Evaluation of Dispatching Algorithms in a Railway Lab Under Realistic Conditions

Von der Fakultät für Bauingenieurwesen

der Rheinisch-Westfälischen Technischen Hochschule Aachen

zur Erlangung des akademischen Grades

einer Doktorin der Naturwissenschaften

genehmigte Dissertation

vorgelegt von

Alexandra Angelika Liebhold, geb. Grub

Berichter: Univ.-Prof. Dr.-Ing. Nils Nießen

Prof. Takafumi Koseki, PhD

Tag der mündlichen Prüfung: 11. November 2024

Contents

Abstract

As the traffic volume operated via rail constantly increases, delayed trains can severely impact the punctuality of other trains within the network. Train dispatching aims at reducing or avoiding the extent of such secondary delays while increasing the overall punctuality by retiming, reordering or rerouting of trains in the network. Today, dispatching is mostly performed manually by experienced human dispatchers. Over the past years, several decision support systems, which provide recommendations to the human dispatchers, have been developed and successfully implemented during real operation. Fully automated dispatching systems are still rarely applied.

Prior to the commissioning of new dispatching systems, extensive testing is required. Theoretical algorithms are first tested for their general feasibility during computer-based case studies and simulations, which are often based on real-world data. After successful simulations, new dispatching systems are normally assessed for their practical usability during field tests on small segments of the real network. However, field tests are extremely costly and long planning horizons are involved. Railway labs can provide an intermediate level testing environment for new dispatching systems under more realistic conditions than pure computer simulations. Tests performed in a railway lab are also more cost-effective than tests performed directly on the real field, while not impeding any real-world operations. However, only few dispatching systems were tested in railway labs so far and previous tests did not incorporate fully automatic operation and thus dispatching decisions were not implemented automatically.

To enable automatic real-time train operation under realistic conditions and incorporate dispatching decisions from an external dispatching system during operation, a railway lab control software was extended, enabling automatic train control and train route setting based on a given timetable. Train speed profiles and speed curves were modeled and implemented to allow for realistic driving characteristics and running times.

Furthermore, a communication interface to connect an external dispatching system was developed. Via the interface, constant information on the current operational situation are provided to the dispatching system. Dispatching measures are forwarded from the dispatching system to the railway lab control software and implemented during operation accordingly. Predefined disturbance scenarios can be imported into the control software and incorporated during operation.

To further allow for energy-efficient train driving, a novel method for the energy optimization of single train runs, based on recommended time corridors for the passing of block signals, was developed and included in the railway lab control software.

The developed software systems provide a realistic testing environment for dispatching algorithms and strategies. During a case study, the validity of the developed testing environment was confirmed. The railway lab control software was used to control real-time train operations on the ELVA, the Railway Signalling Lab of RWTH Aachen University, while the external dispatching system OptDis, which is integrated in the software LUKS®, was connected. As disturbance scenarios can easily be reconstructed, the developed testing environment enables the direct comparison of different dispatching strategies. Several

disturbance scenarios were constructed and dispatching was performed both automatically by OptDis and manually by an experienced human dispatcher for all test cases.

The evaluations show that the developed software systems provide a realistic testing framework for different dispatching strategies, which can serve as an intermediate level testing environment between computer simulations and real field tests. Moreover, the obtained results suggest significant benefits of automated dispatching systems over manual dispatching if operational situations are very complex and decisions must be made quickly. The energy-saving driving method was also tested in a case study on the ELVA network and promising results were obtained.

Kurzfassung

Der Anteil des Verkehrs, der über die Schiene abgewickelt wird, nimmt stetig zu, sodass verspätete Züge die Pünktlichkeit anderer Züge im Netzwerk massiv beeinträchtigen können. Ziel der Zugdisposition ist die Reduktion oder Vermeidung solcher Folgeverspätungen und die Steigerung der Gesamtpünktlichkeit durch Zeitanpassungen, Reihenfolgeänderungen oder räumliche Umleitungen. Die Disposition wird heutzutage meist manuell von erfahrenen Disponenten durchgeführt. Im Laufe der letzten Jahre wurden einige Dispositionssysteme entwickelt und erfolgreich eingesetzt, die Disponenten durch Entscheidungsempfehlungen unterstützen. Vollautomatisierte Dispositionssysteme kommen derzeit nur sehr selten zum Einsatz.

Vor der Inbetriebnahme neuer Dispositionssysteme sind ausgiebige Testungen erforderlich. Theoretische Algorithmen werden in computerbasierten Fallstudien und Simulationen, die oft auf realen Daten basieren, auf ihre generelle Eignung getestet. Nach erfolgreichen Computersimulationen werden neue Dispositionssysteme normalerweise in Feldtests auf kleinen Segmenten des realen Netzwerks auf ihre Praxistauglichkeit sowie Anwenderfreundlichkeit getestet. Feldtests sind jedoch sehr kostspielig und erfordern lange Planungshorizonte. Eisenbahnlabore können als Testumgebung für neue Dispositionssysteme eine Zwischenebene darstellen, die mehr Realitätsnähe als reine Computersimulationen bietet. Neue Dispositionsverfahren zunächst in Eisenbahnlaboren zu erproben ist zudem wirtschaftlicher, als direkt im Anschluss an Computersimulationen die ersten Feldtests durchzuführen und darüber hinaus wird eine Beeinträchtigung des realen Betriebs verhindert. In der Vergangenheit wurden jedoch nur wenige Dispositionssysteme in Eisenbahnlaboren getestet und die durchgeführten Tests beinhalteten keinen vollautomatischen Betrieb und somit keine automatische Umsetzung der Dispositionsentscheidungen.

Um einen automatischen Echtzeitbetrieb unter realitätsnahen Bedingungen zu ermöglichen und Dispositionsentscheidungen von einem externen Dispositionssystem umzusetzen, wurde eine Steuerungssoftware für Eisenbahnlabore weiterentwickelt, sodass die Steuerung der Züge und das Einstellen der Fahrstraßen automatisch auf Basis eines gegebenen Fahrplans möglich ist. Die Geschwindigkeitsprofile der Züge und die Fahrkurven wurden so modelliert und umgesetzt, dass Fahrverhalten und Fahrzeiten realitätsnah abgebildet werden.

Weiterhin wurde eine Schnittstelle zur Kommunikation mit einem externen Dispositionssystem entwickelt. Über die Schnittstelle werden Informationen über die aktuelle Betriebssituation an das Dispositionssystem weitergeleitet. Die Dispositionsmaßnahmen werden wiederum vom Dispositionssystem an die Steuerungssoftware des Eisenbahnlabors übermittelt und von dieser verarbeitet und während des Betriebs umgesetzt.

Um außerdem energiesparendes Fahren zu ermöglichen, wurde eine neue Methode entwickelt und in die Steuerungssoftware integriert, die den Energieverbrauch einer Zugfahrt auf Basis von empfohlenen Zeitkorridoren für die Überfahrt von Blocksignalen optimiert.

Die entwickelten Softwaresysteme bieten eine realitätsnahe Testumgebung für Dispositionsalgorithmen und -strategien. In einer Fallstudie wurde die Anwendbarkeit der entwickelten Testumgebung bestätigt. Die Steuerungssoftware wurde im Echtzeitbetrieb an der ELVA, der Eisenbahntechnischen Lehr- und Versuchsanlage der RWTH Aachen University,

angewendet und das externe Dispositionssystem OptDis, das in die Software LUKS® integriert ist, wurde angebunden. Da Störungsszenarien leicht rekonstruierbar sind, ermöglicht die entwickelte Testumgebung den direkten Vergleich verschiedener Dispositionsstrategien. Verschiedene Störungsszenarien wurden konstruiert und die Disposition wurde für alle Testfälle einerseits automatisch durch OptDis durchgeführt und andererseits durch einen erfahrenen menschlichen Disponenten.

Die Auswertungen zeigen, dass die entwickelten Softwaresysteme eine realitätsnahe Testumgebung als Zwischenebene zwischen reinen Computersimulationen und Feldtests während der Erprobung neuer Dispositionssysteme bieten. Weiterhin lassen die gewonnenen Ergebnisse darauf schließen, dass ein erheblicher Nutzen durch den Einsatz automatisierter Dispositionssysteme, im Vergleich zur manuellen Disposition, zu erwarten ist, wenn die Betriebssituationen sehr komplex sind und Entscheidungen innerhalb kurzer Zeit getroffen werden müssen. Die vorgestellte Methode für energiesparendes Fahren wurde ebenfalls in einer Fallstudie an der ELVA getestet und vielversprechende Ergebnisse wurden erzielt.

1 Introduction

As the traffic volume operated via rail is constantly increasing, railway networks are often operated at their capacity limit. Train delays naturally occur due to internal or external influences and delay propagation to other trains in the network is inevitable.

Train dispatching is essential for reducing the amount of delay propagation in a network area during real-time operation. There are a variety of possible dispatching measures, such as letting trains run faster or slower, changing dwell times, omitting or adding stops or changing the scheduled train routes through the network. The aim is usually the reduction of overall delays in the network area. Nowadays, dispatching is mostly performed by human dispatchers. As operational situations are becoming far more complex and dispatchers need to find resolution strategies fast, computer-based dispatching systems were developed to support the dispatchers. Dispatching systems normally visualize the operational situation, based on information received from the vehicles or infrastructure elements, such as train positions, track occupation or the state of switches and signals. Based on these information, advanced systems can predict train movements and future conflicts between the train runs. Many dispatching systems that are in use today also provide decision support to the human dispatcher. Recommended conflict resolution strategies are provided which the dispatcher can choose to implement during operation. The recommendations can be derived from predefined dispatching rules or complex mathematical algorithms can be integrated into the decision support system, such as mathematical optimization or machine learning approaches. Despite numerous methods and algorithms proposed in the scientific literature, completely automated dispatching systems are very rarely applied during real operation.

Prior to the commissioning and roll-out of new dispatching systems, extensive testing is required, in order not to cause additional delays and impede smooth traffic operation during the testing phase. Developed dispatching algorithms are normally tested during purely computer-based case studies and simulations first, while the input is often based on historical operational data or realistic conditions are simulated via specific simulation environments.

If a railway infrastructure manager decides to apply a new system after promising results were obtained during extensive computer simulations, the next step normally involves field tests performed on a small network segment or a closed system. Field tests still involve immense costs, as they require upgrading and adjusting the existing systems and possibly the vehicles and infrastructure.

Railway labs can serve as an excellent intermediate level testing environment for new dispatching systems. Train operations can be modeled realistically and human operators experience operational situations far more realistically than during pure computer simulations. Thus, rail human factors can be evaluated and accounted for. Disturbances can be manually defined, enabling the exact reconstruction of operational situations to achieve equal setups for the comparison of different dispatching strategies. Furthermore, operation in a railway lab does not disturb or pose any disadvantages to real operation during the testing phase. When testing new dispatching systems in a railway lab, several feedback loops can be performed prior to the first real-world field tests, which can immensely reduce the involved costs. So far, only few research projects assessing the usability of new dispatching systems or

interlocking system interfaces and evaluating rail human factors have been carried out in railway labs. In these projects, however, operation was not performed automatically and thus dispatching strategies generated by the dispatching system were not automatically implemented during operation.

To investigate the potential of railway labs as a testing environment for dispatching algorithms, a generic communication interface between a dispatching system and a railway lab was designed and implemented. An existing fundamental railway lab control software for a railway lab was extended to allow for an automatic operation under realistic conditions, while running according to the dispatching recommendations received from the dispatching system.

Prior to operation, a conflict-free timetable is provided to both the dispatching system and the railway lab control software. The railway lab control software constantly provides real-time information on the current operational situation to the dispatching system which predicts future train movements, based on the predefined timetable and the received information. It is assumed that the dispatching system performs automatic conflict detection and dispatching measures can be transmitted to the railway lab control software. The control software then processes the changes and applies them during operation. Vehicles are automatically controlled via speed levels and run according to individual realistically modeled speed profiles. The trains run according to a predefined timetable and delay data can be imported and considered during operation. Train routes are automatically set for the trains.

Additionally, an energy-saving driving method is proposed, which can be performed after the dispatching process for single trains. The method is based on recommended time corridors for the passing of block signals and was implemented within the railway lab control software. By applying energy optimization after the dispatching process and ensuring that the arrival at the next scheduled stop is not further delayed, its application does not increase the overall delay in the network and does therefore not negatively influence customer satisfaction.

To evaluate the correctness and practical applicability of the developed testing environment, a case study was performed on the ELVA, the Railway Signalling Lab of RWTH Aachen University, and the dispatching system OptDis [1, 2], which is integrated in the software platform LUKS® [3, 4], was connected. Several timetable concepts and delay data were defined, incorporating test cases with low and high traffic volume and a total of six different train classes. All test cases were operated on the ELVA network under automatic operation with both automatic dispatching through OptDis and manual dispatching by an experienced human dispatcher, where OptDis served as user interface for the visualization of train conflicts and for editing manual dispatching measures. Operational data, such as arrival and departure times, as well as train routes taken, were recorded during operation. The data generated during operation under automatic dispatching through OptDis and manual dispatching by the experienced human dispatcher were compared and evaluated, in order to quantify the potential gain in punctuality obtained by implementing an automated dispatching system such as OptDis. Additionally, manually generated conflict situations were used as input to assess and compare the different conflict resolution approaches in specific situations during automatic and manual dispatching. For the proposed energy-saving driving method, a test case was defined and evaluations were performed on the ELVA network.

The thesis is structured as follows. Chapter 2 first gives a general overview of different kinds of dispatching strategies and dispatching systems in section 2.1. Then, a selection of dispatching systems already applied during real operation is presented. As the emphasis of this thesis is put on the testing of dispatching systems, section 2.2 then focuses on ways to test new dispatching systems. A selection of projects is discussed in which dispatching systems were tested during computer simulations, field tests and pilot projects, as well as in railway labs. Section 2.3 presents the state-of-the-art in energy-saving driving methods, focusing on approaches combining energy optimization and train rescheduling. In chapter 3, the fundamentals of train dispatching and railway timetabling, which are relevant for this thesis, are presented.

Chapter 4 describes the newly developed communication interface between a dispatching system and a railway lab. The general requirements of such a communication interface are discussed in section 4.1. Then, in sections 4.1 and 4.2, detailed descriptions of the message exchange from the railway lab to the dispatching system and from the dispatching system to the railway lab are given respectively.

In chapter 5, the requirements both for a physical railway lab and for the control software to serve as a realistic testing environment for real-time dispatching, are discussed in sections 5.1 and 5.2 respectively and an example of an already existing and presently applied railway lab control software is briefly discussed in section 5.3.

The newly developed algorithms for automatic train operation in a railway lab are introduced and described in detail in chapter 6, with special emphasis on their integration into a railway lab control software.

In chapter 7, a new method for energy-saving driving is proposed, which is based on recommended time corridors for the passing of block signals. The method can optionally be applied after the dispatching process and was also integrated into the railway lab control software.

Chapter 8 describes the testing environment that was selected for the case study. First, the ELVA railway lab is described in section 8.1. Section 8.2 then explains how the ELVA network was modeled via a microscopic infrastructure model and the generation of conflict-free timetables is depicted in section 8.3. The dispatching system and its underlying models and optimization approach is described in section 8.4. Finally, section 8.5 discusses the possibilities and limitations of the testing environment selected for the case study. Both the developed software systems for the testing of real-time dispatching algorithms under realistic conditions in a railway lab and the proposed energy-saving driving method were tested and evaluated on the ELVA network. In chapter 9, the test cases defined for the two case studies are presented and the obtained results of both case studies are then discussed.

Lastly, the main contributions and results of this thesis are summarized in chapter 10 and recommendations for future research and the future application of automated dispatching systems are made.

Parts of the work presented in this thesis were already published prior to its submission. In [5] and [6], the developed railway lab control software and the communication interface between

a railway lab and a dispatching system were presented. The energy-saving driving method and the case study performed to evaluate its performance were described in [7].

2 Literature Review and State-of-the-art

This chapter is divided into four sections. Section 2.1 defines train dispatching and dispatching systems in general, gives an overview of the scientific literature and state-of-the-art in the field of train dispatching in 2.1.1 and presents current applications of dispatching systems during real operation in 2.1.2. Section 2.2 then focuses on existing testing environments for dispatching systems and algorithms and in particular the different levels of testing, ranging from case studies based on pure computer simulations in 2.2.1 to field tests in which new systems are evaluated and further optimized during pilot projects on segments of the real railway network in 2.2.2. Special emphasis is also put on the possibilities for the testing of new dispatching systems in railway labs and in particular on historical projects in 2.2.3. As energy-saving driving becomes increasingly important, a variety of approaches aiming at energy optimization during real-time operation were proposed in the scientific literature. Section 2.3 presents the state-of-the-art in energy-saving driving techniques with special emphasis on approaches applied in combination with train rescheduling. Section 2.4 then describes the scientific contribution of this work and outlines the applied methodological research approaches.

2.1 Train Dispatching and Dispatching Systems

During real-time train operation, unexpected events and disturbances cause trains to be delayed. In particular in complex and dense areas with a high traffic volume, knock-on delays likely propagate to other trains in the network. For this reason, train dispatching, also referred to as train rescheduling, is essential to ensure smooth operation and limit the extent of delay propagation. Train dispatching includes the monitoring of train operations, detecting conflicts and solving conflicts through retiming, reordering and rerouting of trains [8].

Retiming refers to the change of scheduled arrival and departure times, including changes in dwell times and the omitting and adding of stops. Train runs can be bent, i.e. running times for a certain line segment are either increased or decreased by a certain amount of time [9]. In case the train shall run faster, the running time reserves can be utilized. Bending train runs to run slower makes sense if future occupation conflicts with other trains can be avoided if the train reaches a certain network section late. Also, energy-saving effects can be observed by letting trains run at lower speeds if future block sections are predicted to be occupied. Note that bending is only possible if the information on the new recommended speed can be transmitted to the train driver who then adjusts the speed accordingly or if trains are controlled automatically. Additionally, train runs can be terminated early, cancelled completely or short turn, i.e. the train is running back in the opposite direction before reaching its final destination [10]. Early termination and short turning are in particular suitable for passenger trains when the next available train does not impose significantly long waiting times on the passengers.

Reordering refers to the change of the order in which trains pass junctions or enter a certain network area [11]. Often, high-speed trains with a high priority shall run first and slow trains then follow after the minimum headway time (see 3.4) has passed.

Rerouting refers to changes in the planned train routes, e.g. changes of destination tracks in stations or running on the opposite track. Occasionally, rerouting includes major changes to the train route, for example if whole line segments are currently blocked. Such major route changes are rather common for freight trains, as they are normally not required to stick to a certain route, as long as the final destination is reached [12].

Normally, dispatching is considered as a multi-objective problem. In [13], the German railway infrastructure manager DB Netz AG defines the four main objectives of dispatching as the re-establishing of regular scheduled operation, the consideration of passenger connections, the minimization of the overall delay and the maximization of network capacity utilization. However, each railway undertaking can have individual objectives and priorities. In general, customer satisfaction is considered to be most important [14, 15]. Thus, the majority of methods primarily aims at maximizing punctuality during the dispatching process.

Dispatching systems contain algorithms or predefined contingency plans [16] which support human dispatchers or are applied during automatic dispatching. Information on the current operational situation, such as train positions and track occupation are required for dispatching systems to represent the actual train operation as close to reality as possible. These information originate from messages received from the vehicles or infrastructure elements. Normally, dispatching systems provide visual interfaces for the human operator to monitor the current operational situation. Often, future train movements are predicted while considering all available information on existing delays. Within dispatching systems, both the infrastructure and rolling stock need to be modeled precisely, in order to obtain reliable predictions. Many state-of-the-art dispatching systems can detect occupation and connection conflicts and provide decision support to the human dispatcher by recommending conflict resolution strategies. Resolution strategies are generated either based on predefined dispatching rules or using optimization algorithms which can either solve the problem heuristically, i.e. the solutions are not guaranteed to be optimal, or optimal solutions can be found using specific solver software. An overview of train dispatching algorithms in the scientific literature is given in section 2.1.1. Currently, only few completely automated dispatching systems exist (see 2.1.2).

2.1.1 Train Dispatching in the Scientific Literature

Over the past decades, numerous methods and algorithms for train dispatching were proposed in the scientific literature. These approaches mostly include mathematical optimization techniques where the optimization problem is modeled via linear constraints and the best solution with respect to an objective function is determined by a solver software. D'Ariano developed the system ROMA (Railway traffic Optimization by Means of Alternative graphs) where the dispatching problem is viewed as an instance of the job shop scheduling problem and formulated by using alternative graphs [17, 18]. Törnquist and Persson describe a mathematical programming model for an n-tracked network, where n is the number of parallel tracks, and propose a heuristic solution approach [19]. Zhu and Goverde present a stochastic optimization approach to model the unknown duration of disturbances [20].

A variety of algorithms also considers passenger satisfaction as objective of the optimization problem during train dispatching. An approach to solve passenger-orientated train rescheduling was first proposed by Schöbel through an integer linear programming formulation based on a space-time network [21]. Kanai et al. describe a train dispatching algorithm aiming at minimizing passenger dissatisfaction [22]. They apply a train traffic simulator and a passenger flow simulator parallelly to predict both future train movements and the number of passengers boarding and leaving trains. The optimization is then achieved by applying a tabu search algorithm. Binder et al. propose a multi-objective optimization method which simultaneously minimizes passenger inconvenience, operational costs and the deviation from the original timetable [23]. Zhu et al. present a mixed-integer linear programming model which aims at increasing passenger satisfaction by minimizing the in-vehicle times, waiting times at stations and the number of transfers [10]. The majority of approaches assume that passengers will take exactly the connections they previously planned to take and thus wait a complete cycle time for the next connection. However, in practice this is normally not the case, as passengers might take an alternative faster connection leading to their destination station. Therefore, Dollevoet et al. propose an event-activity network model and formulate the dispatching problem as an integer linear program to also consider the rerouting of passengers [24]. The origin and destination station of passengers are fixed, however, the lines and intermediate stations along their trip can vary to minimize overall running times. A detailed review on scientific approaches for passenger-orientated train dispatching can be found in [25].

Many algorithms in the scientific literature specifically focus on train dispatching for certain network structures or certain operational cases. As single line sections can be seen as the building blocks of any railway network and most railway networks contain sections with single-track lines between stations, several approaches investigate single-track railway lines. Higgins et al. propose a mathematical programming model for train dispatching on a single-track line that is solved via a branch-and-bound algorithm [26]. Gafarov et al. present several train dispatching algorithms for special cases of single-track lines between two stations by modeling the problem as a single-machine equal-processing-time scheduling problem [27]. Tamannaei et al. describe a mixed-integer linear programming formulation for the train dispatching problem on double-track lines which is solved by a branch-and-bound algorithm [28]. Louwerse and Huisman propose an integer programming model for double-track railway lines in case of partial or complete line blockages, i.e. all tracks of a network segment are completely blocked due to some large disruption [29]. Zhan et al. propose a two-stage optimization algorithm for high-speed lines, based on a mixed-integer programming formulation to minimize the total weighted delay and the number of canceled trains in case of a complete blockage [30]. For the case of a partial station blockage, Wang et al. describe an evolutionary algorithm for the dispatching of high-speed trains [31]. Chen et al. present a modified differential evolution algorithm for real-time train dispatching at junctions [32].

Through mathematical optimization, optimal or near-optimal results are achieved and, in case the optimization process is stopped after a certain amount of time before the optimum was found, the quality of the found solution can be determined. Additionally, solutions produced during the optimization process are non-discriminatory. During manual dispatching, certain railway operating companies might be prioritized, e.g. due to their greater political or business

power, giving them a competitive advantage over smaller companies. During the mathematical optimization process, railway operating companies can easily be prevented from being discriminated, as all constraints and the objective functions are comprehensible and solutions are the result of a deterministic algorithm. Detailed surveys on mathematical optimization models and solution approaches for train dispatching can be found in [33, 34].

More recently, machine learning approaches were proposed in the literature. In particular, the field of reinforcement learning is promising to be applied for train dispatching, as proposed by Zhu in [35] and Šemrov in [36]. Using reinforcement learning, the dispatching system can access previously learned patterns without having to calculate the resulting effects of the numerous possible solutions. Reinforcement learning approaches enable fast decisions, which is advantageous in large network areas and complex operational situations. However, such approaches are not yet applied during real operation, as they might lead to unacceptably long learning times and permanent learning is required in order to react to any changes in the network. As most machine learning tools are constructed as a black box, decisions are not transparent and it is difficult to guarantee that solutions are not discriminating certain railway operating companies. Furthermore, in contrast to mathematical optimization algorithms, machine learning approaches do not provide information on the quality of the found solutions.

Future strategies for train dispatching might therefore incorporate approaches which combine both mathematical optimization and machine learning techniques to achieve reliably good solutions while maintaining fast computation times.

2.1.2 Dispatching Systems used during Real Operation

Dispatching systems that are in use today, normally provide interfaces which visualize the current operational situation and the predicted train movements to the human dispatcher. Often, the prediction of train movements and conflict detection are performed automatically and conflict resolution approaches are suggested to the human dispatcher. Fully automated dispatching systems are rarely applied during real operation and only on small subnetworks of the infrastructure. As many locally applicable constraints, e.g. which tracks can be traversed by certain train types, are not yet digitized, the implementation of a networkwide automated dispatching system still poses great challenges [37]. This section gives an overview of selected dispatching systems that are already used during real operation today.

The dispatching system RCS-Dispo [38] was developed by the Swiss Federal Railways (SBB) and is in operation in Switzerland since 2009. RCS-Dispo automatically detects occupation conflicts between trains and suggests resolution strategies to the human dispatcher. Additionally, the system can predict conflicts arising due to turnarounds or connections between trains. Since 2016, RCS-Dispo is also applied by Infrabel, the Belgian railway infrastructure manager [39]. In 2015, the German railway infrastructure manager DB Netz AG bought the source code of the railway control system RCS, which also includes the dispatching system RCS-Dispo, from SBB [40]. In the project iDis, conducted by DB Netz AG as part of the project PRISMA [41, 42], the new dispatching system LeiDis-D, replacing the previously used dispatching systems LeiDis-S/K and LeiDis-N [43], was developed on the basis of the RCS-Dispo source code. After extensive

field tests, the new system is applied since 2022. The Austrian Federal Railways (ÖBB) use the system ARAMIS-D (Advanced Railway Automation, Management and Information System - Disposition) for train dispatching, which provides automatic conflict detection of occupation and connection conflicts, as well as semi-automatic and automatic conflict resolution [44]. The decision support system COSMOS [45] supports human dispatchers of the East Japan Railway Company in managing the Shinkansen operation. The traffic management system ICONIS [46] by Alstom also provides automatic conflict detection and decision support to human dispatchers and is currently applied by the French railway infrastructure manager SNCF Réseau.

On some small and delimited network segments, fully automated dispatching systems are already in operation. An automated real-time dispatching system presented in [47] is applied in the metro system of Milan since 2007. For the Swiss Lötschberg Base Tunnel, an automated real-time dispatching system was put into operation in 2009 [48].

2.2 Testing of new Dispatching Systems

Before new dispatching systems are commissioned for real operation, extensive testing and readjustments are crucial in order to meet practical requirements and obtain future acceptance from human operators. When dispatching systems are developed, computer simulations using real-world data are a cost-effective way of testing the performance and general applicability of the new methods. Section 2.2.1 gives an overview of existing computer simulation environments.

During field tests or pilot projects, the feedback of experienced human operators is valuable in optimizing and readjusting the software systems to achieve a system that is suitable for productive application. Section 2.2.2 gives an overview of a selection of previously performed field tests and pilot projects.

Railway labs can serve as an intermediate level for the testing of new dispatching systems. However, railway labs mostly serve as training facilities for traffic controllers and their application for testing purposes is still scarce. In section 2.2.3, an overview of past research projects aiming at the testing and evaluation of new dispatching systems in railway labs is given.

2.2.1 Computer Simulations

The first level of testing for new dispatching systems normally involves computer simulations. The majority of dispatching strategies presented in the scientific literature were tested during computer-based case studies with data on small segments of real-world networks. In this section, emphasis is put on computer simulations for advanced dispatching or decision support systems. A selection of computer simulation frameworks and computer simulations, that were conducted to evaluate the performance of dispatching systems, is discussed is this section.

The dispatching system ROMA [17, 18] was tested during computer simulations in the area around Schiphol tunnel while operating a timetable close to the capacity limit. During the EU

project COMBINE [49], a traffic management system with an integrated decision support module for train dispatchers was developed for operation under moving block and ETCS Level 3. Additionally, a computer simulation tool was developed, which was used to evaluate the performance of the traffic management system on a line segment between Belgium and Rotterdam [50]. This traffic management system was further extended during the EU project COMBINE 2 [51] to allow for an application under mixed signaling systems and was tested during simulations in the Schiphol bottleneck area.

The decision support system PANDA (Passenger Aware Novel Dispatching Assistance) [52, 53] supports train dispatchers by providing information on current passenger flows and the predicted impact of potential waiting decisions. The authors test different strategies during computer simulations with real timetable and passenger flow data of 2015 from Deutsche Bahn.

The heuristic dispatching algorithm RECIFE-MILP [54, 55] was evaluated via computer simulations on operational scenarios that were considered as relevant by the French railway infrastructure manager SNCF Réseau and additionally, actually occurred disturbance scenarios were simulated under the application of RECIFE-MILP and later compared to the conflict resolution approaches which were historically chosen by the human dispatchers.

Another option for evaluating the performance of dispatching strategies is via railway traffic simulators, which are software platforms providing realistic simulation environments for train operations. The microscopic railway traffic simulator EGTRAIN [56], which represents the real field, was developed by Quaglietta and provides an interface to incorporate customized real-time dispatching algorithms. EGTRAIN was for example used as a simulation environment to evaluate the stability of the dispatching solutions generated by the dispatching system ROMA [57].

The simulation environment RailSys [58, 59] also provides the possibility to connect an external dispatching system. Another railway traffic simulation tool is OpenTrack [60, 61] which can apply customized dispatching algorithms during simulation. The software tool LUKS® [3, 4] also provides a simulation framework which enables the integration of a dispatching system such as e.g. OptDis [1, 2].

2.2.2 Field Tests and Pilot Projects

Before a new dispatching system can be commissioned on a networkwide scale, the system should first be installed and tested in only a few operating centers controlling small subnetworks. Through feedback loops, experienced human operators can then provide valuable feedback to the developers and the prototypical systems can be adjusted and further extended to obtain systems which can be utilized productively. This way, the future acceptance of new systems will also be increased. In this section, a selection of field tests and pilot projects, which were conducted to evaluate and test new dispatching systems, is presented.

In a previous project of DB Netz AG “Konflikterkennung/Konfliktlösung (KE/KL)“, which translates to “conflict detection and conflict resolution”, a decision support system for human

dispatchers, which detects future conflicts and resolves occurring conflicts based on the iterative application of predefined rules for the resolution of conflicts between two trains, was developed. Recommendations for conflict resolution are provided to the human dispatcher. In 2011 and 2012, a field test was performed in several operating centers where the tool was tested under real-time conditions. In particular, the recommendations of the decision support system were compared to the resolution approaches of experienced human dispatchers and the feedback of the operators was used to improve the existing prototype [62]. Although the decision support system was not applied for real operation in the long run, the developed algorithms are currently applied in the train movement control which provides energy-saving speed recommendations to the trains [63].

During the development of the dispatching system LeiDis-D (see 2.1.2), pilot operations were performed in several operating centers and railway control centers. The feedback obtained from human operators during the test phases was used during the further development of the system, in order to meet practical requirements. Training courses for operators were offered prior to the official launch of LeiDis-D. In the first phase of operation, both LeiDis-D and the previous systems LeiDis-S/K and LeiDis-N are operated parallel to ensure stability and asses the data quality of LeiDis-D compared to the previous systems [41].

The train dispatching assistant ADA-PMB [64], which was developed by DB Netz AG, processes real-time data on the state of the infrastructure, the timetable and current train positions from the operating centers and solves the dispatching problem via a mathematical programming approach utilizing a commercial solver software, while minimizing the sum of all delays in the dispatching area. The tool provides recommendations to the human dispatcher, e.g. on the order in which trains are merged at junctions or on the change of planned tracks. ADA-PMB was tested in a pilot project for the S-Bahn Berlin network, which is controlled via its own operating center. The human operators provided feedback, which was used to further improve the system. The recommendations made by ADA-PMB are considered as reasonable and applicable by the human dispatchers involved in the tests. Therefore, DB Netz AG intends to continue using ADA-PMB for the operation of the S-Bahn Berlin network. To provide reasonable recommendations during mixed traffic operation, the system was adjusted to also incorporate the overtaking of trains. Previously, overtaking was not necessary in the S-Bahn Berlin network, as all trains utilize the same speed profile. The first pilot project under mixed traffic operation started in November 2022 in the operating center of Frankfurt am Main where high-speed and regional passenger trains, as well as freight trains are operated simultaneously [65]. Currently, the tool's running time shall be optimized to be capable of handling larger network areas during real-time operation. In the long run, the integration of the dispatching recommendations from ADA-PMB into LeiDis-D is considered [64].

The optimization algorithm developed in [66] was successfully implemented on the regional line from Trento to Bassano del Grappa in 2011 and later extended to further line segments in Italy [67]. The algorithm was also implemented into a traffic management system that was tested on two line segments of the Norwegian network and put into operation on the line from Stavanger to Moi in 2014. Additionally, it was put into operation on five line segments in

Latvia, where decision support is provided to the human dispatcher and fully automatic dispatching is performed for freight trains [67].

2.2.3 Testing in a Railway Lab

Pure computer simulations might not provide sufficiently realistic setups for human operators to test the new systems reliably. On the other hand, field tests and pilot projects are usually very costly. Railway labs can provide an intermediate level for the testing of new dispatching systems. Operations can be modeled realistically to provide a testing environment that is closer to real operation than pure computer simulations. Feedback loops and further optimization of a system can take place during the testing phase in the railway lab already, before the first field test or pilot project is performed, which is far more cost-effective than going immediately from pure computer simulations to field tests.

There exist a variety of railway labs which are, however, mostly used for the training of traffic controllers [68, 69, 70, 71, 72]. Some research projects have been carried out on railway labs, as they provide an excellent opportunity for including the area of rail human factors into the evaluations, without the necessity to utilize real-world infrastructure and vehicles. This way, new systems, such as dispatching systems or user interfaces for interlocking systems, can be tested intensively without disturbing the real traffic.

During a research project involving the German national railway company Deutsche Bahn AG, RWTH Aachen University, TU Dresden and the University of Göttingen, the dispatching system DisKon [73, 74, 75] was tested at the Integrated Railway Laboratory at TU Dresden [76]. DisKon serves as a real-time decision support system, recognizing occupation and connection conflicts and computing suggested resolution strategies, while also considering connections between trains. While trains were operated automatically on the railway lab, each interlocking system was operated by an experienced human operator from Deutsche Bahn AG who set the required switches, signals and train routes during operation. Certain operational situations were manually created, e.g. situations in which the overtaking or crossing of trains was required and the dispatching decisions suggested by DisKon were later compared to the conflict resolution approaches by an experienced human dispatcher.

In another research project, a dispatching system developed at TU Dresden was tested at the Integrated Railway Laboratory at TU Dresden in cooperation with the German Aerospace Center (DLR) [77]. The focus of this research project was to assess the usability of the user interface and to obtain recommendations on how the interface could be improved, in particular regarding the visual provision of information to human operators.

The Eisenbahn-Betriebs- und Experimentierfeld Berlin (EBuEf) is the railway lab of TU Berlin. Research projects in the area of rail human factors have been conducted at the EBuEF, including the testing of a new interlocking interface “WebSTW” with experienced traffic controllers [78] and participation in the EU project Drive2TheFuture [79]. The Eisenbahn Betriebsfeld Darmstadt (EBD), the railway lab of the Technical University of Darmstadt, provides a testing environment for dispatching systems and algorithms. A software tool supporting dispatchers by visualizing connection conflicts was successfully tested at the EBD and the feedback of the experienced human dispatchers was used to further improve the

tool's usability [80]. The Eisenbahnbetriebslabor Schweiz, which formerly belonged to ETH Zürich, also provides the possibility to be utilized for research purposes [81].

At the Technical University of Braunschweig, a virtual railway lab was developed [82]. The infrastructure and rolling stock are provided via a simulation software and do not exist physically. Interfaces of interlocking systems are connected to the virtual railway lab, thus enabling the investigation of rail human factors.

2.3 Energy-Saving Driving Approaches

Energy-efficient driving is of great importance both from an environmental and financial perspective. This section gives an overview of the state-of-the-art in energy-saving driving techniques found in the scientific literature.

Energy-saving driving can be modeled through a mathematical optimization problem, as introduced by Ichikawa in [83]. A dynamic programming algorithm for the energy-saving driving problem is proposed by Miyatake and Ko in [84] and another approach using Pontryagin's maximum principle is presented by Howlett et al. in [85]. Several approaches considering energy optimization during the timetable generation process prior to operation exist in the literature. For the optimization of multiple train runs, Chevrier et al. propose a multi-objective evolutionary algorithm in [86] and Wang and Goverde present an algorithm adjusting the scheduled arrival and departure times to improve energy efficiency in [87]. For single train runs, the method described by Scheepmaker et al. in [88, 89] optimizes energy consumption during the timetabling process by adjusting the distribution of running time supplements over multiple stops. A nonlinear mixed-integer programming approach combining energy optimization into the timetabling process is proposed by Xu et al. in [90]. A multi-objective programming model optimizing the energy consumption and total passenger trip time as part of the timetabling process is described by Huang et al. in [91] for urban railway traffic. Running time supplements in the timetable are crucial in generating opportunities for energy-saving driving to be applied. Therefore, Miyatake et al. present a method to optimize the distribution of running times along intermediate stations during the timetabling process with respect to energy-saving potentials, while not changing the total running time between designated start and destination stations of trains [92]. For automatic train operation, another approach based on the optimization of running time distributions during timetabling is introduced by Watanabe and Koseki in [93]. During timetable generation, which is often performed annually, not all future long-term disruptions due to e.g. construction works or maintenance are known. As such disruptions do not occur unexpectedly and are known prior to operation, Graffagnino et al. describe an algorithm for the daily optimization of train speed profiles with respect to energy consumption [94].

As many disturbances occur during operation and can thus not be predicted in advance, it is of great importance to react to such situations appropriately, not only with respect to punctuality, but also from an energy-saving driving perspective. There are a variety of methods including centralized energy optimization during the rescheduling process. In particular, several such approaches exist for metro systems. A method applying a deep learning algorithm is presented by Liao et al. in [95]. In [96], Yin et al. describe a dynamic

programming approach that takes uncertain passenger demands into account. Approaches applying mixed-integer programming can be found in [97], where Hou et al. propose a method which simultaneously optimizes the total train delay, the number of stranded passengers and energy consumption, as well as in [98], where Guo and Zhang combine the mathematical programming approach with a trained neural network and dynamic passenger flows and consider random disturbances. For urban rail networks, a method simultaneously optimizing the train circulation plan and the energy consumption is presented by Zhou et al. in [99]. For the rescheduling of high-speed trains within major disruptions, Zhan et al. introduce an optimization strategy applying a multiple-phase optimal control model for the generation of energy-efficient speed profiles [100]. There are also approaches designed for mixed traffic operation that integrate energy optimization within the rescheduling process. A Resource Conflict Graph algorithm is presented by Toletti et al. in [101] and [102], and Naldini et al. describe an Ant Colony Optimization strategy in [103].

Decentralized approaches for energy optimization that also take information on other train runs into account, are mostly proposed for connected driver advisory systems (C-DAS). For the application of C-DAS, all on-board driver advisory systems continuously communicate with a central traffic management system to pass on information such as the train's current location or speed. Based on the received information, the central traffic management system in return predicts future train trajectories and provides the on-board C-DAS with driving recommendations, which can result from dispatching measures or aim to optimize energy consumption. The connected driver advisory system CATO [104] was successfully tested on segments of the Swedish railway network [105]. In the literature, approaches aiming at energy optimization by applying C-DAS can be found. A method reducing the delay at junctions is presented by Galapitage et al. in [106], whereas Wang et al. describe another approach for energy-efficient merging of freight trains at junctions in [107].

In general, the presented approaches found in the scientific literature can be grouped into four categories based on the time at which optimization is performed, as shown in Figure 1. The first category includes all approaches considering the energy optimization during the timetabling process already. The second category includes the method proposed by Graffagnino et al. in [94], which optimizes energy consumption on a daily basis prior to the start of operation. The third category consists of the approaches which incorporate energy optimization into the rescheduling process itself, i.e. energy optimization is performed centrally during real-time operation. The approaches for onboard energy optimization of trains form the fourth category.

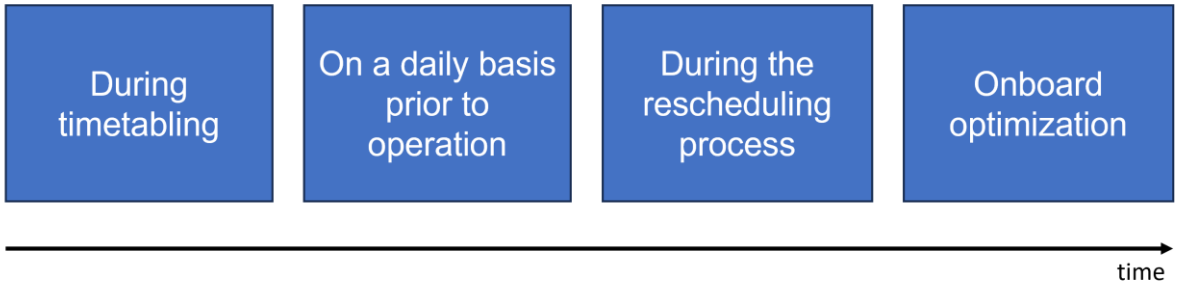


Figure 1: Different categories of energy-saving driving approaches in the scientific literature

When energy optimization is performed during the timetabling process, long computation times do not pose any significant limitation, as timetables are usually generated on an annual basis. The approach re-optimizing the speed profiles of trains with respect to energy-saving driving on a daily basis bears the advantage that long-term disruptions, e.g. due to maintenance, which were not known or considered at the time of timetable generation, can be taken into account and the calculations are not required to be performed in real-time, allowing for longer computation times of the optimization algorithms. However, the designated speed profiles generated prior to operation may no longer be feasible during operation, as unexpected disturbances can occur. Methods integrating energy optimization into the rescheduling process can react to disturbances instantaneously and dynamically adjust speed profiles with respect to energy optimization. On the other hand, the resulting mathematical optimization problems can become extremely complex for large network areas and lead to infeasibly long computation times. Thus, in order to allow for a real-time application, heuristics are often applied, yielding solutions that are not necessarily optimal. The approaches for onboard energy optimization of trains outsource energy-efficient speed profile generation to the trains' driver advisory systems or onboard computers, immensely reducing the complexity and computation time of the rescheduling process. Onboard train speed optimization, however, only optimizes the energy consumption of single trains, thus the overall energy consumption of all trains in the network is not solved holistically. It is also more costly, as connected driver advisory systems or onboard computers and the technical equipment required for communication have to be installed on each vehicle.

