Towards an Optimization Pipeline for the Design of Train Control Systems with Hybrid Train Detection

Stefan Engels[1]
Chair for Design Automation, Technical University of Munich, Germany

Robert Wille
Chair for Design Automation, Technical University of Munich, Germany
Software Competence Center Hagenberg GmbH (SCCH), Austria

Abstract

Increasing the capacity of our railway infrastructure will become more and more essential in coping with the need for sustainable transportation. This can be achieved by intelligently implementing train control systems on specific railway networks. Methods that automate and optimize parts of this planning process are of great interest. For control systems based on hybrid train detection, such optimization tasks simultaneously involve routing and block layout generation. These tasks are already complex on their own; hence, a joint consideration often becomes infeasible. This work-in-progress paper proposes an idea to tackle the corresponding complexity. To this end, we present a pipeline that allows to *sequentially* handle corresponding optimization tasks in a less complex fashion while generating results that remain (close to) optimal. Results from an initial case study showcase that this approach is, indeed, promising. A prototypical implementation is included in the open-source *Munich Train Control Toolkit* available at `https://github.com/cda-tum/mtct`.

2012 ACM Subject Classification Applied computing → Transportation

Keywords and phrases ETCS, MILP, Design Automation, Hybrid Train Detection

Category Short paper

Supplementary Material *Software (Source Code)*: `https://github.com/cda-tum/mtct`

1 Introduction

The demand for railroads is increasing as sustainable transportation becomes more and more important. However, the capacity of existing railway infrastructure is limited. In addition to building new tracks, increasing the capacity of existing lines is crucial to satisfy the growing demand, e.g., by utilizing more efficient train control systems. For reasons of compatibility, international train control systems have been specified in a standardized fashion, e.g., through the *European Train Control System* (ETCS), *Chinese Train Control System* (CTCT), and *Positive Train Control* (PTC) [10]. Each of them comes in different variants. More sophisticated levels allow shorter train following times (headways), and by this, increase the capacity while, at the same time, maintaining a high level of safety.

During the planning process, design choices have to be made that might influence the outcome. Much of the planning relies on manual processes and the work experience of the involved personnel, an expensive and error-prone endeavor. Automating and optimizing specific steps to reduce these costs and ensure the best operational outcome is of great interest [3]. Accordingly, methods have been developed that optimize train operation, such as creating timetables and routing [2]. Optimal routing has recently been considered for trains operating under so-called *Moving Block* [12, 9, 6]. In addition, classical control systems rely on separating the network into blocks, requiring physical hardware for each of them.

[1] Corresponding author

Generating optimal block layouts focused on optimizing general performance indicators independent of specific schedules [7, 13].

Alternatively, modern specifications relying on *Hybrid Train Detection* allow the introduction of purely virtual (sub-)sections. At least in theory, they allow the flexible adjustment of the layout, leading to new design objectives to be optimized [4]. To the best of our knowledge, algorithms tailored to hybrid train detection have first been considered in [14, 11]. While these initial solutions neglect significant modeling details, a more accurate solution method has been introduced in [5]. Unfortunately, these solutions do not scale well. A primary reason for that might be because they combine multiple complex objectives in one task.

In this work-in-progress paper, we propose an optimization pipeline that considers the resulting sub-tasks sequentially. This allows for solving these problems in a less complex fashion while still generating (close to) optimal solutions. Results obtained by initial case studies confirm these premises. A prototypical implementation of the proposed idea is available open-source as part of the *Munich Train Control Toolkit* at `https://github.com/cda-tum/mtct`.

The remainder of this work is structured as follows. Sec. 2 summarizes the relevant background, namely principals of train control systems in Sec. 2.1 and resulting design tasks in Sec. 2.2. Afterward, Sec. 3 describes the proposed optimization pipeline and constitutes the main contribution of this work in progress. A short case study in Sec. 4 demonstrates that this approach is promising, and Sec. 5 concludes this paper.

2 Background

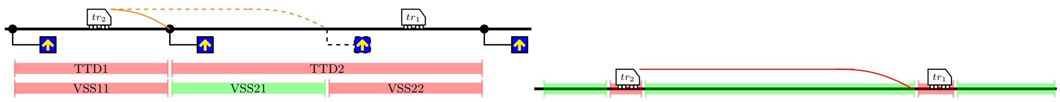
2.1 Moving Block and Hybrid Train Detection

Due to long braking distances, it is not feasible for trains to operate on sight. Instead, signaling systems are implemented to ensure safe operation. Classical control systems divide the network into fixed block sections. A train cannot enter a section that is already occupied by another train. Physical *Trackside Train Detection* (TTD) hardware (e.g., axle counters) at the section borders is used to detect the position of trains.

