Learning to Dispatch: A Reinforcement Learning Framework for Train Dispatch Networks

Andres Espinosa

Industrial and Systems Engineering
University of Florida
andresespinosa@ufl.edu

Abstract.

1 Introduction

The Train Dispatching Problem (TDP) concerns the real-time management of train movements across a rail network, ensuring safe and efficient use of infrastructure. This involves deciding when and where trains should move, stop, or wait, based on factors such as schedules, track availability, and priority rules. Dispatching decisions must be made continuously and quickly, especially in high-traffic networks, making the problem both operationally critical and computationally challenging.

Despite significant technological advances in other areas of transportation and logistics, train dispatching remains heavily reliant on human decision-makers or static rule-based systems. This is largely because the problem is highly combinatorial: at any given moment, there are exponentially many possible sequences of decisions that can lead to different network outcomes. Human dispatchers bring experience and intuition to these situations, but they are limited in how much information they can process and how consistently they can manage large-scale disruptions or optimize traffic flow over time.

Improving train dispatching systems has the potential to reduce passenger delays, minimize dispatching errors, and prevent network deadlocks—situations where no train can proceed without violating safety or scheduling constraints. Optimized dispatching can also improve energy efficiency and capacity utilization, making rail transport more competitive and sustainable. In dense urban transit systems or busy freight corridors, even marginal improvements in dispatching can lead to significant gains in overall network performance and reliability.

Reinforcement Learning (RL) offers a new approach to tackling the TDP by framing it as a sequential decision-making problem, where an agent learns to make dispatching decisions through interactions with a simulated environment. RL is particularly well-suited for problems with delayed consequences, dynamic environments, and large state spaces—all of which characterize train dispatching. With recent successes in games, robotics, and supply chain optimization, RL is emerging as a promising tool for learning policies that outperform hand-crafted heuristics in complex, real-world systems.

This paper focuses on the formulation of the Train Dispatching Problem for RL-based approaches, rather than on developing a fully trained solution. We emphasize how the problem can be encoded as an RL task using graph structures to represent rail networks, define meaningful states, actions, and rewards, and manage constraints inherent to railway systems. Our goal is to provide a robust and extensible framework that future researchers and practitioners can build upon when applying RL methods to train dispatching and related infrastructure scheduling problems.

To provide a thorough understanding of our approach, the remainder of this paper is organized as follows. In Section 2, we present the foundational background necessary for our formulation, beginning with a formal description of the Train Dispatch Problem and its mixed-integer programming (MIP) formulation, as well as the DISPLIB benchmark format (2.1). We then outline key reinforcement learning concepts, including Markov Decision Processes (2.2) and Deep Q-Networks, highlighting their relevance to sequential decision-making in dispatching tasks. This is followed by an introduction to Graph Neural Networks (2.3), which are critical for representing the structure of railway networks. Section 3 provides a review of related literature, including recent RL applications to combinatorial scheduling, graph-based approaches, and prior work on train dispatching. Section 4 details our proposed formulation, introducing Train Operation Graphs (4.1), Resource Conflict Graphs (4.2), and the definitions of our state and action spaces (4.3, 4.4). In Section 5, we demonstrate preliminary results from our Deep Graph Q-Network agent (5.1), evaluating its performance on sample instances and visualizing its learned strategies (5.2). Finally, Section 6 outlines our conclusions and discusses future directions for scaling and refining the framework (6.1, 6.2).

2 Problem Background

2.1 Train Dispatch Problem

2.2 Deep Reinforcement Learning

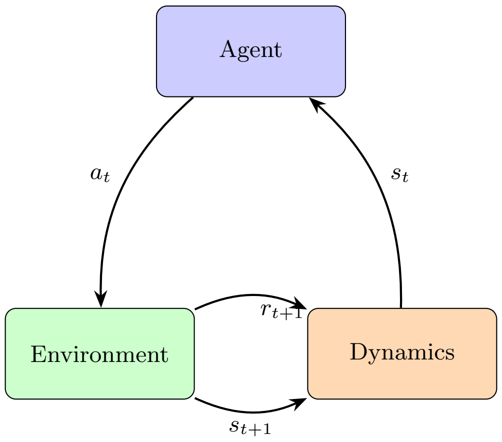


Fig. 1: The simplified markov decision process and reinforcement learning framework.

