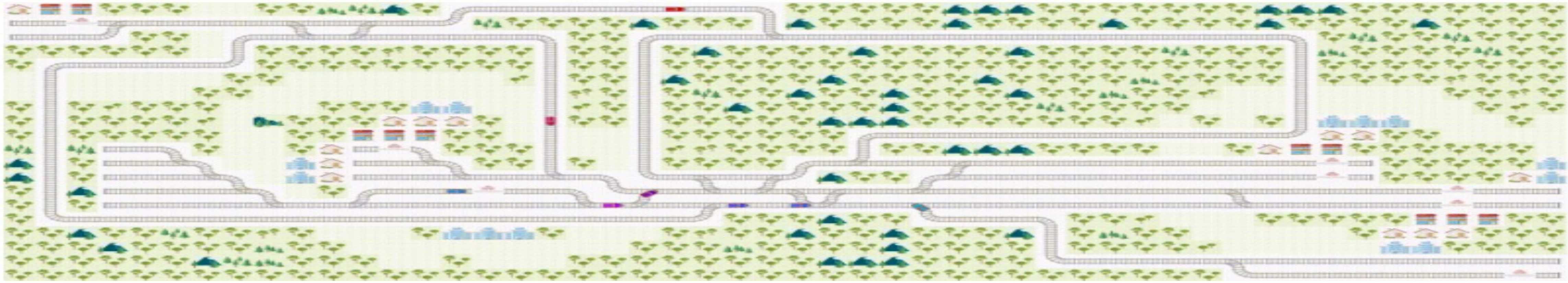
Flatland challenge

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

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

- 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-website.eu-central-1.amazonaws.com/

Conclusions and future research

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

Thank you!