Modern specifications require trains to report their exact positions with certainty. By doing so, TTD hardware is no longer needed. Ideally, a *Moving Block* control system can be implemented in which trains follow each other at their (absolute) braking distance (similar to car traffic) without the need to define block sections.

However, such a system might impose practical problems, especially on lines with mixed traffic where some trains might not be equipped with train integrity monitoring systems to safely report their positions to the control system [1]. As a "compromise" *Hybrid Train Detection* has been specified. For this, existing TTD sections are separated into smaller *Virtual Subsections* (VSS) without the need for additional hardware. This allows for shorter headway times. At the same time, the original TTD sections serve as a backup.

▶ **Example 1.** Consider the scenario shown in Fig. 1 with two trains. In Fig. 1a, the network is divided into block sections TTD1 and TTD2. Because TTD2 is occupied by train tr_1, the following train tr_2 can only advance to the end of TTD1 (solid orange line). Using hybrid train detection, TTD2 might be separated into two virtual subsections. Because of this, train tr_2 is authorized to move until the end of VSS21 (dashed orange line). On the contrary, there are no block sections under moving block control. In Fig. 1b, train tr_2 can advance up to the end of train tr_1 minus a small safety buffer.



(a) Hybrid Train Detection **(b)** Moving Block Control

Figure 1 Schematic drawings of various ETCS levels

2.2 Resulting Design Tasks

In the following, we focus on control systems with hybrid train detection. At least in theory, the virtual block layout can be chosen flexibly. This allows, for the first time, the VSS to be adjusted depending on a specific train schedule. Hence, new design tasks result that utilize this additional degree of freedom [4].

In general, the question is how to separate a given layout into VSS sections to obtain the best operational outcome. Some objectives might be determining a minimal number of subsections to make a previously infeasible timetable possible, minimizing runtimes using a predefined number of VSS, or maximizing the throughput of additional (e.g., freight) trains. Nevertheless, the focus is to achieve this operational benefit by intelligently defining a (virtual) block layout. It can be shown that all of those tasks are NP-hard, even if the routing aspect is fixed. For more details (which are out of the scope of this paper), we refer to previous work [4].

3 Towards an Optimization Pipeline

The design tasks reviewed above have in common that they consist of two main parts, namely train routing and placement of VSS sections. At the same time, they affect each other. The feasibility of a routing depends on the chosen VSS layout, and the necessity of subsections depends on the routing. Both tasks are NP-hard already on their own; hence, a joint consideration often makes solving these tasks infeasible. To cope with the corresponding complexity, we propose using an optimization pipeline to solve the two aspects sequentially while still getting (close to) optimal solutions.

To this end, we use the following observation: Moving block control can be seen as classic block signaling where each section is infinitesimally small. In particular, a routing under moving block is likely feasible if a sufficient amount of block sections is defined. Nevertheless, finding such a routing is easier on moving block systems because the optimization model does not have to generate a (virtual) block layout simultaneously. Moreover, there is already promising work for time-optimal routing on moving block controlled networks [12, 9, 6]. Thus, we propose a two-step approach, quasi an “optimization pipeline”:

1. The trains are routed as if they were to operate under moving block control using the approaches mentioned above.
2. The routing obtained from Step 1 is fixed, and VSS sections are then generated based on this assumption.

Based on the above reasoning, we conjecture that Step 1 will likely choose the same routes as a combined optimization model would have produced (even though there is no theoretical guarantee). In this case, Step 2 outputs the same optimal solution, but the sequential approach substantially reduces the complexity compared to the joint consideration.

In order to implement that idea, we can utilize the approach proposed in [5], which is based on a *Mixed Integer Linear Program* (MILP) that (in principle) jointly models Steps 1 and 2. At the same time, this approach offers an option to additionally constrain trains

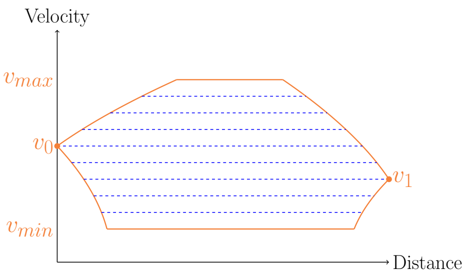


Figure 2 Velocity Profiles

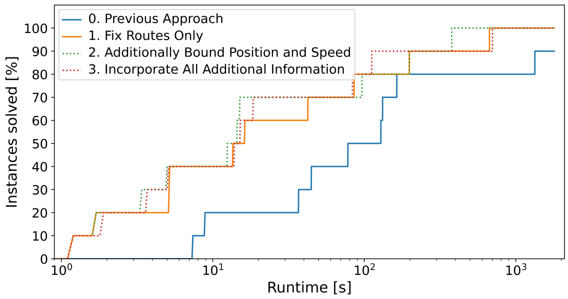


Figure 3 Experimental Evaluation

to use predefined routes; hence, it can be used in Step 2. In [5], it was already shown that this option is beneficial under the assumption that these routes are available "for free". Unfortunately, the question of how (and at what additional cost) to obtain this information has not been investigated before.