For application of energy-saving approaches during operation, many objectives such as computation time, the feasibility and the quality of the generated solutions, need to be considered. In general, the consideration of energy consumption during timetable generation and optionally the daily re-optimization of speed profiles are advisable, as long computation times are acceptable. Additionally, real-time energy optimization is reasonable to react to unexpected disturbances. Whether the optimization should be performed by a central traffic management and rescheduling system or onboard trains depends on multiple factors, such as the network size, the number of vehicles operated within the network or whether trains are already equipped with the required technology. Large networks with many vehicles lead to complex operational situations, increasing computation times during rescheduling. By performing energy optimization solely onboard trains, computation times are reduced, yielding faster rescheduling solutions and thus a higher punctuality. On the other hand, such systems involve more costs, thus tradeoffs must be made when deciding whether to perform energy optimization centralized or decentralized.

2.4 Scientific Contribution

The literature research shows that new dispatching strategies are hardly ever tested in railway labs, despite their potential to serve as a cost-effective and realistic testing framework. Currently, no testing environment for automatic train dispatching exists for railway labs. Moreover, a direct comparison between the performance of an automated dispatching system and an experienced human dispatcher is still challenging during real operation, as disturbance scenarios cannot easily be reconstructed to create equal conditions.

The scientific contribution of this thesis is the introduction of a novel testing environment for automatic train dispatching algorithms in a railway lab during real-time operation under realistic conditions. For this purpose, communication interfaces between a dispatching system, a physical railway lab and a railway lab control software are formally defined. A railway lab control software enabling automatic operation during real-time rescheduling is developed and its structure and underlying algorithms are described.

This testing environment is used to compare the dispatching strategies of an automated dispatching system to those of an experienced human dispatcher. A case study is set up, connecting an existing dispatching system to an existing railway lab. Both the dispatching system and the human dispatcher are confronted with the same operational situations and disturbances. The resulting delays during operation are recorded and evaluated.

Furthermore, a novel approach for onboard energy optimization of trains is proposed and its performance is assessed in a case study.

3 Fundamentals of Railway Timetabling

This chapter presents the fundamentals of railway timetabling which are relevant for this thesis. First, section 3.1 describes the fundamentals of running time calculation. An overview of definitions regarding stops and dwell times is given in section 3.2 and different types of delays are categorized in section 3.3. The foundations of blocking time stairways which are commonly applied in railway timetabling and dispatching are then provided in section 3.4. Lastly, railway infrastructure models with different granularities are introduced in section 3.5.

3.1 Running Time Calculation

In particular for timetable generation or train dispatching, precise information on the minimum required running times of trains between a starting and a destination point are required. During timetable generation, a running time reserve of a few percent is normally added to this minimum technically possible running time to allow for the reduction of small delays during regular operation.

The braking deceleration is normally assumed to be constant [108]. Running times during cruising at constant speed and braking are therefore easy to compute. Thus, to obtain precise information on minimum possible running times, information on acceleration processes are crucial.

To compute the momentary acceleration of a train, the tractive effort and the total resistance on the vehicle are required. The tractive effort of a railway vehicle corresponds to the maximum tractive force the vehicle can generate and is normally modeled through a tractive effort curve [109] as a function of the current train speed which is composed of a constant part and a hyperbola, as shown in Figure 2.

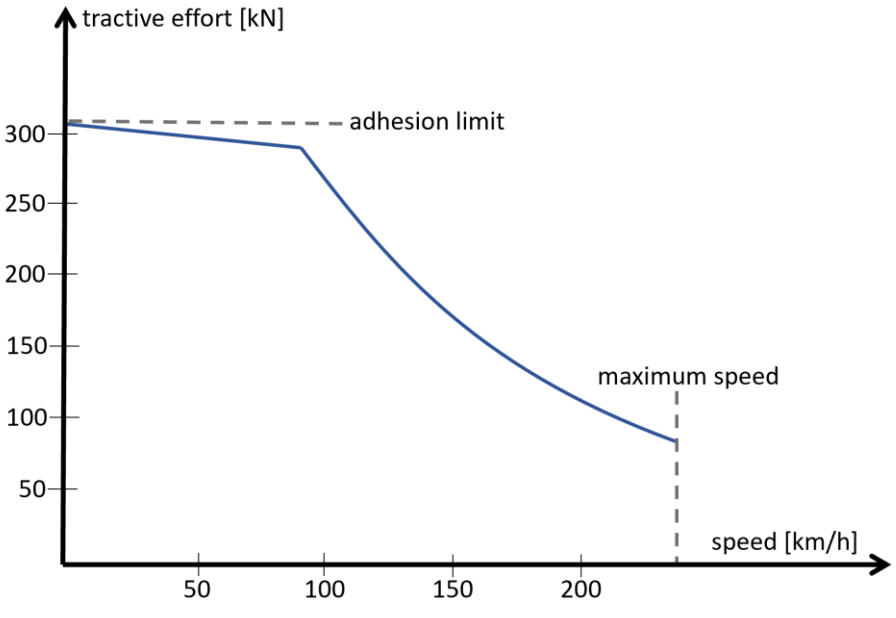


Figure 2: Tractive effort as a function of the train speed (according to [110])

During acceleration, different resistances affect the railway vehicle. The overall resistance on the vehicle is the sum of the running resistance and the track resistance. The running resistance is the vehicle-specific resistance, e.g. due to rolling resistance or air resistance, whereas the track resistance results from track-specific parameters, such as gradients or curve radiuses.

The running resistance R_r of a train can generally be modeled as a quadratic function of the train speed v [km/h] by

$$R_r = A + Bv + Cv^2$$

where A [N] , B [N · s / m] and C [N · s² / m²] are coefficients which depend on the vehicle [111].

For line segments through tunnels, specific empirical formulas for R_r have been derived through experiments for passenger trains, freight trains and rail cars respectively [112].

The track resistance R_t is the sum of the gradient resistance R_g and the curve resistance R_c [113]:

$$R_t = R_g + R_c.$$

When a train runs on a track segment with an inclination, a gradient resistance is effective. This gradient resistance is calculated as

$$R_g = m \cdot g \cdot \sin \alpha$$

where m $[t]$ is the mass of the vehicle, $g = 9.81$ m/s² is the acceleration due to gravity and α is the inclination angle [113]. Note that for tracks going downhill, the resistance is negative, thus acceleration of the train is increased.

In track curves, a curve resistance applies. Different empirical formulas exist for the curve resistance [114, 115] which is generally calculated as

$$R_c = m \cdot g \cdot \frac{A' + B' \cdot a_R}{r}$$

where a_R is the axle spacing of the wagons in meters, r is the radius of the curve in meters and A' and B' are coefficients depending on e.g. the current weather conditions.

The total resistance on a railway vehicle is calculated as the sum of the running resistance and the track resistance:

$$R = R_r + R_t.$$

Let F_{te} be the momentary tractive effort of a train, as in Figure 2. Then the resulting force F_a which the train can use for acceleration is given by

$$F_a = F_{te} - R.$$

The second Newton's law of motion states that the net force equals to product of mass and acceleration [116]. To account for the rotationally accelerated mass of the train (e.g. its wheels), a mass factor λ which depends on the specific vehicle must be considered [117].

It follows that

$$F_a = \lambda \cdot m \cdot a$$

and therefore, the momentary acceleration of a train at a given velocity is described through

$$a = \frac{F_{te} - R}{\lambda \cdot m}.$$

There are different approaches to approximate the acceleration of a train as a function of the velocity, e.g. the Δv micro-step method [118]. For this method, the available speed range is divided into small intervals and a constant acceleration is assumed within each of these intervals. Figure 3 illustrates one such interval from v_i to v_{i+1}. The acceleration a_i at v_i and a_{i+1} at v_{i+1} can be computed by the above formulas. The average acceleration is chosen as constant acceleration $\bar{a}_i$ for this interval and is given by

$$\bar{a}_i = \frac{a_i + a_{i+1}}{2} = \frac{F_{te_i} - R_i + F_{te_{i+1}} - R_{i+1}}{2 \cdot \lambda \cdot m}$$

where F_{te_i} and $F_{te_{i+1}}$ are the tractive effort and R_i and R_{i+1} are the total resistance on the railway vehicle at a_i and a_{i+1} respectively.

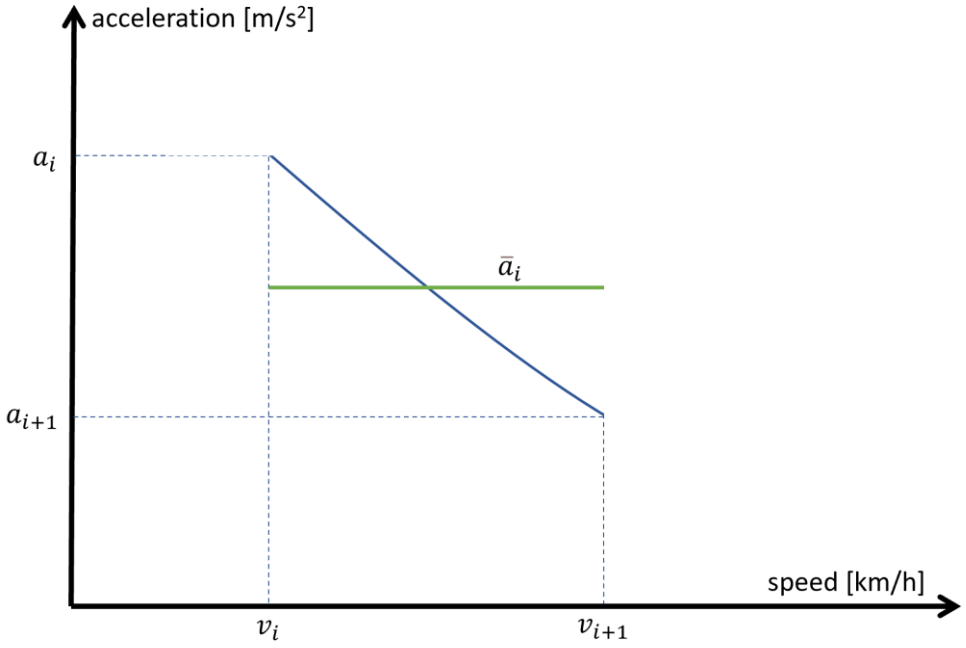


Figure 3: Approximation of the acceleration via the Δv micro-step method (according to [118])

The precision of the method strongly depends on the size of the speed intervals and can be increased by decreasing the interval size.

The time t_i required to accelerate from v_i to v_{i+1} with constant acceleration $\bar{a}_i$ can now easily be computed as

$$t_i = \frac{v_{i+1} - v_i}{3.6 \cdot \bar{a}_i} \text{ [sec]}$$

and the total time required for acceleration is obtained as the sum over all t_i corresponding to the intervals.

3.2 Stops and Dwell Times

Train stops are divided into scheduled stops and operating stops [113]. The purpose of scheduled train stops includes the entry and exit of passengers for passenger trains and the unloading and loading of goods for freight trains. Scheduled stops normally occur at designated stopping positions, e.g. platforms in passenger railway stations. Operating stops are stops that are necessary for a smooth operation. Operating stops can be required to enable the overtaking by another train or when trains meet on a one-directional line segment. Additional operating stops can also be necessary as a dispatching measure in case of delays.

The dwell time is the time interval designated for a train stop. For scheduled stops, minimum dwell times can be defined in the timetable, as passenger exchange or the unloading and loading of goods requires a certain amount of time. In case of delays, dwell times can be reduced, however, the minimum dwell times must be followed. In general, no minimum dwell times are defined for operating stops, thus these stops can be omitted, if required due to the operational situation.

3.3 Delays

During operation, trains normally do not run precisely as originally scheduled. Delays are divided into primary and secondary delays [119]. Primary delays are delays which are caused by the train itself, e.g. due to engine failure or a high volume of passengers, or by external factors, such as weather conditions or people on the tracks. Secondary or knock-on delays are delays caused by another train being delayed. Train delays can lead to occupation conflicts with succeeding trains, resulting in the succeeding trains not being able to run according to schedule. Furthermore, train delays can cause connection conflicts, i.e. certain connections between different trains are no longer possible, and circulation conflicts, i.e. train runs cannot start due to rolling stock or staff not being available [120]. If delays are only considered within a certain network area, the delay that a train already has when entering this area is called delay at entry.

3.4 Blocking Time Stairways

For operation under a fixed block signaling system, lines are divided into consecutive block sections to enable safe train operation. Block sections can only be exclusively occupied by one train at a time. Thus, a train can only enter a new block section if the section is currently not occupied. Often, clearing points are defined behind block signals, which must be completely cleared by all axes of the preceding train in order for a new train route to be set.

The blocking time of infrastructure segments was first described by Müller in [121] and Happel in [122]. It refers to the time interval the segment is blocked due to a train running on this segment, i.e. the infrastructure segment cannot be occupied by another train simultaneously. The blocking time not only includes the pure running time of the train on the infrastructure segment, but also the following time intervals [123]:

- The **route setup time** is the time required for the train route to be set. This time strongly depends on the type of interlocking system used, as well as the number of switches that need to be set for the train route.
- The **signal watching time** is the time required for the train driver to recognize the signal aspect and react to it, e.g. by braking.
- The **approach time** is the running time between the pre-signal and the corresponding block signal.
- The **clearing time** is the time required for all axes of the train to clear the block section or, if applicable, the clearing point.
- The **route release time** is the time required for the interlocking system to unlock the train route after clearance. This time depends on the type of interlocking system.

Figure 4 visualizes the components of the blocking time of a block section. First, the train route is set, then the signal watching time starts, in which the train driver recognizes the signal aspect. Next, the train approaches the main signal delimiting the block section. After running through the block section, the train must clear the section between the next main signal delimiting the block section and its corresponding clearing point completely. Once all axes passed the clearing point, the route is released and the section between the two main signals is no longer blocked and can thus be occupied by a succeeding train.

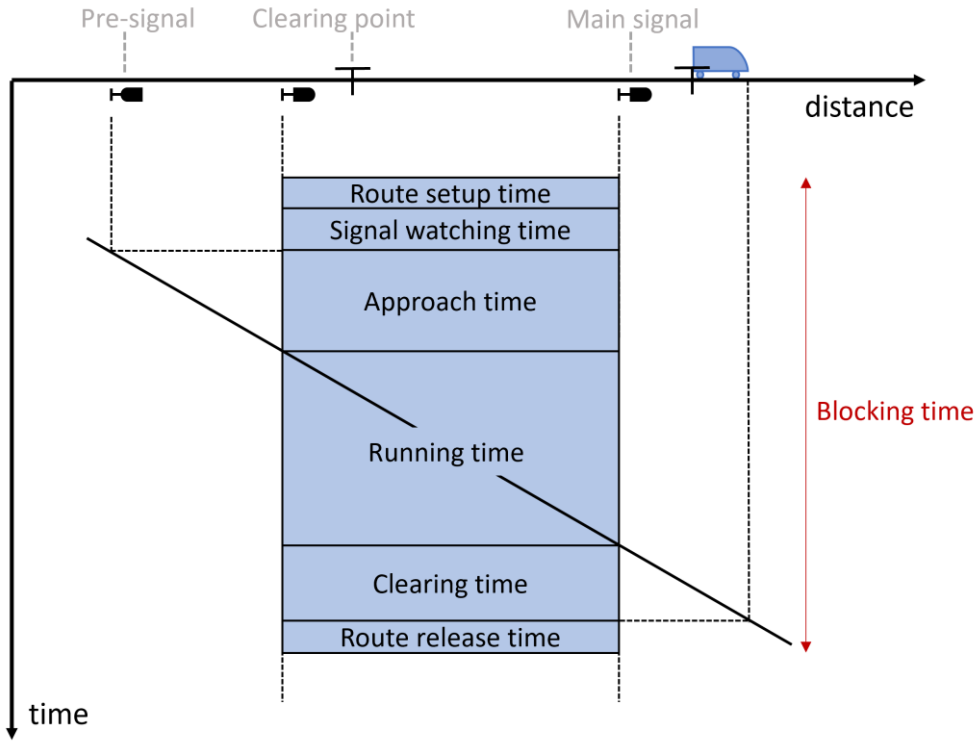


Figure 4: Components of the blocking time of a conventional fixed block section (according to [123])

Train routes inside station areas are often released partially, in order not to block the complete train route for the passage of other trains [113]. The train routes are divided into partial train routes which are released once the train passes and clears the corresponding infrastructure section. Switches along a train route that have already been cleared completely can thus be used by other trains. Partial route releases are essential to ensure a smooth operation by letting multiple trains enter or exit parallel tracks simultaneously, in particular in case of scheduled or operating stops in stations.

Figure 5 shows an example of a partial route release. A train runs along the train route highlighted in green. Once the switch marked in yellow is cleared by the train, the route is partially released and the new train route highlighted in orange can be set.

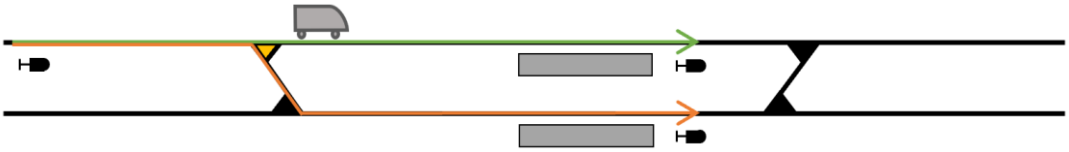


Figure 5: Partial route release

Train routes usually have an overlap length beyond the signal until which they are set [124]. The train route can only be set if the overlap length beyond the destination signal is not occupied by another train to avoid safety hazards if trains unintendedly do not stop before the signal. For this purpose, clearing points which have to be cleared by a preceding train to allow for the safe route setting for the succeeding train are defined. The overlap length can vary, depending on the permitted speed at which signals can be approached. An example is shown in Figure 6 where the train running along the route highlighted in green must first pass the clearing point before the train route highlighted in orange can be set to ensure that the complete overlap length is not occupied.

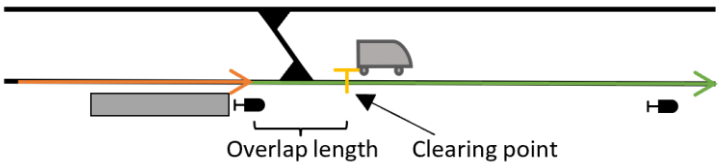


Figure 6: Route setting after clearance of overlap length

A sequence of consecutive blocking times along the block sections which a train passes, is called blocking time stairway. An example of a blocking time stairway between two overtaking sections is shown in Figure 7, where the blue line represents the time distance line of the train run.

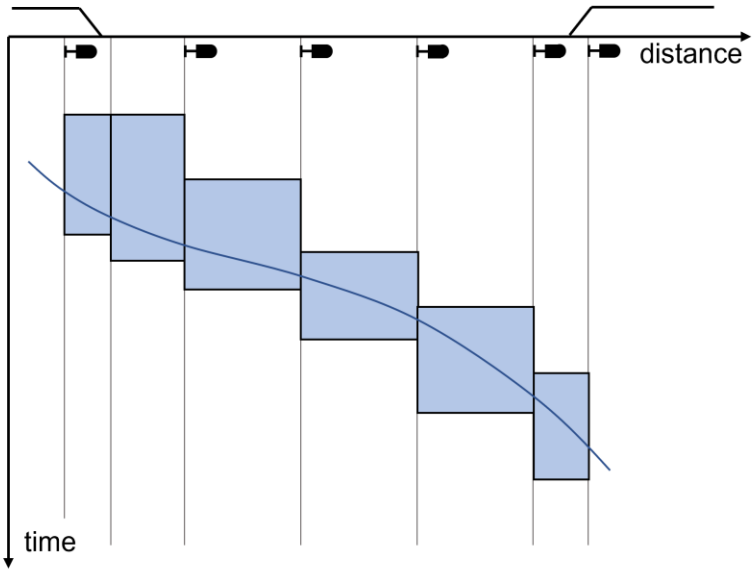


Figure 7: Blocking time stairway

Note that the section between the first two signals in Figure 7 includes the overlap length after the first signal. As described above, a train route until the first signal can only be set once the overlap length was cleared. Therefore, the first part of the section between the first two signals has a shorter blocking time than the remaining part.

By modeling scheduled train runs via blocking time stairways, occupation conflicts between two trains occur if their blocking time stairways overlap. In order to obtain a feasible timetable, it is thus essential that all blocking time stairways do not have any overlaps with one another. Many computer-based scheduling systems, e.g. RUT-K [125] or TPS [126], apply blocking time theory to generate conflict-free train timetables.

Let T_1 and T_2 be two consecutive train runs, where T_1 precedes T_2. The minimum headway time between two overtaking sections is defined as the minimum time interval at which T_2 can follow T_1 without restrictions, i.e. without causing an occupation conflict, thus the trains' blocking time stairways do not overlap. The minimum headway time between the two trains can be determined as the time between the beginning of the blocking time stairway of T_1 and the beginning of the blocking time stairway of T_2 in the first common block section, as shown in Figure 8. If several consecutive overtaking sections are passed in succession, the maximum minimum headway time over all sections is considered for timetable construction.

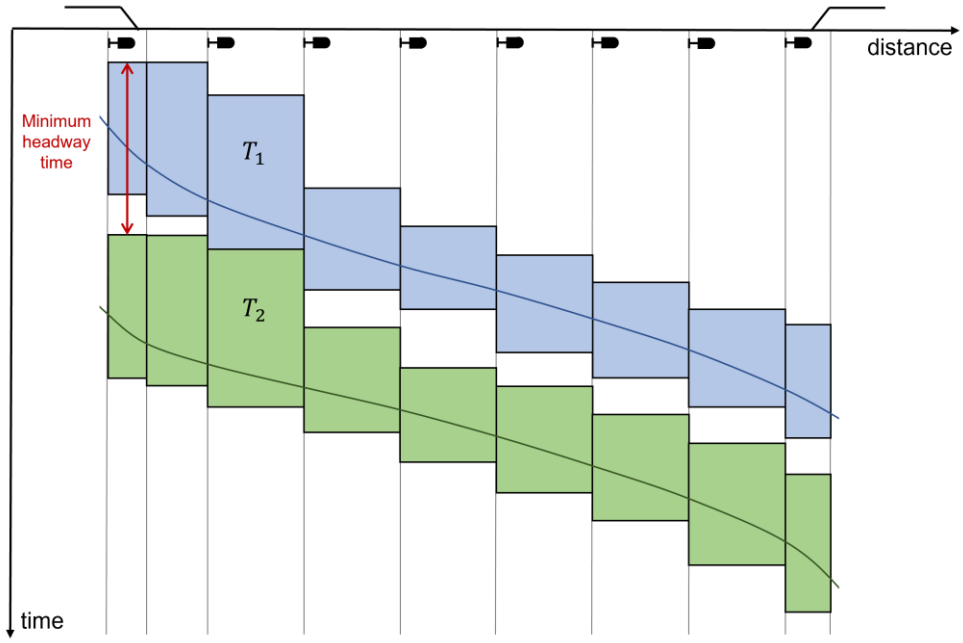


Figure 8: Minimum headway time

In order to reduce the propagation of delays, it is reasonable not to let a train follow another train by only the minimum headway time. Otherwise, a delay received by the first train would directly propagate to the second train. To obtain better timetable robustness, buffer times are added to the minimum headway time. Buffer times usually depend on the probability of train delays, the train types following each other, maximum permitted speeds and the train protection system in place. Commonly, minimum buffer times lie in the range of one to three minutes [127]. For a homogeneous train mix, buffer times tend to be smaller, whereas for a heterogenous train mix, larger buffer times are usually applied. For instance, in a metro system, where all trains have the same priorities and speed profiles, buffer times of one

minute might be reasonable, whereas a buffer time of three minutes between a high-priority high-speed train following a low-priority freight train might be applied. By utilizing buffer times during timetable construction, delay propagation is reduced during operation and might possibly be completely prevented in case of small delays.

3.5 Railway Infrastructure Models

The modeling of railway infrastructure is essential and is required for the generation of conflict-free timetables and during real-time train dispatching. As state-of-the-art, railway infrastructure is modeled using graph theory. Utilizing graphs as underlying data structure allows for the application of a versatile set of efficient computer algorithms developed over the past decades. Various infrastructure models based on graphs have been proposed with different data granularities [128].

Graphs are data structures containing a set of vertices V, also called nodes, and a set of links E between pairs of nodes, also called edges [129]. If a pair of vertices v_1 and v_2 is connected via an edge, the tuple (v_1, v_2) is in E and the edge represents an existing relation between these two vertices. If, for example, two railway stations are directly connected to each other, this relation can be modeled by defining the stations as vertices and the connection between the stations as edge of a graph. There are numerous extensions of the basic graph model. E.g. the edges can be directed [130], which can be used to model railway tracks that can only be traversed in one direction, or additional attributes and variables can be assigned to both vertices and edges. Multigraphs can contain multiple edges between the same vertices, which allows for the modeling of parallel tracks. Depending on the granularity of the railway infrastructure model, the vertices and edges can represent different elements, e.g. a vertex can represent a whole railway station or only a certain infrastructure element, such as a main signal.

Macroscopic infrastructure models have a coarse granularity and represent railway stations and junctions as nodes and tracks between stations as lines connecting the nodes [128]. Although macroscopic representations of the infrastructure require less data and storage capacity, track occupation cannot be represented with sufficient precision, thus for timetabling and dispatching infrastructure models with a finer granularity need to be utilized.

For reliable timetabling and dispatching, precise running time calculation is crucial. Occupation times of infrastructure segments, such as tracks in a station or segments between switches, are required. Additionally, information on e.g. maximum permitted speeds or train protection systems need to be modeled. Microscopic infrastructure models contain precise information on the infrastructure elements, including signals, switches, speed limits, signaling systems, clearing points, station platforms and gradients [131]. Each element can additionally store specific parameters, such as the maximum speed a switch can be traversed at or the length of a platform. The precise position within the network topology is assigned to each element. Microscopic infrastructure models are required for the blocking time theory (see 3.4) to be applied.

Mesoscopic infrastructure models possess a data granularity that is in-between macroscopic and microscopic, e.g. the different tracks within a station might be modeled separately, whereas the open track and junctions might be modeled as lines and nodes respectively [132].

Which model is appropriate to model the infrastructure strongly depends on the use case and a tradeoff between precision and the computation time associated with the data size of the model must be made. Microscopic models are essential for the application of blocking time theory and are for instance implemented in simulation environments such as RailSys [59], OpenTrack [61] or LUKS [3], whereas macroscopic models are e.g. utilized when performing route search algorithms in large network areas.

Figure 9 shows examples of the three types of railway infrastructure models.

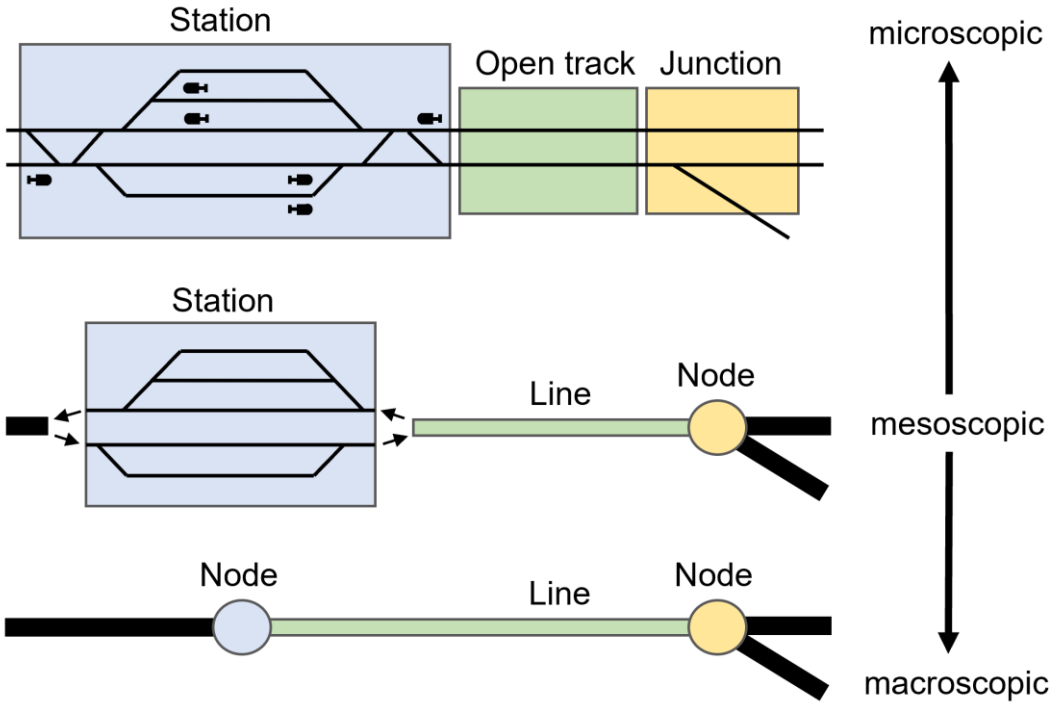


Figure 9: Railway infrastructure models with different levels of detail (according to [133])

4 Definition of a Communication Interface between a Dispatching System and a Railway Lab Control Software

To establish a testing framework for real-time dispatching algorithms in a railway lab, a communication interface between the dispatching system in which these algorithms are applied and the railway lab control software must be defined and mutually utilized. This way, trains on the railway lab network can be operated according to a newly generated dispatching timetable. This chapter defines the requirements for such a communication interface and discusses which information are essential for both systems during operation respectively in section 4.1. Section 4.2 then describes one possible way to achieve communication between these two systems.

It is assumed that the dispatching area is subdivided into several operating stations which can either be whole railway stations or line segments within the network. Train runs thus pass a predefined sequence of operating stations along their journey. Signals in the network are referenced by their name and the name of their operating station. It is assumed that no two operating stations with the same name exist in the dispatching area and no two signals with the same name exist within an operating station.

Dispatching measures considered in this thesis are bending of train runs between operating stations, changes of scheduled dwell-times, the cancellation or adding of stops and route changes through an operating station. Note that the cancellation of stops is only considered for operating stops or for scheduled stops that do not have a minimum dwell time.

4.1 Requirements for the Communication Interface

This section gives an overview of the requirements necessary to establish a shared communication interface between a dispatching system and a railway lab control software to be used during real-time operation.

Firstly, mutual communication between these two systems requires a common protocol for message exchange and the definition of a shared structure to encode information.

The realistic prediction of future train movements by the dispatching system is key to ensure that good quality dispatching measures are obtained during the conflict resolution process. Thus, the dispatching system needs to be provided with frequent updates on the current operational situation. The provided information should include both current train positions, as well as routes set by the interlocking system. During real-world operation, information on delays at entry of a train entering a dispatching area are normally available prior to its time of entry. Thus, to provide a testing environment resembling realistic conditions, such predictions have to be provided to the dispatching system in advance.

Conversely, the dispatching measures determined by the dispatching system during conflict resolution have to be implemented during operation on the railway lab infrastructure. Therefore, continuous transmission of dispatching measures from the dispatching system to the railway lab control software is necessary. These measures should ideally be provided directly after modifications to the dispatching timetable have been performed.

Furthermore, in a realistic real-time testing environment, it is crucial for both system times to be synchronized, i.e. the systems are using the same internal time in case a virtual time is used instead of the actual real-world time.

Table 1 summarizes the requirements for a communication interface between a dispatching system and a railway lab control software, the possible implementations of these requirements and the respective necessary information.

Table 1: Requirements for a communication interface between a dispatching system and a railway lab control software

Requirement	Possible implementation	Necessary information
Possibility for mutual communication	Common protocol for message exchange	Definition of the respective message formats
Realistic prediction of future train movements by the dispatching system	Updates on the current operational situation; information on trains entering the dispatching area	Train positions, set train routes; predicted times of entry into the dispatching area
Implementation of the dispatching measures on the railway lab infrastructure	Continuous transmission of dispatching measures	All information required to uniquely encode the dispatching measures: e.g. new arrival times, new routes
Synchronization between system times of the dispatching system and the railway lab	Use of the same internal time; transmission of the shared internal time at constant intervals	Real-world time or virtual time (in case the real-world time is not used)

4.2 Description of the Message Exchange

This section describes how the message exchange between a dispatching system and a railway lab control software can be achieved to meet the requirements defined in section 4.1 and how all relevant information can be encoded.

As strings provide a convenient data format to exchange short and customized messages which are easily readable for both humans and machines, all messages for the communication interface are defined via strings. These strings follow a certain predefined format consisting of several message parts separated by a delimiter symbol, e.g. "$part_1$| $part_2$ |… | $part_n$" where n is the number of message parts the specific message format contains. Different message formats must be used, depending on which type of information shall be transmitted. Each message part within a message format contains a certain predefined information, e.g. a train number or an arrival time. To enable fast and efficient communication, all message formats are defined in such a way that redundant message parts, not required to transmit the information, are avoided. All messages are defined to have a binary symbol as first message part, indicating the direction of message transmission, i.e. the first message part is "1" if the message is transmitted from the railway lab control software to the dispatching system and "0" if it is transmitted from the dispatching system to the railway lab control software. The second message part always indicates the type of message transmitted (e.g. a new train route), in order to enable unique identification of a received message. The remaining message parts contain all parameters required to uniquely transmit the desired information.

For the mutual communication between the two systems, one way to establish message exchange is via the Transmission Control Protocol (TCP) [134]. The TCP provides a framework for fast and efficient data exchange between computers connected via a network and can also be used for data transmission between different programs running on the same computer.

Therefore, it can be applied regardless of whether the two systems are running on the same or two separate computers. TCP permits reliable bi-directional data exchange, i.e. data is guaranteed to be transmitted through retransmission of lost packets [135]. In particular during dispatching, reliability is of great importance. Lost packets from the railway lab to the dispatching system can result in infeasible dispatching measures, as important information on the current operational situation is missing. On the other hand, lost packets from the dispatching system to the railway lab can lead to the computed dispatching measures not being applied, resulting in unintended delay propagation. As a drawback, TCP involves latency times resulting from the retransmission of lost packets, thus the protocol does not operate entirely in real-time. As the messages defined in this chapter only contain short message strings, the amount of data transferred via the protocol is very small. During practical tests on a railway lab (see chapters 8 and 9), latency times in the low millisecond range were observed. These extremely small latency times are negligible during train dispatching, therefore TCP is suitable to serve as means of reliable message exchange for the communication interface between a dispatching system and a railway lab control software.

The following two subsections discuss the definition of message formats for encoding the relevant information to achieve the requirements described in section 4.1 and explain why the formats were defined a certain way.

4.2.1 Messages from the Railway Lab Control Software to the Dispatching System

As described in 4.2, the second message part defines the type of transmitted information and the remaining message parts contain the parameters specific to the predefined message formats. Five different types of messages are defined to be transmitted from a railway lab control software to a dispatching system and their respective message formats are explained below.

The first three messages provide information on the reaching of main signals, the setting of train routes and the prediction of entry times into the dispatching area and enable the dispatching system to predict future train movements realistically, as defined in the requirements in section 4.1. Furthermore, the synchronization of the mutual system time, as well as the starting of the conflict resolution process, are initiated by the railway lab control software and communicated to the dispatching system via designated messages.

Main signal reached by a train

The precise prediction of the current positions of all trains within the dispatching area is crucial for reliable conflict detection and thus for the quality of the resulting dispatching measures determined by the dispatching system. Although continuous train position detection is possible with ETCS Level 2 [136], network-wide localization of trains is still a challenge the railway sector is facing. Therefore, train positions are defined to be transmitted to the dispatching system each time the first axle counter after a main signal is reached by a train, as a tradeoff resembling realistic conditions. In between main signals, the current position of a train has to be predicted by the dispatching system, based on its scheduled trajectory. The time the head of the train reaches the first axle counter located after the main signal corresponds to the time it enters the first clearing section of the route beginning at this signal,

thus the time that the signal aspect changes from “proceed” to “stop”. Train localization in railway labs is often performed via axle counters and it is assumed that for each main signal, an axle counter is located closely after the signal in the respective running direction. The time the first axle of the train reaches this axle counter is defined as reference time for reaching the signal, as this time can be measured precisely.

The type of message to transmit the reaching of a main signal by a train is called “reachedSignal”. As parameters it contains the train number TN, the operating station OS of the reached main signal, the name of the reached main signal S, as well as the time stamp t at which the first axle of the train reached the main signal. The message format is thus defined as “1 | reachedSignal | TN | OS | S | t”.

Once a message of this format is received, the dispatching system has to synchronize its internal train position, e.g. by bending the train run in the corresponding section such that the main signal S is reached at time t in the predicted trajectory. Figure 10 shows an example. The orange trajectory shows the previously predicted trajectory of the train run within the dispatching system. After receiving a message that signal S was reached at time t, which is later than previously predicted, the dispatching system internally adjusts the trajectory in the affected section by bending it, resulting in the green trajectory.

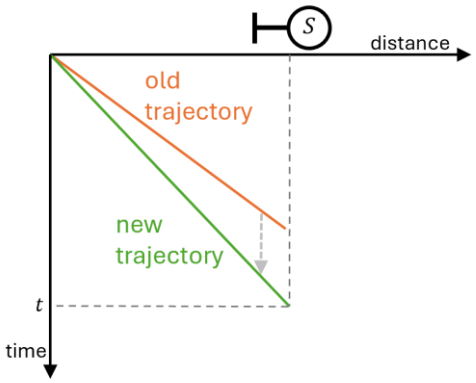


Figure 10: Synchronization performed by the dispatching system upon receiving information on a reached main signal

Route set for a train

The dispatching system has to be informed about routes that have already been set by the interlocking system for a specific train. This ensures that future dispatching measures for this train, containing route changes conflicting the already set route, can be avoided. Additionally, the knowledge of a set route that differs from the one in the current dispatching timetable enables the dispatching system to detect possible resulting occupation conflicts early and consider them during conflict resolution.

In addition to the train number, the dispatching system requires information on which route was claimed for the train, hence which route was set by the interlocking system for this specific train number. The starting and destination signal of the train route are suitable to describe the train route and are thus chosen for this purpose.

This type of message is called “trainRouteClaimed”. As parameters, it contains the train number TN, the operating station OS_s and the name of the starting signal S_s, as well as the

operating station OS_d and the name of the destination signal S_d associated with the route claimed for the train.

Thus, the message format is defined as "1 | trainRouteClaimed | TN | OS_s| S_s| OS_d| S_d". Note that train routes are not necessarily uniquely defined by their starting and destination signals. If there exist multiple paths from the starting signal to the destination signal via different combinations of switches, the dispatching system will assume that the train route has been set according to the route scheduled in the current dispatching timetable. Therefore, it is essential for the railway lab control software to always consider the latest dispatching measures when setting train routes.

When receiving a message of this format, the dispatching system must disable future changes to the respective train route.

Prediction of a train's time of entry into the dispatching area

This type of message is called "predictedTimeOfEntry". To uniquely encode an entry of a train into the dispatching area, the train number needs to be provided. As the location of entry into the dispatching area can differ from the scheduled location, information on the predicted location of entry should also be provided to the dispatching system, e.g. via the main signal at which the train is predicted to enter.

The transmitted message contains four additional parameters: the train number TN, the predicted time of entry t_p, the operating station OS at which the train enters the dispatching area and the name of the main signal S at which the train enters the dispatching area. The message format is thus defined as "1 | predictedTimeOfEntry | TN | t_p | OS | S".

During operation, this message can be sent prior to the time of entry. If predicted times of entry into the dispatching area change after the message was sent, the message "predictedTimeOfEntry" should be resent with the updated prediction.

When receiving a "predictedTimeOfEntry" message, the dispatching system has to process this information. This way, the train's adjusted predicted trajectory can be updated to further improve the quality of early conflict detection and conflict resolution.

Synchronization of the system time

As defined in the requirements in section 4.1, the system times of both the dispatching system and the railway lab control software are required to be synchronized. Therefore, prior to the start of operation, the system time has to be initially synchronized. Additionally, the system time should be synchronized in frequent time intervals to avoid divergent internal times within the dispatching system and the railway lab control software. Time synchronization can theoretically be triggered by either the dispatching system or the railway lab control software. To achieve a broader compatibility with different dispatching systems, time synchronization was chosen to be performed by the railway lab control software (see section 6.1 for details).

The type of message used for time synchronization is called "synchronizeTime" and the message format consists of one parameter t containing the system time used within the railway lab control software. Thus, the message format to transmit the current system time is "1 | synchronizeTime | t" where the time t is given in the format "hh:mm:ss" with "hh"

representing the hours in a 24-hours-format, “mm” representing the minutes and “ss” representing the seconds.

Before real-time operation in the railway lab is started, the message “synchronizeTime” has to be initially sent to the dispatching system to synchronize both internal system times. Additionally, this information should be resent at frequent time intervals, e.g. once every 60 seconds, to avoid small time deviations during operation over longer periods of time and to ensure that both system times are constantly synchronized.

Once a “synchronizeTime” message is received by the dispatching system, its internal system time has to be set to the received time. The first “synchronizeTime” message can also be used to indicate the start of operation to the dispatching system, thus no additional message format is required for this purpose.

Request dispatching

In order for the dispatching system not to constantly recalculate its internal dispatching timetable, conflict resolution and dispatching should be explicitly requested by the railway lab control software.

This type of message is called “requestDispatching” and has no parameters. The message format is given by “1 | requestDispatching”.

When receiving this message, the dispatching system has to perform occupation conflict resolution and forward the resulting timetable changes to the railway lab control software.

Table 2 summarizes the five presented types of messages sent from the railway lab control software to the dispatching system, their message format and required parameters. The column condition describes when the respective message is sent.

Table 2: Messages sent from the railway lab control software to the dispatching system

Type of message	Message format	Parameters	Condition
reachedSignal	1 | reachedSignal | TN | OS | S | t	TN: train number, OS and S: operating station and signal name of reached signal t: time the train reached the signal	The first axle of a train reaches the first axle counter after a main signal
trainRouteClaimed	1 | trainRouteClaimed | TN | OS_s | S_s | OS_d | S_d	TN: train number, OS_s and S_s: operating station and signal name of starting signal, OS_d and S_d: operating station and signal name of destination signal	A route for a train is set by the interlocking system
predictedTimeOfEntry	1 | predictedTimeOfEntry | TN | t_p | OS | S	TN: train number, t_p: predicted time of entry, OS and S: operating station and signal name at entry position into the dispatching area	Prior to the predicted time of entry; the predicted time of entry is updated
synchronizeTime	1 | synchronizeTime | t	t: current system time	Start of operation; updates on the system time during operation (e.g. at constant time intervals)
requestDispatching	1 | requestDispatching	none	Dispatching is requested by the railway lab control software (e.g. at constant time intervals)

4.2.2 Messages from the Dispatching System to the Railway Lab Control Software

In order to implement the dispatching measures determined by the dispatching system on the railway lab infrastructure (see 4.1), message formats for messages transmitted from the dispatching system to the railway lab control software must be defined.

