Train Scheduling with Hybrid ASP

Dirk Abels[2], Julian Jordi[2], Max Ostrowski[1], Torsten Schaub[1,3⋆], Ambra Toletti[2], and Philipp Wanko[1,3]

[1] Potassco Solutions, Germany
[2] SBB, Switzerland
[3] University of Potsdam, Germany

Densely-populated railway networks transport millions of people and carry millions of tons of freight daily; and this traffic is expected to increase even further. Hence, for using a railway network to capacity, it is important to schedule trains in a flexible and global way. This is however far from easy since the generation of railway timetables is already known to be intractable for a single track [2]. Hundreds of trains on a densely connected railway network lead to complex inter-dependencies due to connections between trains and resource conflicts.

The train scheduling problem can essentially be divided into three distinct tasks: routing, conflict resolution and scheduling. First, trains are routed through a railway network. Second, parts of the network are assigned resources representing, for example, the physical tracks or junctions that can only be passed by a single train at once. Whenever several trains are routed through the same resource, they have to be serialized in order to avoid collisions. Finally, a schedule has to be created reflecting when the train arrives at certain points in its path. A valid schedule has to respect a variety of timing constraints, ranging from earliest and latest arrival times, traveling and waiting times, resource conflicts between trains, to connections between trains. After obtaining a valid routing and scheduling, the solution is evaluated regarding the delay of the trains and the quality of the paths they have taken.

We take up this challenge and show how to address real-world train scheduling with hybrid Answer Set Programming (ASP [5]). More specifically, we implement our approach with the hybrid ASP system *clingo*[DL] [4], an extension of *clingo* [3] with difference constraints. Our hybrid approach allows us to specifically account for the different types of constraints induced by routing, scheduling, and optimization. While we address routing and resource conflicts with regular ASP, we use difference constraints (over integers) to capture fine timings. Difference constraints are of the form $x - y \leq k$, where x and y are integer variables and k an integer constant, and constitute a subset of Integer Linear Programming that is solvable in polynomial time. Furthermore, encoding timing constraints for scheduling as difference constraints gives us the obvious advantage that integer variables are not subject to grounding.

Our hybrid ASP-based encoding for solving the train scheduling problem relies heavily on dedicated preprocessing techniques to reduce the problem size as well as additional constraints and domain-specific heuristics to reduce the search space and improve solving performance. This constitutes a significant improvement over the previous approach [1] that, while similar in principle, does not scale to the largest instances

⋆ Affiliated with Simon Fraser University, Canada, and Griffith University, Australia.

available. The preprocessing techniques mainly exploit redundancy in the resource distribution in the railway network. That is, we remove resources that pose constraints which are subsumed by other resources, and identify large areas for which a single decision suffices to serialize trains which are in conflict. We furthermore reduce the amount of integer variables and timing constraints by using a compressed representation of the railway network. The additional constraints restrict the search space in the hope of improving solving performance. We hereby rely on the fact that *clingo*[DL] consists of a Boolean search engine (*clingo*) and a dedicated difference logic propagator. While some atoms are purely Boolean, others have a semantics in terms of difference constraints. We transfer some knowledge represented in these difference constraints back to the logic program, thereby improving the search in the ASP part of the problem and leveraging *clingo*'s effective propagation techniques. Finally, a domain-specific heuristic is used that prefers sequences of trains in a way that reduces likelihood of conflicts.

We evaluate our train scheduling solution on 25 real-world instances crafted by domain experts from Swiss Federal Railways (SBB). The instances capture parts of the railway network between three Swiss cities, namely Zurich, Chur and Luzern, and vary in number of trains, size and depths of railway network and timing constraints. The biggest instances contain the whole railway network and up to 467 trains taken from long distance, regional, suburban and freight traffic between those three cities. Thus, we tackle instances with approximately six hours of the full train schedule on a railway network covering approximately 200 km. We show the vital importance of a deep understanding of the problem to enable effective preprocessing techniques and simplifications in the problem encoding. For instance, for those 25 instances, we were able to reduce the amount of resources by 34%, the number of decisions to serialize the trains by 92%, and the number of integer variables by 25% on average. This enables us to solve all available instances within minutes. Furthermore, the train schedules created were of high quality. Our approach found a train schedule without delay for all instances were this was possible, and otherwise, the quality of the results was sanctioned by SBB.

References

1. Abels, D., Jordi, J., Ostrowski, M., Schaub, T., Toletti, A., Wanko, P.: Train scheduling with hybrid ASP. In: Balduccini, M., Lierler, Y., Woltran, S. (eds.) Proceedings of the Fifteenth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'19). Lecture Notes in Artificial Intelligence, vol. 11481, pp. 3–17. Springer-Verlag (2019)
2. Caprara, A., Fischetti, M., Toth, P.: Modeling and solving the train timetabling problem. Operations Research **50**, 851–861 (10 2002)
3. Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T.: Multi-shot ASP solving with clingo. Theory and Practice of Logic Programming **19**(1), 27–82 (2019), `http://arxiv.org/abs/1705.09811`
4. Janhunen, T., Kaminski, R., Ostrowski, M., Schaub, T., Schellhorn, S., Wanko, P.: Clingo goes linear constraints over reals and integers. Theory and Practice of Logic Programming **17**(5-6), 872–888 (2017)
5. Lifschitz, V.: Answer set planning. In: de Schreye, D. (ed.) Proceedings of the International Conference on Logic Programming (ICLP'99). pp. 23–37. MIT Press (1999)