Given a solution obtained by Step 1, we can extract the used edges and even more information to narrow down the search space and guide the optimization algorithm in Step 2. To this end, observe that the approaches in [12, 9, 6] (which can be used for Step 1) only model the times and velocities when entering and leaving specific track segments. Say a train enters a given track segment with velocity v_0 and exits at speed v_1. The intermediate positions and velocities can only be interpolated and might not be uniquely defined. To this end, consider Fig. 2. The orange lines denote the (two) extreme velocity profiles that might occur on the track segment and correspond to the min- and max-time profiles in [12]. If we assume that trains only accelerate/decelerate close to the ends and travel at constant line speed in between, we obtain the dashed blue profiles. Doing so allows assigning a well-defined approximate velocity profile for any possible timing. Keeping this in mind, we concretize:

- *Fix Train Orders:* To ensure train separation, every formulation has to somehow model in which order trains traverse specific track segments. This usually adds complexity to the underlying model. However, we can extract those train orders from a Step 1 solution and fix it for Step 2, which reduces the feasible region, prunes the search space, and might lead to faster solving times.
- *Fix Train Positions and Velocities:* Solving Step 2 with the method proposed in [5], position and velocity are modeled at a discretized set of time points. Using the above observation, we can extract lower and upper bounds at every time point using the extreme profiles and add this information as constraints. Theoretically, this could cut off the optimal solution. However, we conjecture this to be unlikely due to the aforementioned reasoning that train routes are likely equivalent under both controlling principles. To be on the safe side, we add an additional tolerance of the distance traveled in one time step to reduce a possibly negative effect of discretization errors.
- *Hint Approximate Train Positions:* Using the specific timings from Step 1, we can map precisely one of the approximate velocity profiles mentioned above. At any time, we can easily calculate exact positions and velocities; however, the actual trajectory might differ. Because of this, we are not sure enough to add this information using equality constraints. However, we can pass these as a variable hint to the MILP solver, indicating that we believe the optimal solution is close to that approximated trajectory. Some solvers, e.g., Gurobi, can use this information to speed up the optimization process by adapting heuristics and branching decisions [8].

Overall, the above ideas allow for a sequential (rather than joint) consideration of the corresponding design aspects. This may provide the path towards efficiently handling those design tasks while still maintaining (close to) optimal results.

4 Case Study: Generation of Minimal VSS Layouts

To preliminary evaluate the proposed approach, we tested it on one of the more straightforward design tasks: generating minimal VSS layouts to make a specific timetable possible. Our implementation is based on [12, 6] for Step 1 and on [5] for Step 2, which was extended to include the additional information described in Sec. 3. The code is available as an open-source implementation on GitHub at `https://github.com/cda-tum/mtct`. We used the same benchmark as in [5] and an Intel(R) Xeon(R) W-1370P system using a 3.60GHz CPU (8 cores) and 128GB RAM running Ubuntu 20.04 and Gurobi version 11.0.2 [8].

The resulting runtimes[2] are plotted in Fig. 3 (see previous page). The x-axis shows timeouts in seconds, whereas the y-axis represents the percentage of instances solved within the given time or faster. By design, all lines are monotonously increasing, and being on the left/top is considered to be "better". For comparison, we solved the benchmark using the previous approach [5], which jointly considers all decision aspects. Additionally, all instances were solved using the proposed sequential approach. In Step 2, three variants have been considered, namely,

1. only fixing the routes (i.e., used edges) without any additional information,
2. additionally, constraining position and speed at every time step, and,
3. finally, incorporating all information described in Sec. 3.

The depicted runtimes are total times, i.e., the sum of both Step 1 and Step 2, as well as model creation times.

Overall, these initial case studies clearly show the benefit of the proposed pipeline. Even though this includes two optimization steps, it is consistently and significantly faster than the previous approach. On the other hand, we can observe that most of this improvement is due to the separation of routing and VSS placement (orange solid line). The additional information described in Sec. 3 (dotted lines) seem to further improve the runtime in most cases; however, the difference is not as big. Moreover, it is not immediately apparent which of the described information are best to be included. Still, we can conclude that adding all additional information is never a bad idea. It is just that, in some cases, almost the entire runtime benefit might be due to fixing edges and not due to additional information. Finding the best set of parameters within the proposed optimization pipeline is left to future research.