These messages are defined to start with the message part “0”, to indicate the direction of information exchange. The type of message is encoded in the second message part and the remaining message parts contain the parameters specific to the respective type of message. The dispatching measures considered in this research work are bending, changes in dwell times, cancellation or adding of stops and route changes through operating stations. Short turning, early termination, cancellation of train runs or major changes in train routes along a different set of stations are not considered, as quantifying the delay of a train at a station which it never reached can be very complex, in particular if there is no later train run serving

as an alternative. Note that, although major rerouting of passenger trains is normally not intended during real operation, it can theoretically be applied for freight trains in case only the final destination of the freight train is relevant.

Two types of information suffice to encode the utilized dispatching measures. A detailed discussion of these two message types and how they allow encoding of these dispatching measures is given below.

Changes in arrival or departure times

It is assumed that for each operating station along a train run through the dispatching area, there is a unique reference element in the infrastructure (e.g. a main signal or a predefined stopping position) serving as reference point for the train's arrival in the operating station. The time the train arrives at the respective infrastructure element is thus considered as its arrival time in the operating station. In case the train does not stop at this infrastructure element, its departure time in the operating station equals its arrival time. In case the train stops at the reference infrastructure element, its scheduled departure time is considered as its departure time in the operating station. Arrival times are determined by the time the train's first axle reaches the reference element.

Figure 11 illustrates an example with two consecutive operating stations and two different train routes through these operating stations. The train routes' respective reference elements within the operating stations are displayed in the same color as the train routes. In each operating station every train route has a unique reference element. The train route associated with the orange line stops in the left operating station and its stopping position serves as reference point, thus its arrival time in the left operating station coincides with its arrival time at the stopping position. For the train associated with the green line, which is not scheduled to stop in either of the two operating stations, the respective arrival times coincide with the times it reaches the main signals marked in green, i.e. the times at which its first axle passes the axle counters located after the respective main signals.

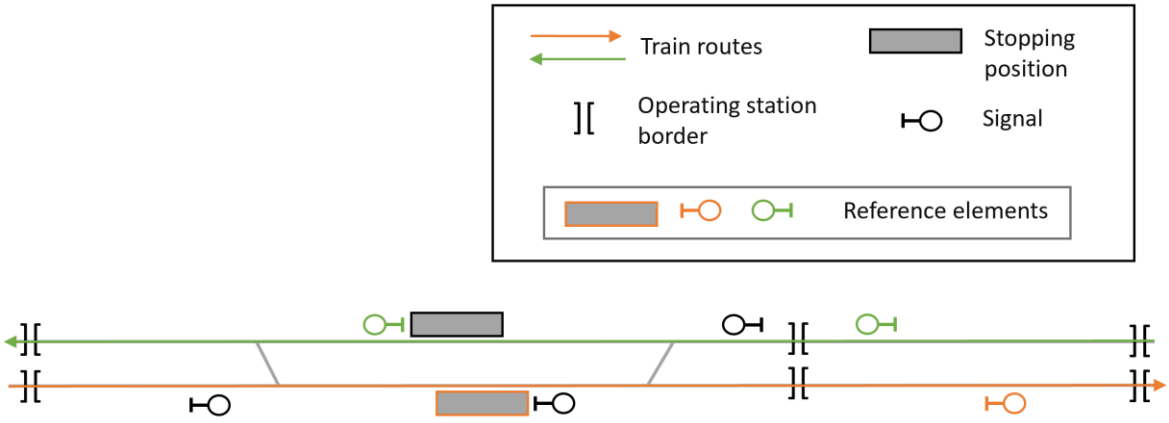


Figure 11: Two train routes through two consecutive operating stations and their respective reference elements

As bending of a train run is considered between subsequent operating stations, this dispatching measure can easily be transmitted to the railway lab control software by sending the trains' new arrival times in the affected operating stations. An advantage of this kind of data transmission is that the information about new arrival times is already available by the end of the dispatching system's optimization process and can directly be sent to the railway

lab control software, avoiding any additional time-consuming calculations. New arrival times can then be processed by the railway lab control software for the adjustment of the trains' speed profiles and thus suffice to precisely transmit information about bending.

In case a train's dwell time in a station is changed during the dispatching process, the train has to depart in the station earlier or later than scheduled. A changed dwell time can easily be forwarded to the railway lab control software by sending the new departure time in the respective station.

The cancellation of an operating stop or a scheduled stop without a minimum dwell time in a station results in the train's departure time equaling its arrival time. In this case, only the information on the cancellation of the stop does not suffice, as the braking process, which was previously necessary before stopping, becomes redundant, thus enabling the train to optionally achieve an earlier arrival time if needed. Hence, in case a scheduled train stop is cancelled, the new arrival time in the station should be sent from the dispatching system to the railway lab control software. The new departure time has to be sent as well. By equaling the arrival time, it indicates to the railway lab control software that the stop is omitted. The main signal which the time refers to becomes the new reference element within the operating station, replacing the stopping position which previously served as reference element.

When an additional stop in a station was scheduled by the dispatching system, the train's arrival time will change due to the required additional braking process. Also, its departure time in the station will differ from its arrival time where the time difference between the two times equals its dwell time. Therefore, in case of an additional stop both the new arrival and departure time of the train in the station need to be forwarded to the railway lab. Instead of the main signal, which was previously defined as reference element, the new stopping position now serves as reference element within the operating station.

Bending, i.e. increasing or decreasing the running time along a certain section, changes in dwell times, as well as the cancellation or adding of stops, all yield changes in arrival or departure times. One single message format suffices to transmit these four dispatching measures, making use of their similarities. As many dispatching systems are applying a mathematical programming approach for holistically solving occupation conflicts (see also [33] and [34] for a detailed overview on mathematical programming approaches for train dispatching), they do not necessarily apply or distinguish between any of these dispatching measures, but rather find optimal values for the arrival and departure times of all trains, following the predefined constraints. It can therefore be computationally difficult to reconstruct explicit dispatching measures from time changes originating from the solution of a mathematical programming problem. Hence, having only one message format for any change in arrival or departure times, independent of the type of dispatching measure, ensures efficient transmission of the relevant information, regardless of the solution algorithm implemented within a certain dispatching system. Note that it can be necessary for the dispatching system to send multiple messages at once, in case more than one operating station or more than one train is affected.

The type of message that is transmitted in case of bending, changes in dwell times and the cancellation or adding of stops is called "timeChange". Its four additional parameters are the train number TN, the operating station OS the time change is applied to, as well as the new arrival time t_a and departure time t_d at the reference element. In case only one of the times is changed, the other one remains the same, but is nonetheless transmitted to comply with the message format. The message format is thus given by "0 | timeChange | TN | OS | t_a | t_d".

The railway lab control software needs to process incoming time changes and implement them during train operation. Section 6.5 describes in detail how time changes can be processed by the railway lab control software.

Change of a train route

Changes of train routes are another option to solve occupation conflicts, especially in railway stations where trains are scheduled to stop on the same track or to enable overtaking of a slower train by a faster train.

There are different possible alternatives for transmitting route changes from the dispatching system to the railway lab control software. One way is to define a train route by its starting and destination signal and, if needed for uniqueness, the switches along the route. Changes of train routes can then be transmitted by providing a (partially) updated list of all signals and switches along the train's designated path. This section presents another possible implementation of the transmission of route changes, which is based on operating stations. The whole network is assumed to be subdivided into operating stations. These operating stations include train stations, block sections, crossovers and junctions [137]. A train route then corresponds to a feasible path through such an operating station.

Figure 12 shows an example of a train route through an operating station that has been changed from the previously planned route.

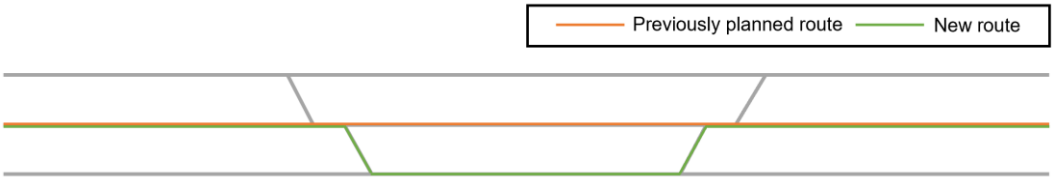


Figure 12: Change of a train route through an operating station

It is possible for train routes to be changed over multiple consecutive operating stations, as long as the resulting train run remains feasible, i.e. all routes of the train run have to be connected and possible to be set by the interlocking system. This way, running on the opposite track is possible. Figure 13 illustrates an example of a train route that was changed over three consecutive operating stations. In this case, each of the route changes in one of the operating stations must be forwarded to the railway lab.

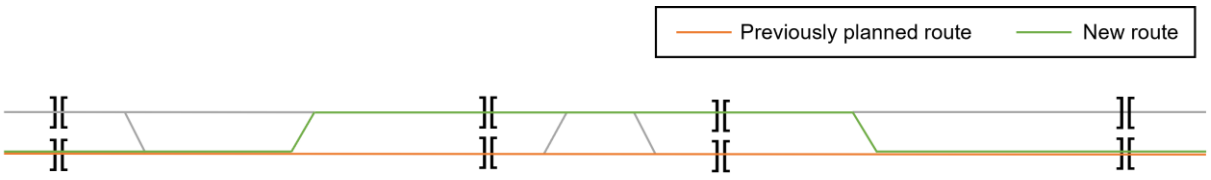


Figure 13: Change of a train route over multiple consecutive operating stations

It is assumed that each route through an operating station is uniquely encoded via a string. One possible way to encode a route through an operating station is via a string that is composed of the main track number the route passes through, as well as the names and neighboring track numbers of the preceding and succeeding operating stations. Optionally, the switches along the route can also be encoded in this string if uniqueness of the route is

not given. Additionally, the name of the operating station the route change is applied for should be transmitted to the railway lab control software.

As an example, consider the setup shown in Figure 14. Assume that a train route through track 1 of operating station OS_1 originates from track 2 of operating station OS_o and leads to track 1 of operating station OS_2. The train route can be encoded by "OS_o (2) – 1 – OS_2 (1)". However, uniqueness of the train route is not guaranteed, as a train route fulfilling these conditions can either run along the straight paths of switches Sw_3 and Sw_5, as indicated by the orange dashed line, or it can run along the diverging paths of switches Sw_3, Sw_4, Sw_6 and Sw_5, as indicated by the green dashed lines. In such a case, uniqueness can be achieved by appending a list of switches that are required to be passed along their diverging path at the end of the string. The train route along the green dashed lines can e.g. be encoded by "OS_o (2) – 1 – OS_2 (1); (Sw_3, Sw_4, Sw_6, Sw_5)" while the above encoding without the list at the end suffices to encode the train route along the orange dashed line.

Figure 14: Train routes along different sets of switches

The type of message is called "routeChange". Its three parameters are the train number TN, the name of the operating station OS and the string str_{route} encoding the new train route for TN through OS. The message format is given by "0 | routeChange | TN | OS | str_{route}". As this message transmits only single route changes in one operating station, separate messages must be sent for the change of train routes over multiple operating stations or in case of route changes for several trains simultaneously.

Incoming route changes must be processed and implemented by the railway lab control software during operation. Section 6.5 describes in detail how this is achieved.

Table 3 summarizes the two types of messages sent from the dispatching system to the railway lab control software. The messages should be sent right after a conflict resolution was performed by the dispatching system.

Table 3: Messages sent from the dispatching system to the railway lab control software

Type of message	Corresponding dispatching measure	Message format	Parameters
timeChange	Bending between operating stations; change of dwell time; cancellation of a train stop; adding of a train stop	0 | timeChange | TN | OS | t_a | t_d	TN: train number, OS: operating station, t_a: new arrival time at the reference element, t_d: new departure time at the reference element
routeChange	Change of a train route through an operating station	0 | routeChange | TN | OS | str_{route}	TN: train number, OS: operating station, str_{route}: string encoding the new route through the operating station

5 Requirements for a Testing Environment and Existing Fundamental Railway Lab Control Software

The general requirements for a physical railway lab and a railway lab control software, required for a realistic testing environment of dispatching algorithms, are discussed in sections 5.1 and 5.2 respectively.

In section 5.3, the framework and main functionalities of existing fundamental railway lab control software are described to give an overview of presently applied railway lab control software.

5.1 Requirements for a Physical Railway Lab

This section briefly describes the requirements for a physical railway lab in order to provide a testing environment for dispatching algorithms while a railway lab control software is connected to the railway lab. It is presumed that the railway lab provides a physical infrastructure containing tracks, switches, signals, as well as sufficiently many physical vehicles and power supply is provided to both the infrastructure and the vehicles.

Automatic setting of switches and signals via a software interface is necessary to avoid timely manual setting. One possibility to achieve this is via the setup of a CAN bus network [138] where all switches and signals are connected to the network and bi-directional message exchange enables communication between the infrastructure and the control software. This way, switches and signals can be set and conversely, the current state of these infrastructure elements can be reported back to the control software.

Speed control of the vehicles at different speeds is essential to allow for realistic driving dynamics. Normally, railway lab vehicles provide a range of speed levels where each speed level corresponds to a different speed. Again, a possible implementation is the use of a CAN bus network enabling the control software to set these speed levels.

Moreover, a possibility for track vacancy detection must be provided to safely set and release train routes. There are two main types of track vacancy detection. The first one is vacancy detection via track circuits [139], as shown in Figure 15. A power supply provides electrical current which flows through the rails. If no train is present within the track section, the electrical current reaches a signalling relay which results in the section being detected as vacant. In case a train is present within the track section, its axes will generate a short circuit, hence the electrical current does not reach the signalling relay and the section is detected as occupied. The track sections are separated by small, insulated gaps and the track circuits of neighboring track sections have reverse polarity.

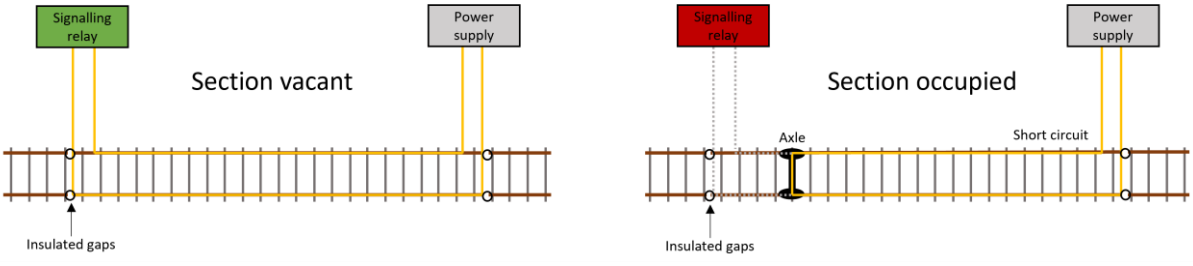


Figure 15: Vacancy detection via track circuits

The second option for track vacancy detection is via axis counting circuits [140]. As shown in Figure 16, the infrastructure needs to be subdivided into axis counting circuits which are delimited by axle counters. For each axis counting circuit, the number of currently present axes within the circuit is monitored by counting the number of axes entering and leaving the circuit respectively. If the total number of axes is 0, the axis counting circuit is vacant, otherwise it is occupied.

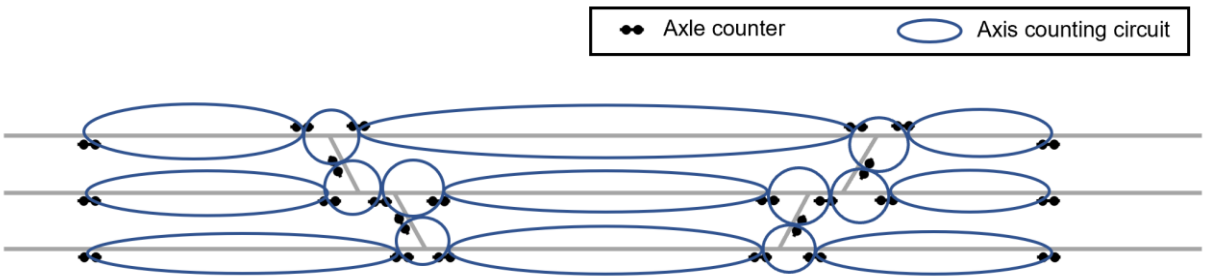


Figure 16: Division of a station into axis counting circuits

From a technical point of view, both forms of track vacancy detection can be implemented in railway labs. However, the installation of power supplies and signalling relays for each track section is complex and incorporates immense costs. Thus, it is reasonable to achieve track vacancy detection via axis counting circuits. Note that the physical vehicles must be detectable by the axle counters, e.g. via magnets attached to the bottom of the vehicles.

Additionally, a possibility for train localization is required, however, it does not necessarily need to be continuous. If axle counters are installed on the railway lab network, these can serve as possible synchronization points of the current train position, when passed by a train.

Table 4 summarizes the requirements for a physical railway lab described above and possible implementation approaches.

Table 4: Requirements for a physical railway lab

Requirement	Possible implementation
Control of switches and signals via a software interface	Setup of a CAN bus network; control of the infrastructure elements via messages sent over the network
Speed control of the vehicles via a software interface	Available speed levels set by the control software via a CAN bus network
Track vacancy detection	Installation of axle counters; definition and supervision of axis counting circuits
Train localization	Synchronization of train positions when axle counters are passed

5.2 Requirements for a Railway Lab Control Software

This section describes the requirements for a railway lab control software to provide a testing framework for dispatching algorithms. It is assumed that a physical railway lab and vehicles, as described in 5.1, exist and communication interfaces to control the switches, signals and vehicles, as well as track vacancy detection are available.

A microscopic infrastructure model of the railway lab network, information on all vehicles, as well as the applied timetable concept, serve as input for the control software. These information need to be provided to the control software. One possibility for the data provision is via XML (Extensible Markup Language) files [141] which were first introduced by the World Wide Web Consortium in 1998. XML files are a widely used file format to store and exchange data. The main advantage of this file format is its platform independency, making it suitable to be utilized for a railway lab control software, as input data can originate from different sources, e.g. different tools for editing railway infrastructure or for railway timetabling. No specific software is required to open XML files, as they only contain text and can thus be read and edited with any text editor. Moreover, they are easily readable and editable by humans, as the names and tags used in the files normally describe the content. Small instances of data can thus be edited or modified manually. XML files also allow for the definition of own customized tags, allowing for an extremely versatile use. Examples for XML file formats are given in Appendix A for infrastructure data, in Appendix B for train data and in Appendix C for timetable data. The railway lab control software has to read and process the provided data.

The state of the physical railway lab infrastructure needs to be constantly synchronized and supervised within the control software. As described in section 5.1, a CAN bus network can provide a possible way of data transmission. The received information then need to be stored and updated within the control software.

To operate trains on the network, a possibility to set and release train routes, while preventing conflicting routes from being set, is required. Using the information from the train vacancy detection, vacancy of a new train route can be checked prior to the route being set.

In order to provide a realistic testing environment, acceleration and deceleration processes of different types of trains have to be modeled realistically through the available speed levels. Speed profiles need to be determined and dynamically updated, based on restrictions through the existing infrastructure or dispatching measures. To ensure the reliable control of the trains on the network, current train positions should be constantly predicted and frequent synchronizations between the vehicles' actual positions on the physical infrastructure and their positions predicted by the control software, need to be performed. If axle counters are installed, synchronization of train positions can be performed upon the passing of axle counters and in-between axle counters, the position of trains can be predicted via the applied speed levels.

Trains on the railway lab infrastructure must be operated according to a given timetable. One possible implementation to achieve operation based on a given timetable is the utilization of a specific timetable module within the railway lab control software which ensures that both scheduled or rescheduled arrival and departure times are met and the correct train routes are set at the right time. Train routes set too early prevent other routes from being set, potentially causing further delays, whereas train routes set too late cause the train to brake unnecessarily. To avoid such inaccuracies caused by train routes being set manually, the scheduled train routes should be set automatically.

Dispatching measures from a dispatching system (see 4.2.2) need to be processed and considered during operation. Conversely, the information required by the dispatching system (see 4.2.1) have to be continuously provided by the control software.

Furthermore, it should be possible to consider delay data on delays at entry and primary delays during operation, while ensuring that the delays are not accessible to the dispatching

system. Additionally, to perform evaluations and compare different dispatching approaches, the recording and exporting of delays achieved during operation is required.

Table 5 summarizes the requirements for a railway lab control software, possible implementation strategies and the sections in which these implementations are described in detail.

Table 5: Requirements for a railway lab control software

Requirement	Possible implementation	Sections
Data on the infrastructure, vehicles and timetable	Provision of data via XML file interfaces	5.3.1, 5.3.5, 6.5
Synchronization and supervision of the state of the infrastructure	Processing of messages on the state of the infrastructure received via a CAN bus network	5.3.2
Prevention of the setting of conflicting train routes	Processing of the information from a track vacancy detection	5.3.2, 5.3.3
Realistic train speeds	Realistic modeling of acceleration and deceleration curves through speed levels; definition and dynamic adjustment of speed profiles	6.3, 6.4
Frequent synchronization and constant prediction of train positions	Synchronization of the current train position when axle counters are passed; prediction via the applied speed levels in-between axle counters	6.4
Operation according to a given timetable	Management of departure and arrival times as well as automatic setting of train routes at the correct time	6.5, 6.6
Continuous communication with a dispatching system	Setup of a communication interface between the railway lab control software and a dispatching system	4, 6.2
Consideration of delay data during operation	Provision of delay data via a file interface and processing by the control software during operation	6.5
Recording and exporting of delays achieved during operation	Storing of achieved arrival and departure times; file interface to export the delays	6.5

5.3 Fundamental Railway Lab Control Software

The following sections describe a fundamental software framework as an example of an existing presently applied railway lab control software. The control software contains different software modules, each serving a specific purpose. The focus of this section is on the general functionalities required for the setup of the testing environment for dispatching algorithms, hence additional functions existing within these software modules, which are not serving the testing framework, are not discussed.

A railway lab control software requires various types of data. Storing the data in external files enables flexible adjustments of the setup, e.g. the infrastructure, the vehicles or the timetable can be interchanged, while the software and its logic remain unchanged. XML files provide one possible file format (see 5.2).

5.3.1 Infrastructure Model

A fundamental railway lab control software requires information on the positions of certain infrastructure elements, such as signals, switches and axle counters. Thus, a microscopic model [131] of the railway lab infrastructure is essential. The infrastructure data and information on the physical vehicles can be provided to the control software via an XML interface which uses the German infrastructure graph format proposed in [142]. The infrastructure model contains several infrastructure elements with specific parameters and their respective position on the network infrastructure. An example for such a format is given in Appendix A. Elements in the microscopic infrastructure model include main signals, switches, axle counters, operating station borders, stopping positions and speed limit signs. The network is subdivided into operating stations. Operating stations are further subdivided into so-called inter-switch segments corresponding to the part of the infrastructure between two neighboring switches.

Within the microscopic model, all elements possess a unique integer ID. Within an inter-switch segment, the ID of neighboring microscopic infrastructure elements in direct succession either increases or decreases by 1, depending on the current counting direction. A switch itself is modeled as three separate infrastructure elements: the beginning of the switch, as well as its straight and diverging paths. Before the beginning of a switch, as well as from its straight and diverging path on, the IDs are increased or decreased by 1 respectively, until the next switch. This property is used to uniquely identify neighboring infrastructure elements. Figure 17 shows an example extract of such an infrastructure model.

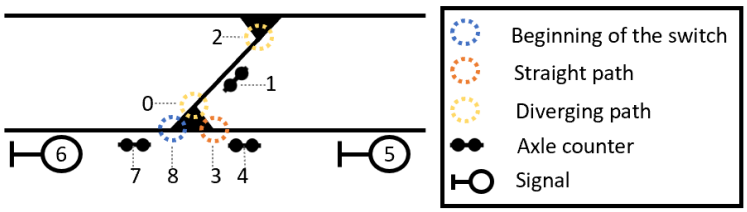


Figure 17: Unique IDs of infrastructure elements in the microscopic model

In the XML file, which is imported into the railway lab control software, for each beginning of a switch, the corresponding IDs of its straight and diverging paths are defined. Thus, the control software can easily identify neighboring infrastructure elements along a train route, based on the current or intended positions of switches.

It is assumed that the network infrastructure is subdivided into disjoint axis counting circuits. Note that a division of the network into axis counting circuits does not necessarily require all axle counters to serve as delimiters of an axis counting circuit.

5.3.2 Field Level

In a fundamental railway lab control software, the field level is a software module which supervises and modifies the state of the network infrastructure and provides these information to other software modules. It stores the current state of all switches and signals. Whenever a switch position or signal aspect on the physical network infrastructure changes, this information is sent to the field level, e.g. via a CAN bus network and processed by the field level. Conversely, the field level also sends commands to set switch positions or signal aspects on the physical network infrastructure that are required e.g. for route setting or route releasing.

Furthermore, each passing of an axle counter and the direction of passing is transmitted to the field level. When an axle counter delimiting two axis counting circuits is passed, the number of axes in the respective circuits is updated. Figure 18 visualizes an example: a train is entering an axis counting circuit. Each time one of its axes passes the delimiting axle counter, a message containing the information on the direction of passing is transmitted from the axle counter to the field level. The number of axes in both affected neighboring axis counting circuits is updated, increasing the number of axes in the circuit the train is entering by 1 and decreasing the number of axes in the circuit the train is leaving by 1.

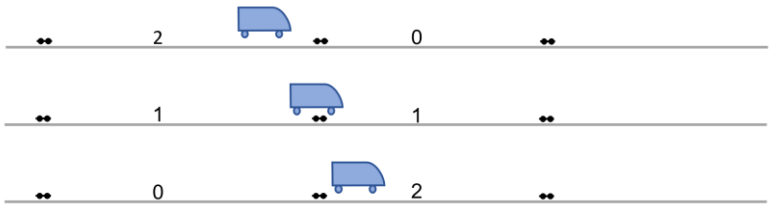


Figure 18: Train entering an axis counting circuit and number of axes stored in the field level

The information on switch and signal states, as well as on the axis counting circuits, are thus managed by the field level. Information on the current infrastructure state are required by other software modules and must therefore be accessible from the field level. These information can e.g. be provided via the User Datagram Protocol (UDP) [143]. States of switches and signals are provided to the interlocking system module (see 5.3.3) and the train control module (see 5.3.5). Passed axle counters are also communicated to the train control module to enable the synchronization of train positions (see 6.4). Additionally, states of axis counting circuits are provided to the interlocking system module such that train routes can be released and new routes can be set, given that all axis counting circuits along the designated route have been cleared, and to the automatic route setting module (see 6.6). Figure 19 summarizes which information are exchanged between the field level, the physical infrastructure of the railway lab and the software modules of the railway lab control software.

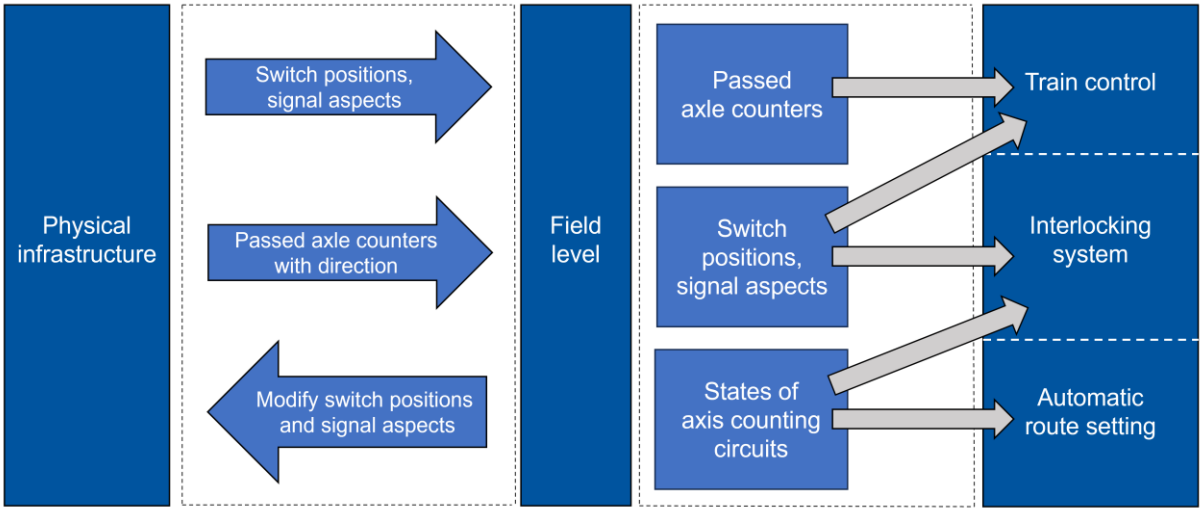


Figure 19: Overview of the data exchange between the field level, the physical infrastructure and other software modules

5.3.3 Interlocking System

In railway labs, train routes can normally be set manually through original interlockings or requested to be set via an interlocking system module within the railway lab control software. Both the original interlockings and the interlocking system module set train routes while ensuring rail safety by avoiding the simultaneous setting of conflicting routes.

The states of switches and signals, as well as the information from the train vacancy detection, are synchronized with the field level (see 5.3.2). When a train route is requested, the interlocking system utilizes a train route search module (see 5.3.4) to receive lists containing all switches and their intended positions, intermediate signals, as well as axis counting circuits along the requested route. In case none of the switches or signals are currently locked and all axis counting circuits along the route are cleared, the train route can be set. Switch positions are changed by forwarding the intended positions to the field level and the switches are locked. All intermediate signals along the route are also locked. Locking the switches and intermediate signals ensures that no other train route can be set along these switches, until the train route has been released or partially released. Partial releasing unlocks switches or intermediate signals such that other, non-conflicting train routes can be set before the complete train route is released. Once all switches and intermediate signals are locked, the signal aspect of all intermediate signals and the starting signal is changed to “proceed”. Signal aspects indicating speed restrictions, e.g. “proceed at restricted speed” can also be set. As soon as the axis counting circuit behind the starting signal is occupied by the train, the signal aspect is changed to “stop”. Additionally, trains can run in shunting mode and train routes for shunting can be set by the interlocking system.

If a route requested by the automatic route setting (see 6.6) cannot be set, it can be stored in a waiting list and set as soon as setting is possible. In case of several conflicting routes in the waiting list, the routes can e.g. be set in the order in which they were requested or they can be given a priority, e.g. routes for high-speed passenger trains could be prioritized over routes for freight trains.

5.3.4 Train Route Search

In railway labs, train routes can normally be set manually via original interlocking systems. Additionally, a fundamental railway lab control software often provides the possibility to set requested train routes, which are usually referenced by their starting signal and destination element, automatically. For this functionality, all switches, axis counting circuits and intermediate signals along the requested route have to be determined. This task can be performed by an integrated train route search module.

This section describes a train route search module of a fundamental railway lab control software, required to search for train routes on the network. The infrastructure of the railway lab is modeled through a microscopic infrastructure model (see 5.3.1) and provided to the railway lab control software e.g. via an XML file interface where the infrastructure elements are stored in such a way that neighboring infrastructure elements can easily be determined. The train route search is a software module designated to determine all switches, intermediate signals and axis counting circuits along train routes and to provide these information to the interlocking system (see 5.3.3). This section describes two possible search algorithms that can be implemented within a train route search module of a fundamental railway lab control software and the communication with the interlocking system module.

Requests for train routes that are not set manually via original interlockings, are sent to the railway lab control software's interlocking system module. The interlocking system module first forwards the request to the train route search module which determines all switches with their respective target positions, signals and axis counting circuits along the route. Train routes are identified via their starting signal and destination infrastructure element. A destination infrastructure element can either be a signal, a border to another operating station or a bumper. The infrastructure is required to be modeled such that no switches or intermediate signals exist along a train route after an operating station border until the next main signal, at which the next axis counting circuit begins. Thus, the remaining train route from the operating station border until the next main signal of the adjacent operating station does neither contain any switches or intermediate signals, nor does a new axis counting circuit start until this next main signal. When a route having an operating station border as destination element is set for a train, the train will therefore continue running beyond the operating station border until the next main signal. Hence, the starting signal and destination element for the train route search can be assumed to belong to the same operating station and it is feasible for the search algorithm to terminate at the operating station border, which reduces the complexity of the search algorithm. Figure 20 shows an example. Both the starting signal and the destination element are highlighted in green, where the destination element corresponds to an operating station border. A train route, which is highlighted in orange, from the starting signal until the destination element was found by the train route search. This shortened route suffices to provide all information on signal states and target switch positions, as well as the axis counting circuits intersecting the train route, required for the setting of the train route, to the interlocking system. When the train route is set, the train will continue running along the train route highlighted in blue until reaching the next main signal.

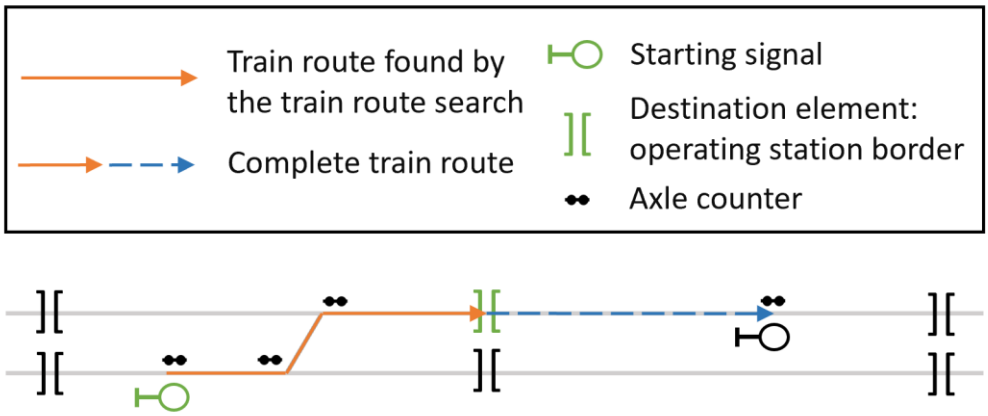


Figure 20: Train route search from a starting signal until an operating station border

Note that the train route search is only performed from the starting signal until the nearest destination element, e.g. the nearest main signal or operating station border. If train routes shall be set via multiple destination elements simultaneously, the train route search needs to be called separately for each pair of starting signal and destination element.

A train route search can use a depth-first search algorithm [144] to generate all feasible train routes with the given starting signal and destination element. During the route search, temporary lists of all infrastructure elements, all switches and their positions, all intermediate signals and all axle counters along the current route are stored and maintained. For each switch, a binary variable is used to store whether this switch has already been searched completely. Once a feasible route is found, the current temporary lists are stored in a list of found train routes.

The search algorithm starts at the starting signal and stores the neighboring infrastructure element, in the direction of the starting signal, in the temporary list of infrastructure elements. The infrastructure element is then checked for the following conditions:

1. **The element is a final element**

   A final element is an infrastructure element at which the current iteration of the search algorithm terminates. This can be a main signal, in case the route does not cross any border to another operating station, an operating station border or a bumper. In case the element coincides with the destination element, a feasible train route from the starting signal to the destination element was found and the temporary lists are stored in the list of found train routes. Otherwise, if the main signal or operating station border is not the destination element, or if the element is a bumper, the route is not a feasible train route.

2. **The element is a switch that can be passed in two diverging directions**

   The search is continued with the next neighboring element along the straight path of the switch and the switch together with its position “straight“ is stored in the temporary list of switches.

3. **The element is an intermediate signal**

   The element is stored in the temporary list of intermediate signals and the search is continued with the next neighboring element.

4. **The element is an axle counter**

   The element is stored in the temporary list of axle counters and the search is continued with the next neighboring element.

5. **The element is neither of the above infrastructure elements**

   The search is continued with the next neighboring element.

Once a final element has been reached, regardless of whether the route was feasible or not, backtracking is performed. The last switch within the current temporary list is considered. If it has not been completely searched, all elements in the temporary lists that were traversed after this switch, are removed and the search is continued with the next neighboring element of the switch along its diverging path. The position of the switch stored in the list of switches is changed to “diverging” and the switch is considered as completely searched. All succeeding infrastructure elements after this switch are handled as described above. If the switch has already been completely searched, backtracking is further continued until the next switch in the temporary list of switches. Note that switches removed from the temporary list of switches during backtracking are not considered as completely searched anymore. Backtracking is repeated, until all switches in the temporary list are completely searched.

Figure 21 shows an example of the depth-first search algorithm from signal A to signal B. The train routes between signals A and B are determined as described above. The circled switches mark the completely searched switches after the respective iteration of the search algorithm. First, for switches 1, 4 and 5, the search is continued along the straight path until a final element, the main signal C, is reached. This route is not feasible, as the final element does not coincide with the destination element. Backtracking is performed until switch 5, the last switch along the route, and the diverging path of switch 5 is searched until signal B is reached, resulting in a feasible train route from signal A to signal B. To search for more routes, backtracking is again performed until switch 1 and the diverging path is searched until signal D is reached. As this route is not feasible, backtracking is performed again, until switch 3 and the diverging path is searched, until signal C is reached. Again, backtracking is performed until switch 5, continuing the search along its diverging path until signal B is reached and the second feasible train route is found. As all switches along the route, which can be passed in two diverging directions, have been completely searched after the last step, the algorithm terminates and the search is completed.

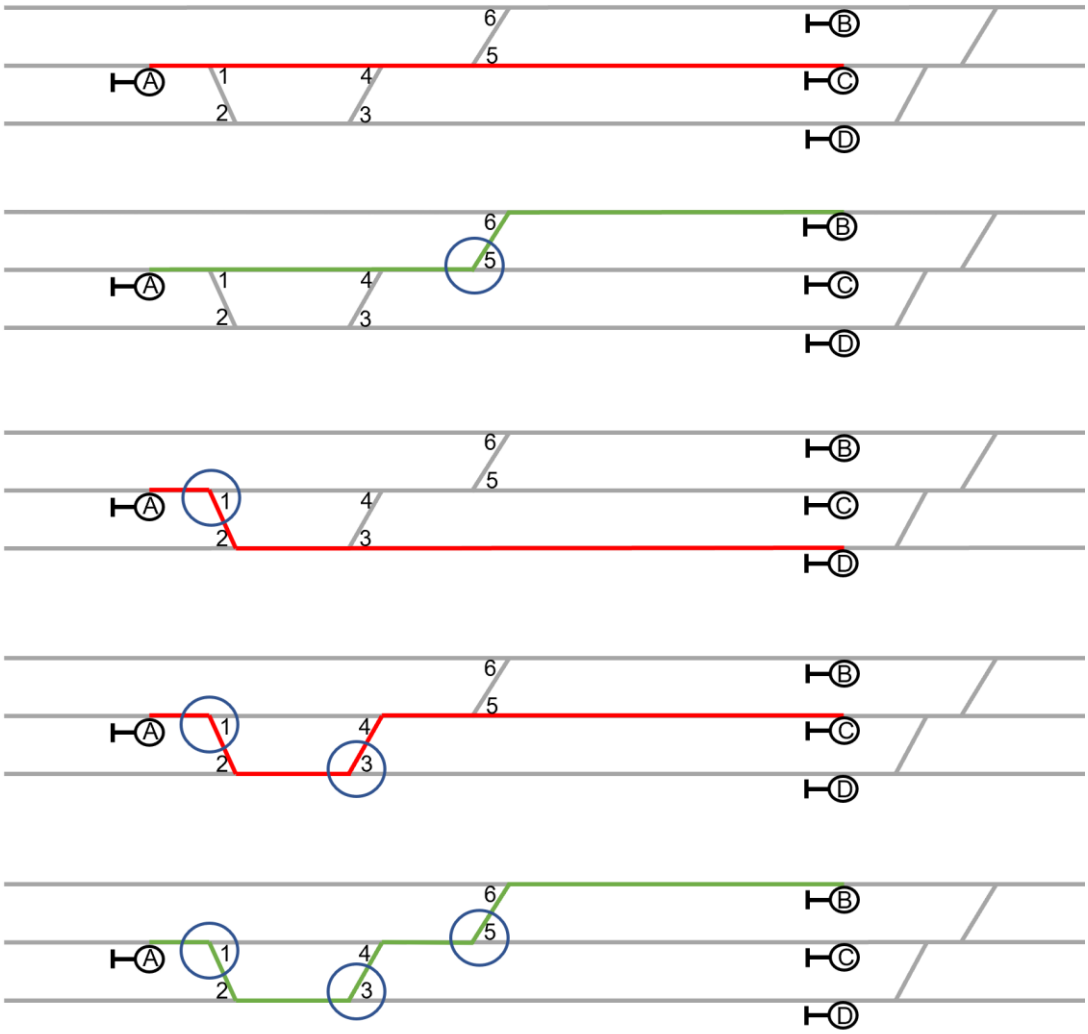


Figure 21: Train route search from signal A to signal B via a depth-first search algorithm along the switches

Alternatively, a breadth-first search algorithm [144] can be applied to find all feasible train routes. Starting at the first switch after the starting element, the idea is to explore both the straight path and the diverging path of the switch during the same iteration of the algorithm and then continue the search from the neighboring switches along these paths. This way, no backtracking is required within the search algorithm.

An example is shown in Figure 22. Starting at switch 1, both the straight and diverging path are followed until the next switches where two paths can be taken, indicated by the red and blue routes in the second step. This leads to switches 5 and 3 respectively. At switch 5, the straight path leads to signal C, indicated by the red route in the third step. The diverging path of switch 5 leads to signal B, as indicated by the yellow route in the third step. At switch 3, the straight path leads to signal D, indicated by the blue route in the third step. The diverging path of switch 3 leads to switch 5, as indicated by the orange route in the third step. From switch 5, the straight path then leads to signal C and the diverging path leads to signal B, as indicated by the orange and purple routes in the fourth step respectively.