Markov Decision Processes

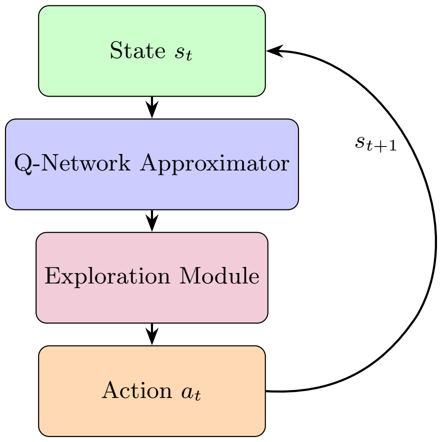


Fig. 2: Simplified Deep Q-Network action selection and feedback loop.

Deep Q-Network

2.3 Graph Neural Networks

3 Related Work

I am [1]

4 Formulation

4.1 Train Operation Graphs

4.2 Resource Conflict Graphs

4.3 State Space

4.4 Action Space

5 Preliminary Agent Results

5.1 Deep Graph Q-Network Agent

5.2 Solutions

6 Conclusion and Future Work

6.1 Future Work

6.2 Conclusion

7 Appendix

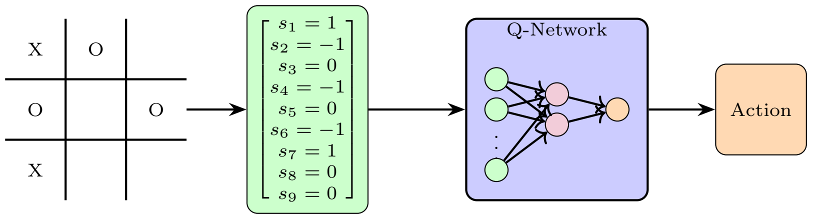
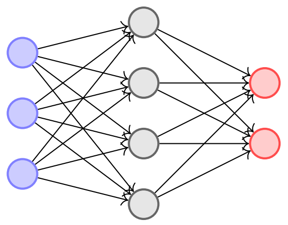


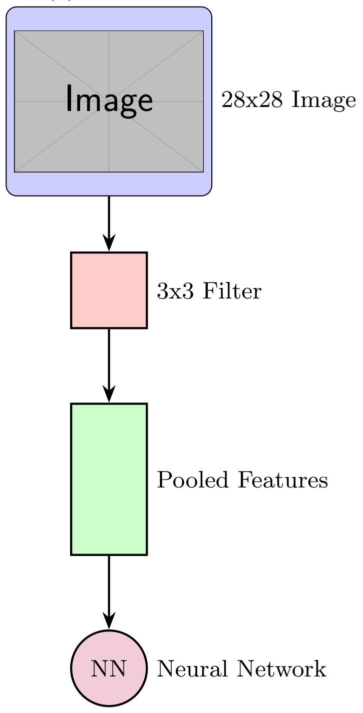
Fig. 3: Illustration of a Q-Network processing a Tic-Tac-Toe board state.

References

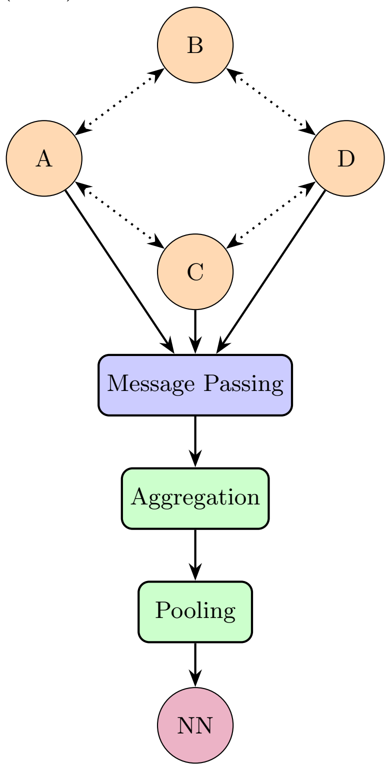
1. F.-X. Devailly, D. Larocque, and L. Charlin, “Ig-rl: Inductive graph reinforcement learning for massive-scale traffic signal control,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, p. 7496–7507, Jul. 2022. [Online]. Available: http://dx.doi.org/10.1109/TITS.2021.3070835



(a) Fully Connected ANN



(b) Convolutional Neural Network (CNN)



(c) Graph Neural Network (GNN) with Message Passing and Aggregation Steps