A Planning-based Approach for In-Station Train Dispatching

Matteo Cardellini,[1] **Marco Maratea,**[1] **Mauro Vallati,**[2] **Gianluca Boleto,**[1] **Luca Oneto**[1]
[1] DIBRIS, University of Genoa, Italy
[2] School of Computing and Engineering, University of Huddersfield, UK
matteo.cardellini@edu.unige.it, marco.maratea@unige.it, m.vallati@hud.ac.uk,
gianluca.boleto@edu.unige.it, luca.oneto@unige.it

Abstract

In-station train dispatching is the problem of optimising the effective utilisation of available railway infrastructures for mitigating incidents and delays. In this paper, we describe an approach for dealing with the in-station dispatching problem by means of automated planning techniques.

Introduction

Railways play a significant economical role in our society for transporting either goods or passengers, but the increasing volume of people and freight transported on railways is congesting the networks (Bryan, Weisbrod, and Martland 2007). Within a railway network, stations are probably the most critical points for interconnecting trains' paths. This is because, in a station, a number of trains need to stop according to a given timetable, and a number of possible routes are thus occupied, with the concrete risk of accumulating delays, that may boil down to cost penalties and inconveniences for passengers. In this context, in-station train dispatching plays a central role in maximising the effective utilisation of available railway infrastructures and in mitigating the impact of incidents and delays.

In this abstract we briefly describe an approach for performing in-station dispatching by means of automated planning techniques. Given the mixed discrete-continuous nature of the problem, we employ PDDL+ (Fox and Long 2006) for the specification of the model, and we extend the state-of-the-art planning engine ENHSP (Scala et al. 2016, 2020) with customised techniques such as an adaptive notion of delta, a domain-specific heuristic, and a set of ad-hoc constraints. The interested reader is referred to Cardellini et al. (2021a,b) for additional details and analysis.

We carry out an experimental analysis using real data of a station of the North West of Italy, provided by Rete Ferroviaria Italiana (RFI), that shows the contribution that the implemented domain-specific techniques may have in efficiently solving the various instances of the problem.

The PDDL+ Domain Model

Informally, the in-station train dispatching problem can be defined as follows. Given a station infrastructure, a set of trains and their current position in the station or expected time of arrival, find a route for every train that allows to respect the provided timetable, as much as possible.

A railway *station* can be represented as a graph, composed by a set of connected *track segments*, the minimal controllable rail units. Sequences of connected track segments are organised in *itineraries*; this is manually done by experts of the specific railway station. A track segment can be occupied by a single train at the time. For safety reasons, a train is required to *reserve* an itinerary, and this can be done only if the itinerary is currently not being used by another train. A train traverses the railway station by reserving an itinerary and moving through the corresponding track segments.

The core of the proposed PDDL+ encoding is the way in which the movement of trains in the railway station is modelled and controlled. A set of dedicated operators allow a train to reserve an itinerary and to start moving on it. The use of such operators triggers a process that models the time needed by the train to reach the end of the itinerary. Over time, track segments of the itinerary that are not occupied by the train anymore are released via events. Additional operators allow a train that reached an exit point to leave the station, and to stop a train at a platform to allow the disembarking/embarking of passengers. The duration of the stop is variable: each type of train has a minimum time that is required to stop in order to safely allow the movement of passengers; further, a train is not allowed to leave a platform before its timetabled leaving time. Finally, an additional operator is used to model the fact that a train has reached its final destination, and should not move any further.

Dedicated processes are used to keep track of the time spent by a train (i) navigating an itinerary; (ii) moving from one itinerary to the next, and (iii) stopping at a platform.

A pivotal point of the PDDL+ model relates to the way in which the structure of the railway station is encoded. To avoid the well-known issues of a potentially huge ground problem (Scala and Vallati 2020), due to the presence of itineraries, trains, track segments, etc. the domain model is fully ground for a given problem to solve. The grounding is done via a pre-processing step that takes into account the actual structure of the network, and the considered trains. The structure of the railway station is directly encoded in the ground actions provided in the domain model.

A planning problem is specified in terms of the initial lo-

cation of trains, and of the required goal state defined as the positions that trains must reach. Example models can be found at https://github.com/matteocarde/icaps2021.

Heuristic Search for Train Dispatching

The ENHSP planning engine (Scala et al. 2016, 2020) has been used to solve in-station train dispatching problems encoded in PDDL+. It is a forward search planning engine that deals with continuous processes using the Discretise and Validate approach. ENHSP has been optimised in three ways: adaptive delta, heuristic, and constraints.

Adaptive Delta. Traditionally, PDDL+ planning engines exploit a fixed time discretisation step for solving a given instance. In the proposed model, it is possible to know a-priori when events will be triggered, so it is possible to exactly predict when the world will change. It is therefore possible to exploit a dynamic time discretisation. This is similar in principle to the approach exploited by decision-epoch planning engines for dealing with temporal planning problems (Cushing, Kambhampati, and Weld 2007).

Specialised Heuristic. Following the traditional A* search settings, the cost of a search state z is calculated as $f(z) = g(z)+h(z)$, where $g(z)$ represents the cost to reach z, while $h(z)$ provides an heuristic estimation of the cost needed to reach a goal state from z. In our specialisation, $g(z)$ is calculated as the elapsed modelled time from the initial state to z. $h(z)$ is a domain-specific heuristic calculated as follows:

$$h(z) = \sum_{t \in T(z)} \rho_t(z) + \pi_t(z) \quad (1)$$

where $T(z)$ is the set of trains of the given problem that did not yet achieve their goals at z. $\rho_t(z)$ is a quantity that measures the time that, starting from the current position, the considered train needs to reach its final destination. The method $\pi_t(z)$ gives a very high penalisation for each goal of the considered train t that has not yet been satisfied at z.