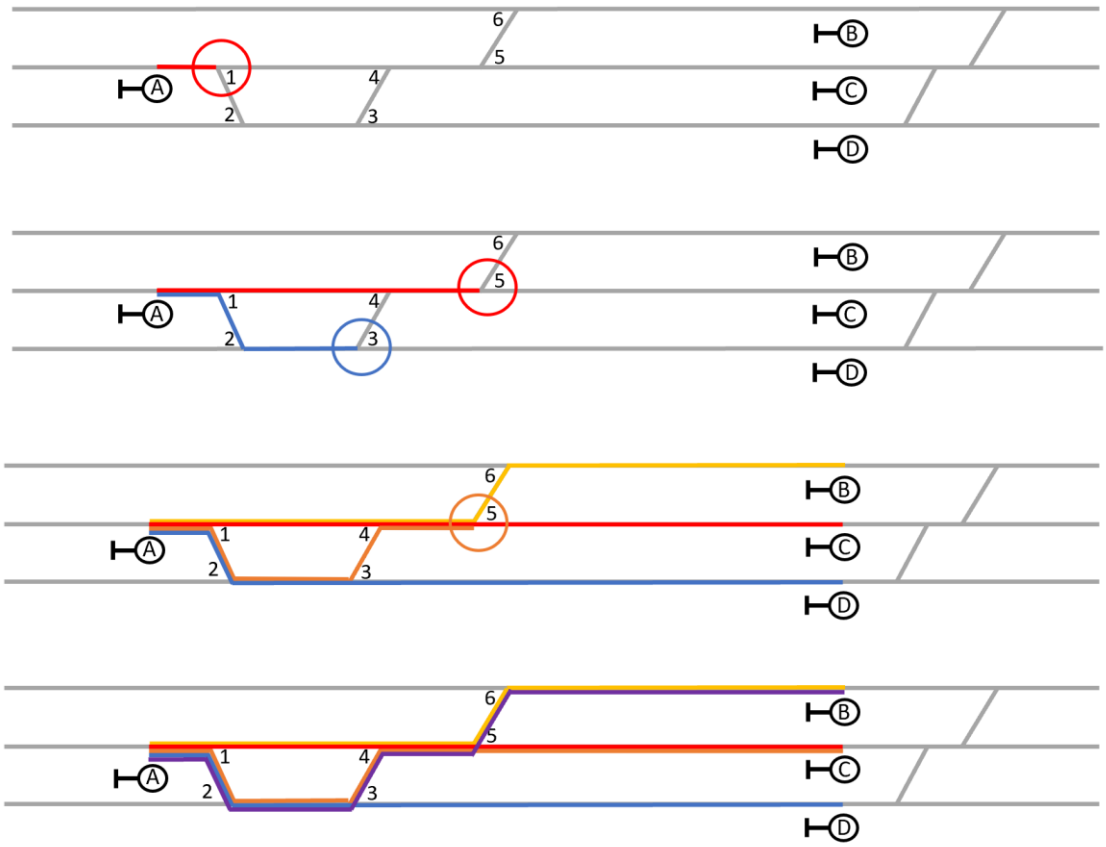


Figure 22: Train route search from signal A to signal B via a breadth-first search algorithm along the switches

As both search algorithms explore all train routes from the starting element until a final element, they have the same running time complexity. However, the breadth-first search algorithm requires more storage capacity, as all train routes, including the infeasible ones, have to be stored simultaneously. The depth-first search algorithm only needs to store the train routes leading to the destination element and the current temporary lists of switches, intermediate signals and axle counters, as well as the binary variables indicating whether the switches in the temporary list have been completely searched. In particular in large stations with many switches, the number of possible train routes is huge. As most of the train routes from the starting element to a final element will not lead to the destination element, these routes are stored unnecessarily and complicate the search algorithm. Therefore, a depth-first search algorithm is recommended for the train route search.

If more than one feasible train route was found between the starting signal and destination element, only one of them can be set. The routes can e.g. be sorted and ranked based on the number of diverging paths taken along switches in an ascending order, i.e. routes with fewer diverging paths are preferred. For all feasible train routes, the axis counting circuits along the respective route are determined. These axis counting circuits are required by the interlocking system to check for vacancy prior to route setting. Only if all axis counting circuits of the route are vacant, the route can be set.

The routes in the sorted list are provided to the interlocking system module via the lists of switches and their target positions, as well as the lists of intermediate signals and axis counting circuits. Upon receiving a train route defined via these lists, the interlocking system module (see 5.3.3) must check if the route can be set.

For a train route to be set, all axis counting circuits along the route need to be vacant, i.e. no other train is allowed to occupy an axis counting circuit along the route. Additionally, all switches along the train route must not be locked. Switches of a train route are locked after route setting until they have been cleared by a train to prevent them from being falsely changed while a train is running along its set route. Lastly, the starting signal and all intermediate signals along the train route must show a “stop” aspect for the new route to be set.

If a route cannot be set by the interlocking system, the next route in the list, that was generated by the train route search algorithm, can be provided to the interlocking system, until either a route is found that can be set or no more feasible route is available. In case none of the routes can be set and the route search was initiated by the automatic route setting (see 6.6), route setting must be performed as soon as there is a feasible route that can be set, i.e. as soon as one of the found routes is cleared (see 5.3.3).

5.3.5 Train Control

This section describes the functionalities of a basic train control software module for a fundamental railway lab control software.

Data on all trains, including the train number, the vehicle’s maximum permitted speed and the number of axes are provided e.g. via an XML file interface which is described in Appendix B.

The train control module uses a microscopic infrastructure model of the railway lab network (see 5.3.1). The field level (see 5.3.2) transmits all state changes of switches and signals to the train control. For each train, an initial starting position in front of a signal within the network is defined. The train control module has to monitor train movements through the network. This can be achieved by maintaining a dynamic “train route list” for each train, containing upcoming infrastructure elements along the route set for the train until the next signal indicating “stop”. The list contains only those infrastructure elements that are relevant for the functionalities within the train control module which include the supervision of trains along their routes, localization of trains on the network and information on whether a train shall currently stop, e.g. in front of a signal or bumper. Signals, switches, axle counters and bumpers are infrastructure elements that are stored in the train route list. Signals are essential, as train route extensions are triggered by signals changing their aspect to “proceed” and the signal aspect indicates whether a train can pass or has to stop in front of the signal. Additionally, information on the state of switches are required for the train control to monitor along which segments of the network trains are running. Axle counters installed on the lab infrastructure network can serve multiple purposes, e.g. the localization of trains or the supervision of speed curves, and are therefore of great importance for the train control. In front of a bumper, the train control needs to ensure that trains come to a complete halt, hence information on bumpers are also essential to the train control. Furthermore, to uniquely identify the current train position within the network infrastructure, information on the line number, the operating station and the kilometrage of each infrastructure element are required. Thus,

positions at which the line number, the counting direction or the kilometrage changes and operating station limits at which the operating station changes are also stored in the train route list. Note that for a fundamental train control, it suffices to let trains run at a designated constant speed, as described below, thus no speed limit signs indicating different maximum permitted speeds are required.

Table 6 shows a train route list as an example. The element ID uniquely identifies the element within the network infrastructure model. Furthermore, the operating station, as well as the name and type of the infrastructure element, are stored. The direction, encoded as a binary variable, refers to the train's driving direction the infrastructure element is associated with. For instance, signals are only valid for one driving direction, whereas axle counters or switches are valid regardless of the direction. In the example below, the main signal N1 lies in the opposite direction and is thus not considered during operation, hence changes of its signal aspect do not influence the current train. The last column in the list below represents the distance between the element and its predecessor. The distance value of the first element refers to its distance from the current train position.

Table 6: Example of a train route list

Element ID	Operating station	Name	Type	Direction	Distance [m]
87	OS_1	F	Main signal	1	46
86	OS_1	G51/31B301	Axle counter	-	1
82	OS_1	W4/G51	Axle counter	-	200
80	OS_1	W4	Switch	-	21
30	OS_1	G41/W4	Axle counter	-	36
28	OS_1	G31/G41	Axle counter	-	70
27	OS_1	N1	Main signal	0	0
21	OS_1	P1	Main signal	1	355

Initially, the train route list contains only the starting signal in front of which the train is located on the infrastructure. As soon as the signal aspect of the last signal in the list changes to "proceed", the list is extended by the infrastructure elements along the newly set route. As all switch positions in the infrastructure model are synchronized with the actual positions, the extension of the train route list is performed by iterating over neighboring infrastructure elements in the direction of the last signal, while considering the current positions of the switches, until a signal indicating "stop" is reached.

Trains on the network are controlled via speed levels. The trains' speed levels are determined by the train control module and forwarded to a digital command station, e.g. via an interface using the UDP (User Datagram Protocol) for communication. The digital command station then sets the speed levels of the physical trains. The railway lab infrastructure is assumed to represent a real rail network in a certain scale. The speed of a train running at a certain speed level and the scale of the railway lab are combined to calculate the actual resembled speed. For instance, a train that runs at 10 cm/s on a railway lab infrastructure in the scale 1:100 is assumed to run at a model speed of 36 km/h.

In a fundamental version of a train control software module, a train starts running as soon as its starting signal changes its aspect to "proceed" and runs at its predefined standard speed until it reaches a signal showing "stop" where its speed is set to 0 km/h. Once this signal shows

"proceed", the train continues its journey. Figure 23 shows an example speed profile of a train where the train's predefined standard speed is 160 km/h. When not stopping, it always runs at its standard speed.

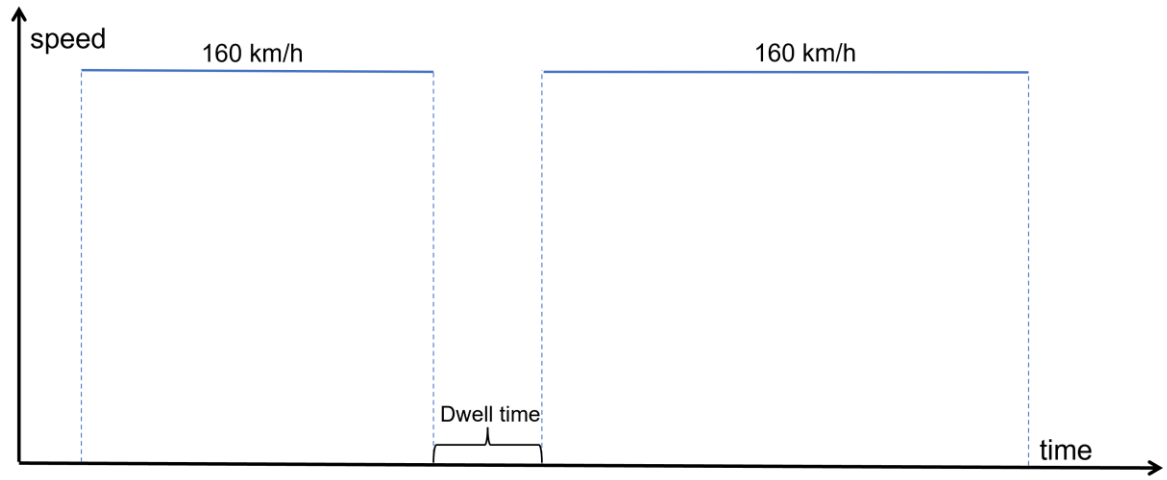


Figure 23: Train speed profile in a basic version of a train control module

Acceleration and deceleration processes are normally not implemented in a basic version of a train control module. Also, different speed restrictions due to certain infrastructure properties are not considered. However, these features are crucial for a realistic testing of dispatching algorithms and were therefore implemented during this research work (see 6.3 and 6.4).

Train positions on the network infrastructure are constantly predicted by the train control module. For this reason, a constant time interval is used. A timer is started for each train, constantly emitting time-out signals in the chosen time interval. For every time-out signal emitted by the timer, the traveled distance of the train is calculated. Let v be the current speed of the train in km/h and t the time-out interval of the timer in milliseconds. Then, the distance in meters traveled by the train during the time-out interval is given by $\frac{v \cdot t}{3600}$. Every time the traveled distance is calculated, it is subtracted from the first entry in the distance column of the train route list. If an element in the train route list was reached by the train, it is removed from the train route list and the remaining distance is subtracted from the distance to the next element.

6 Development of the Software for the Testing Environment

To provide a realistic testing environment for dispatching algorithms, a fundamental railway lab control software (see 5.3) must be extended, and new software modules must be introduced. Figure 24 gives an overview of the architecture of the newly required extended railway lab control software to serve as a testing environment for real-time dispatching algorithms during fully automated operation. The figure includes the information exchanged between the respective systems, software modules and physical components of the railway lab.

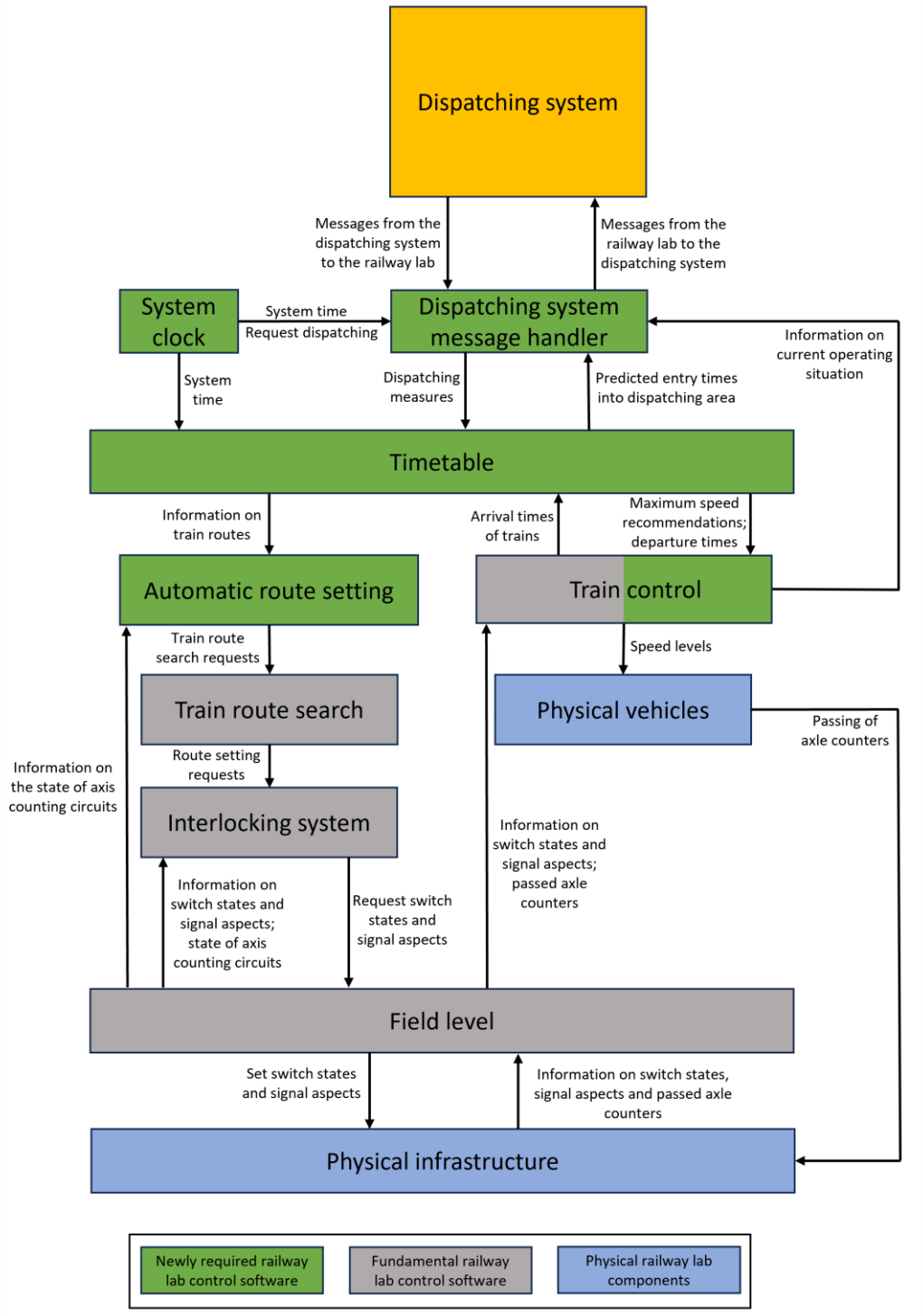


Figure 24: Architecture of an extended railway lab control software

A system clock (see 6.1) provides the mutual system time for both the railway lab control software and the dispatching system. A dispatching system message handler (see 6.2) processes incoming messages from the dispatching system and forwards the received dispatching measures to a timetable module (see 6.5). Conversely, information that will be provided to the dispatching system is first sent to the dispatching system message handler which then forwards the messages to the dispatching system. The timetable module, which manages all information on scheduled and rescheduled train runs, provides information on new recommended maximum speeds and departure times to the train control (see 6.4) which controls the physical vehicles via speed levels. The approaches for realistically modeling acceleration and deceleration, which are implemented within the train control, are described in 6.3. To enable automatic route setting, the timetable module provides information on planned train routes to an automatic route setting (see 6.6) which relies on information about the current state of axis counting circuits that are provided by the field level. When a train route shall be set, the automatic route setting requests the search for feasible train routes from the train route search module (see 5.3.4) which then requests route setting from the interlocking system (see 5.3.3). The field level (see 5.3.2), serving as a communication interface between the physical railway lab and the control software, sets switch states and signal aspects received from the interlocking system. It also provides the interlocking system, the train control and the automatic route setting with information on the current state of the physical infrastructure.

The following sections describe the software modules that were newly developed in addition to a fundamental railway lab control software in detail.

6.1 System Clock

The system clock is a software module within the extended railway lab control software which provides the possibility to set a shared virtual system time between the control software and the dispatching system, independent of the actual system time within the computer, on which the programs are running.

Initially, a starting time can be set manually. Once the starting time has been set, the new time is forwarded to the dispatching system message handler (see 6.2), which then transmits the time to the dispatching system. Additionally, to prevent deviations between the system times within the control software and the dispatching system over long periods of time, the system clock sends the current time to the dispatching system message handler at constant time intervals.

Dispatching needs to be requested frequently from the dispatching system to ensure early conflict resolution. Thus, dispatching requests are sent to the dispatching system via the dispatching system message handler at constant time intervals.

6.2 Dispatching System Message Handler

Incoming and outgoing messages from and to the dispatching system must be coordinated. This task is performed by the dispatching system message handler which is a software module within the extended railway lab control software. This way, only one single communication interface has to be set up, instead of separate interfaces between the dispatching system and each software module exchanging information with it.

For each type of outgoing message, the dispatching system message handler provides a function that can be called by the respective software modules (see 4.2.1 for details on the message definitions). The functions $synchronizeTime(t)$ and $requestDispatching$ can be called by the system clock. The function $predictedTimeOfEntry(TN, t_p, OS, S)$ can be called by the timetable module, whereas $passedSignal(TN, OS, S, t)$ and $trainRouteClaimed(TN,\ OS_s, S_s, OS_d,\ S_d)$ can be called the by train control. When one of these functions for outgoing messages is called, the dispatching system message handler generates a message according to the specific message format definition (see 4.2.1), based on which function was called and the provided parameters. The generated message is then transmitted to the dispatching system via the communication interface.

Incoming messages from the dispatching system are distinguished by their type (see 4.2.2 for details on the message definitions). All parameters are extracted from the received string, e.g. "0 | timeChange | TN | OS | t_a | t_d", which is achieved by separating the message parts delimited by the vertical lines. Then, the dispatching system message handler calls the respective function $timeChange$ or $routeChange$ provided by the timetable module (see 6.5) with the parameters extracted from the message, enabling the timetable module to process the received dispatching measures and provide them to the train control and the automatic route setting.

6.3 Modeling of Acceleration and Deceleration

Train operations in the railway lab should resemble reality as good as possible. Thus, realistic acceleration and deceleration processes must be defined, maintained dynamically and implemented during operation. For each physical train, finitely many discrete speed levels are available where each speed level corresponds to a certain train speed. The number of available speed levels depends on the vehicle and normally ranges from about 50 to 200. Thus, realistic acceleration and deceleration processes have to be modeled via the available speed levels during operation.

The speeds at all available speed levels are assumed to be measured for each train and stored in a train-specific lookup table. The measured speeds are further assumed to be given in km/h with respect to model speed (e.g. for a scale of 1:100, the measured speed 10 cm/s corresponds to a model speed of 36 km/h). Note that this lookup table is specific to a certain vehicle, i.e. even two trains representing the same series with the same parameters can have different lookup tables, based on the measured speed levels of the physical vehicles. Table 7 shows an example extract of a train's speed levels lookup table.

Table 7: Extract of a speed levels lookup table

Speed level	Speed [km/h]
⋮	⋮
14	39
15	41
16	43
17	46
18	50
⋮	⋮

Modeling of the trains and their speed curves

Acceleration curves can be modeled as a composition of small intervals with constant acceleration, as in the running time calculation described in section 3.1. These approximated acceleration curves can e.g. be stored in CSV files. For each interval with constant acceleration, the file contains the lower and upper interval limits.

Table 8 shows an example extract for an IC train with total mass $m = 450$ t, running resistance $R_r = m \cdot 9.81 \cdot (1.67000 + 0.00984 \cdot v + 0.27949 \cdot v^2)$ N, gradient resistance $R_g = 0$ N and curve resistance $R_c = 0$ N. The speed in the i^{th} row represents the lower interval limit for the acceleration value in the i^{th} row and the speed in the $(i+1)^{th}$ row represents the upper interval limit. For instance, an acceleration of 0.498 m/s² is assumed between a speed of 5.00 and 7.98 km/h.

Table 8: Intervals with constant acceleration

Speed v [km/h]	Acceleration a [m/s²]
0	0.503
5.00	0.498
7.98	0.494
11.30	0.490
13.83	0.487
15.95	0.485
⋮	⋮
139.99	0.195
144.99	0.183
145.38	0.182
150.13	0.171
155.13	0.159
156.10	0.158
160	0

This information on the intervals with constant acceleration are used to derive the speed curves for the extended railway lab control software. To achieve synchronization between the running times on the physical railway lab network and the running times assumed by the dispatching system, it is essential that the same acceleration curves are internally applied within the dispatching system.

The elapsed time and covered distance during each interval with constant acceleration can be determined. Let v_{l_i} and v_{u_i} be the speeds in km/h at the lower and upper interval limits of interval i respectively. Further, let a_i be the constant acceleration during this interval in m/s^2. The elapsed time t_i in seconds during interval i is calculated as

$$t_i = \frac{v_{u_i} - v_{l_i}}{3.6 \cdot a_i}.$$

(1)

The covered distance d_i in meters during this interval is calculated as

$$d_i = \frac{v_{l_i}}{3.6} \cdot t_i + \frac{1}{2} a_i \cdot {t_i}^2.$$

(2)

The elapsed time or covered distance while accelerating from a speed v_0 to a speed v_1 can be calculated by adding the elapsed times or covered distances during all respective intervals. Let I be the set of intervals intersecting $[v_0, v_1]$. The elapsed time $t(v_0, v_1)$ and the covered distance $d(v_0, v_1)$ are given by

$$t(v_0, v_1) = \sum_{i \in I} \frac{{v_{u_i}}' - {v_{l_i}}'}{3.6 \cdot a_i} \text{ and } d(v_0, v_1) = \sum_{i \in I} {v'}_{l_i} \cdot \frac{{v_{u_i}}' - {v_{l_i}}'}{3.6 \cdot a_i} + \frac{1}{2} a_i \cdot \left(\frac{{v_{u_i}}' - {v_{l_i}}'}{3.6 \cdot a_i}\right)^2$$

$$\text{where } {v_{u_i}}' = \begin{cases} v_{u_i} \; if \; v_{u_i} \leq v_1 \\ v_1 \; if \; v_{u_i} > v_1 \end{cases} \text{ and } {v_{l_i}}' = \begin{cases} v_{l_i} \; if \; v_{l_i} \geq v_0 \\ v_0 \; if \; v_{l_i} < v_0 \end{cases} \text{ for all } i \in I.$$

(3)

The braking deceleration is assumed to be constant and determined by the respective vehicle parameters. E.g., for an IC train, a braking deceleration of $a = -0.7$ m/s^2 is assumed [108]. Thus, the elapsed time $t(v_0, v_1)$ and covered distance $d(v_0, v_1)$ during deceleration from a speed v_0 to a speed v_1 are calculated as

$$t(v_0, v_1) = \frac{v_1 - v_0}{3.6 \cdot a}$$

(4)

and

$$d(v_0, v_1) = \frac{v_0}{3.6} \cdot t(v_0, v_1) + \frac{1}{2} a \cdot (t(v_0, v_1))^2.$$

(5)

Approximation of the speed curves via speed levels

To operate trains on the railway lab according to the continuous acceleration and deceleration described above, these functions must be approximated via the train's available speed levels. Target speeds in km/h that cannot exactly be matched with a certain speed level are matched with the speed level closest to the target speed. If there are two speeds that are equally close to the target speed, the higher speed is chosen. For example, if Table 7 is taken as lookup table and the target speed is 40 km/h, the train's speed level is set to 15. As only finitely many speed

levels exist, acceleration and deceleration must be approximated by letting trains run constantly at their available speed levels for predefined time intervals. Figure 25 shows an example of such an approximation via speed levels. The blue lines represent the continuous speed curves and the green lines represent the time intervals and the constant speed levels the train runs at during acceleration and deceleration. Depending on the vehicle and the railway lab infrastructure, a minimum speed, e.g. 10 km/h, can be necessary, as the train might not run reliably at lower speeds on the physical infrastructure.

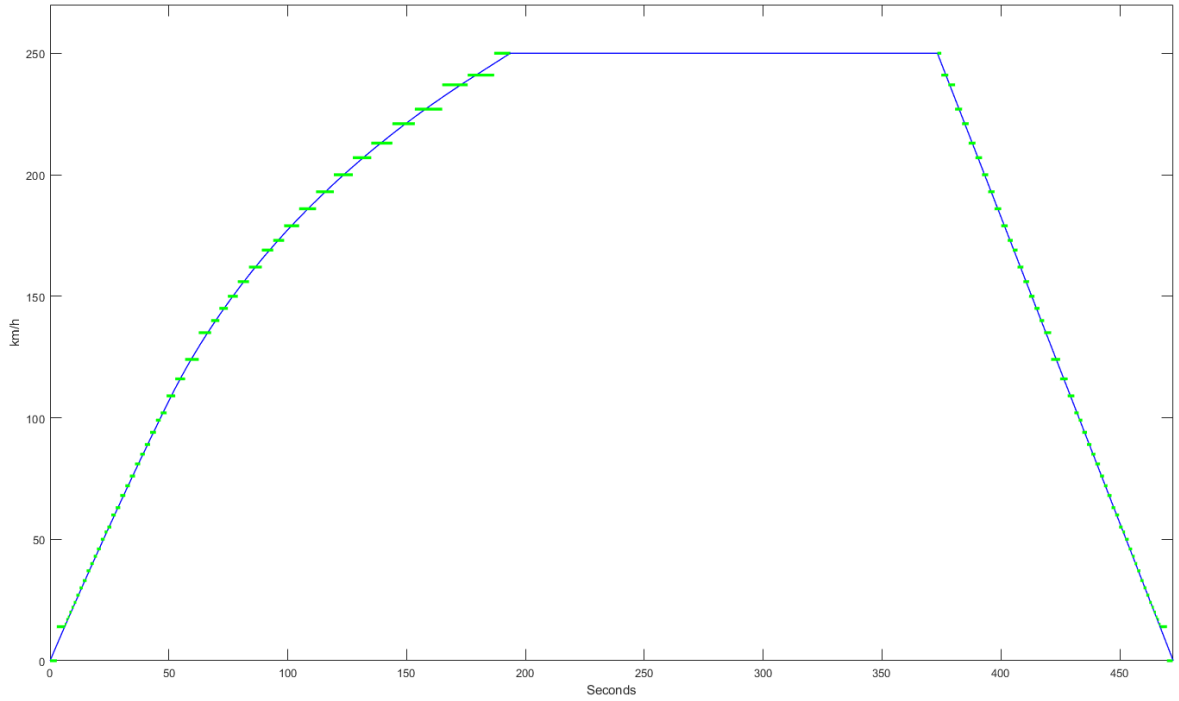


Figure 25: Approximation of the acceleration and deceleration curves of an ICE train via its speed levels

The time intervals during which the speed levels are applied, must be determined in such a way that realistic approximations are achieved.

Consider an acceleration or deceleration process. Let v_i denote the speed in km/h associated with the i^{th} speed level, where the 0^{th} speed level is defined to equal 0 km/h. Further, let $T_i = T_{i_1} + T_{i_2}$ be the time interval that the i^{th} speed level is used. All T_{i_1} and T_{i_2} are determined such that the distance covered when running at speed level $i-1$ for T_{i-1_2} seconds and at speed level i for T_{i_1} seconds, equals the distance covered when running according to the continuous acceleration curve from v_{i-1} until v_i. For deceleration curves, the intervals T_{i_1} and T_{i_2} are determined such that the distance covered when running at speed level i for T_{i_2} seconds and at speed level $i-1$ for T_{i-1_1} seconds, equals the distance covered when running according to the continuous curve from v_i until v_{i-1}.

To ensure that a speed curve approximated via speed levels starts and ends at the same time as its continuous counterpart, the intervals T_{0_1} and T_{n_2} are defined to be 0 for acceleration, where n is the number of available speed levels. For deceleration, the intervals T_{0_2} and T_{n_1} are defined to be 0.

An example is given in Figure 26. The blue curve shows an extract of an acceleration curve where the intervals T_{i_1} are displayed in orange and the intervals T_{i_2} are displayed in green. Consider the dashed rectangle in which the speed change from speed level 42 to speed level

43 occurs. By definition of T_{42_2} and T_{43_1}, the distance covered when running at speed level 42 for T_{42_2} seconds and at speed level 43 for T_{43_1} seconds equals the distance covered during continuous acceleration along the blue curve from the speed associated with speed level 42 until the speed associated with speed level 43. Therefore, the total distances covered by running constantly at the two speed levels and by running along the continuous acceleration curve, are equal inside the dashed rectangle.

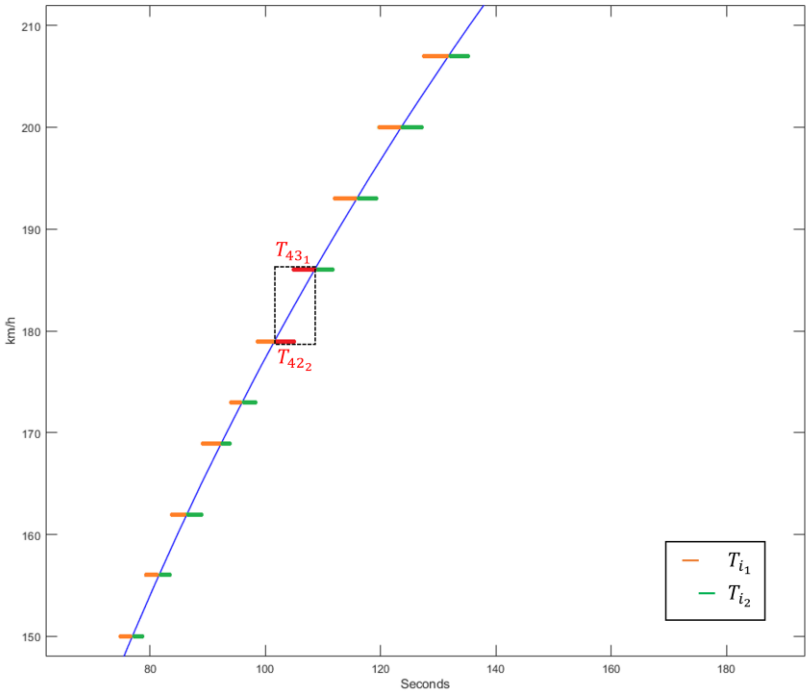


Figure 26: Time intervals for the speed levels

To determine the values of the time intervals, let v_{i-1} and v_i be the speeds associated with two consecutive speed levels of a speed curve. Further, let T_i be the time in seconds and d_i be the distance in meters that is required to accelerate from v_{i-1} to v_i or to decelerate from v_i to v_{i-1} according to the continuous speed curve.

In case of acceleration, by the definition of the time intervals, the speed v_{i-1} is run at for T_{i-1_2} seconds and the speed v_i is run at for T_{i_1} seconds during the continuous acceleration from v_{i-1} to v_i. Therefore,

$$T_i = T_{i-1_2} + T_{i_1} \tag{6}$$

and

$$d_i = T_{i-1_2} \cdot \frac{v_{i-1}}{3.6} + T_{i_1} \cdot \frac{v_i}{3.6}. \tag{7}$$

Transforming equation (6) yields $T_{i_1} = T_i - T_{i-1_2}$. Inserting T_{i_1} into equation (7) then yields

$$d_i = T_{i-1_2} \cdot \frac{v_{i-1}}{3.6} + (T_i - T_{i-1_2}) \cdot \frac{v_i}{3.6}$$

$$\Leftrightarrow d_i = T_{i-1_2} \cdot \frac{v_{i-1}}{3.6} + \frac{T_i \cdot v_i}{3.6} - T_{i-1_2} \cdot \frac{v_i}{3.6}$$

$$\Leftrightarrow d_i - \frac{T_i \cdot v_i}{3.6} = T_{i-1_2} \cdot \frac{v_{i-1} - v_i}{3.6}$$

$$\Leftrightarrow T_{i-1_2} = \frac{3.6 \cdot d_i - T_i \cdot v_i}{v_{i-1} - v_i}.$$

(8)

Similarly, for deceleration, $T_i = T_{i_2} + T_{i-1_1}$ and $d_i = T_{i_2} \cdot \frac{v_i}{3.6} + T_{i-1_1} \cdot \frac{v_{i-1}}{3.6}$, yielding the time intervals $T_{i_2} = T_i - T_{i-1_1}$ and $T_{i-1_1} = \frac{3.6 \cdot d_i - T_i \cdot v_i}{v_{i-1} - v_i}$.

(9)

These calculations can be performed for all pairs of consecutive speed levels in the speed curve to obtain the complete set of values for the time intervals.

By defining the intervals as described above, all points where the approximated speed curve, composed of the constant running at the available speed levels, and the continuous speed curve intersect, serve as supporting points at which both the approximation through speed levels and the continuous curve coincide in the total covered distance.

Prior to operation on the railway lab, the time intervals for each speed level during acceleration and deceleration are calculated according to the formulas derived above for each physical vehicle that is used during operation. The resulting tuples of speed, speed level and time intervals T_{i_1} and T_{i_2} are stored in lookup tables which are accessed by the train control module during operation (see 6.4). Table 9 shows an example extract of such a lookup table for acceleration.

Table 9: Extract of a lookup table for acceleration

Speed [km/h]	**Speed level**	**Time interval T_{i_1} [sec]**	**Time interval T_{i_2} [sec]**
⋮	⋮	⋮	⋮
39	14	0.323	0.975
41	15	0.323	0.987
43	16	0.329	1.319
46	17	0.660	1.671
50	18	1.003	1.351
⋮	⋮	⋮	⋮

When accelerating or decelerating from a speed level i to a speed level k, the train will continue to run at speed level i for T_{i_2} seconds, then run at all intermediate speed levels j for $T_j = T_{j_1} + T_{j_2}$ seconds respectively and finally run at speed level k for T_{k_1} seconds. Note that the approximated acceleration and deceleration curve contains time intervals during which the starting and target speed are run at, although the train is not physically accelerating or decelerating during these time intervals. For example, a train accelerating after stopping, will continue to stop for a short time interval, as $T_{0_2} > 0$. This is necessary to make both the approximation and the continuous curve coincide at all supporting points, increasing the precision of the approximation.

As an example, Figure 27 shows a train acceleration process from 162 km/h to 200 km/h. The blue curve represents the continuous reference curve and the green lines represent the speed levels and time intervals used by the physical train during operation.

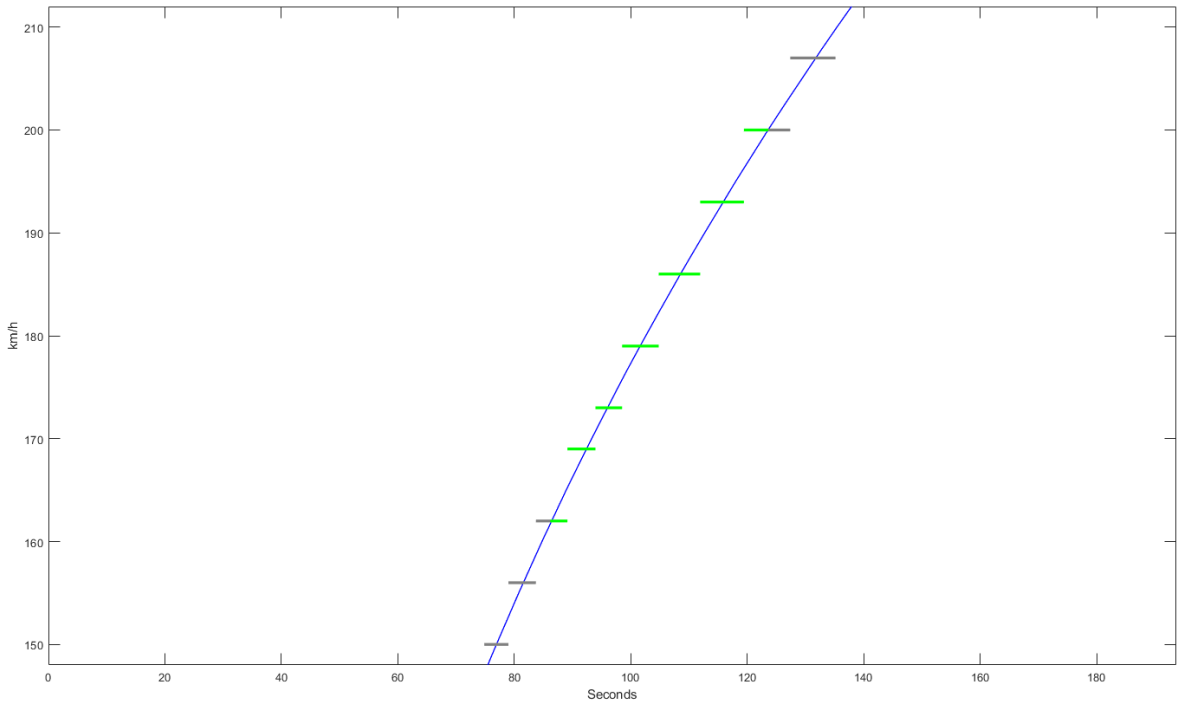


Figure 27: Speed levels and time intervals during train acceleration

Estimation of the error during approximation

During an acceleration or deceleration process, the traveled distances at a certain time, according to the continuous reference curve and the time intervals with constant speed levels respectively, only coincide at the intersection points between the reference curve and the constant speed levels. Thus, only in between two intersection points an error occurs. The error can be measured as the deviation between the actually traveled distance through the speed levels and the theoretically traveled distance of the reference curve.

Consider a time interval between two such intersection points. From the beginning of the interval until the speed level change occurs, the train runs at a speed that is lower than the corresponding speed of the reference curve at the respective time. Conversely, after the speed level change, the train runs at a higher speed than the corresponding speed of the reference curve. The absolute error in the interval is thus 0 at the beginning of the interval, increases until it reaches its maximum value at the time of the speed level change and then decreases again until it is 0 at the end of the interval. Therefore, all errors only occur in between two consecutive intersection points and do not add up over time.

The error occurring in an interval between two intersection points can be computationally determined. For acceleration curves, consider the time interval $[p_{i-1},\ p_i]$ between two intersection points at v_{i-1} and v_i, as shown in Figure 28. The maximum error within this interval occurs at time $p_{i-1} + T_{i-1_2}$ at which the speed change from v_{i-1} to v_i occurs. Consider the intervals with constant acceleration of the reference curve. Let the j^{th} interval

with lower interval limit speed v_j be the interval in which this maximum error occurs, i.e the j^{th} interval is the interval for which $t_j \leq p_{i-1} + T_{i-1_2} \leq t_{j+1}$, where t_j and t_{j+1} are the times required to accelerate from the initial speed of the curve to v_j and v_{j+1} respectively. Further, let a_j be the constant acceleration in the j^{th} interval.

Then the traveled distance of the reference curve in the interval from p_{i-1} to $p_{i-1} + T_{i-1_2}$ is given by

$$d_{ref} = d(v_{i-1},\ v_j) + \frac{a_j \cdot (p_{i-1} + T_{i-1_2} - t_j)^2}{2} + v_j \cdot \left(p_{i-1} + T_{i-1_2} - t_j\right),$$

(10)

i.e. the distance traveled during acceleration from v_{i-1} to v_j plus the distance traveled with constant acceleration a_j for the time interval $p_{i-1} + T_{i-1_2} - t_j$ when starting at speed v_j. The formula for calculating $d(v_{i-1}, v_j)$ is given by equation (5).

The traveled distance in the interval from p_{i-1} to $p_{i-1} + T_{i-1_2}$ for the constant speed levels is given by

$$d_{const} = \frac{v_{i-1}}{3.6} \cdot T_{i-1_2}.$$

(11)

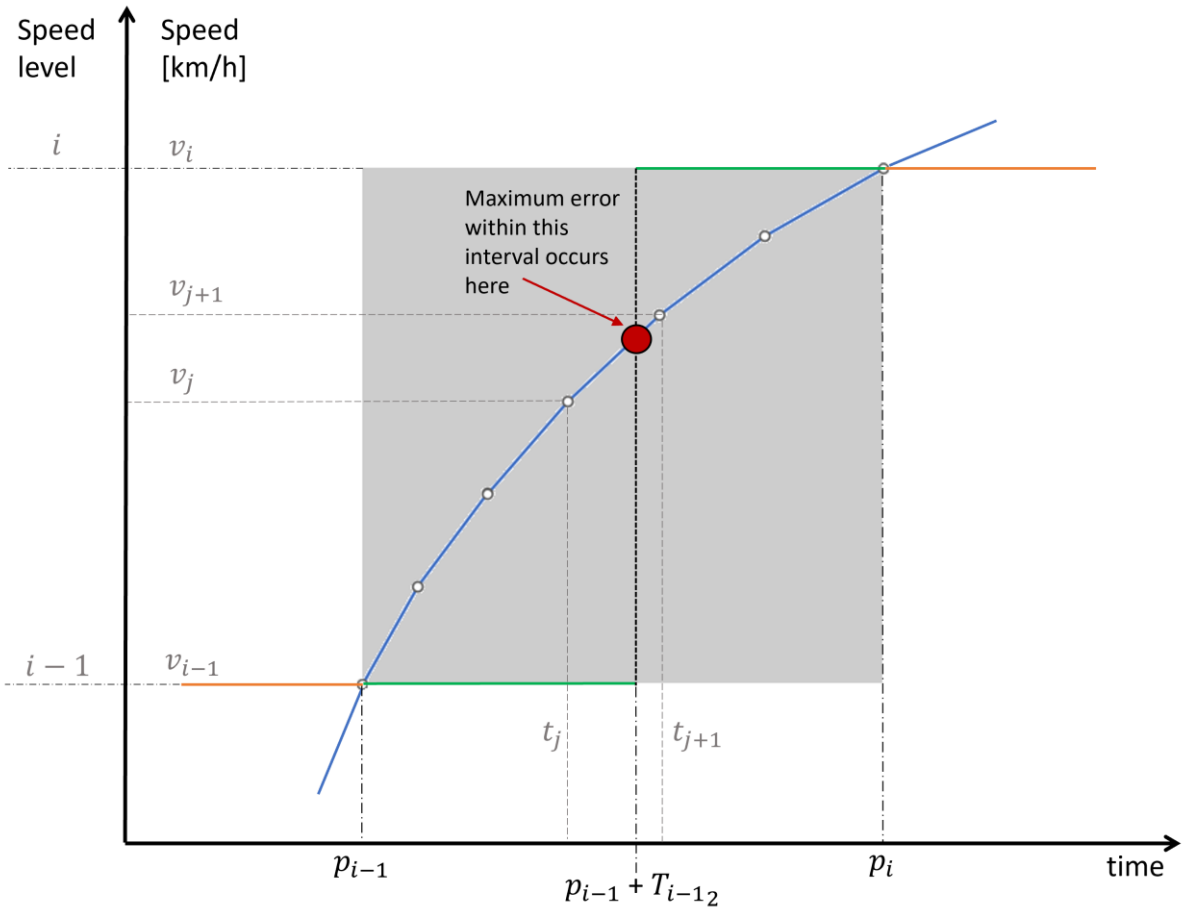


Figure 28: Error between the approximation via constant speed levels and the reference curve

Table 10 shows an extract of the errors occurring during the approximation of an example acceleration curve from 0 km/h to 150 km/h. Note that the distances do not refer to the distances on the physical railway lab, but to the real-world distances that are being modeled.