5 Conclusions

With this work, we proposed a step towards an efficient optimization pipeline for designing railway networks based on train control with hybrid train detection. We demonstrate how routing information from a different control principle, namely, moving block, can significantly simplify the optimization model. Even though the resulting approach consists of two optimization steps, the runtime is significantly reduced. The resulting prototypical

[2] We do not show the objective values in detail. The optimal solution was returned independently of the chosen algorithm in all but one instance. In this one case, fixing position bounds led to an increase in VSS sections from 6 to 13, even though the route itself was optimal. All other parameters did not have any effect on the objective. Overall, this shows that the proposed approach often yields (close-to) optimal results (even though there is no theoretical guarantee).

implementation is available open-source and included within the Munich Train Control Toolkit at https://github.com/cda-tum/mtct. Future work focuses on a more sophisticated implementation and evaluation of the idea presented in this paper. This includes the extension to more complex design tasks and objectives as well as the development of algorithms tailored to this framework to use more information on the relevant problem structure already at the core of their development.

References

1 Maarten Bartholomeus, Laura Arenas, Roman Treydel, Francois Hausmann, Nobert Geduhn, and Antoine Bossy. ERTMS Hybrid Level 3. *SIGNAL + DRAHT (110) 1+2*, 2018. URL: https://www.eurailpress.de/fileadmin/user_upload/SD_1_2-2018_Bartholomaeus_ua.pdf.

2 Ralf Borndörfer, Torsten Klug, Leonardo Lamorgese, Carlo Mannino, Markus Reuther, and Thomas Schlechte, editors. *Handbook of Optimization in the Railway Industry*. 2018. doi:10.1007/978-3-319-72153-8.

3 Stefan Dillmann and Reiner Hähnle. Automated planning of ETCS tracks. In *Reliability, Safety, and Security of Railway Systems. Modelling, Analysis, Verification, and Certification*. 2019. doi:10.1007/978-3-030-18744-6_5.

4 Stefan Engels, Tom Peham, Judith Przigoda, Nils Przigoda, and Robert Wille. Design tasks and their complexity for the European Train Control System with hybrid train detection. 2024. arXiv:2308.02572.

5 Stefan Engels, Tom Peham, and Robert Wille. A symbolic design method for ETCS Hybrid Level 3 at different degrees of accuracy. In *23rd Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS)*, 2023. doi:10.4230/OASICS.ATMOS.2023.6.

6 Stefan Engels and Robert Wille. Comparing lazy constraint selection strategies in train routing with moving block control. 2024. arXiv:2405.18977.

7 D.C. Gill and C.J. Goodman. Computer-based optimisation techniques for mass transit railway signalling design. *IEE Proceedings B Electric Power Applications*, 139(3), 1992. doi:10.1049/ip-b.1992.0031.

8 Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023. URL: https://www.gurobi.com.

9 Torsten Klug, Markus Reuther, and Thomas Schlechte. Does laziness pay off? - a lazy-constraint approach to timetabling. In *22nd Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS)*, 2022. doi:10.4230/OASIcs.ATMOS.2022.11.

10 Jörn Pachl. *Railway Signalling Principles: Edition 2.0*. 2021. doi:10.24355/dbbs.084-202110181429-0.

11 Tom Peham, Judith Przigoda, Nils Przigoda, and Robert Wille. Optimal railway routing using virtual subsections. In *Reliability, Safety, and Security of Railway Systems. Modelling, Analysis, Verification, and Certification*. 2022. doi:10.1007/978-3-031-05814-1_5.

12 Thomas Schlechte, Ralf Borndörfer, Jonas Denißen, Simon Heller, Torsten Klug, Michael Küpper, Niels Lindner, Markus Reuther, Andreas Söhlke, and William Steadman. Timetable optimization for a moving block system. *Journal of Rail Transport Planning & Management*, 22, 2022. doi:10.1016/j.jrtpm.2022.100315.

13 Valeria Vignali, Federico Cuppi, Claudio Lantieri, Nicola Dimola, Tomaso Galasso, and Luca Rapagnà. A methodology for the design of sections block length on ETCS L2 railway networks. *Journal of Rail Transport Planning & Management*, 13, 2020. doi:10.1016/j.jrtpm.2019.100160.

14 Robert Wille, Tom Peham, Judith Przigoda, and Nils Przigoda. Towards automatic design and verification for Level 3 of the European Train Control System. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2021. doi:10.23919/date51398.2021.9473935.