Constraints. The planning engine has been extended by explicitly considering 3 set of constraints. The first set limits the time a train is allowed to stay in the controlled station. The other sets limit, respectively, the time passed from the arrival of the train in the station, and the time spent stopping at a platform. The idea behind such constraints is to avoiding situations where trains are left waiting for long periods of time, occupying valuable resources. The maximum times are calculated a-priori, according to historical data, and depends on the structure of the railway station.

Evaluation

We tested the proposed PDDL+-based approach by exploiting 5 months (January to May 2020) of real-world data of one medium-sized railway station provided by RFI. The station includes 130 track segments (34 track switches), 107 itineraries, 10 platforms, 3 entry and 3 exit points.

Here we focus on assessing the usefulness of the domain-specific improvements. To perform this analysis, we selected the day –in February 2020, before the start of the COVID-19 lockdown in Italy– with the minimum mean squared deviation of recorded train timings from the official timetable.

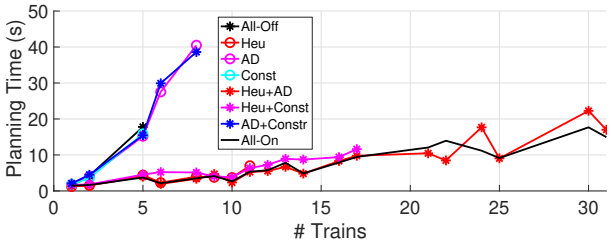

Figure 1: The CPU-time needed for planning instances with an increasing number of trains using different combinations of domain-specific optimisation techniques.

This was done to guarantee that no emergency operations were executed by the human operators. We considered the evening peak hour of the day, 17:00-20:00. During the selected time-slot 31 trains moved through the station. We considered different time windows of the 3-hours period to generate instances with an increasing number of trains.

Figure 1 shows the CPU planning time required by the 8 combinations to deal with increasingly large instances. The combination labelled *All-Off* considers ENHSP run using the default settings, and none of the optimisation techniques introduced in this paper. The *All-On* label indicates instead the planning engine run using all the optimisations. Finally, labels *AD*, *Heu*, and *Const* are used to indicate the use of, respectively, adaptive delta, domain-specific heuristic, and constraints. The analysis was performed with a cut-off time of 60 CPU-time seconds, on a 2.5GHz Intel Core i7 Quad-processors with 16GB of memory made available.

Summarising, the performed experiments indicate that the use of the specialised heuristic is the single most important component of the domain-specific planning engine. The adaptive delta plays an important role as well, but their synergic combination allows to solve all evaluated instances. The use of constraints provides some improvement, but not as significant as the other elements.

Conclusion

We presented an automatic solution to the in-station train dispatching problem. We modelled the problem in PDDL+, and designed a set of domain-specific enhancements that allow the ENHSP planning engine to solve complex instances.

As future work, we are interested in extending our approach to support maintenance operations and preferences, and to consider different and more complex scenarios. Finally, we would like to consider alternative approaches, such as Answer Set Programming (Abels et al. 2020).

Acknowledgements

This work has been partially funded by Hitachi Rail STS through RAIDLab, a joint laboratory with University of Genoa. This work has been supported by Rete Ferroviaria Italiana who provided the data for the analysis (we sincerely thank Renzo Canepa for his support). Mauro Vallati was supported by a UKRI Future Leaders Fellowship [grant number MR/T041196/1].

References

Abels, D.; Jordi, J.; Ostrowski, M.; Schaub, T.; Toletti, A.; and Wanko, P. 2020. Train Scheduling with Hybrid Answer Set Programming. *Theory and Practice of Logic Programming* 1–31.

Bryan, J.; Weisbrod, G. E.; and Martland, C. D. 2007. *Rail freight solutions to roadway congestion: final report and guidebook*, volume 586. Transportation Research Board.

Cardellini, M.; Maratea, M.; Vallati, M.; Boleto, G.; and Oneto, L. 2021a. An Efficient Hybrid Planning Framework for In-Station Train Dispatching. In *Proceedings of ICCS*.

Cardellini, M.; Maratea, M.; Vallati, M.; Boleto, G.; and Oneto, L. 2021b. In-Station Train Dispatching: A PDDL+ Planning Approach. In *Proceedings of ICAPS*.

Cushing, W.; Kambhampati, S.; and Weld, D. S. 2007. When is temporal planning really temporal? In *Proceedings of IJCAI*, 1852–1859.

Fox, M.; and Long, D. 2006. Modelling Mixed Discrete-Continuous Domains for Planning. *Journal of Artificial Intelligence Research* 27: 235–297.

Scala, E.; Haslum, P.; Thiébaux, S.; and Ramírez, M. 2016. Interval-Based Relaxation for General Numeric Planning. In *Proceedings of ECAI*, 655–663.

Scala, E.; Haslum, P.; Thiébaux, S.; and Ramírez, M. 2020. Subgoaling Techniques for Satisficing and Optimal Numeric Planning. *Journal of Artificial Intelligence Research* 68: 691–752.

Scala, E.; and Vallati, M. 2020. Exploiting Classical Planning Grounding in Hybrid PDDL+ Planning Engines. In *Proceedings of ICTAI*, 85–92.