Table 10: Errors occurring during acceleration via speed levels

v_{i-1} [km/h]	v_i [km/h]	Total distance traveled until p_{i-1} [m]	T_{i-1_2} [sec]	d_{const} [m]	d_{ref} [m]	Absolute error [m]	Relative error over all previous intervals [%]
0	3	0	0.827	0	0.172	0.172	100
3	6	0.691	0.828	0.690	0.862	0.172	11.075
6	9	2.080	0.847	1.412	1.825	0.413	10.576
9	12	3.524	0.796	1.991	2.384	0.393	6.652
12	15	5.013	0.856	2.855	3.034	0.179	2.224
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
149	150	570.981	0.826	34.187	34.255	0.068	0.011

At the beginning of a speed curve, the total relative error is large and decreases over time. Note that, despite larger absolute errors, higher starting speeds of a speed curve result in smaller relative errors. In general, the error strongly depends on the respective speed curve and the speed levels via which it is approximated. The smaller the intervals between consecutive speed levels are and the more speed levels are available, the smaller the errors become. At all intersection points with the reference curve, and in particular at the end of a speed curve, the error is 0.

For deceleration curves, the traveled distance is considered in the interval from p_i to $p_i + T_{i_2}$. For the constant speed levels, it is given by

$$d_{const} = \frac{v_i}{3.6} \cdot T_{i_2}$$

(12)

and the traveled distance of the reference curve with a constant braking deceleration a is given by

$$d_{ref} = \left(\frac{v_i}{3.6}\right) \cdot T_{i_2} + \frac{1}{2} \cdot a \cdot {T_{i_2}}^2.$$

(13)

The errors described above serve as an upper bound for the occurring errors. Most of the time, the actually occurring errors are smaller.

A precision of 1 millisecond can be achieved during operation, e.g. by using the QTimer Class of Qt [145]. For a speed curve, let $step_{max}$ be the maximum speed difference in km/h between two consecutive speed levels. Then $\frac{step_{\max}}{3600}$ corresponds to the maximum speed difference in m/ms. With a precision of 1 millisecond, an additional error of at most $\frac{step_{\max} \cdot 1\,ms}{3600}$, can thus occur at every speed level change. Let n be the total number of speed level changes of the

acceleration or deceleration curve. Then, the total additional error due to timer imprecision is at most

$$\frac{n \cdot step_{\max} \cdot 1\,ms}{3600}.$$

As an example, for $step_{max} = 10$ km/h this corresponds to an error of less than 3 millimeters per speed change. Assuming a total of 50 speed level changes for a speed curve, a total additional error of at less than 15 centimeters occurs due to timer imprecision. Note that this is a theoretical upper bound and the actually occurring error will likely be smaller.

6.4 Train Control

The functionalities of a basic version of the train control described in 5.3.5 must be extended to meet the requirements to operate trains on the railway lab infrastructure realistically during real-time dispatching, thus enabling the reliable testing of dispatching algorithms. This section describes how the train control software module was extended to meet this goal.

Generation of speed profiles

To operate trains according to the speed curves described in section 6.3, speed profiles are derived from the infrastructure model and updated dynamically. The train control module accesses a representation of the network infrastructure that is constantly synchronized with the current element states through the field level (see 5.3.2). The train route list (see 5.3.5) was extended to also contain speed limit signs and stopping positions. Both speed limit signs and stopping positions are only valid for trains running in the same direction as the element's direction within the infrastructure.

Speed limit signs indicate a new maximum permitted line speed and optionally a train protection system the train is required to be compatible with for the speed limit to apply. E.g., there are speed limit signs that are valid for all trains or speed limit signs that are only valid for trains that are compatible with the LZB train protection system. For each vehicle, its highest compatible train protection system is stored in the XML file containing the train parameters (see Appendix B). A speed limit sign is only added to the train route list if the protection system is compatible.

Stopping positions are located in front of main signals within stations and indicate the position where a train is supposed to stop in front of the signal. Stopping positions can be suitable for passenger or freight trains and are only added to the train route list if their type coincides with the train type.

Based on the infrastructure elements along the designated train route, a speed profile is defined through a list of tuples (d_i, v_i), where for all tuples the speed v_i is a maximum permitted speed starting at distance d_i from the current train position. As an example, consider the setup in Figure 29. A train shall run along the blue route. Along this route, different maximum permitted speeds apply. The green signals show a proceed aspect and a maximum permitted speed of 40 km/h applies at these signals, i.e. the speed of the train must not exceed 40 km/h when passing the signal until a new maximum permitted speed applies. The red signal, on the other hand, shows a stop aspect, thus the train must reach 0 km/h prior to reaching the signal. Along the route, there are also two speed limit signs, indicating that the

train may run at up to 60 or 160 km/h respectively once the complete length of the train has passed the speed limit sign. The distance axis below the infrastructure gives the distances of each maximum permitted speed from the current train position. The speed profile is thus defined by the following list of tuples: [(0, 40), (1000, 60), (1500, 40), (1800, 160), (7000, 0)] where the first entry of each tuple represents the distance from the current train position in meters and the second entry the maximum permitted speed in kilometers per hour.

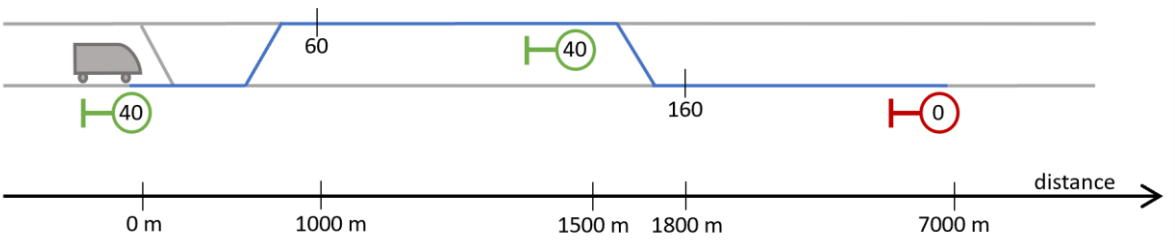


Figure 29: Maximum permitted speeds and distances from the current train position

Every time the train route list is extended until the next signal indicating “stop”, the speed profile is adjusted. Let S denote the signal that triggered the generation or extension of the speed profile. The speed 0 km/h at S is removed from the speed profile. The infrastructure elements considered during speed profile generation or extension are the signal S itself, as well as all signals, switches, speed limit signs, stopping positions and bumpers along the train route after the signal S. The elements are stored in a list. Let d_i be the distance in meters between S and the i^{th} infrastructure element of the list where d_0 is the distance between the current train position and the position of S. The generated list of relevant infrastructure elements for speed profile generation is processed in the order in which the infrastructure elements occur in the train route list. For each infrastructure element, depending on its type, the following six cases are distinguished.

1. Signal

Each signal provides information on maximum line speeds or speed restrictions starting at its position. To resemble realistic operation, trains are defined to stop d_S meters in front of signals indicating “stop”, e.g. $d_S = 10$.

Let v_{signal} be the maximum line speed or speed restriction provided by the signal. The distance from the current train position until the signal is given by $d_0 + d_i$. Thus, if the signal aspect shows “proceed”, the tuple $(d_0 + d_i\ , v_{signal})$ is added to the speed profile. In case signal indicates “stop”, the tuple $(d_0 + d_i\ - d_S, 0)$ is added instead, as the train shall stop d_S meters in front of the signal.

2. Switch along the diverging path

Each switch is associated with a maximum speed v_{div} at which its diverging path can be traversed. If a switch along the train route is passed along its diverging path, v_{div} must be adhered to. In case the train protection system is below ETCS Level 2, this speed restriction must start at the signal preceding this switch. Thus, if the speed associated with the preceding signal is greater than v_{div}, it is changed to v_{div}. In case of several switches taken along their diverging path following a signal, the maximum permitted speed at the signal is set to the minimum speed v_{div} over all switches. From ETCS Level 2 on, it suffices to locally restrict the speed to v_{div} before the switch instead of from the preceding signal on,

as trains constantly receive information on maximum permitted speeds, e.g. via GSM-R [136].

3. Speed limit sign

A speed limit sign contains information on a maximum permitted speed v and optionally on the train protection system the vehicle must be compatible with for the speed limit to apply. If the speed limit sign applies for the train, the sign is considered for speed profile generation and the tuple $(d_0 + d_i\,, v)$ is added to the speed profile.

4. Stopping position

A stopping position in the train route list is only considered during speed profile generation if the train is supposed to stop due to a scheduled or an operating stop. If the signal right after the stopping position indicates "stop", a query is sent to the timetable module (see 6.5) which returns if the train is supposed to stop or not. In case it is supposed to stop, the tuple $(d_0 + d_i\ - d_{SP}, 0)$ is added to the speed profile, where d_{SP} is the distance in meters which the train is assumed to stop in front of the stopping position, e.g. $d_{SP} = 40$. The signal after the stopping position showing "stop" is not added to the speed profile list. Stopping positions thus enable the train to not stop directly in front of the signal, but instead at predefined positions. If the train is not supposed to stop, the stopping position is ignored during speed profile generation. There can be distinct stopping positions for passenger and freight trains, thus the train type is compared with the type of the stopping position and only matching stopping positions are considered. In case there is more than one stopping position of the designated type, e.g. the nearest one from the current train position can be chosen if no particular stopping position is specified. Note that the stopping position can also be defined through the timetable instead.

5. Bumper

A train must stop in front of a bumper. For a bumper, $(d_0 + d_i - d_B, 0)$ is added to the speed profile, where d_B is the distance in meters that the train is supposed to stop in front of the bumper, e.g. $d_B = 20$.

If two or more consecutive tuples in the speed profile have the same speed value, all but the first one are deleted, as they do not provide any additional informational value.

Speed profiles can also be reduced, if the signal aspect of a signal along the train route changes from "proceed" to "stop" prior to the passing of the train. In this case, all tuples associated with infrastructure elements after this signal are removed from the speed profile, the speed of the tuple associated with the signal itself is changed to 0 km/h and the distance of the tuple is reduced by d_S.

Combination of speed profiles and speed curves

The list of tuples through which the speed profile is defined yields a speed-distance diagram which shows the local maximum permitted speeds along a section of the network infrastructure. Figure 30 shows an example of a speed-distance diagram.

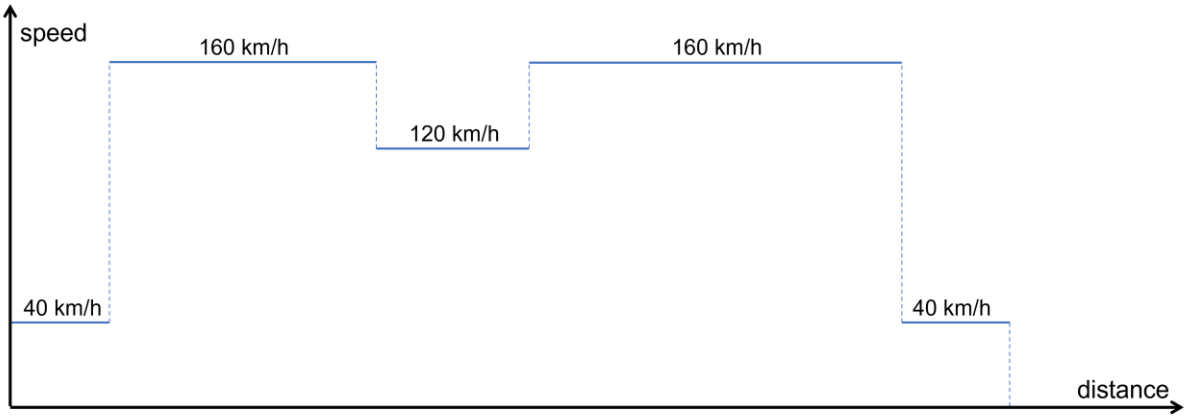


Figure 30: Speed-distance diagram of a speed profile

To resemble realistic train operation, acceleration and deceleration curves, as described in 6.3, need to be added to the speed-distance diagram of the speed profile. Maximum permitted speeds change at certain positions within a speed profile. When a maximum speed is higher than the previously permitted speed, acceleration can be started once the complete train passed this position. Thus, the train length must be added to the respective position of the speed profile, to obtain the starting position for acceleration which the head of the train must have reached. If the new maximum speed is lower than the previous one, deceleration to the speed must be completed when the train reaches this position. Figure 31 shows an example. The green points represent the positions along the speed profile where acceleration to a higher speed can be started, whereas the red points represent the positions where deceleration to a lower speed must be completed. Note that, if the train already has a speed not higher than the new lower speed limit, no deceleration process is required.

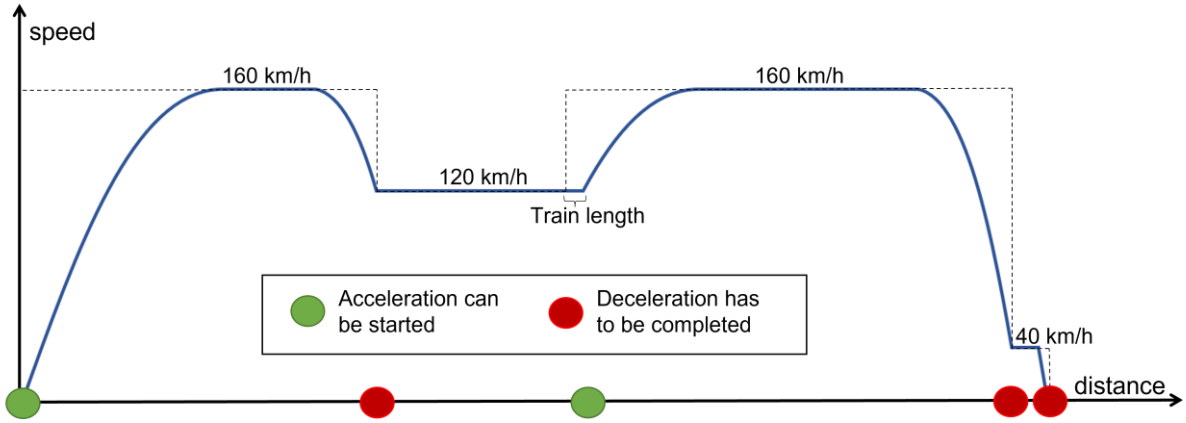


Figure 31: Speed profile with acceleration and deceleration curves and the starting points for acceleration curves and end points for deceleration curves

In some cases, speed curves can overlap. Thus, acceleration to a speed v would be completed after the position where subsequent deceleration from v would start. This is resolved by making the train only accelerate to and thus decelerate from some speed lower than v. The speed v is iteratively decreased by one speed level of the train, until the overlap is resolved. Figure 32 shows an example: The two blue curves in the diagram on the left overlap. Thus, the maximum speed level that is accelerated to and decelerated from is reduced iteratively by one speed level until the conflict is resolved. The new resulting curves are displayed in the diagram

on the right. Note that, as abrupt train movements are often perceived as uncomfortable by passengers, a minimum distance traveled at constant speed is normally required after acceleration prior to the start of deceleration [113].

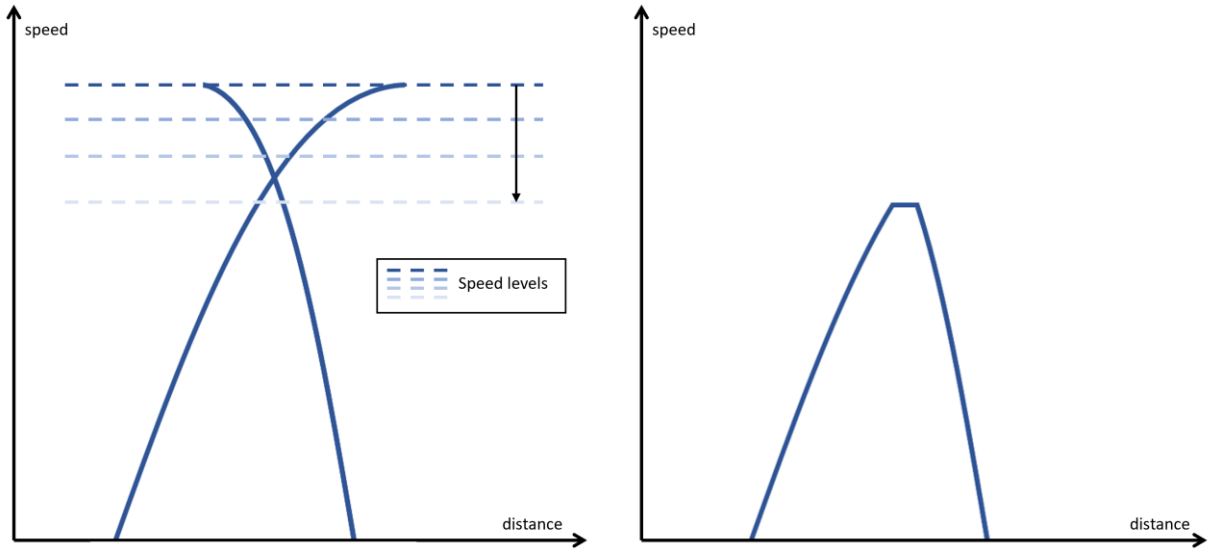


Figure 32: Overlap of an acceleration and deceleration curve is resolved

Distance points in the speed profile might be very close to each other, such that acceleration to a certain speed is not yet finished when acceleration to a higher speed is already permitted. Figure 33 shows an example: A train is accelerating from 0 km/h to 120 km/h. Before it reaches 120 km/h, the maximum permitted speed is increased to 160 km/h. In this case, the starting point of the acceleration from 120 km/h to 160 km/h is set to the end point of the previous acceleration curve from 0 km/h to 120 km/h.

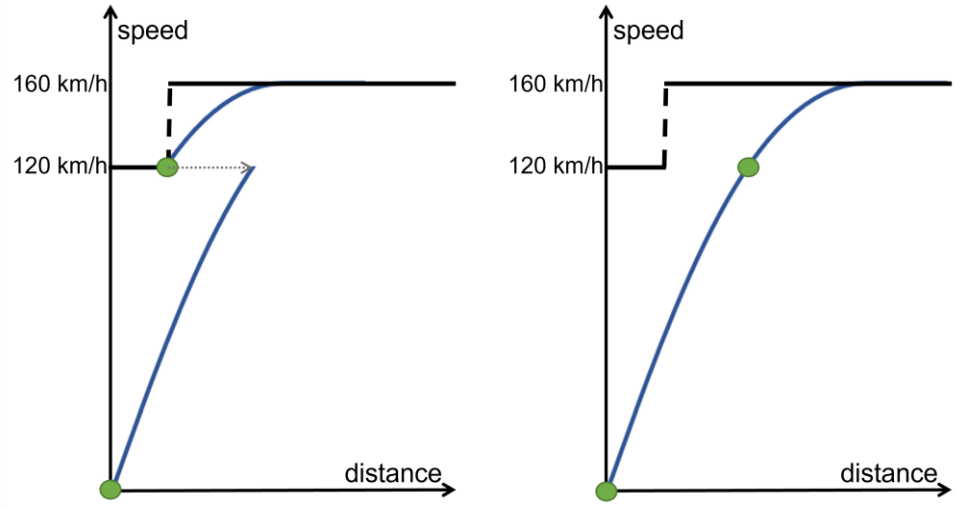


Figure 33: The starting point of an acceleration curve is adjusted to match the preceding acceleration curve

Also, a braking curve might have to start prior to its preceding braking curve. Figure 34 shows an example: The braking curve from 120 km/h to 0 km/h would have to start before the braking curve from 160 km/h to 120 km/h is completed. Thus, the end point of the deceleration curve from 160 km/h to 120 km/h is set to the starting point of the succeeding deceleration curve from 120 km/h to 0 km/h.

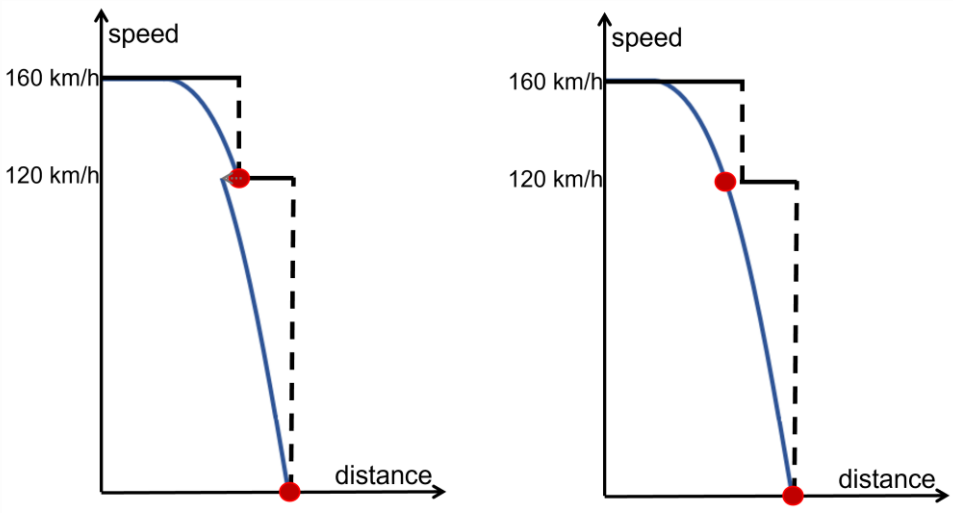


Figure 34: The end point of a deceleration curve is adjusted to match the succeeding deceleration curve

For each speed curve, its length is determined by the time intervals of all applied speed levels. Thus, the end point of an acceleration curve is determined by its starting point plus its length and the starting point of a deceleration curve is determined by its end point minus its length.

Management and supervision of speed curves

Let $d_{i_{start}}$ be the distance until the starting point of the i^{th} speed curve and let $d_{i_{end}}$ be the distance until the end point of the i^{th} speed curve. By resolving overlaps and adjusting acceleration and deceleration curves whenever necessary (see Figure 32, Figure 33 and Figure 34), all intervals $[d_{i_{start}}, d_{i_{end}}]$ are disjoint, i.e. no curve is supposed to start while another one is still active.

Within the train control software module, a speed curve is assigned the following information:

- The list of its tuples (speed, speed level, time interval T_{i_1}, time interval T_{i_2}), as in Table 9.
- The distances from the current train position until the starting and end points of the curve.

During operation, this information is dynamically updated. Tuples are removed from the list upon their completion and distances are updated as the current train position is updated.

Speed curves are started when their starting point is reached. Let d_0 be the distance in meters until the starting point of the first curve, v the current train speed in km/h and t the time interval in seconds in which traveled distances are updated (see 5.3.5). As soon as at most one more train position update will occur before the starting point of the first curve will be reached, i.e. $d_0 < 2 \cdot \frac{v \cdot t}{3.6}$, a timer is started. The timer is set to time out after $T = \frac{d_0 \cdot 3.6}{v}$ seconds, which is the time required for the train to reach the starting point of the curve assuming a constant speed v.

Even in case the curve starts directly after another curve with last time interval T_{n_2}, the speed v will be constant if t is sufficiently small such that $2 \cdot t \leq T_{n_2}$. Since $T = \frac{d_0 \cdot 3.6}{v} < 2 \cdot t \leq T_{n_2}$,

the final speed of the previous curve would have been set to v already when the timer starts. Note that during real operation, however, this case is extremely unlikely to occur.

As described in section 6.3, speed curves consist of a list of consecutive speed levels that are run at for certain predefined time intervals. Each curve begins at its starting speed. A timer is set, timing out after the time interval that the starting speed shall be run at. Each time, the timer times out, the tuple associated with the current speed level is removed from the list, the next speed level in the list is set, and the timer is restarted with the time interval associated with the new speed level, until the speed curve is finished.

Dynamic adjustment of speed curves due to train route extension or reduction

When a speed profile is updated due to a train route extension or reduction resulting from a changed signal aspect, the speed curves must be readjusted dynamically to be synchronous with the current speed profile.

For train route extensions, two cases are distinguished. Let v_1 and v_2 be the maximum permitted speeds that apply before and after the last signal S respectively and let v be the current speed of the train.

1. **The final deceleration curve towards signal S has not yet been started**

   In this case, the deceleration curve towards signal S is removed from the speed curve list.

   If $v_1 < v_2$, an acceleration curve from v_1 to v_2 is added at the position of S plus the train length. An example for this case is shown diagram 1 of Figure 35.

   Note that in case an acceleration curve was previously reduced due to an overlap with the removed deceleration curve, an acceleration until v_1 is added first, starting at the current train position.

2. **The final deceleration curve towards signal S has been started**

   The deceleration curve towards signal S is stopped and removed and an acceleration curve from v to v_1 is added at the current train position.

   If $v_1 < v_2$, an acceleration curve from v_1 to v_2 is added at the position of S plus the train length. An example for this case is shown diagram 2 of Figure 35. If the acceleration curves from v to v_1 and from v_1 to v_2 conflict one another, this conflict is resolved, as shown in Figure 33. An example for this case is shown diagram 3 of Figure 35.

   If $v_1 > v_2$, a deceleration curve from v_1 to v_2 is added at the position of S minus the length of the curve. An example for this case is shown diagram 4 of Figure 35. If the acceleration curve from v to v_1 and the deceleration curve from v_1 to v_2 overlap, this overlap is resolved, as shown in Figure 32. An example for this case is shown diagram 5 of Figure 35.

Then, for both cases, the speed profile is extended as described above, until the next signal indicating “stop”.

Note that by setting train routes early, unnecessary braking can often be avoided. Setting train routes at the right time is achieved by the automatic route setting described in 6.6.

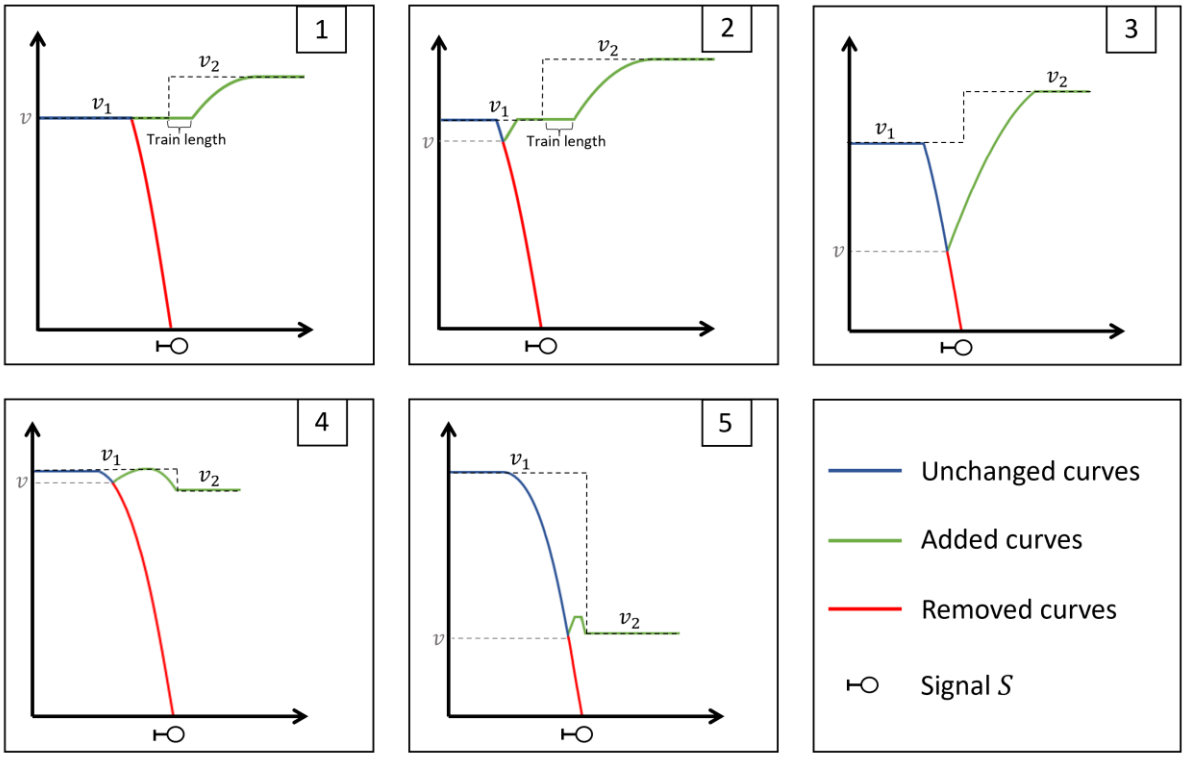


Figure 35: Extension of the speed profile and adjustment of the speed curves

For train route reductions, it is assumed that the signal aspect is changed to "stop" early enough to enable braking in front of the signal. The target speed at this signal is changed to 0 km/h, thus a deceleration curve is added and conflicts between the new curve and already existing curves, that have not yet been started, are resolved (see Figure 32 and Figure 34). If the newly added deceleration curve intersects an already started deceleration curve, the currently active deceleration curve is removed. In case the new deceleration curve intersects an already started acceleration curve, an overlap is possible (see Figure 32). If such an overlap occurs, the current train position has not yet reached the position where the acceleration and deceleration curves intersect, as otherwise braking to 0 km/h until the signal would not be possible anymore. The conflict between the newly added deceleration curve and the already started acceleration curve is resolved, as shown in Figure 32.

Dynamic adjustment of speed curves through dispatching measures

The timetable module (see 6.5) processes all information provided by the dispatching system and provides the train control module with new maximum recommended speeds which e.g. result from bending. Bending is assumed to always apply between reference elements of operating stations (see Figure 11). Shortly after a reference point is passed or the train starts accelerating after stopping at a stopping position, the current maximum recommended speed is requested from the timetable module. Between reaching a reference point, a minimum time, e.g. 2 seconds, should be defined until the new maximum recommended speed is requested from the timetable module to ensure that the necessary computations within the timetable module terminated completely.

Let v_t be the new maximum recommended speed received from the timetable module and let v_{max} be the current maximum permitted speed, resulting from the infrastructure and the

vehicle's maximum speed, at the current train position. Further, let v be the current train speed. The speed curves are adjusted to follow the new maximum recommended speed v_t.

If $v > v_t$, a deceleration curve is added from v to v_t, starting at the current train position and in case another speed curve is currently active, it is stopped and removed. If $v < v_t$ and $v_t \leq v_{max}$, an acceleration curve from v to v_t is added, starting at the current train position and in case a speed curve is currently active, it is stopped and removed.

All other speed curves are reduced to v_t, i.e. all speed levels corresponding to speeds greater than v_t are removed. In case any of these curves was previously reduced due to some maximum recommended speed from the timetable smaller than v_t, the curve is extended again until $\min(v_t, v_{max})$. The starting and end points of the modified curves are adjusted accordingly.

Figure 36 shows an example of an adjusted speed profile, based on the maximum recommended speeds provided by the timetable module. At the first reference point, the new maximum recommended speed is 140 km/h. Thus, the currently active acceleration curve is stopped and the train decelerates to 140 km/h and continues at this speed until braking to 120 km/h which is the speed restriction given by the infrastructure. It then accelerates to 140 km/h again and runs constantly at this speed until the next reference point. The new maximum recommended speed according to the timetable module is 150 km/h, thus the train accelerates to 150 km/h.

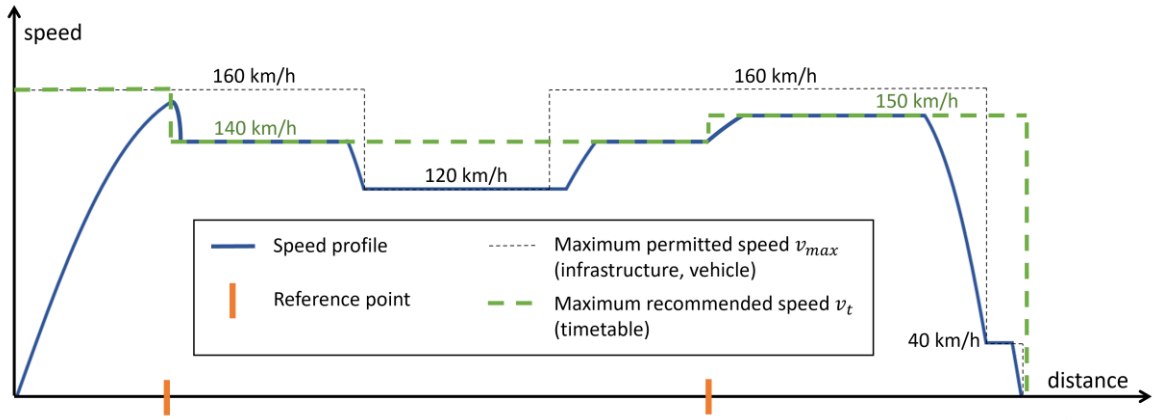


Figure 36: Adjusted speed profile following maximum recommended speeds from the timetable module

Synchronization between the predicted and physical train positions

Predicted train positions do not always precisely coincide with the actual train positions on the physical network infrastructure. Due to small deviations between the assumed model speed and the actual speed of the vehicle, inaccuracies naturally occur, especially during operation over long distances. Therefore, axle counters are used to synchronize the predicted positions within the train control software module with the actual positions of the physical vehicles on the railway lab infrastructure. If an axle counter is passed by a physical train and the predicted train position has not yet reached this axle counter, the predicted position is set to this axle counter. Let d be the distance between the physical train's position and the position previously predicted by the train control software module, as shown in Figure 37.

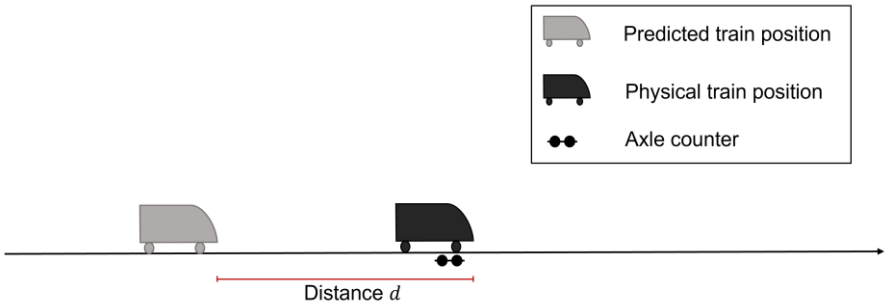


Figure 37: The physical train reaches an axle counter before its prediction

The distances in the train route list are updated (see 5.3.5) and the distances to the starting and end points of all speed curves are reduced by d. The train position within the speed profile is shifted by the distance d. If the current speed v_1 of the train differs from the speed v_2 the train should have been running at upon reaching the axle counter, there are two possible resolution approaches.

For the first approach, the current train speed is not changed. Then, if the speed curve during which the speed v_2 will be reached has not yet started, the curve is started immediately. However, this approach bears the major disadvantage that maximum permitted speeds might be exceeded locally. Figure 38 shows an example where the dashed lines indicate the locally applicable maximum permitted speeds. When the physical vehicle reaches the axle counter, the deceleration curve from v_1 to v_2 has not yet started. Upon synchronization, the train speed is not adjusted to v_2 and instead deceleration until v_2 is initiated immediately. For the duration of this deceleration curve, however, the local maximum permitted speed of v_2 is exceeded.

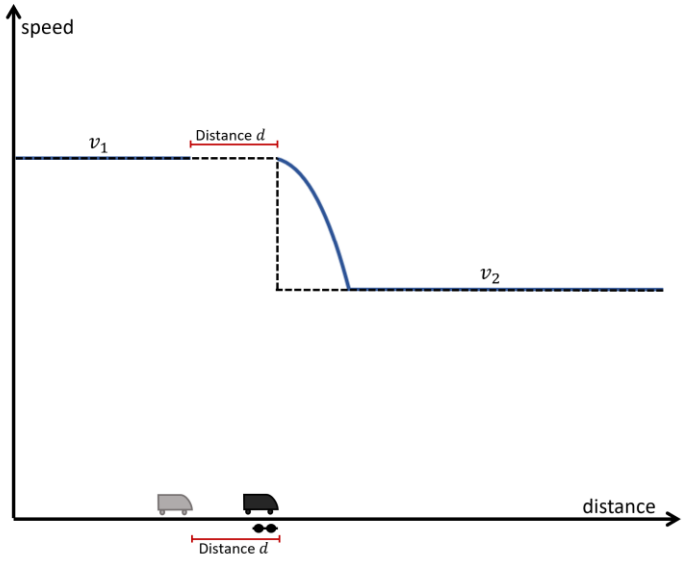


Figure 38: Synchronization of the train position without adjusted current speed

Therefore, the preferable resolution approach is to adjust the train speed to the speed it should have, according to the speed profile, when reaching the axle counter. An example is shown in Figure 39. The speed is set directly to v_2 and the acceleration curve is continued from there. Thus, the acceleration from v_1 to v_2 is skipped. If the gap between the two speeds v_1 and v_2 is significantly large, unrealistic jumps in the train speed can occur. However, if

sufficiently many axle counters serve as synchronization points and if the speed levels of the trains are correctly calibrated, the distance d will be small enough to avoid such jumps.

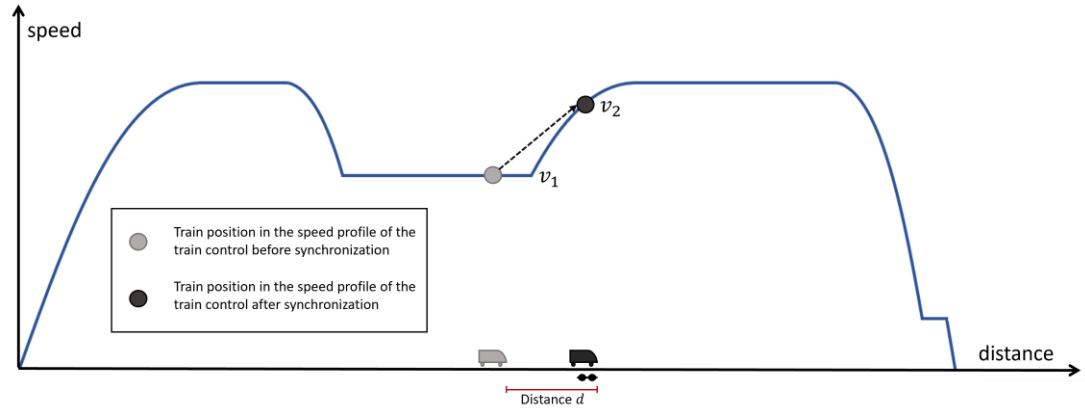


Figure 39: Synchronization of the train position with adjusted current speed

At the time the physical train reaches the axle counter, the speed v_2 is unknown and has to be determined to perform the synchronization. To determine v_2, consider the train's current speed profile prior to synchronization. For this speed profile, the remaining distance the train shall run at the current speed v_1 is determined as follows:

- If the train is currently cruising at constant speed: the distance until the starting point of the next speed curve plus the distance covered at speed v_1 during the first time interval T_{i_2} of the next speed curve.
- If a speed curve is currently active: the distance $\frac{v_1 \cdot (T_{i_1} + T_{i_2} - t)}{3.6}$ covered during the remaining time until the timer for v_1 times out where t is the time in seconds since v_1 was set.

For all succeeding speeds v_i, the distance the train shall run at v_i is determined as follows:

- If the train shall cruise at v_i: the distance between the end point of the preceding speed curve and the starting point of the succeeding speed curve plus the last time interval of the preceding curve and the first time interval of the succeeding curve during which speed v_i is applied.
- If v_i is part of a speed curve, the distance is given by $\frac{v_i \cdot (T_{i_1} + T_{i_2})}{3.6}$.

For the determination of v_2, it suffices to look at the distances the train shall run at its current speed and its succeeding speeds until a total distance of at least d is reached.

An example is shown in Table 11. A train is currently running at 40 km/h and the physical train reaches an axle counter that is 135 meters ahead of the predicted train position. The distances the train shall run at each speed are computed until a total distance of at least 135 meters is reached. After synchronization, the train speed will thus be set to 50 km/h. This speed will then be kept for $145.67 - 135 = 10.67$ meters, hence after $\frac{10.67 \cdot 3.6}{50} = 0.768$ seconds, the next speed level is set.

Table 11: Speeds within a speed profile and the distances they are run at

Speed [km/h]	Distance at this speed [m]	Total distance [m]
40	100	100
43	2.39	102.39
46	29.39	131.78
50	13.89	145.67
⋮	⋮	⋮

If a speed curve is currently active when the train position and speed are synchronized, the timer of the curve is stopped. In case d is greater than or equal to the remaining distance of the speed curve, the curve is removed. Normally, rather small deviations d between the train positions that are being synchronized occur, thus the currently active speed curve will likely not be finished after the synchronization. This case occurs if d is smaller than the remaining distance of the currently active curve. In this case, parts of the current speed curve must be skipped before the curve is resumed. All tuples until the one associated with speed v_2 are removed from the list and the timer is restarted with the remaining time interval the train shall run at v_2.

If the train's new position within the speed profile coincides with a speed curve that was not active before synchronization, all tuples until speed v_2 are removed and the speed curve's timer is started with the remaining time interval the train shall run at speed v_2.

Conversely, if the predicted train position reaches an axle counter that has not yet been reached by the physical train, the timer calculating the traveled distances is paused and continued once the axle counter is passed on the physical infrastructure of the railway lab. In case a timer of a speed curve is currently running, it is as well paused until the axle counter is passed by the physical train.

In case a physical train passes the axle counter after a signal indicating "stop", emergency braking is performed by setting its speed to 0. This rarely happens, for instance if a train's wheels are extremely dirty, impeding its electrical contact with the tracks and therefore the reception of commands from the digital command station.

Refinement of the distance calculation

To further avoid imprecisions in between position synchronizations, the calculation of the traveled distance described in 5.3.5 can be refined. As in the basic version of the train control module, a timer timing out at constant time intervals is started once the train starts running and after every time-out, the covered distance is calculated based on the current train speed. However, when applying realistic speed curves, the speed levels change frequently, thus this approach needs to be extended to achieve sufficient accuracy. Each time a speed level is changed, the distance calculation timer is stopped and the traveled distance is computed based on the previous speed level and the time interval since the last distance calculation was performed, thus since the last time-out or speed level change. Then the timer is restarted with its predefined time-out interval. Especially for high speeds, the gained accuracy is crucial for the precise prediction of train positions.

For instance, consider the extract of an acceleration curve given in Table 12. Let the time-out interval for the distance calculation be 300 milliseconds. Once the train starts running at 14 km/h, the distance calculation timer will time out $\left\lfloor \frac{T_{1_1} + T_{1_2}}{300\ ms} \right\rfloor = \left\lfloor \frac{(2.87 + 0.44)}{0.3} \right\rfloor = 11$ times and each time a traveled distance of $\frac{0.3 \cdot 14}{3.6} \approx 1.167$ meters is calculated. Once the speed changes to 15 km/h, the distance calculation timer ran for $1000 \cdot (2.87 + 0.44) - 11 \cdot 300 = 10$ milliseconds since its last timeout, thus a traveled distance of $\frac{0.01 \cdot 14}{3.6} \approx 0.039$ meters is calculated. After the new speed was set to 15 km/h, the timer is restarted again to time out after 300 milliseconds. Note that in this example, the first two speed levels are not utilized, as some physical trains do not run reliably at very low speeds. If these low speeds are not considered during the approximation of acceleration and deceleration curves, the overall time and distances required for the speed curves remains unchanged. Thus, inaccuracies resulting from low speeds being omitted during acceleration and deceleration have a negligible impact on achieved arrival times during operation and therefore do not distort the evaluation of dispatching strategies.

Table 12: Extract of an acceleration curve

Speed [km/h]	Speed level	Time interval T_{i_1} [sec]	Time interval T_{i_2} [sec]
0	0	0	2.86
14	3	2.87	0.44
15	4	2.22	0.67

Note that small time-out intervals yield a higher precision, yet also require higher computational capacity, especially in case several trains are operated at once. Thus, a tradeoff must be made.

Change of direction

Another extension to the basic version of the train control software module is the possibility to change the driving direction. Thus, a train can run back and forth between a starting and target station. Each time a train passes a signal that is valid for the opposite direction, the signal is temporarily stored. If the train passes another signal lying in the opposite direction, the stored signal is replaced by the new one. Once the train leaves the clearing section the signal in the opposite direction is located in, this signal is no longer valid for the train and is therefore removed. Consider the situation shown in Figure 40. The last signal in the opposite direction of the orange train, indicated by the dashed rectangle, is no longer valid once the orange train leaves the clearing section of the signal, as a conflicting train route, like the one for the yellow train can now be set. If the signal in the opposite direction was still valid for the orange train, an infeasible operational situation could occur.

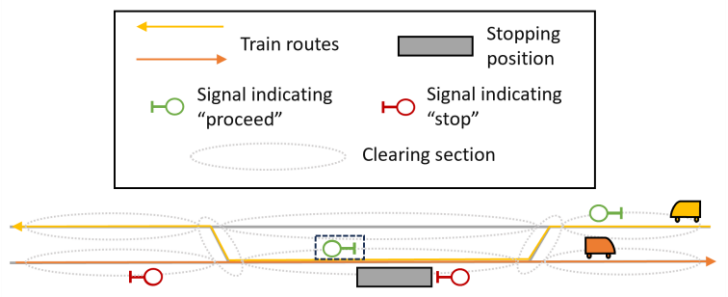


Figure 40: Signal in the opposite direction is no longer valid

When the signal aspect of the stored signal changes from "stop" to "proceed", the train first continues its current journey until it stops, in case that it is currently still running. As soon as the train stops or if it was already stopping when the signal aspect changed to "proceed", a timer is started to time out after a predefined minimum time required before the driving direction can change, e.g. 120 seconds. All remaining elements in the train route list, e.g. the signal in front of which the train is stopping, are removed. Then, the train route is generated, starting from the current train position, after the change of direction, via the signal in the opposite direction showing "proceed" until the next signal indicating "stop". An example is shown in Figure 41. A train is entering a station and stopping at a stopping position. When the signal in the opposite direction changes its aspect to "proceed", the train route is generated in the opposite direction.

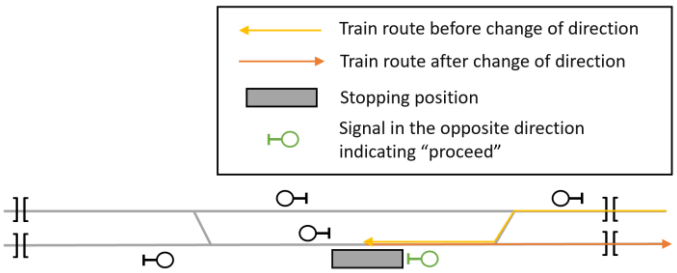


Figure 41: Change of a train's driving direction in a station

To avoid imprecisions, the current train position needs to be adjusted whenever a change of direction occurs. Let d_1 be the distance between the train head and the signal S_1 in front of which the train stopped. Further, let d_2 be the distance between S_1 and the signal S_2 in the opposite direction which triggered the change of direction. After the change of direction, the distance between the train head and S_2 will be $d = d_2 - d_1 - l$, where l is the train length. Figure 42 illustrates this situation.

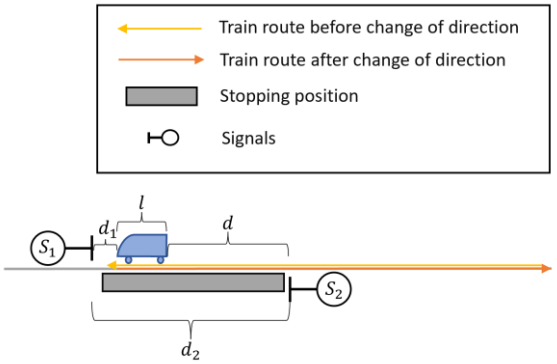


Figure 42: Adjustment of the current train position after a change of direction

The new speed profile and speed curves are generated, as described above. Then, the direction of the train is changed both within the train control module and for the physical vehicle via a command from the digital command station. Afterwards, the train starts running normally according to its speed curves.

Communication with the timetable module

For the railway lab control software, a train run (see 6.5) is defined as the trajectory of a train that either enters the dispatching area or starts from some starting station within the dispatching area and either exists the dispatching area or terminates at some destination station lying within the dispatching area. Each train line within the timetable can have multiple disjoint train runs that pass through the same set of operating stations. For each train run, a unique internal train number is defined in the timetable, as it is crucial to uniquely map a train number with its scheduled and actually achieved arrival times to later evaluate punctuality. Thus, a physical vehicle can change its corresponding train number during operation.

Each time a vehicle starts running after stopping in a station due to some signal S triggering the extension of the train route, the train control software requests the train number, for which the train route starting from signal S was set, from the timetable module (see 6.5). If the train number differs from the currently set train number due to the beginning of a new train run, the train number is updated in the train control module.

Whenever a train stops at a stopping position where it is scheduled to stop or reaches its designated reference point in an operating station, the train control module forwards this information, together with a time stamp of the current internal system time (see 6.1), to the timetable module.

Communication with the dispatching system

Whenever a main signal in the train route list is removed due to the first axle of the train reaching the signal, the message “reachedSignal” is transmitted to the dispatching system message handler (see 6.2) to be forwarded to the dispatching system. As described in 4.2.1, the message contains information on the train number, the current operating station, the signal name and the current system time within the railway lab control software. Note that for operation under ETCS Level 2 or higher, the transmission of train positions is continuous and thus not limited to main signals. In this case, the train positions must be transmitted to the dispatching system in constant time intervals.

Each time a train route is set for a train, i.e. the train route list is extended, the message “trainRouteClaimed” is transmitted to the dispatching system message handler, containing the train number, as well as the operating stations and names of the starting and target elements.

Providing train numbers to the field level

The field level was extended to store the current train numbers of all trains within axis counting circuits.

Each time an axle counter is passed by a physical train, the train control sends a message, containing the operating station and name of the axle counter, the train number and the direction of passing to the field level. If the axle counter delimits an axis counting circuit, the received train number is stored for the axis counting circuit in the received direction. As soon as an axis counting circuit is cleared, i.e. the axis count is 0, the stored train number is deleted again. The information on the train numbers within the axis counting circuits are utilized by the automatic route setting (see 6.6).

Every axis counting circuit can only be occupied by one train at a time. Therefore, if a train number is stored in the field level, for some axis counting circuit, it is always unique.

6.5 Timetable

The main task of the timetable module is the administration of a predefined timetable. It provides relevant information to the train control (see 6.4) and the automatic route setting (see 6.6). Furthermore, it processes the incoming messages from the dispatching system message handler (see 6.2). This section describes the functionalities of the timetable module and its communication with other software modules.

Import of timetable data and internal timetable data structure

Timetables used during operation can be provided to the railway lab control software e.g. via an XML file interface which is described in Appendix C. A train run describes the operation from a starting point on the infrastructure along a designated route until an end point on the infrastructure and is defined through the timetable. A train run is always associated with a unique train number that is independent of any physical vehicle. Thus, a physical vehicle can be used to operate multiple train runs. Dependencies between train runs resulting from the same physical vehicle being used for two or more different train runs during operation can be modeled in the timetable through succession constraints, i.e. a train run can be defined not to start prior to the termination of some other train runs.

For each train run, the timetable data provides information on the train number, whether the train run enters the dispatching area, as well as a list of traversed operating stations. Additionally, for each of these operating stations, the scheduled train route, arrival and departure times are provided.

The file containing the timetable data is opened by the timetable module and imported into its own data structure. The internal timetable data structure within the timetable module contains a train run map. A train run map is an object mapping train numbers from the timetable to their train runs. A train run is modeled as an object storing information on its operating stations, train routes and scheduled times in the order of passing. The order in which operating stations are passed by a train run is assumed to be fixed and thus cannot be changed. The train run object contains a list of train run items, each representing an event occurring during the train run, in chronological order. An event always refers to a certain operating station. Events are defined to be either the arrival, the departure or the passing through an operating station without stopping. A special case of the departure event is the start of a train run in an operating station. Similarly, the end of a train run in an operating station is a special case of the arrival event. For each event, the train route through the operating station, as well as the time of the event, are stored. For each operating station, the time refers to the reference point of the particular train route. In case a train run is scheduled to stop in an operating station, two train run items exist for this operating station, one of which represents its arrival, whereas the other one represents its departure. In case the train is scheduled to stop in an operating station, its dwell time is given as the time difference between its arrival and departure time. Additionally, a minimum dwell time can be defined for any scheduled stop of a train run. Figure 43 gives an overview of the timetable data structure.

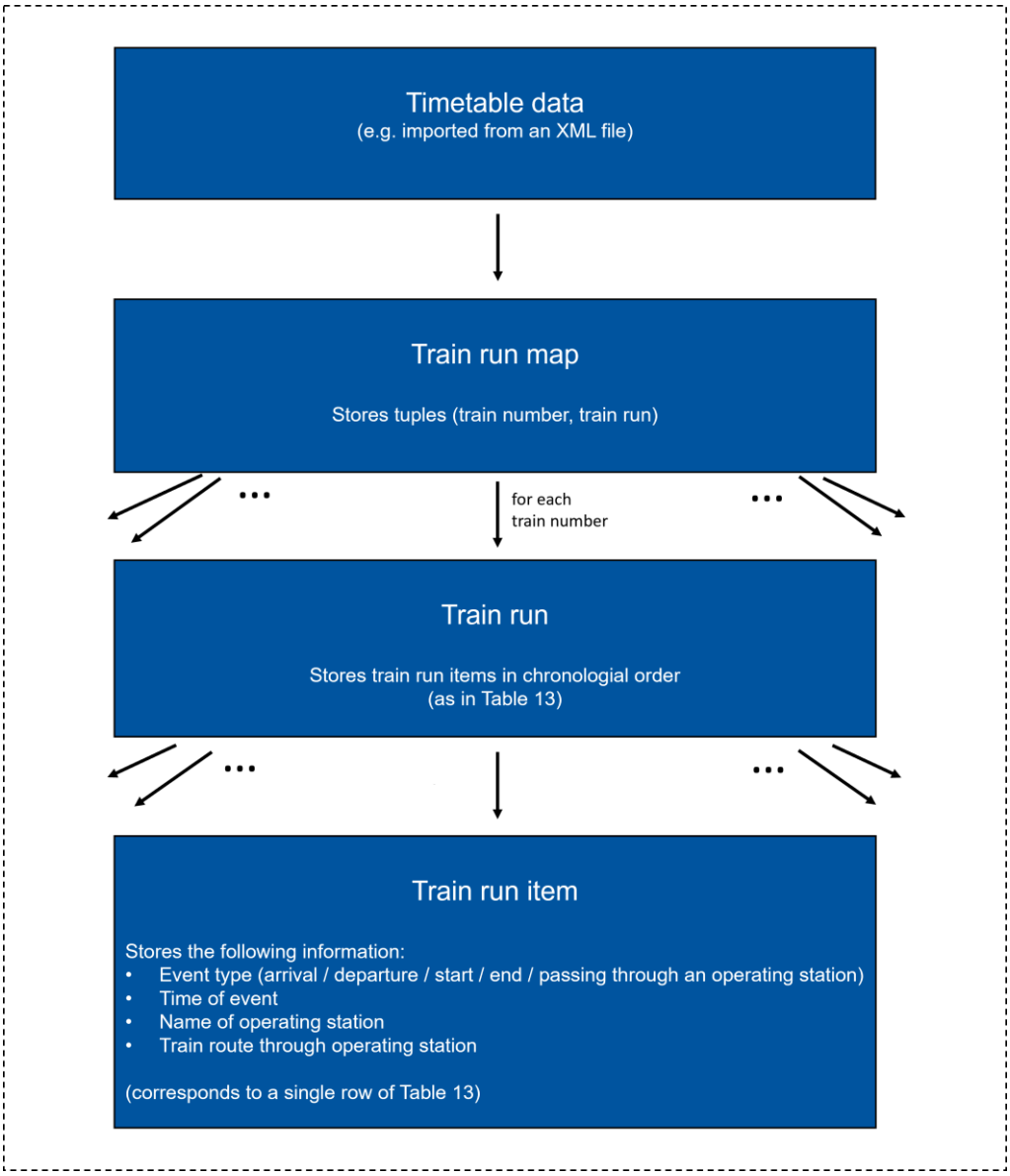


Figure 43: Data structure of the timetable

As an example, consider the extract of a train run shown in Figure 44. The train run starts in operating station OS_1, then stops in operating station OS_2 for 2.5 minutes, passes through operating station OS_3 and finally arrives in operating station OS_4, where it terminates.

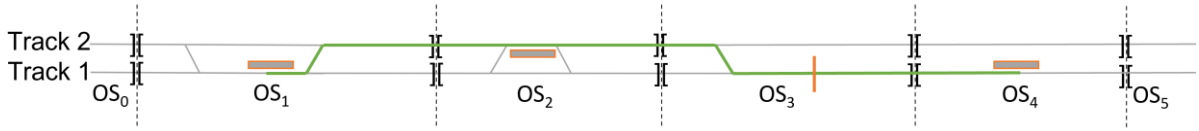


Figure 44: Train run through consecutive operating stations

The corresponding list of train run items contained in the train run is given in Table 13, where each row represents a train run item with its corresponding parameters. The train routes are encoded as described in section 4.2.2. Additionally, for train runs starting inside the dispatching area, the string encoding the first train route begins with the track number it starts from and conversely, for train runs ending inside the dispatching area, the string encoding the last train route ends with the track number it terminates on.

Table 13: List of train run items of a train run

Event	Time	Operating station	Train route
start	08:00:00	OS_1	1 – OS_2 (2)
arrival	08:15:30	OS_2	OS_1 (2) – 2 – OS_3 (2)
departure	08:18:00	OS_2	OS_1 (2) – 2 – OS_3 (2)
passing through	08:25:12	OS_3	OS_2 (2) – 1 – OS_4 (1)
end	08:33:40	OS_4	OS_3 (1) – 1

Due to the integrity of physical vehicles, certain train runs can only start once their preceding train run has terminated. Thus, relations on the succession of train runs relying on the same physical vehicle are also stored in the timetable module.

Train runs entering or exiting the dispatching area are assumed to start or end in another operating station outside of the dispatching area that is connected to all tracks entering and exiting the area. This operating station outside of the dispatching area is called fiddle yard. Figure 45 illustrates this concept. In reality, multiple operating stations are connected to the dispatching area. However, for evaluating dispatching algorithms within the dispatching area, no distinction between the operating stations outside this area is necessary, thus the simplified model, merging all these operating stations into one single station, is legitimate.

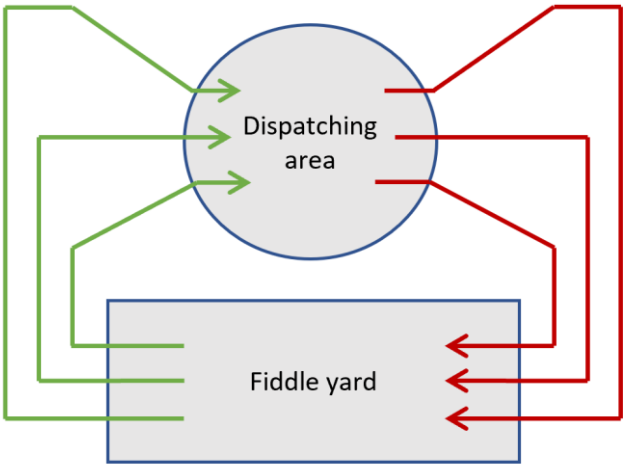


Figure 45: Overview of the relation between the dispatching area and the fiddle yard

The fiddle yard provides a place outside of the dispatching area where trains can disappear to and emerge from. Without a fiddle yard, all physical vehicles would have to be kept somewhere inside the dispatching area after the termination of train runs that are supposed to leave the dispatching area, potentially blocking parts of the infrastructure for other trains. However, as the fiddle yard can only store a finite number of trains at a time, a physical vehicle reaching the fiddle yard late might lead to the delayed entry of another train run that uses the same vehicle.

Synchronization of the system time

During operation, it is crucial for the timetable module to always know the current system time within the railway lab control software. For this reason, the internal time within the

timetable module is synchronized with the time provided by the system clock (see 6.1). Through a connection between the timetable module and the system clock module, the internal time within the timetable module is updated once the system time changes, e.g. once every second. Thus, the timetable does not have its own clock, but instead uses the system time which is centrally provided by the system clock module. This way, inaccuracies caused by small time deviations between two distinct clocks are avoided.

Processing of dispatching measures

The dispatching system message handler (see 6.2) forwards dispatching measures to the timetable module by transmitting the relevant parameters (see 4.2.2). Transmission of these information is achieved by two functions provided by the timetable module.

The function processing time changes is defined as $timeChange(TN \mid OS \mid t_a \mid t_d)$ where TN is the train number, OS is the operating station and t_a and t_d are the new arrival and departure times at the reference element within OS. The second function, which is processing route changes, is defined as $routeChange(TN \mid OS \mid str_{route})$ where TN is the train number, OS is the operating station and str_{route} the new train route through OS. The dispatching system message handler calls the respective function of the timetable module when receiving a message from the dispatching system.

The timetable module then processes the received information and provides it to the train control and automatic route setting during operation. For this purpose, a second timetable is administrated. The timetable read from the timetable data file is used to later evaluate achieved punctuality and is therefore never changed. The second timetable is used both to store dispatching measures and to store operational data during operation and is thus dynamically adjusted, as new information become available. It is used as dynamic reference timetable during operation, i.e. the trains run according to this timetable instead of to the timetable scheduled prior to real-time operation.

In case of a route change, the parameters TN (the train number), OS (the operating station) and str_{route} (the string encoding the new train route) are provided. In the dynamic reference timetable, within the train run associated with train number TN, all train routes of train run items with operating station OS, are changed to str_{route}.

In case of a time change, the parameters TN (the train number), OS (the operating station), as well as t_a (the new arrival time at the reference element) and t_d (the new departure time at the reference element) are provided. Let T_a and T_d be the currently scheduled arrival and departure times for the train run of TN in OS. In case the train passes through OS, $T_a = T_d$. Note that T_a and T_d might already result from a previous dispatching measure. The following four cases are distinguished.

1. $\boldsymbol{T_a = T_d}$ **and** $\boldsymbol{t_a = t_d}$

   In this case, the train is currently scheduled to pass through OS without stopping and no additional stop is scheduled as a dispatching measure. The dispatching measure provides the new time t_a for passing through.

   The time of the train run item is updated to t_a.

2. $T_a \neq T_d$ and $t_a \neq t_d$

In this case, the train is currently scheduled to stop in OS and the stop is not omitted as a dispatching measure. The dispatching measure provides a new arrival or departure time.

The time of the train run item with the event "arrival" or "end" in OS, is update to t_a and the time of the train run item with the event "departure" or "start" in OS, is update to t_d.

3. $T_a = T_d$ and $t_a \neq t_d$

In this case, the train is currently scheduled pass through OS without stopping, however, due to a dispatching measure, a new stop is added.

Let (passing through, $T_a, OS, str_{route},$) be the current train run item in OS. This item is replaced by (arrival, t_a, OS, str_{route}) and (departure, t_d, OS, str_{route}).

4. $T_a \neq T_d$ and $t_a = t_d$

In this case, the train is currently scheduled to stop in OS, however, the stop is cancelled as a dispatching measure.

Let (arrival, T_a, OS, str_{route}) and (departure, T_d, OS, str_{route}) be the current train run items in OS. These items are replaced by the new item (passing through, t_a, OS, str_{route}).

Management and storage of operational data

The timetable module stores operational data generated during real-time operation on the railway lab. When a train arrives at its designated stopping position or reaches its reference point in an operating station, the train control provides this information to the timetable module, together with a time stamp. The time stamp always refers to the current system time within the railway lab. These arrival times are stored in the dynamic reference timetable and cannot be changed anymore, thus enabling later evaluations of punctuality by comparison with the original timetable.

Once a train run arrives at its final destination, the train run is considered as completed. In case the train run has a succeeding train run that can only be started once its preceding train run terminated, the succeeding train run is unlocked.

If a train arrives at a stopping position where it is scheduled to stop and the difference between its actual arrival time and its departure time in the dynamic reference timetable is less than the minimum dwell designated for the train run in this operating station, the departure time in the dynamic reference timetable must be adjusted. Let t_a be the actually achieved arrival time and T_d the departure time in the dynamic reference timetable. Further, let T_{dwell} be the minimum dwell time. Then the new departure time in the dynamic reference timetable is set to $t_d = \max(T_d,\ t_a + T_{dwell})$.

Providing information to the automatic route setting

For a fully automated train operation, train routes need to be set automatically, depending on the scheduled or rescheduled routes of the trains. For this reason, the timetable module provides information on intended routes to the automatic route setting (see 6.6). The automatic route setting can send requests for the setting of train routes to the interlocking

system, while avoiding unnecessary deceleration in front of signals a train is not scheduled to stop at. A detailed description of this software module and its functionalities can be found in 6.6. To make use of the automatic route setting, precise information on which route shall be set needs to be provided.

As in 5.3.4, starting elements for routes are defined as main signals and target elements are defined as main signals or borders between operating stations. Note that for the route from a given main signal via a given operating station border, the destination signal is uniquely defined by the first entry signal after the operating station border in the respective direction.

For each train route through an operating station OS_1 of a train run, a list of the target elements along the route, in chronological order, is generated. The border to the succeeding operating station OS_2 is encoded via "$OS_1\_OS_2\_n$" where n is the is the track number leading to the border. As the train route search algorithm (see 5.3.4) can only process starting and target elements within the same operating station, this string is considered as the target element when setting the train route leading from OS_1 to OS_2 and is added at the end of the list of target elements. As an example, consider the station layout shown in Figure 46. The train passes through operating station OS_1 via the signals A and B and continues towards operating station OS_2 on track 1. Thus, for this train run, the list of target elements in OS_1 is given by [A, B, $OS_1\_OS_2\_1$].

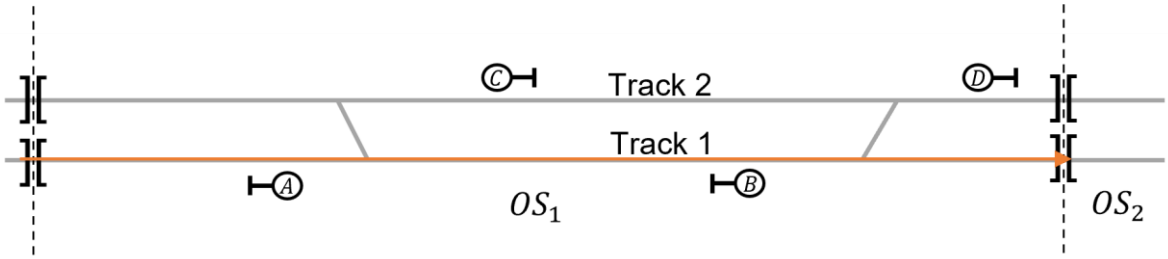


Figure 46: Example of a station layout

If a train is not scheduled to stop in front of a signal, the train route should ideally be set before the train starts decelerating towards the signal. The time at which the train route should be set depends on the speed and the braking distance of the train. For this reason, the automatic route setting requests the current maximum speed v_{max}, as well as the braking distance from v_{max} to 0 km/h from the timetable module which provides this information. The braking distance is computed using the functionalities provided by the train control.

If a train run shall initially start or continue its journey after a scheduled or operating stop, the timetable module sends information on the intended train route to the automatic route setting. The starting times of all train runs that were not yet initially started, are sorted in ascending order. Whenever the system time is updated and the starting time of a train run is reached, the timetable module either sends the request to set the initial train route of the train run to the automatic route setting, if all preceding train runs arrived at their final destination, or otherwise it stores the train run in a waiting list and the request to the automatic route setting is postponed until all preceding train runs terminated. When a train stops at a stopping position where it is scheduled to stop, the current system time is compared with its departure time, every time the system time is updated. Once the system time reaches the departure time, the next train route for this train run is transmitted to the automatic route setting. Note that the dwell time might still change after arrival when new dispatching measures are received. By always comparing the system time with the current departure time,

which is stored in the dynamic reference timetable, consideration of the latest dispatching measures is ensured.

All train route requests are transmitted to the automatic route setting by providing the starting signal and the target element of the train route, where the target element can either be another signal or the border to another operating station.

Calculation of new maximum speeds

In order to meet the arrival times given through the dynamic reference timetable, a train's speed profile is adjusted dynamically during operation. For this purpose, the timetable module calculates new maximum permitted speeds which a train shall not exceed. As the dynamic reference timetable provides arrival times at the reference points, a new maximum permitted speed is valid along an infrastructure segment between two consecutive reference points. A train's speed profile is therefore adjusted with a new maximum permitted speed whenever the train reaches a reference point and starts approaching its next reference point. This way, the maximum permitted speed until the next reference point is adjusted as early as possible while not affecting the train's arrival time at the previous reference point.

To compute such a new maximum permitted speed, the timetable module requires information on the speed profile (see 5.3.5) that is given through the infrastructure elements between the two reference points. For any candidate for a new maximum permitted speed, the time required to run from the starting reference point until the destination reference point can be computed, thus the optimal new maximum permitted speed can be determined iteratively.

To obtain this speed profile, a train route search similar to the one described in section 5.3.4 is performed, yielding the list of infrastructure elements required for the speed profile generation. However, the route search within the timetable module terminates at the next reference point, instead of at the terminal elements within the train route search module. Note that the train routes until the destination reference point might not be completely set by the time the new maximum speed is computed.

Applying the train route search method, the list of traversed infrastructure elements and their distances, starting from the current train position until the reference point, is generated. In case the train changes its running direction when departing, the current train position from which the train route search is started, is adjusted accordingly, as described in section 6.4. Based on this list of infrastructure elements and the current train speed, the speed profile including acceleration and deceleration curves is generated with respect to the locally permitted maximum speeds, as described in section 6.3. Note that the speed profile does not necessarily have to end at 0 km/h if the train passes through the next operating station without stopping at the next reference point.

The overall time consumption, when running according to this generated speed profile, is calculated. The time required for acceleration and deceleration respectively is determined by the time intervals the speed levels are applied. Assume that there are n stretches with constant speed and m speed curves within the speed profile. Let (l_i, v_i) for $i \in \{1, \dots, n\}$ represent the stretches with constant speed where l_i is the length of the i^{th} stretch with constant speed v_i. Note that the running at v_i as the beginning or end of a speed curve is not

considered to be part of these stretches. Further, let t_i be the time required for the i^{th} speed curve for $i \in \{1, \dots, m\}$. Then the overall running time for the speed profile is calculated as

$$\sum_{i=1}^{n} \frac{3.6 \cdot l_i}{v_i} + \sum_{i=1}^{m} t_i.$$

For the stretches with constant speed, the lengths l_i are determined as the difference between the end and starting points of their preceding and succeeding speed curve respectively. In case the stretch with constant speed does not have a preceding speed curve, it starts at the current train position and in case it does not have a succeeding speed curve, it ends at the next reference point.

Let t_{sys} be the current system time within the railway lab, let t_{run} be the time required for the train to run according to the speed profile and let t_{ref} be the arrival time at the next reference point according to the dynamic reference timetable.

If $t_{sys} + t_{run} \geq t_{ref}$, the new overall maximum recommended speed until the next reference point is set to the maximum speed occurring within the speed profile. If $t_{sys} + t_{run} < t_{ref}$, a binary search algorithm (see Figure 48) is applied to find a reduced overall maximum recommended speed for the speed profile, such that $t_{sys} + t_{run}$ is as close to t_{ref} as possible. Figure 47 shows an example of a speed profile adjusted with a reduced new maximum recommended speed. Note that locally permitted maximum speeds smaller than the new maximum recommended speed still apply.

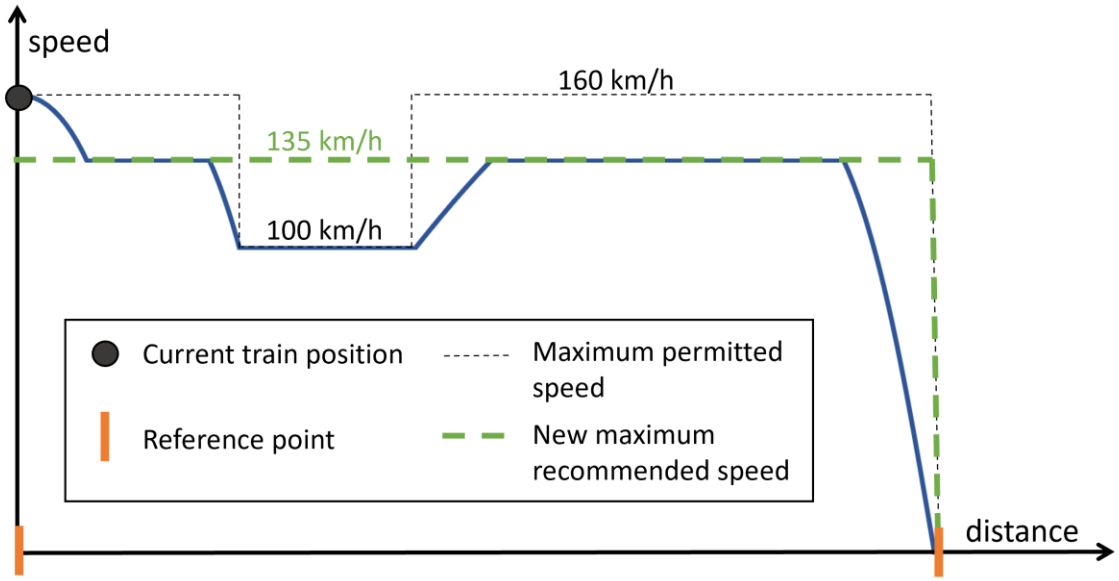


Figure 47: Speed profile with reduced overall maximum speed

For the binary search algorithms, let i_{min} and i_{max} be the minimum and maximum index of the speed level whose corresponding speed will be taken as new maximum speed for the speed profile. The indices i_{min} and i_{max} will iteratively be adjusted until $t_{sys} + t_{run} = t_{ref}$ or $i_{max} - i_{min} = 1$.

Initially, $i_{min} = 1$ and $i_{max} = n$ where n is the number of available speed levels whose corresponding speeds are in the interval (0, v_{max}] and v_{max} is the maximum overall speed occurring in the speed profile. For each iteration, let

$$i = i_{min} + \left\lceil \frac{i_{max} - i_{min}}{2} \right\rceil.$$

The original speed profile is adjusted, as shown in Figure 47, with the new maximum speed v_i corresponding to the i^{th} available speed level as new overall maximum speed. In the example shown in Figure 47, the new overall maximum speed is 135 km/h. All local maximum speeds greater than 135 km/h are set to 135 km/h, the speed curves are reduced accordingly and a deceleration curve from the current train speed to 135 km/h is added. For the stretch with a local maximum speed of 100 km/h, the speed limit remains unchanged. For details on how speed profiles are adjusted to meet a new overall maximum recommended speed, see also 6.4.

Set t_{run} as the time required for the train to run according to the adjusted speed profile, as described above. In case $t_{sys} + t_{run} = t_{ref}$, the new overall maximum recommended speed until the next reference point is set to v_i. Otherwise, if $t_{sys} + t_{run} < t_{ref}$, the train would arrive at the next reference point too early, so the overall maximum recommended speed has to be decreased. Thus, i_{max} is set to i. Conversely, in case $t_{sys} + t_{run} > t_{ref}$, the train would arrive at the next reference point too late, so the overall maximum recommended speed has to be increased. Thus, i_{min} is set to i.

The search terminates as soon as $t_{sys} + t_{run} = t_{ref}$ or $i_{max} - i_{min} = 1$ during some iteration step. In the first case, the speed v_i from this iteration step is set as new overall maximum recommended speed. In the second case, let t_{run} and $t_{run}{}'$ be the times required for the speed profiles with a maximum speed of $v_{i_{max}}$ and $v_{i_{min}}$ respectively.

Then, the overall maximum recommended speed for the speed profile is set to $v_{i_{max}}$ if

$$\left|t_{sys} + t_{run} - t_{ref}\right| \leq \left|t_{sys} + t_{run}{}' - t_{ref}\right|$$

and otherwise to $v_{i_{min}}$. Thus, the speed that yields the arrival time that is closest to t_{ref} is chosen and if the optimal speed lies exactly between two available speeds, the higher speed is chosen. Alternatively, the higher speed or the lower speed can always be chosen to prevent a train from arriving too late or too early at the reference point. Note that, however, for sufficiently small speed level steps, this will not significantly impact the overall travel time.

Figure 48 gives an overview of the binary search algorithm described above.

Instead of decreasing the new maximum speed, the dwell time of the train could alternatively be extended at the current stopping position, if the train currently stops, to achieve a later arrival at future operating stations. This option is not considered, however, in order to maintain existing running time reserves which might be used to reduce delays in case of unexpected future events. Additionally, from an energy-saving perspective, lower cruising speeds are preferrable.

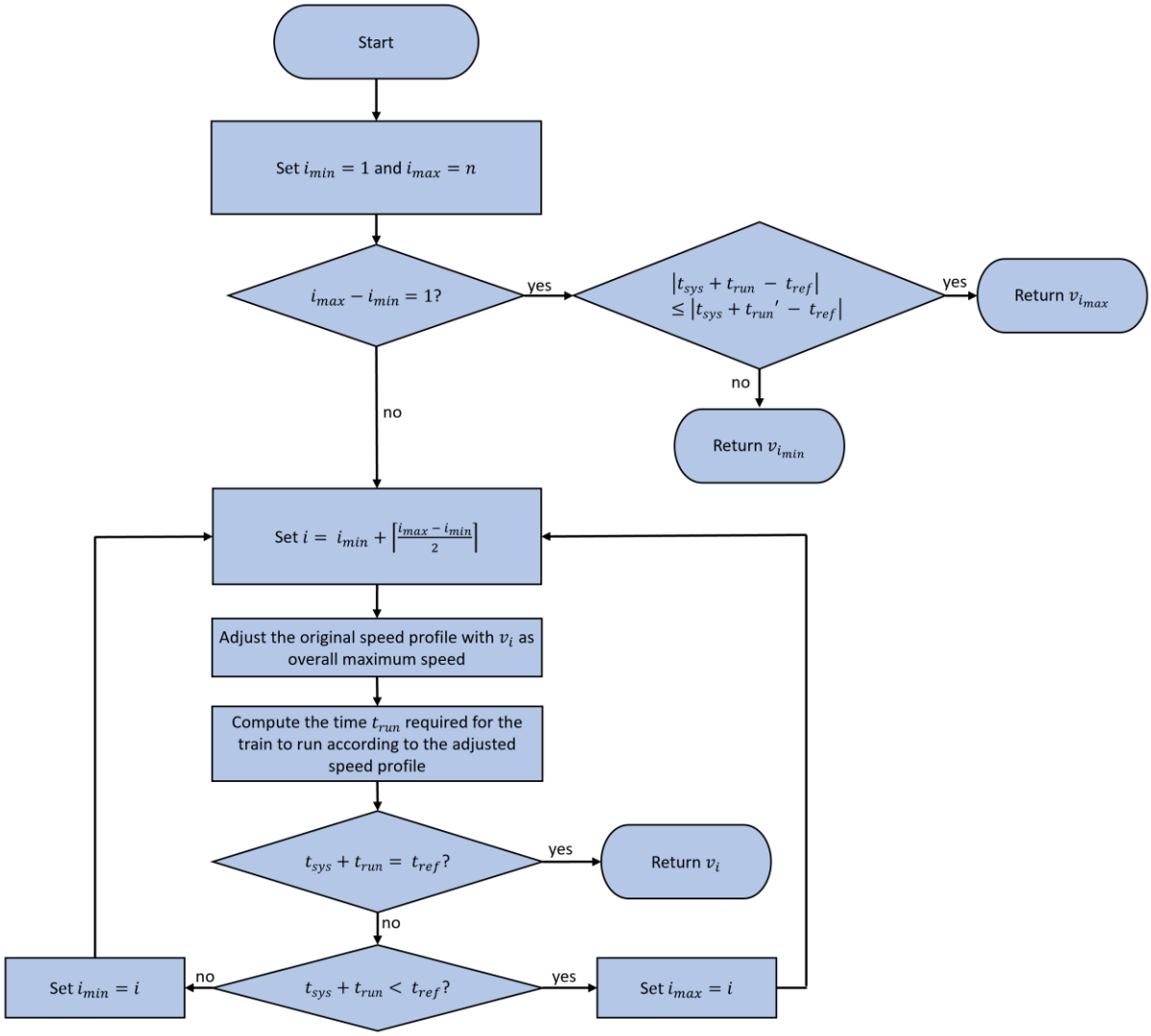


Figure 48: Overview of the binary search algorithm to find a new overall maximum recommended speed meeting the given arrival time at the next reference point

Providing predicted times of entry to the dispatching system

The timetable module provides predicted times of entry into the dispatching area to the dispatching system message handler (see 6.2), which forwards the information to the dispatching system. For a train number TN_i of a train run entering the dispatching area, let t_{a_i} be the time of entry that was originally scheduled prior to operation and let $d_{i,0}$ denote the predefined delay at entry. Initially, when the timetable and the delay data are processed, the predicted time of entry is set to $t_{a_i} + d_{i,0}$ for all trains entering the dispatching area.

During operation, the predicted time of entry is transmitted to the dispatching system via the dispatching system message handler (see 6.2) in advance. Fixed time intervals can be set, at which predictions are transmitted prior to entry, to resemble realistic conditions, as delays at entry usually develop over time and information are available before the actual time of entry. Additionally, if the time until the predicted time of entry is less than the set time interval for a train run, when the operation is initially started, the prediction is transmitted to the dispatching system message handler as well.

Structure and processing of delay data

To enable operation of delayed trains that require dispatching measures, delays have to be defined and implemented during operation accordingly. Lists with data on delays at entry of trains entering the dispatching area, as well as primary delays, originating at scheduled and operating stops during operation, can be provided to the timetable module. These delay data are only accessible by the timetable module. The delay values can either be generated randomly, e.g. based on the delay distribution function with changeable parameters or edited manually. By using the delay data from files that were created prior to operation, the files can be reused to evaluate different dispatching strategies under the same operational conditions.

Let $\{TN_1, \dots, TN_n\}$ be the set of all train numbers within the timetable and further, let $\{OS_1, \dots, OS_m\}$ be the set of all operating stations in the dispatching area. The delay data is structured in a table, as shown in Table 14, where $d_{i,0}$ represents the delay at entry of TN_i in minutes and $d_{i,j}$ represents the primary delay of TN_i in OS_j in minutes. If TN_i does not enter the dispatching area or TN_i is not scheduled to stop in OS_j, $d_{i,0}$ or $d_{i,j}$ are not defined respectively, thus the respective cells remain empty.

All delay values that are defined are non-negative. The timetable module reads the delay data from the provided file and stores it.

Table 14: Structure of the delay data

Train number	**Delay at entry**	**Primary delay OS_1**	**Primary delay OS_1**	**…**	**Primary delay OS_m**
TN_1	$d_{1,0}$	$d_{1,1}$	…	…	$d_{1,m}$
TN_2	⋮	⋱			
TN_3	⋮		⋱		
⋮	⋮			⋱	
TN_n	$d_{n,0}$	…	…	…	$d_{n,m}$

In case a train TN_i enters the dispatching area and $d_{i,0} > 0$, the request to the automatic route setting to set the initial route for TN_i and thus the departure of TN_i is postponed by the delay at entry $d_{i,0}$.

If $d_{i,j} > 0$, for a train TN_i in an operating station OS_j, two cases are distinguished during operation. Let T_a and T_d be the arrival and departure times according to the dynamic reference timetable respectively. Note that T_d already includes past dispatching measures, as well as the minimum dwell time. Further, let T'_d be the departure time that was initially scheduled prior to operation and T_{dwell} be the minimum dwell time. The departure time T_d is not changed if $\max(T'_d,\ T_a + T_{dwell}) + d_{i,j} \leq T_d$. Otherwise, T_d is set to

$$\max(T'_d,\ T_a + T_{dwell}) + d_{i,j}.$$

Thus, the primary delay is added to the departure time without any dispatching measures and only the part of a dwell time prolongation above the primary delay is considered, to account for the fact that the dispatching system is not aware of primary delays beforehand. The train

route request that is required prior to the departure of TN_i in OS_j is sent to the automatic route setting at time T_d. Thus, if T_d was increased due to the primary delay, the departure is postponed accordingly.

Generation of randomized delay data

Delay data are structured and stored in tables as described above. Delays at entry into the dispatching area and primary delays can either be edited manually or generated randomly, e.g. according to the delay distribution function.

The delay distribution function was first introduced by Schwanhäußer in [146] and is defined as

$$F_T(t) = \begin{cases} 0, & if\ t < 0 \\ 1 - p_d \cdot e^{-\lambda \cdot t}, & if\ t \geq 0 \end{cases}$$

where p_d is the probability of a train delay and $\frac{1}{\lambda}$ is the average delay of a delayed train. For both delays at entry into the dispatching area and primary delays, the parameters p_d and λ can be individually set, depending on the type of train. The train types distinguished in the timetable module are long-distance passenger trains, fast regional passenger trains, regional passenger trains, long-distance freight trains and regional freight trains.

The delay distribution function is composed of a one-point distribution (either a train is delayed or not) and an exponential distribution function. Using e.g. the “mt19937” pseudo-random generator [147] with the “random” class provided within C++ [148], random delay data can be generated, based on the respective parameters of the delay distribution function.

For each entry in the delay data table, the random generator determines, whether a delay occurs, using the discrete distribution function defined as

$$F(x) = \begin{cases} p_d, & if\ x = 1 \\ 1 - p_d, & if\ x = 0 \end{cases}$$

where the train is considered as delayed, if $x = 1$, and punctual, if $x = 0$.

If a delay occurs, the amount of delay is determined by the random generator, using the exponential distribution function described by

$$F(t) = \begin{cases} 0, & if\ t < 0 \\ 1 - e^{-\lambda \cdot t}, & if\ t \geq 0. \end{cases}$$

6.6 Automatic Route Setting

The automatic route setting is a module within the extended railway lab control software that enables route setting during automatic operation. Its task is to request scheduled train routes from the interlocking system.

Route setting plan

A fixed route setting plan defines route setting rules containing conditions and interdependencies for the setting of train routes. The route setting plan is e.g. stored in an XML file format and initially read when the automatic route setting module is started. An example for this XML format can be found in Appendix D.

Each route setting rule is associated with one specific main signal and its triggering axis counting circuit on the infrastructure. Note that a main signal can have more than one triggering axis counting circuit and an axis counting circuit be associated with more than one signal.

A route setting rule in the route setting plan is associated with a specific signal and contains the following information:

- the operating station and name of the associated signal,
- the distance between the beginning of the axis counting circuit and the signal,
- whether or not there is a stopping position in front of the signal,
- the operating station and name of the signal‘s triggering axis counting circuit,
- optionally the operating station and name of a condition signal.

The beginning of an axis counting circuit is defined as the position of its first delimiting axle counter that is passed when approaching the signal in its respective direction.

Route setting in case of no scheduled or operating stop

In order to set train routes on time and avoid unnecessary braking, the time at which a train would start braking in front of a signal if the signal aspect did not show “proceed” is essential. However, a physical railway lab normally provides no possibility for continuous train localization. The locations of the axle counters are often the only positions where exact train localization can be performed and therefore the concept of triggering axis counting circuits is applied for the automatic route setting. When approaching a main signal, the train enters a triggering axis counting circuit associated with this main signal. This information is used to predict when a train would start braking and when to ideally set the new train route.

Whenever the train number within a triggering axis counting circuit A for a route setting rule changes in the field level (see 5.3.2), a time $t[sec]$ is computed as a lower bound for the time until deceleration would have to start, if the signal S in the operating station OS associated with the route setting rule does not change its aspect to “proceed”. The current maximum speed $v_{max}[km/h]$, as well as the braking distance $d_{br}[m]$ required to brake from v_{max} until 0 km/h, are requested from the timetable module (see 6.5). Let $d[m]$ be the distance between the beginning of A and S.

Figure 49 illustrates this setup. The time t is computed as

$$t = \frac{3.6 \cdot (d - d_{br})}{v_{max}}.$$

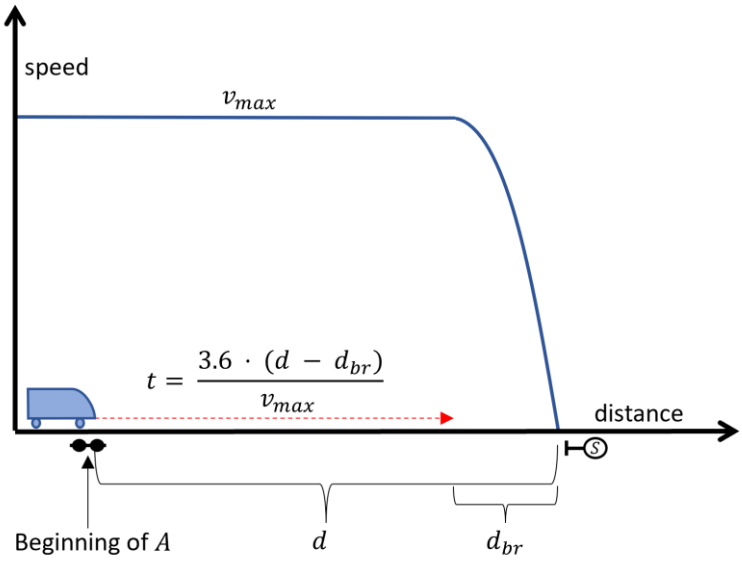


Figure 49: Route setting on case of no scheduled or operating stop

Note that the train might currently run at a speed lower than v_{max}, however v_{max} serves as an upper bound for the train speed. Thus, t is the minimum time required until the starting point of the deceleration is reached. The triggering axis counting circuit of a main signal, defined in the route setting plan, is chosen such that $d > d_{br}$ for all admissible speeds and possible braking decelerations, i.e. the distance until the main signal is larger than the braking distance to avoid unnecessary braking.

A timer is started, timing out after $t - 3$ seconds, where 3 seconds is the assumed route setup time. Once the timer times out, the current list of target elements for OS and the information on whether or not the train is scheduled to stop in OS, are requested from the timetable module. The request is thus made at the latest possible time to ensure that up-to-date information from the dynamic reference timetable is used while still ensuring the prevention of unnecessary deceleration. This way, route changes or the adding or cancellation of stops, received from the dispatching system after the timer was started, can still be considered. If S is not in the signal list, the route setting rule is ignored. If S is in the signal list and the train is not scheduled to stop, it is taken as starting element and its successor in the list is taken as target element for route setting. The resulting route request is sent to the interlocking system. If the train is scheduled to stop, the route setting rule is ignored. Note that for each route setting rule, whose signal is associated with the axis counting circuit A, these steps are performed simultaneously, however, only the route setting rule whose associated signal is in the signal list of the train will lead to a route setting request being sent to the interlocking system.

There are cases where a route setting rule for a signal S must also have an associated condition signal C to prevent false route setting, as in Figure 50. If such a condition signal exists, the train route starting from S can only be set if the aspect of signal C was changed to “proceed” after the train entered the triggering axis counting circuit of S. Every time a train route from signal C to signal S is set, the train number of the train for which the route was set is stored

in the variable $lastTrainNumber$ for this route setting rule. The route starting from S will only be requested for a train with train number TN, if $lastTrainNumber = TN$. Once the route was requested, $lastTrainNumber$ is reset to an empty string.

As an example, consider the operational scenario shown in Figure 50. The distance between the two consecutive signals C and S is not sufficient for the axis counting circuit of the route setting rule for S to start after C and the signals share the same triggering axis counting circuit A. When train T_2 approaches C and enters A, the timers for the route setting rules associated with C and S are both started. When the timer for C times out, the route request from C to S is sent to the interlocking system. However, as train T_1 is currently stopping in front of S, the route for T_2 from C to S cannot be set. As soon as the timer for signal S times out, a route request to the interlocking system would falsely result in the route starting from signal S being set for train T_1. However, since $lastTrainNumber = T_1 \neq T_2$, the train route will not be set for T_2 until the route from C to S has been set for T_2 first.

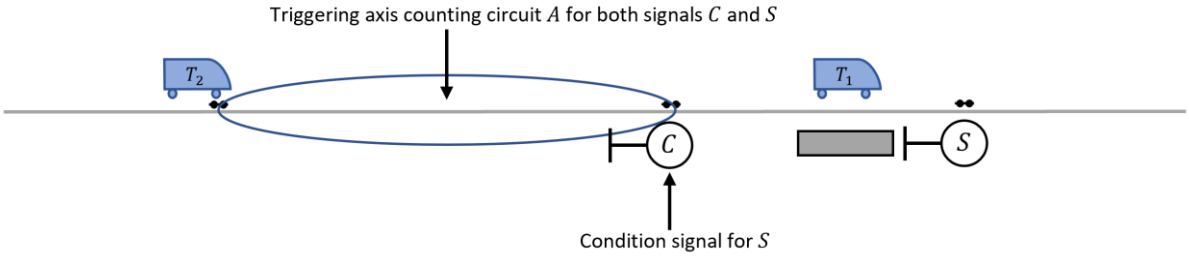


Figure 50: Condition signal preventing a false route setting

Route setting after a scheduled or operating stop

If a train run shall initially start or continue after a scheduled or operating stop, the timetable module sends a request to the automatic route setting, containing the operating station OS_1 and names of the starting and target elements for the route (see 6.5). Starting elements are always signals. Target elements can either be signals or operating station borders, encoded as $OS_1\_OS_2\_n$ where OS_1 is the current operating station, OS_2 is the destination operating station and n is the track number within OS_1 leading towards OS_2. Upon receiving such a request from the timetable module, the automatic route setting forwards the request to the interlocking system (see 5.3.3) which sets the route, if possible. If route setting is not possible, e.g. due to an occupied axis counting circuit, the interlocking system module adds the train route to its waiting list such that the automatic route setting only has to send the request once.

7 An Approach Combining Real-Time Dispatching with Energy-Efficient Driving

This chapter presents a method for the optimization of energy consumption for single train runs within mixed traffic operation under a fixed block signaling system, after the dispatching process was performed. The proposed method was already published in [7].

It is assumed that the primary goal of the dispatching process is the minimization of weighted delays and energy consumption is not considered during dispatching. The method aims at reducing unnecessary braking and acceleration of vehicles and thus increasing cruising times, as suggested in [88, 149], since particularly acceleration contributes to the overall energy consumption immensely. It is further assumed that information on the predicted block clearing times of preceding and succeeding trains are used by a central traffic management system to provide the trains with recommended signal passing times which are used for energy optimization by onboard adjustment of target cruising speeds. The presented method was also implemented within the railway lab control software and a case study was performed to evaluate its performance (see 9.2). While the extended railway lab control software (see chapter 5 and chapter 6) and the communication interface with a dispatching system (see chapter 4) provide a framework for the realistic operation of trains under real-time rescheduling, the energy consumption of trains operated on a railway lab usually does not allow for conclusions about the energy consumption of a corresponding real train. Thus, for evaluations, the energy consumption is calculated based on speed data recorded during the operation of trains on a railway lab. Realistic assumptions on driving dynamics have to be made to determine the corresponding energy consumption (see 9.2.2).

Many existing approaches for energy-saving of mixed traffic during rescheduling include energy optimization within the rescheduling process itself, e.g. [102, 103]. The proposed method is applied after the rescheduling process was completed. By optimizing the energy consumption in such a way that arrival times and scheduled stops will not be delayed any further, no additional delays occur due to energy-saving and customer satisfaction is not negatively affected. Another advantage of onboard energy optimization is the reduction of computational complexity during the rescheduling process. Specifically for large dispatching areas with a high number of trains, the onboard energy optimization, which is performed detached from the dispatching system, thus bears significant advantages. There are also approaches for onboard energy optimization in the scientific literature. However, the focus is mainly on certain operational situations, such as e.g. energy-efficient merging at junctions [107]. The newly proposed method, on the other hand, is not restricted to any particular operational situation and can thus be applied universally.

7.1 Prerequisites for the Method

The developed method for energy saving aims at the onboard optimization of energy consumption by adjusting cruising speeds to prevent unnecessary braking.

It is assumed that all vehicles constantly transmit information on their current position and speed to a central traffic management system. In case of occupation conflicts, rescheduling is performed centrally and recommended time corridors for the passing of block signals are

forwarded back to the vehicles which in return perform individual onboard energy optimization. It is assumed that energy optimization was not considered during the rescheduling process already. The energy saving approach is thus decentralized and builds on the idea of connected driver advisory systems.

Let $B_1, \dots, B_n$ denote the block sections a train is scheduled to pass until its next scheduled or operating stop, in chronological order. Further, let $S_1, \dots, S_{n+1}$ be the block signals delimiting these block sections, i.e. block section B_i is delimited by the block signals S_i and S_{i+1}. The earliest possible time the train can pass signal S_i is the predicted block clearing time of block section B_i by the preceding train plus some overhead for the route setup time, the signal watching time and the approach time (see 3.4). The latest time the train should pass signal S_i is the predicted starting time of block occupation of B_{i-1} by the succeeding train minus some overhead for the clearing time and route release time (see 3.4). These times yield a recommended time corridor for the train to pass signal S_i.

Figure 51 shows an example of a recommended time corridor for a train T, marked by the red line. The earliest possible signal passing for signal S_2 is the time at which B_2 is cleared by the preceding train, plus the route setup time, the signal watching time and the approach time for T in B_2. The latest recommended signal passing time for signal S_2 is the time at which the block occupation of B_1 by the succeeding train starts, minus the clearing time and route release time for T in B_1.

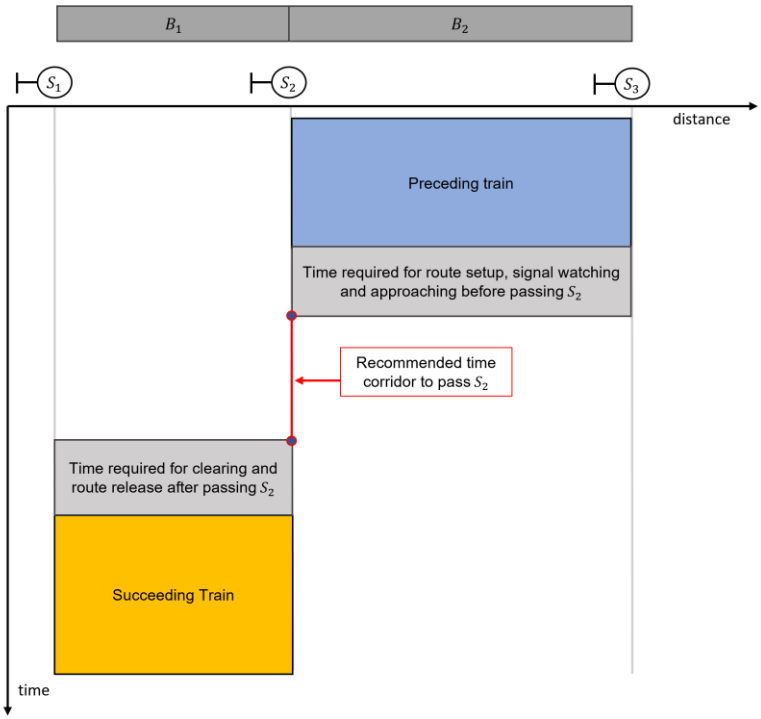


Figure 51: Recommended time corridor for signal passing

It is assumed that a central traffic management system provides such recommended time corridors to the trains during operation. Depending on the desired level of robustness, the recommended time corridors can be shortened both at the beginning and the end, such that small deviations from the predicted running times do not diminish energy-saving disproportionately. The method proposed in section 7.2 is applicable independently of how

the recommended time corridors are computed. Through future research, several different approaches of how to compute such recommended time corridors could be proposed. Time corridor generation might e.g. include the consideration of multiple trains simultaneously or different levels of robustness might be taken into account. The proposed method is not tied to any specific way of computing these time corridors, thus no precise specification on time corridor generation is made here to allow for a general applicability.

7.2 Description of the Method

For the proposed method, punctuality is assumed to be the primary objective. As energy-efficient driving normally implies a tradeoff between punctuality and saved energy, the method is performed after the rescheduling process in which punctuality is optimized by minimizing the overall weighted train delay. Customer satisfaction is not affected by the presented energy saving method, as (re)scheduled arrival times at scheduled stops will not be further delayed by the algorithm.

The method aims at maximizing cruising distances and thus minimizing the amount of unnecessary deceleration and acceleration. In particular, unnecessary stopping in front of signals shall be avoided whenever possible. This is achieved by dynamically adjusting the maximum cruising speed of the train's speed profile, leading to optimized speed profiles through which the train possibly runs at a lower speed than locally permitted. As the speed profile is never adjusted beyond the next scheduled or operating stop and the earliest possible arrival time at this stop is prioritized by the method, arrival at scheduled or operating stops will not be further delayed for energy-saving purposes, aligning with the primary objective of punctuality.

To achieve the earliest possible arrival time at the next scheduled or operating stop, the new maximum cruising speed v is determined to be as high as possible while following the recommended time corridors in between. Similar to the calculation of new maximum speeds within the timetable module (see 6.5), only those maximum speeds in the speed profile that are greater than v are set to v. This way, local maximum permitted speeds smaller than v still apply. It is assumed that v is an integer number, as other speeds are hard to communicate and implement during real operation.

Consider the speed profile of a train until some signal $S_i \in \{S_1, \dots, S_n\}$ where $S_1, \dots, S_n$ are the signals along the route the train is planned to take until its next scheduled or operating stop, for which a recommended time corridor is available. Let t_{e_i} denote the earliest recommended passing time for signal S_i, whereas t_{l_i} denotes the latest recommended passing time for S_i. Further, let t_i be the actually achieved signal passing time for S_i during operation.

The method iteratively optimizes the new cruising speed. For the first iteration, define $t = t_{e_n}$ and $i = n$. The new cruising speed v until S_i is set as the maximum speed such that $t_i \geq t$. The fact that the new cruising speed is assumed to be integer allows for the computation of v using e.g. a binary search method. If the current train speed is higher than v, the train is assumed to brake from its current speed until v and if the current train speed is lower than $\min(v, v_{max})$, where v_{max} is the maximum permitted speed at the current train position, the train is assumed to accelerate from its current speed until $\min(v, v_{max})$. Once v has been

computed, the passing times of the signals $S_1, \dots, S_{i-1}$ are calculated according to the adjusted speed profile with new maximum cruising speed v. If all arrival times lie within their respective recommended time corridors, the method terminates and the cruising speed is adjusted to v until S_i. If there are violations with the recommended time corridors, let signal S_j be the signal with maximum index j for which a violation occurs. If the violation is caused by the train passing S_j too early, t is set to t_{e_j}. Conversely, if the train arrives at S_j too late, t is set to t_{l_j}. The next iteration of the algorithm is then started with the new value for t and $i = j$.

Whenever the train reaches the signal S_i until which the adjusted cruising speed was calculated or if recommended time corridors change or new ones become available, the algorithm is started again and the currently active adjusted cruising speed is possibly changed by a new one. In case the train reaches signal S_i and no recommended time corridors are available for the upcoming signals along its scheduled route, the train can accelerate to its maximum permitted speed. Note that this maximum permitted speed might be lower than the minimum of the line speed and vehicle speed, based on the (re)scheduled arrival time at the next scheduled or operating stop (for details on the adjustment of maximum speeds based on arrival times see also 6.5). Figure 52 gives an overview of the method described above.

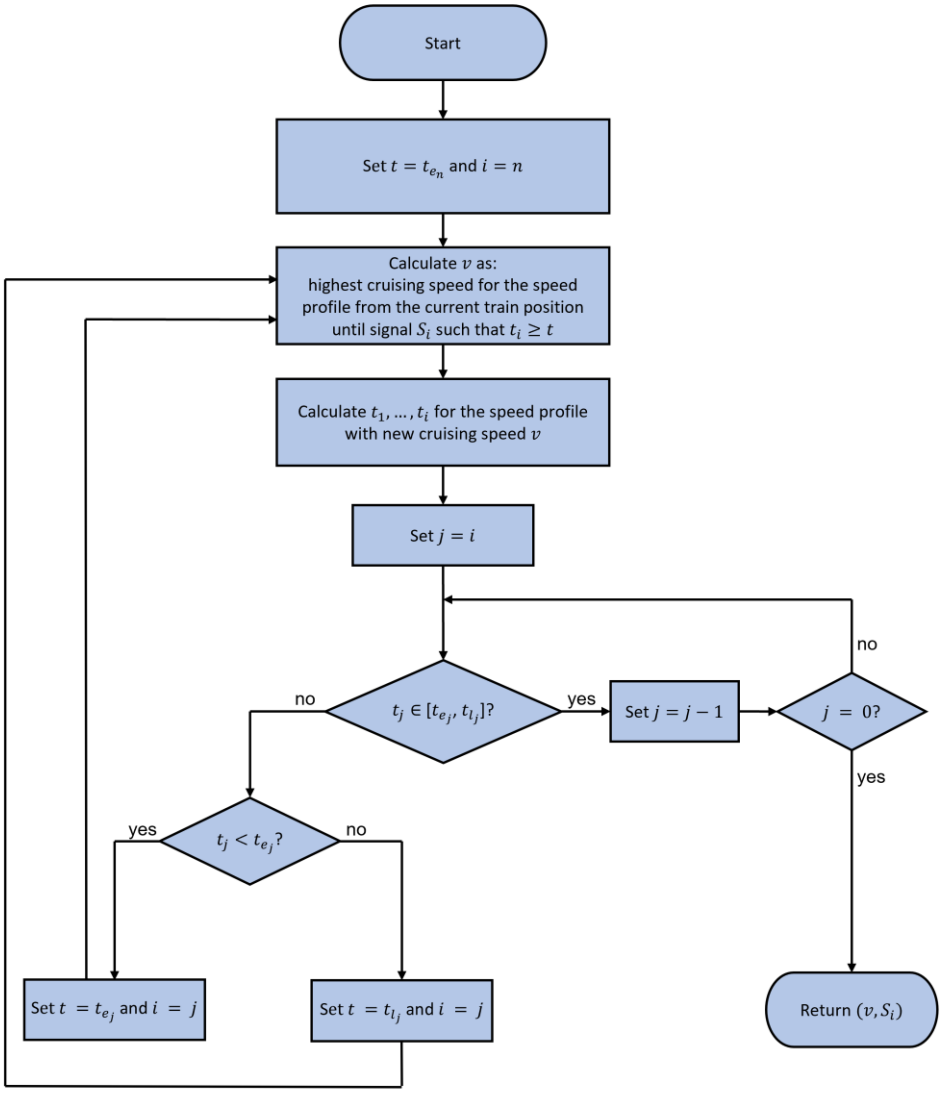


Figure 52: Overview of the method for the energy-saving adjustment of cruising speeds

As the method is not performed beyond the next scheduled or operating stop and guarantees the earliest possible arrival time at this stop, punctuality and thus passenger satisfaction is not affected. By applying the energy optimization described above, arrival times can merely be further delayed at intermediate block signals at which the train is not scheduled to stop.

An example of the method is shown in Figure 53. The red lines indicate the recommended time corridors for passing the respective block signals. In the first iteration, the new cruising speed is set such that the train arrives at signal S_6 at the earliest recommended time t_{e_6}, indicated by the dashed line 1. This would, however, cause a conflict with the recommended time corridor for signal S_5, as the train would arrive at S_5 before t_{e_5}. Therefore, in the second iteration, the speed profile is only considered until S_5 and the new cruising speed is calculated such that the train arrives at signal S_5 at t_{e_5}, indicated by the dashed line 2. This would yield a conflict at signal S_3, where the train would arrive after t_{l_3}. In the third iteration, the new cruising speed is computed such that the train arrives at signal S_3 at t_{l_3}, indicated by the green line until S_3. No more violations with recommended time corridors occur, thus the determined cruising speed is applied until the train reaches S_3 or the information on succeeding time corridors are updated. The green lines from S_3 until S_5 and from S_5 until S_6 indicate the further trajectory of the train. However, note that the last two green lines can still change during operation in case the recommended time corridors are updated before S_3 or S_5 have been reached respectively.

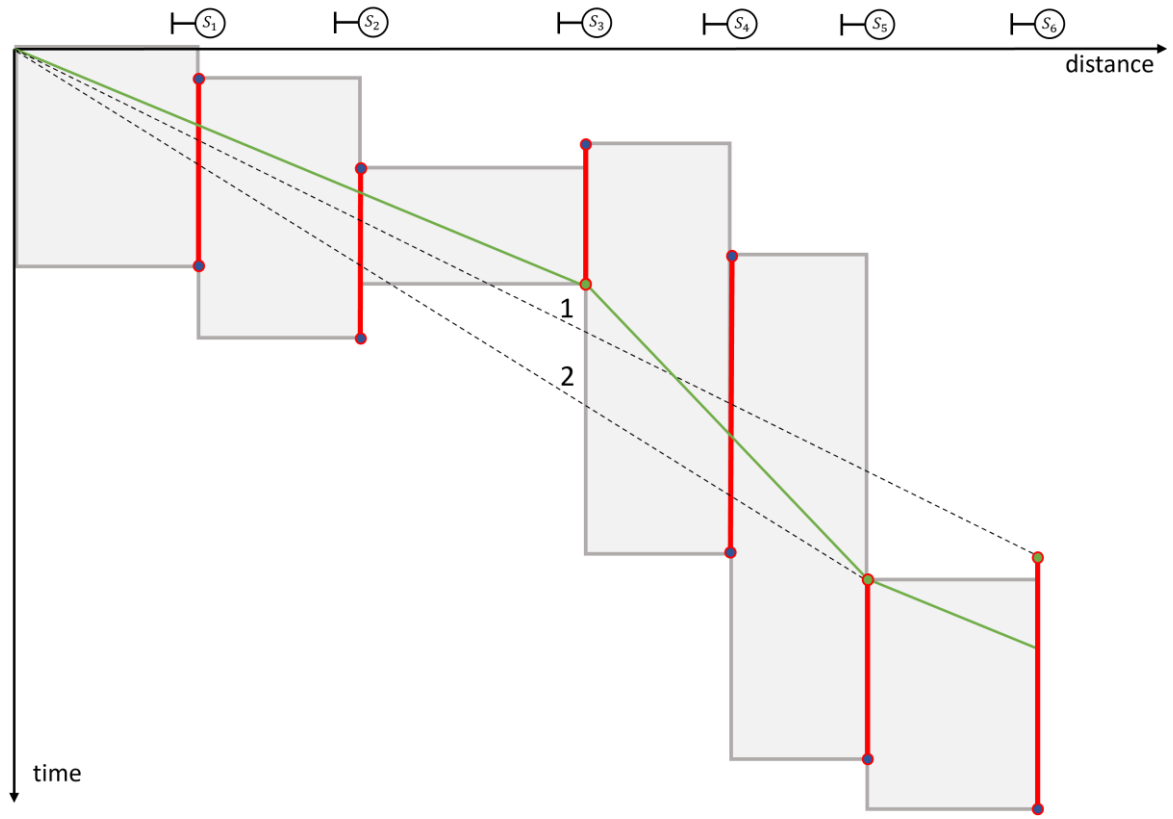


Figure 53: Example of the energy optimization method

8 Application and Testing of the Developed Software Systems

The communication interface with the dispatching system (see chapter 4) and the railway lab control software (see chapter 5 and chapter 6) are suitable to be applied on general dispatching systems and railway labs. During this research work, the developed software systems were applied and tested on the Railway Signalling Lab (ELVA) of RWTH Aachen University, using the dispatching system OptDis [1, 2] which is integrated in the software LUKS® [3, 4]. This chapter describes both the railway lab ELVA in section 8.1 and its underlying infrastructure model in section 8.2. The generation of feasible timetables is described in section 8.3. The dispatching system OptDis is introduced in section 8.4. Finally, the possibilities and limitations of this testing environment are discussed in section 8.5.

8.1 Description of the Railway Signalling Lab ELVA

The ELVA (Eisenbahntechnische Lehr- und Versuchsanlage) is the Railway Signalling Lab of RWTH Aachen University. It provides a scale model network, resembling a real infrastructure network. The rolling stock and physical tracks are modeled in the commercial scale of 1:87. Distances and speeds of the vehicles, however, are modeled in a scale of 1:200, to allow for the representation of a larger infrastructure in the limited space available within the lab.

The ELVA infrastructure contains a total of twelve operating stations. There are six regular stations with stopping positions, e.g. for passengers to exit or board a passenger train or for goods to be unloaded or loaded onto a freight train. Two operating stations serve as crossovers, one operating station models a junction and two operating stations represent sections of open track between operating stations. Additionally, a fiddle yard provides a total of 15 tracks to store vehicles. It is connected to three of the stations in both directions, i.e. vehicles can exit the fiddle yard and enter the dispatching area in any of these three stations, as well as leave the dispatching area from one of these three stations towards the fiddle yard. Figure 54 shows in overview of the ELVA infrastructure, where the arrows represent the allowed directions of travel.

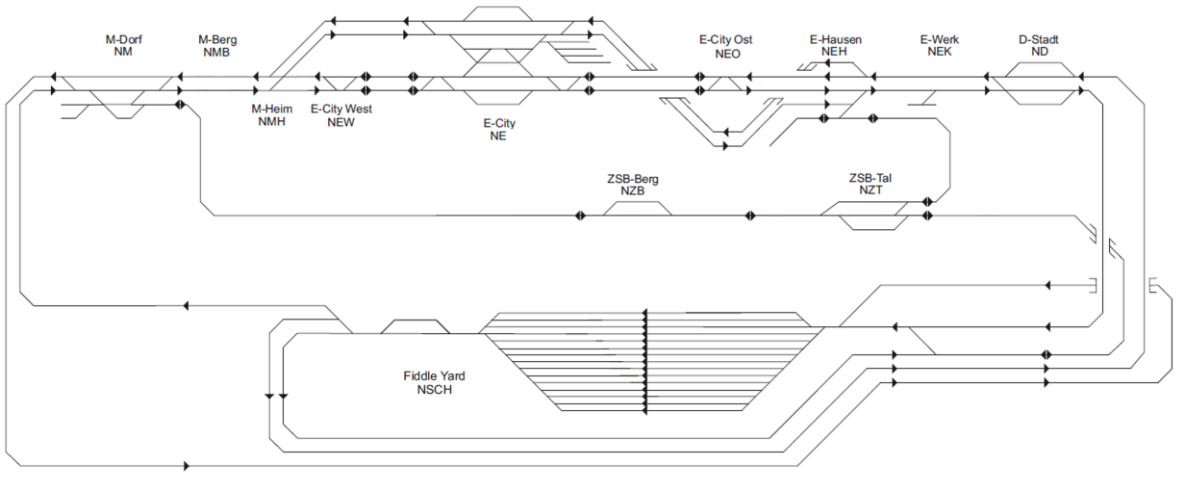


Figure 54: Overview of the ELVA infrastructure (Institute of Transport Science, RWTH Aachen University)

The scale infrastructure of the ELVA models a real network of about 100 km total line length and about 240 km total track length. Original mechanical, electro-mechanical, relay and electronic interlocking systems are installed and connected to the railway lab. Train routes can be set manually via the original interlocking systems. Note that the control software was extended such that route setting is possible automatically, without the need to use the original interlocking systems (see 5.3.3 and 6.6 for details). In this case, the intended switch positions and the new signal aspects are forwarded to the ELVA infrastructure via UDP messages and applied respectively. New positions of switches and signal aspects are transmitted from the lab to the control software via UDP messages. All rolling stock is equipped with magnets underneath each axle. Axle counters are installed on the tracks, allowing for the detection of passed axles and the direction of passing. These information are also sent to the control software via UDP messages.

The ELVA infrastructure is divided into axis counting circuits, delimited by axle counters (see 5.1 for details). Track occupancy is supervised in the field level (see 5.3.2) and track clearance is checked before train routes are set (see 5.3.3). Additional axle counters, not delimiting axis counting circuits, are installed and serve e.g. for the synchronization of train positions (see 6.4).

Operating stations with stopping positions have both entry signals, separating the open track from the station, and exit signals, positioned directly after the stopping positions, for all feasible train routes through the operating station. In the biggest operating station E-City (see Figure 54), four additional intermediate signals exist, one in each direction of the two central tracks, positioned between the two exit signals. Shunting signals are present in some operating stations, enabling operation in shunting mode.

The trains on the ELVA are controlled via their finite set of speed levels, where each speed level corresponds to a different model speed. The speed at each speed level was measured for each physical train. For this purpose, a long stretch between two axle counters was chosen. For each speed level of a train, the times required to run back and forth between these two axle counters were measured several times for both running directions. The average time was used to calculate the average speed of the vehicle at the respective speed level in cm/s. The model speed represented by the speed level was computed by converting this average speed to km/h, while also considering the scale of 1:200. For each train, a lookup table was generated, containing the corresponding model speed for each speed level. Small speeds greater than 0 km/h, but below 10 km/h were excluded for operation, as trains do not run reliably at lower speeds on the physical network infrastructure.

The speed levels are set via a digital train control system Z21 by Roco [150]. Each train has a unique address. Whenever a train shall change its speed, the train address and the intended speed level are sent to the Z21 by the train control module (see 5.3.5) via an UDP message. The Z21 then forwards the speed command to the respective vehicle. The speed level 0 always corresponds to 0 km/h and is set when a train shall stop.

8.2 Microscopic Infrastructure Model of the ELVA

The infrastructure of the scale network of the ELVA is modeled with the software tool LUKS® [3, 4], using a microscopic infrastructure model. Line segments, switches, signals and axle counters of the ELVA infrastructure and their exact positions are represented in the microscopic infrastructure model. The model is subdivided into the operating stations shown in Figure 54.

Stopping positions for passenger and freight trains are defined in the microscopic infrastructure model and are positioned 5 meters in front of exit and intermediate signals respectively. Freight trains have stopping positions in front of all exit and intermediate signals. There are two tracks within the station E-City on which no station platforms are located, hence on which stopping is not possible for passenger trains. If both passenger and freight trains are allowed to stop, two stopping positions are modeled, one for each train type.

For each operating station, the set of all feasible routes through the station is stored within LUKS®. For each route through an operating station, a unique reference point is defined. Thus, each route through an operating station passes through exactly one reference point (see also 4.2.2 for a detailed explanation of reference points).

The kilometrage of infrastructure elements along line segments can either increase or decrease. Certain positions where the counting direction of the kilometrage switches need to be defined to obtain a feasible infrastructure model, due to the specific circular setup of the ELVA infrastructure. As an example, a train can enter the station M-Dorf, coming from the fiddle yard. However, it can also leave the fiddle yard in the direction of the station ZSB-Tal and continue until it enters M-Dorf in the opposite direction. Hence, the counting direction has to be inverted between the fiddle yard and M-Dorf. In addition to the counting direction, there are also cases where the value of the kilometrage itself has to change. Reaching a certain point in the infrastructure from two different directions would otherwise cause a conflict, e.g. when reaching M-Dorf directly from the fiddle yard or via ZSB-Tal and ZSB-Berg, the traveled distances will differ. Thus, in between these stations, there needs to be a point where the value of the kilometrage jumps. The positions where the counting direction is inverted or the value of the kilometrage jumps, are therefore modeled within the LUKS® infrastructure model. Figure 55 shows an example of an infrastructure segment where both the counting direction is inverted and there is a jump in the value of the kilometrage.

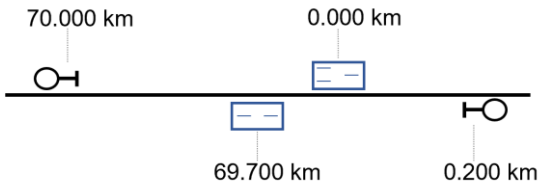


Figure 55: Inversion of counting direction and position jump in the infrastructure model

Maximum permitted line speeds are also defined in the LUKS® infrastructure model. Note that during operation, if the maximum permitted speed of a vehicle is lower than the line speed, the maximum speed of the vehicle is used. Speed limit signs, indicating a maximum permitted speed, are positioned within the infrastructure model. The maximum permitted speeds are

valid from their respective position on, for their respective direction. The diverging path of a switch may only be allowed to be traversed at a reduced maximum speed. In this case, this speed is valid from the previous main signal until the next speed limit sign. Note that from ETCS Level 2 on, the trains can receive information on maximum permitted speeds at any time [136], thus the modeling of speed limits from the last main signal on is no longer necessary. As operation under ETCS is currently not possible on the ELVA network, this case is not considered.

Maximum speeds for the diverging path of a switch are modeled in LUKS® via speed limit signs. The sign is placed in direct succession of the diverging path and its name is of the format "SpeedLimit_Sw_x" where Sw_x is the name of the switch the speed limit refers to. Figure 56 shows an example of how the maximum speeds for traversing diverging switch paths are modeled. For a train passing the main signal S and running along switches Sw_1 and Sw_2, a maximum permitted speed of 40 km/h applies from signal S on.

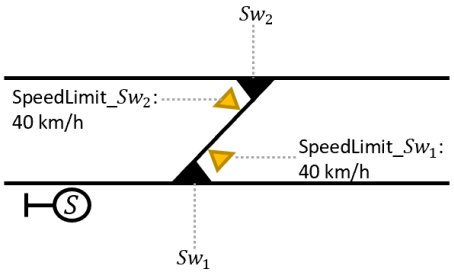


Figure 56: Modeling of speed limits for diverging switch paths

This way, when the infrastructure XML file (see Appendix A) exported from LUKS® is read by the control software, the speed limits for passing the diverging path of a switch can be uniquely assigned to the speed value of the corresponding speed limit sign.

On the ELVA network, the default line speed is set to 160 km/h. There are sections, however, with a lower or higher maximum permitted speed, depending on the train protection system in place. On the one-directional line from M-Dorf until ZSB-Tal, the maximum permitted speed is set to 100 km/h. Along the diverging route from M-Heim until E-City and from E-City until E-Hausen, the speed limit is 120 km/h in both travel directions respectively. Between E-City West and E-City Ost, a maximum speed of 250 km/h is permitted in both directions, for vehicles compatible with the LZB train protection system. Diverging paths of switches are generally allowed to be passed at a maximum speed of 40 km/h. Exceptions are the switches in ZSB-Berg and ZSB-Tal on the one-directional line segment, which can be passed at 60 km/h along their diverging paths, and the switches in the crossovers E-City West and E-City Ost, which can be traversed at 100 km/h along their diverging paths.

8.3 Timetable Generation

Feasible timetables are generated with the software tool LUKS®. The rolling stock is modeled realistically using parameters of the corresponding real train series, which are stored in a database within LUKS®. Driving dynamics and thus running times are modeled based on each vehicle's specific parameters, including its weight and power efficiency. For each vehicle,

acceleration curves were computed with LUKS® based on the running time calculation (see 3.1) integrated within LUKS®. Tables containing information on these intervals with constant acceleration (see 6.3) were exported from LUKS® and used as input to create the acceleration curves that are applied by the railway lab control software. This way, synchrony of the running time calculation within LUKS® and the railway lab control software is achieved.

Train runs can be defined by selecting train routes through operating stations. A train run can either start in the fiddle yard or at a stopping position within a station. For each pair of consecutive operating stations of a train run, a runtime margin can be defined, e.g. if a runtime margin of 5 % is defined, the running time is assumed to be 1.05 times the minimum possible running time. Additionally, connections between trains or physical continuity can be represented.

Blocking time stairways are computed for each train run, based on the infrastructure and vehicle properties and the predefined train routes. In case of occupation conflicts, LUKS® provides options for both manual and automatic conflict resolution. Once a conflict-free timetable (as shown in Figure 57) is obtained, it can be exported into an XML file format (see Appendix C) which can then be imported by the timetable module (see 6.5).

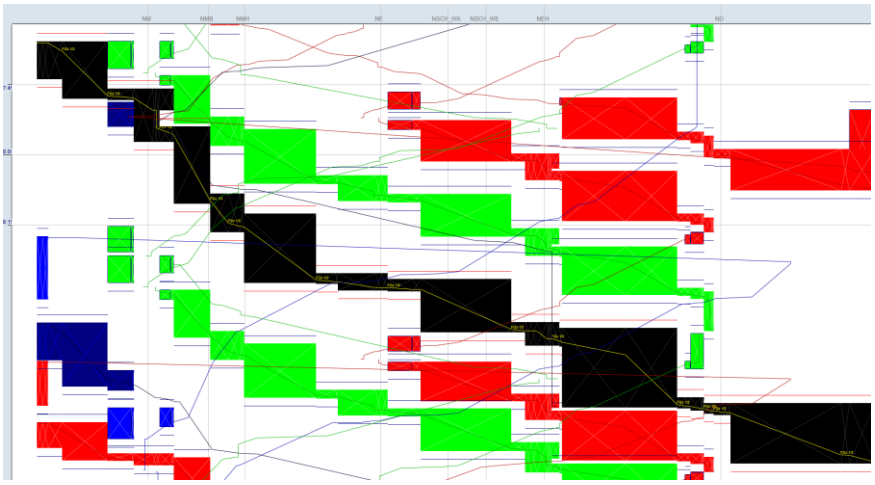


Figure 57: Blocking time stairways of a conflict-free timetable

8.4 The Dispatching System OptDis

The dispatching system OptDis was developed at the Institute of Transport Science at RWTH Aachen University during the DFG project “Optimierungsverfahren bei Disposition und Fahrplanerstellung im Eisenbahnwesen” [1, 2] and is integrated in the software LUKS®.

OptDis uses a conflict-free timetable in LUKS® as its initial dispatching timetable. The timetable is visualized by the blocking time stairways of the trains. Messages from the railway lab are processed by OptDis. For details on the different messages, see also 4.2.1. Received messages on the current state of operation are incorporated into the current dispatching timetable by dynamically adjusting the blocking time stairways model.

When a train reaches a signal, the blocking times are adjusted to meet the received arrival time, i.e. blocks are bent or moved, or dwell times are adjusted. When receiving an information about a set train route, OptDis freezes the train route to prevent future changes

of this route during the dispatching process. Upon receiving a “synchronizeTime” message, OptDis synchronizes its internal system time with the received time.

If a train is delayed due to a primary delay, OptDis will not receive the expected message on the next reached signal on time. As long as OptDis is waiting for a message of a delayed train, the blocking time stairway is adjusted in constant time intervals by either bending the current blocking time, if the train is currently not stopping, or by increasing the current dwell time of the train, in case the train is currently stopping. Thus, information on primary delays are only implicitly provided to OptDis through missing expected messages. A more detailed discussion on how OptDis handles missing expected messages or messages that diverge from the expected information, is given below.

As OptDis receives the “requestDispatching” message, conflict resolution and dispatching are performed. Occupancy conflicts between the blocking time stairways are automatically detected by identifying overlapping blocks, i.e. segments of the infrastructure would be occupied by more than one vehicle at the same time. To find a new optimal dispatching timetable, the problem is converted into a mixed-integer linear programming model. The constraints within the model ensure that occupation conflicts are resolved and all chosen train routes are feasible, i.e. they have to form a connected, feasible train run. If minimum dwell times are defined, they have to be followed. Additionally, the parts of train trajectories lying in the past, including received train positions and set train routes, cannot be changed anymore. Furthermore, the constraints ensure that the arrival and departure times are realizable, i.e. speed limits, vehicle dynamics or train protection systems are taken into account. Realistic running times are obtained by the running time calculation integrated within LUKS® and used by OptDis during the optimization process. Additionally, modeled connections between train runs are maintained during optimization. E.g., connection constraints can be necessary to model physical availability of vehicles operated for several distinct train runs, or to ensure the transfer of passengers between different trains in a station.

A certain time interval can be defined as dispatching horizon. Thus, the operational situation is only evaluated up to the predefined time horizon into the future. As trajectories of train runs cannot be predicted precisely in the far future, the definition of such a time horizon does not decrease the quality of the computed dispatching measures significantly. Especially for large dispatching areas, computational complexity can be immensely reduced by restricting the optimization to a certain dispatching horizon.

The objective function primarily minimizes overall weighted delays among all train runs. Secondly, it minimizes the number of dispatching measures, since during real operation these measures have to be forwarded to the responsible person, if the operation is not completely automated. In particular the reversal or change of previous dispatching measures shall be kept at a minimum. Lastly, the maximization of buffer times is prioritized to obtain a higher robustness of the new dispatching timetable.

The resulting mixed-integer linear program is passed on to the commercial solver software Gurobi [151]. Gurobi searches for an optimal solution with respect to the objective function, while meeting all defined constraints, yielding a new optimized dispatching timetable. The decision variables of the optimal solution are then adopted in the dynamic dispatching

timetable maintained by OptDis. After completion, all changes in arrival times, departure times or train routes are sent to the railway lab via the message formats defined for the shared communication interface (see 4.2.2).

Note that during dispatching, an infrastructure database is used which is restricted to only the dispatching area. In particular, it does not include the fiddle yard. For timetable generation (see 8.3), however, a database containing the complete infrastructure of the railway lab is utilized.

Missing or divergent messages

OptDis expects messages on reached main signals at the time the train is predicted to reach the respective signal. Messages on claimed train routes are expected at the time route setting is predicted to take place according to the blocking time stairways model of the currently active dispatching timetable (see 3.4). In constant time steps, OptDis checks if any expected messages were not yet received. In case of missing messages, OptDis makes assumptions on the possible reasons for the missing messages and internally adjusts the blocking time stairways of the current dispatching timetable to meet these assumptions.

A missing message on a reached main signal S is processed by bending the trajectory of the train run in the block section preceding S in constant timesteps, such that the new arrival time at S coincides with the current system time. The remaining part of the blocking time stairway is shifted accordingly. This process is repeated until either the message is received or another message on a reached main signal is received, indicating that either the message on the reaching of S was lost or the train ran along a different route than scheduled by the dispatching system and the message on the claimed train route was lost.

For messages on reached main signals diverging from their prediction, five cases are distinguished.

1. **A main signal, which does not lie on a train's scheduled route, is reached**

   This case can occur if a different train route than the one according to the current dispatching timetable was set by the interlocking system and the message on the claimed train route was lost. OptDis adjusts the train route such that it passes through the new main signal whose reaching was transmitted.

2. **The arrival time at an exit signal after a scheduled or operating stop in a station differs from the predicted arrival time**

   This case can occur if a train stops for a longer or shorter period of time. Longer stops can be caused by unexpected disturbances. OptDis adjusts the dwell time at the last scheduled or operating stop before the exit signal accordingly to meet the given arrival time at the exit signal.

3. **The arrival time within the dispatching area differs from its prediction**

   If the running time of a train until it enters the dispatching area is shorter or longer than predicted, such a deviation can occur. OptDis processes this information by moving the complete train's blocking time stairway on the time-axis to meet the new entry time.

4. **A train arrives late at a main signal during operation**

   This scenario can occur if e.g. a train route was set too late, causing the train to brake unnecessarily or the train ran slower than predicted. OptDis adds a new braking process into the train's speed profile such that the predicted arrival time coincides with the transmitted one. Note that this case does not refer to exit signals after a scheduled train stop.

5. **A train arrives early at a main signal during operation**

   This case can occur if a train used its running time reserves without being required to, e.g. during manual operation by a train driver. Upon receiving such an information, OptDis reduces the running time reserves in the previous block section which were apparently not used by the train, shortening the block occupation time of the previous block, such that the new arrival time is met. Again, this case does not refer to exit signals after scheduled or operating stops.

Normally, upon receiving a message on a claimed train route, OptDis internally freezes the route such that future changes to the route during the conflict resolution process are disabled. As long as a message on a claimed train route does not arrive, OptDis does not freeze the route, hence the route can still be changed as a dispatching measure. If OptDis receives a train route that differs from the currently expected one, but coincides with a future train route for the train, OptDis assumes that previous messages on claimed train routes were lost. If OptDis receives a train route that does not coincide with any currently scheduled train route, OptDis assumes that the interlocking system set a different route for the train and the train routes in the dispatching timetable are updated to match the newly received claimed route. All train routes up to the newly received one are frozen to disable route changes during the dispatching process.

Manual dispatching with OptDis

Beside the automatic conflict detection and resolution, manual dispatching is also possible with OptDis. In this case, the messages received during operation from the railway lab are processed and OptDis can be used for visualization of the currently predicted time-distance graphs and blocking time stairways. A user interface is provided to manually submit bending, dwell time adjustments, including the adding or cancellation of stops and the changes of train routes through operating stations. OptDis always ensures the feasibility of manual dispatching measures, e.g. train runs can only be bent such that maximum permitted speeds are still followed, minimum dwell times are considered, additional stops can only be used if an appropriate stopping position is available for the train and routes can only be changed to feasible routes through the respective operating station.

For bending, OptDis provides the possibility to manually drag the corresponding block of the blocking time stairway (see Figure 58). Bending can also be entered by the human dispatcher by providing the percentage by which the running time between operating stations shall be increased or decreased (see Figure 60 (d)) or by providing either the amount of time to be added through bending, the total running time for the designated section or the arrival time

at the reference point of the operating station until which bending shall be applied (see Figure 60 (e)).

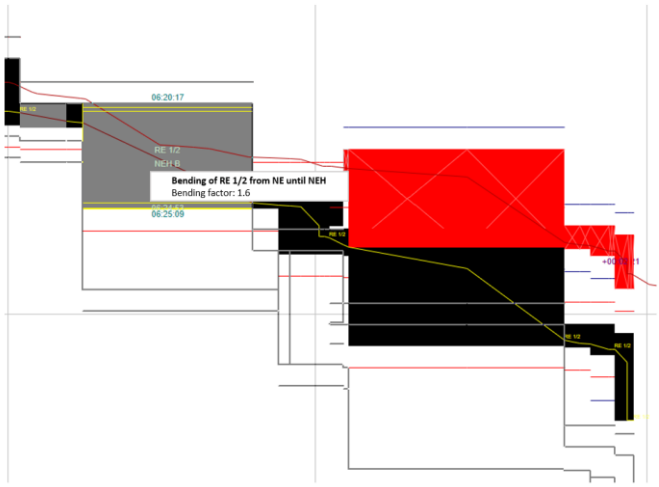


Figure 58: Manual bending [Screenshot from LUKS® (edited)]

Dwell time adjustments, including the omitting of a stop, can also be entered by dragging the corresponding block (see Figure 59) or by manually entering the new dwell times via an input box (see Figure 60 (a)). It is also possible to add additional stops via this input box. The departure time in an operating station can be provided via an input box (see Figure 60 (b)) and optionally, the departure time in some subsequent operating station can be provided (see Figure 60 (c)). Note that there are several possible ways of entering bending or dwell time adjustments which the manual dispatcher can choose from.

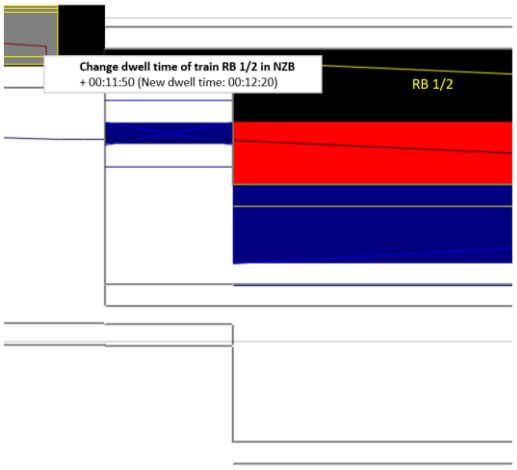


Figure 59: Manual dwell time change [Screenshot from LUKS® (edited)]

For route changes, OptDis provides a list with all feasible train routes for each operating station of a train run, from which the dispatcher can choose a new route via a drop-down menu (see Figure 60 (f)). If a route change requires route changes in multiple consecutive operating stations (see Figure 13), neighboring train routes are automatically adjusted by OptDis to support the dispatcher and to ensure connectivity and feasibility of the train routes.

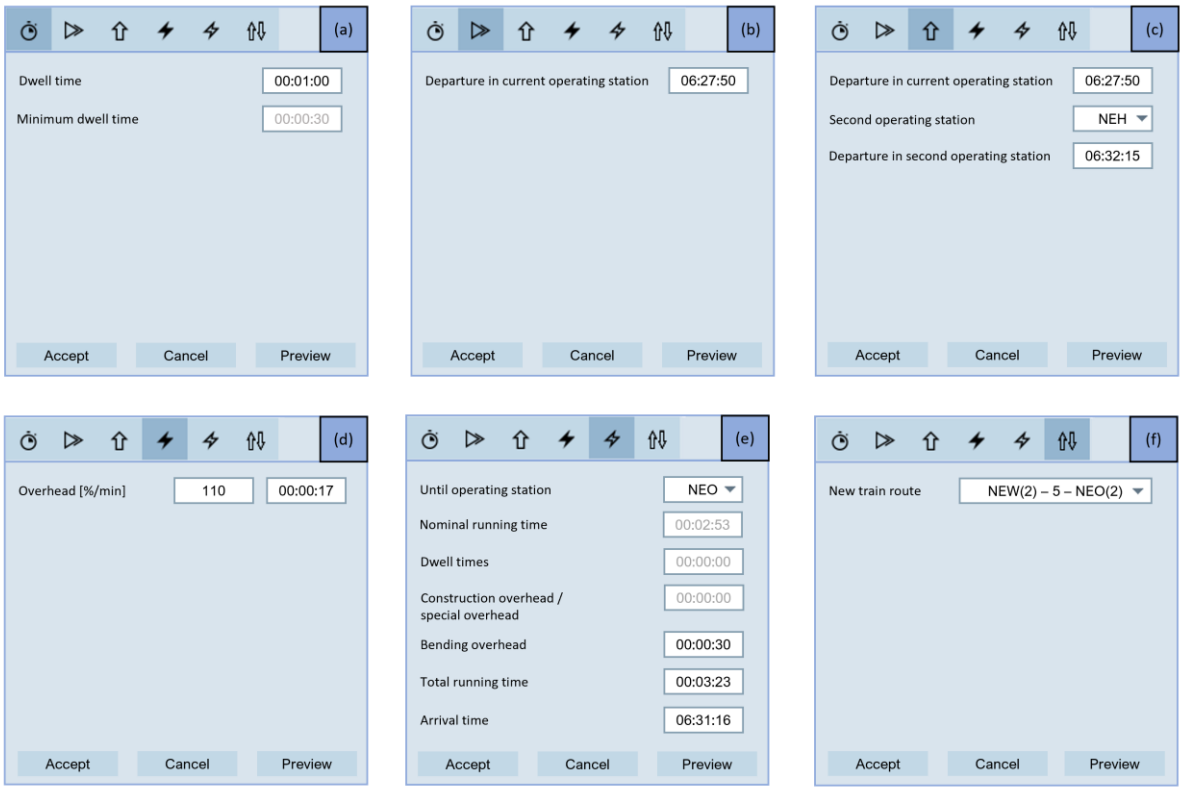


Figure 60: Input boxes for manual dispatching [Screenshots from LUKS® (edited)]

When manual dispatching measures are entered by the human dispatcher, OptDis adjusts the predicted blocking time stairways accordingly and the resulting changes are visualized for the dispatcher. The resulting messages are only sent to the railway lab once the dispatcher manually presses a button to submit the changes. The dispatcher can also choose to discard the changes. Hence, OptDis provides the opportunity for a human dispatcher to try out and visualize different conflict resolution approaches first before they are submitted and forwarded to the railway lab.

8.5 Possibilities and Limitations of the Testing Environment

In chapter 4, a communication interface between a dispatching system and a railway lab control software was defined. The extended railway lab control software required for automatic operation under real-time dispatching was described in chapter 5 and chapter 6. These software systems serve as testing environment and were tested on the Railway Signalling Lab ELVA at RWTH Aachen University (see 8.1) with the dispatching system OptDis (see 8.4). This section gives an overview of the possibilities, as well as limitations which the developed systems and their application on the ELVA with OptDis provide.

Possibilities

The implemented railway lab control software and the communication interface defined for communication between a railway lab and a dispatching system serve as realistic testing environment for dispatching algorithms.

The general definition of the communication interface between the software and hardware components allows for a versatile use. If another dispatching system than OptDis or another railway lab than the ELVA shall be used, merely the processing of received information and the provision of information to other systems has to be adjusted. E.g., in order for another dispatching system to be used, the system needs to be adjusted to process the received messages in the specific predefined formats and send messages to the railway lab following the specific message formats. For another railway lab to be utilized, the communication between the control software and the physical railway lab, which is used to transmit states of switches and signal aspects in both directions, as well as passed axle counters from the infrastructure to the software and the setting of speed levels from the software to the physical trains, has to be made compatible. Thus, for the dispatching system or the railway lab to be exchanged, only the communication interfaces to the control software have to be adjusted, in order to be compatible. All internally implemented algorithms remain the same.

The ELVA network resembles a realistic railway network of about 100 km line length and eleven operating stations within the dispatching area. The fiddle yard, which is connected to the dispatching area, provides the possibility for train runs to enter and exit the dispatching area with predefined delays. All physical vehicles provide an average of 60 different speed levels, thus enabling accurate approximations of realistic speed curves via the available set of discrete speeds. Train routes can either be set automatically by the automatic route setting (see 6.6) or manually via the connected original interlocking systems.

OptDis provides the visualization of the current operational situation and allows for both automatic conflict detection and resolution within predefined time intervals, as well as the processing of manually submitted dispatching measures. When a human dispatcher selects a certain train, the visualization is limited to only the operating stations relevant to the specific train and only the parts of other trains' blocking time stairways which share mutual segments of the infrastructure are displayed. This way, OptDis provides a simple representation of the current operational situation to human operators and prevents potential confusion caused by too many overlapping blocking time stairways, e.g. when trains are running on different tracks of the same operating station. Dispatching measures available in OptDis include the bending of train runs, changes of dwell times, the adding or cancellation of stops and the change of train routes through operating stations. Train routes can also be changed via multiple consecutive operating stations, enabling the running on the opposite track. Before conflict resolutions are submitted by the human dispatcher, the predicted effects of the dispatching measures are visualized by OptDis, thus different approaches can be tested virtually before they are finally submitted and sent to the railway lab.

Limitations

There are, however, also limitations to the developed testing environment. Evaluations on the ELVA are restricted to its network size and the number of available physical vehicles. Furthermore, for each vehicle, one single speed curve is defined both for acceleration and deceleration respectively. In the current implementation, trains on the ELVA are operated under a fixed block signaling system and only the signaling systems PZB and LZB are applied.

The use of moving block during operation is therefore also not possible yet. This would additionally require constant localization of trains, which is currently only done when axle counters are passed that are sometimes several kilometers apart. Also, special driving strategies, such as coasting, emergency braking or different braking curves under ETCS are currently not modeled.

Although the testing environment provides realistic conditions, operation in a railway lab can never model real train operation entirely accurate. As trains are operated automatically via the train control module (see 5.3.5 and 6.4), imprecisions in driving behavior and deviations from the ideal speed curves, occurring during manual train control by a train driver, are not considered during evaluations. Thus, trains are assumed to always accelerate and decelerate precisely as modeled. Starting and end points of speed curves might also differ when trains are operated manually. Furthermore, during real operation, the acceptance of dispatching measures by train drivers is not guaranteed. If, for instance, a speed of 61 km/h is given, the train driver might instead run at 60 km/h, as speed restrictions are normally provided as multiples of 5 km/h.

Moreover, energy consumption cannot be realistically evaluated with the model vehicles. Energy consumption of a train on the ELVA is approximately constant, regardless of the speed level it currently runs at. In particular, the model trains do not allow for conclusions about the energy consumption during acceleration or deceleration. Thus, to evaluate energy consumption, an external model has to be applied, where the speed curves during operation are used as input.

OptDis provides a huge range of dispatching measures. However, the current underlying mathematical optimization model does not yet incorporate the possibility to change the order in which operating stations are passed or to reroute trains through alternative operating stations. Thus, dispatching is limited to the fixed order of operating stations a train run is scheduled to pass. Cancellation, early termination of a train run or making a train change its direction prior to reaching its destination, are therefore also not possible. Additionally, passenger connections between trains are not considered in the optimization model implemented within OptDis.

Conflict resolution within OptDis is performed by generating a mixed-integer linear programming model (MILP) and passing it to the solver Gurobi. The optimal values for the decision variables, e.g. arrival and departure times or train routes, are determined by the solver and returned to OptDis. As Gurobi holistically solves the MILP, it does not return the specific motivation behind dispatching measures that a human dispatcher would apply, e.g. “let train 1 stop at track 1 instead of track 2 and increase its dwell time, so that train 2 can overtake train 1”. Also, several conflicts might be resolved during one single optimization process. Therefore, the resulting dispatching measures, which are potentially complex and involve several trains, cannot be separated or attributed to a specific conflict which they solve.

9 Evaluation of Operational Test Cases and Results

This chapter presents the case studies defined to evaluate the developed software systems and methods.

Section 9.1 describes how the developed software environment and algorithms described in chapters 4 to 6 were tested on the Railway Signalling Lab of RWTH Aachen University (ELVA) with the dispatching system OptDis, which is integrated within LUKS® (see section 8.4). Several timetable concepts and operational test cases were defined for the case study and the performance of the algorithms was evaluated. To assess the dispatching algorithms within OptDis on the ELVA, the results were compared to manual dispatching by an experienced human dispatcher. For manual dispatching, OptDis was connected to the ELVA control software and messages were exchanged, as described in section 4.2, however, the message "requestDispatching" was not sent. During manual dispatching, OptDis did not suggest any conflict resolutions, however it supported the human dispatcher by visualizing the current operating situation, predicting future trajectories of trains, displaying occupation conflicts and providing an interface for the dispatcher to manually solve conflicts by bending train runs, adjusting dwell times or changing train routes. Once the human dispatcher finished their conflict resolution and committed the changes, the required messages (see 4.2.2) were generated by OptDis and sent to the ELVA control software. The detailed setup of the test cases is described in section 9.1.1. Three timetable concepts are introduced and the generation of the delay data is described. The results of the case study are then presented and discussed in section 9.1.2.

The energy-saving driving method proposed in chapter 7, which was integrated within the railway lab control software, was also tested on the ELVA network in a case study. Section 9.2.1 describes the test case used for this case study. As energy consumption cannot directly be derived from the operation on the railway lab, a model for the calculation of the energy consumption based on the speed profiles of the trains is given in section 9.2.2. In section 9.2.3, the obtained results are presented and discussed.

9.1 Evaluation of Dispatching Algorithms in a Railway Lab

This section presents the case study conducted to validate the developed software systems on the Railway Signalling Lab ELVA of RWTH Aachen University and to evaluate the performance of the dispatching system OptDis compared to an experienced human dispatcher. In section 9.1.1, the detailed setup of the case study is described. Three timetable concepts, which were defined for the evaluations, are presented. Additionally, four manually constructed conflict situations are introduced. In section 9.1.2, the results of the case study are presented and discussed.

9.1.1 Setup of the Case Study

Definition of the test cases and general setup

To obtain reliable results, evaluations were performed for a total of three timetable concepts under different investigation periods and traffic volumes. Detailed information on all

timetable concepts can be found in Appendix E. This section describes the three timetable concepts constructed for the test cases, the generation of delay data, as well as general settings and assumptions that were made. For all test cases, dispatching was performed both through OptDis and by an experienced human dispatcher. By using predefined delay data, operational situations could be precisely reconstructed, enabling the comparison of the results obtained by the two strategies.

Both the ELVA control software and OptDis were executed on the same computer, which is connected to the physical railway lab. Apart from the information exchanged solely through the defined communication interface (see chapter 4), these two systems run completely detached. Thus, parallel computations within both the control software and OptDis do not affect each other and are therefore feasible.

Each physical vehicle operates one single train line, i.e. all train runs resulting from the physical vehicle reentering the dispatching area at certain time intervals or from turnarounds in an operating station, belong to the same train line. Each train run of a train line is assigned a train number with a different ending, such that train numbers uniquely identify train runs. The arrival time of each train run at the reference points of operating stations were recorded during operation to enable the future evaluation of delays.

In total, 40 evaluations were performed, using 20 different test cases. Operation was performed automatically for all test cases, i.e. all trains were automatically controlled by the train control software (see 5.3.5 and 6.4) and train routes were set automatically by the automatic route setting (see 6.6). On the one hand, all test cases were operated under automatic dispatching through OptDis. On the other hand, all 20 test cases were repeated under automatic operation, but with manual conflict resolution through an experienced human dispatcher during the dispatching process. During manual dispatching, LUKS® with OptDis was still connected to the ELVA, while providing a visualization of the operational situation and displaying occurring occupation conflicts. An example is shown in Figure 61.

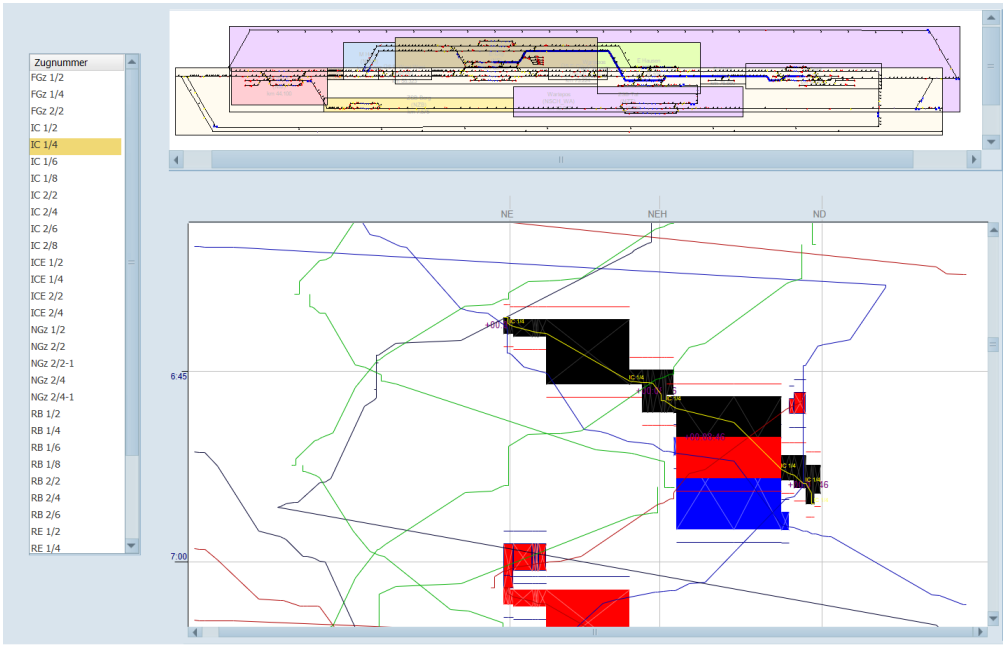


Figure 61: Visualization of the operational situation and occupation conflicts during manual dispatching [Screenshot from LUKS® (edited)]

Additionally, the human dispatcher was provided with a real-time overview of the ELVA infrastructure, showing all current track occupations with the respective train numbers in red, as well as all set train routes in green. Signals are visualized as triangles and their aspect is indicated by their color. An example is shown in Figure 62.

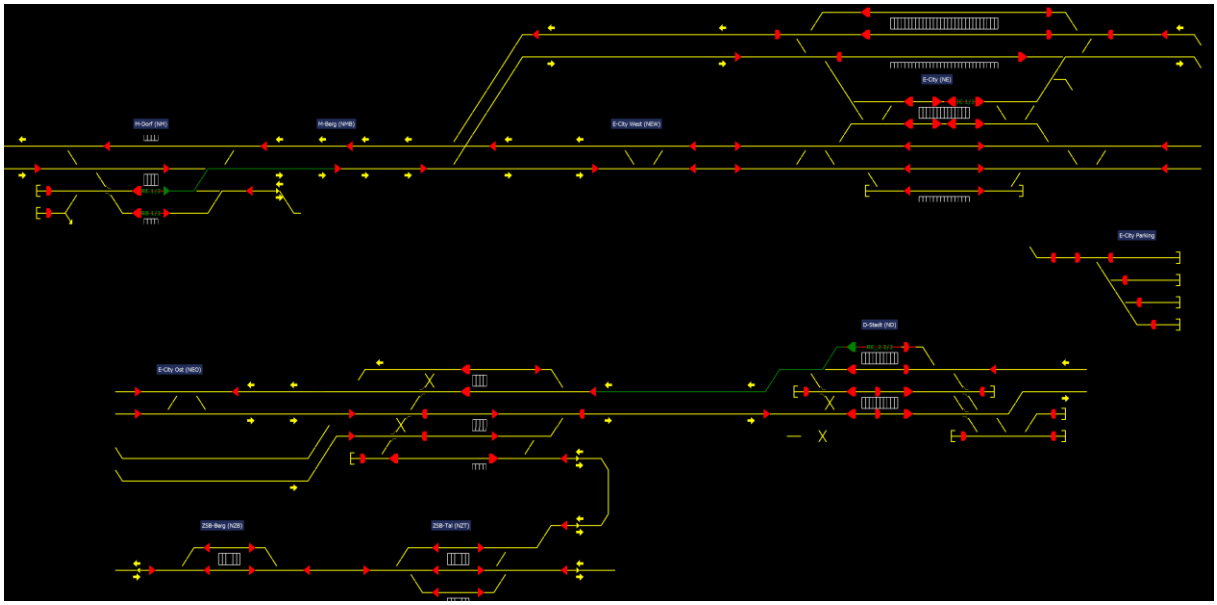


Figure 62: Overview of the ELVA infrastructure during operation

For all operating stations, the dispatcher could optionally open a more detailed view, as shown in Figure 63.

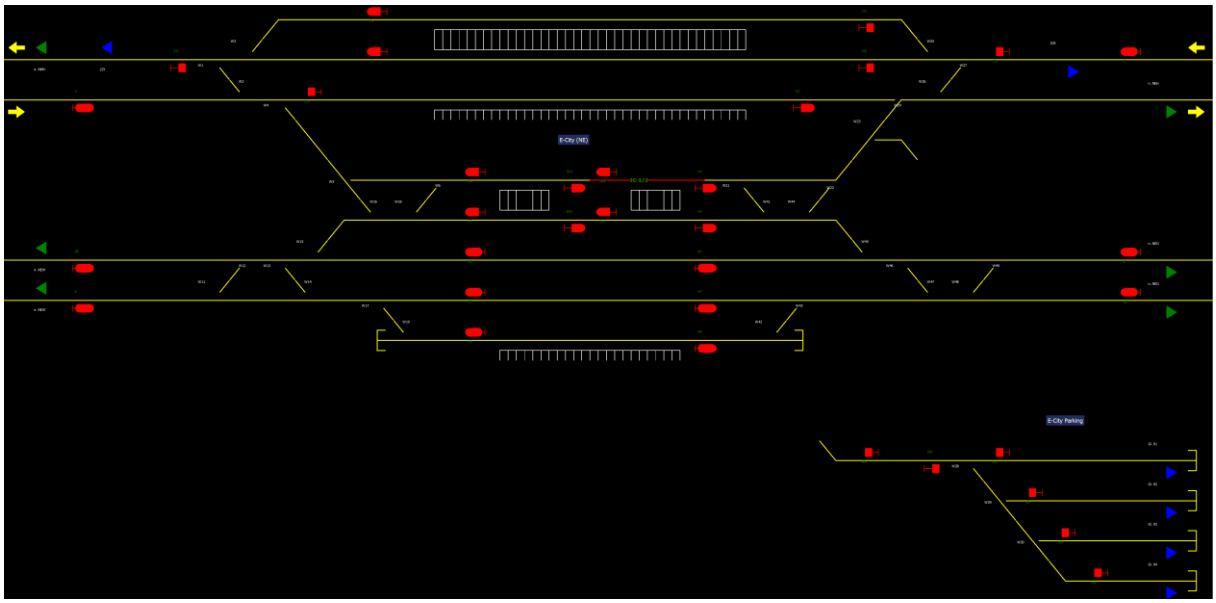
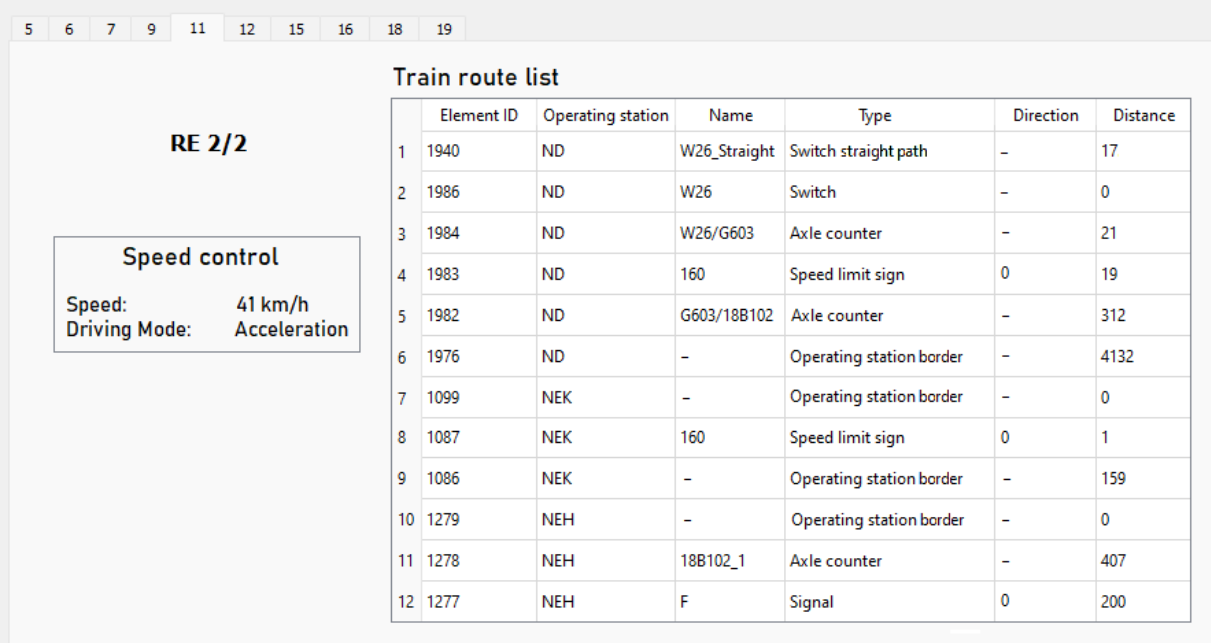


Figure 63: Detailed overview of the operating station E-City

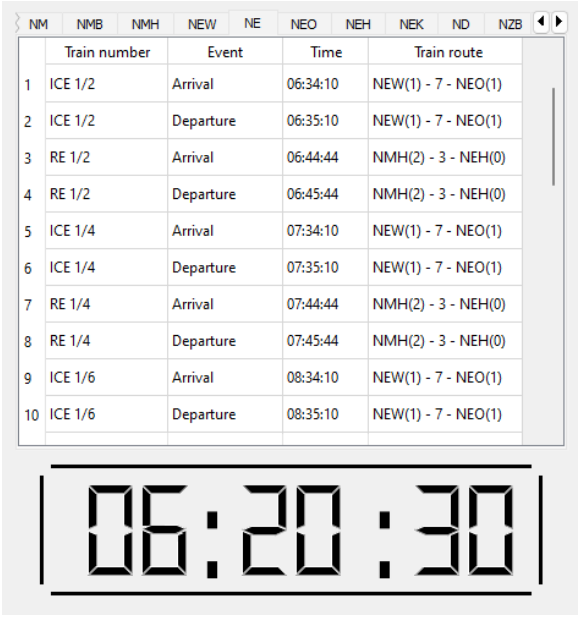
Furthermore, information on current train positions and speeds were provided by the train control module and could be viewed during operation. An example is shown in Figure 64. In the upper left corner, the dispatcher could switch between all physical vehicles active in the current timetable concept, referenced by their physical address. The current speed of the train in km/h, its driving mode (standstill, acceleration, deceleration or cruising) and the train route list (see 5.3.5) were displayed.



	Element ID	Operating station	Name	Type	Direction	Distance
1	1940	ND	W26_Straight	Switch straight path	–	17
2	1986	ND	W26	Switch	–	0
3	1984	ND	W26/G603	Axle counter	–	21
4	1983	ND	160	Speed limit sign	0	19
5	1982	ND	G603/18B102	Axle counter	–	312
6	1976	ND	–	Operating station border	–	4132
7	1099	NEK	–	Operating station border	–	0
8	1087	NEK	160	Speed limit sign	0	1
9	1086	NEK	–	Operating station border	–	159
10	1279	NEH	–	Operating station border	–	0
11	1278	NEH	18B102_1	Axle counter	–	407
12	1277	NEH	F	Signal	0	200

Figure 64: Information on the current train status during operation

Lastly, an overview of the dynamic reference timetable (see 6.5) and the current system time was provided to the dispatcher, as shown in Figure 65. The dispatcher could switch between operating stations and all scheduled events in the chosen operating station could be viewed. The provided information for each event included the train number, the type of event (arrival, departure, passing through, start or end), the time of the event and the currently scheduled train route through the operating station. In case of dispatching measures, the timetable overview was updated accordingly.



	Train number	Event	Time	Train route
1	ICE 1/2	Arrival	06:34:10	NEW(1) - 7 - NEO(1)
2	ICE 1/2	Departure	06:35:10	NEW(1) - 7 - NEO(1)
3	RE 1/2	Arrival	06:44:44	NMH(2) - 3 - NEH(0)
4	RE 1/2	Departure	06:45:44	NMH(2) - 3 - NEH(0)
5	ICE 1/4	Arrival	07:34:10	NEW(1) - 7 - NEO(1)
6	ICE 1/4	Departure	07:35:10	NEW(1) - 7 - NEO(1)
7	RE 1/4	Arrival	07:44:44	NMH(2) - 3 - NEH(0)
8	RE 1/4	Departure	07:45:44	NMH(2) - 3 - NEH(0)
9	ICE 1/6	Arrival	08:34:10	NEW(1) - 7 - NEO(1)
10	ICE 1/6	Departure	08:35:10	NEW(1) - 7 - NEO(1)

Figure 65: Timetable overview during operation

Conflict resolution was performed manually within OptDis (see 8.4) and committed dispatching measures were then sent to the ELVA using the communication interface (see 4.2, and 6.2), similar to the case under automatic dispatching. In total, six monitors were provided to the dispatcher, such that all windows could be displayed simultaneously.

Three timetable concepts were defined, which are described below. For both the first and the second timetable concept, five test cases with different delay data were created and operation was performed both under automatic and manual dispatching. For the third timetable concept, ten different test cases were created and again, operation was performed under both automatic and manual dispatching.

Low traffic volume and investigation period of four hours (timetable concept 1)

For the first timetable concept defined for the evaluations, a low traffic volume and an investigation period of four hours are applied. In total, nine different vehicles are used for this timetable concept: two high-speed passenger trains, one long-distance passenger train, one regional express passenger train, one local regional passenger train, two long-distance freight trains and two regional freight trains.

Table 15 gives an overview of the different train lines used in timetable concept 1. All trains that do not enter the dispatching area will run back and forth between stations within the dispatching area. The frequency of trains which turn around within the dispatching area refers to the frequency of a complete circulation, i.e. until a train run in the same direction is started again. In total, an average number of nine train runs is operated in a time frame of one hour.

Table 15: Overview of the train lines in timetable concept 1

Train line	Category	Maximum train speed [km/h]	Train length [m]	Highest compatible train protection system	Entering the dispatching area	Frequency [min]
ICE 1	high-speed long-distance passenger train	300	226	LZB	yes	120
ICE 2	high-speed long-distance passenger train	300	146	LZB	yes	120
IC	long-distance passenger train	160	220	PZB	no	60
RE	regional express passenger train	160	220	PZB	no	60
RB	local regional passenger train	100	154	PZB	no	60
LDF 1	long-distance freight train	100	196	PZB	yes	120
LDF 2	long-distance freight train	100	242	PZB	yes	120
RF 1	regional freight train	100	194	PZB	yes	120
RF 2	regional freight train	100	182	PZB	yes	120

Figure 66 shows the scheduled train routes through the dispatching area for the ICE trains and the stations where the trains are scheduled to stop. The train line ICE 1 enters the dispatching area in M-Dorf and exits the dispatching area in D-Stadt, whereas the train line ICE 2 runs in the opposite direction, entering the dispatching area in D-Stadt and exiting it in M-Dorf. Both train lines pass through the crossovers E-City West and E-City Ost, traversing the part of the infrastructure which permits high-speed operation for trains that are compatible with the LZB train protection system.

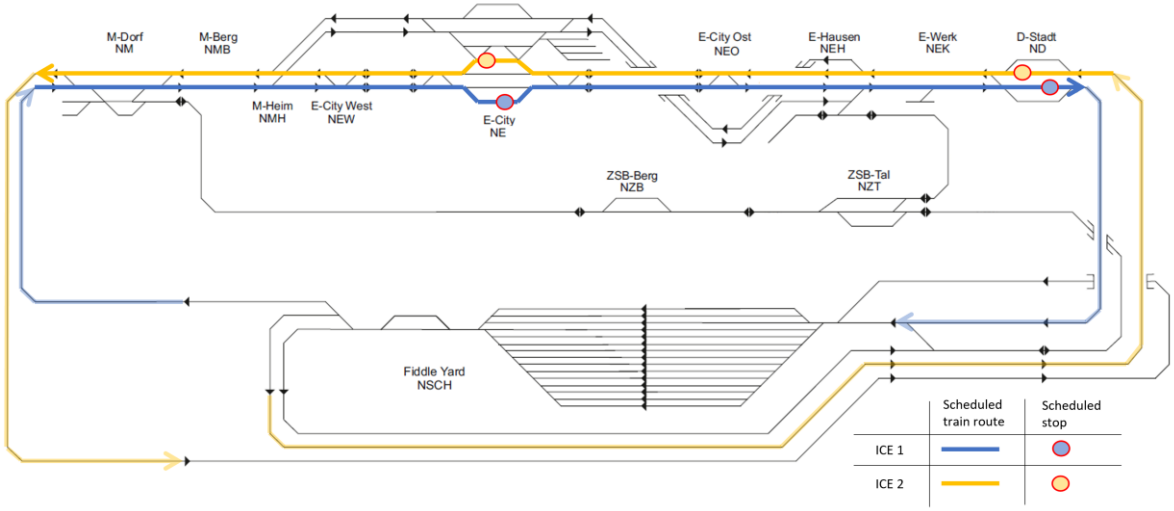


Figure 66: Scheduled train routes of the ICE train lines

In Figure 67, the scheduled train routes for the IC train line in both directions are shown. It starts in the station E-City and runs until the station D-Stadt. In D-Stadt it changes its direction and runs back until E-City where it terminates. Train stops are scheduled in the station E-Hausen and the terminal station of each direction.

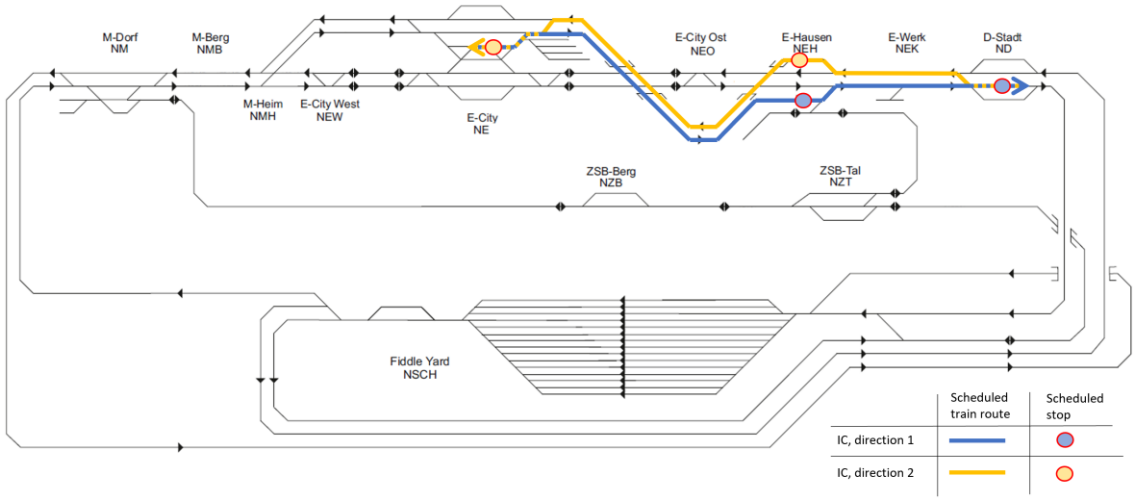


Figure 67: Scheduled train routes of the IC train line

Figure 68 gives an overview of the scheduled train routes for both directions of the RE train line which starts in M-Dorf and runs via the stations E-City and E-Hausen until D-Stadt. There it changes its driving direction and runs back until it terminates in M-Dorf. It is scheduled to stop in E-City and E-Hausen along its route and at the terminal stations of each direction.

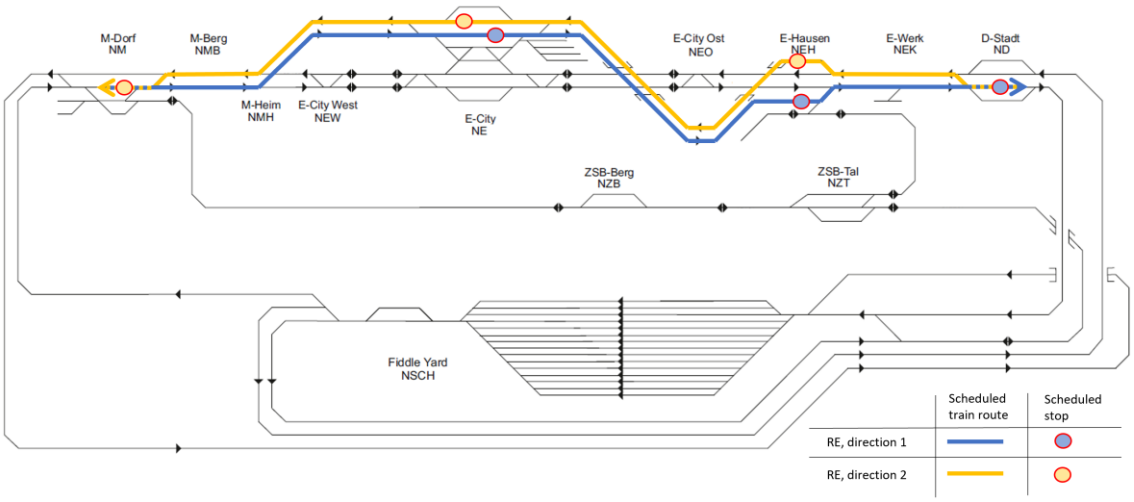


Figure 68: Scheduled train routes of the RE train line

The scheduled train routes for both directions of the RB line are presented in Figure 69. It is scheduled to run along the single-track line and starts in the station M-Dorf, passes through the station ZSB-Berg and changes its direction after stopping in the station ZSB-Tal, running back until M-Dorf. Stops are scheduled in ZSB-Berg and in the terminal stations of each direction.

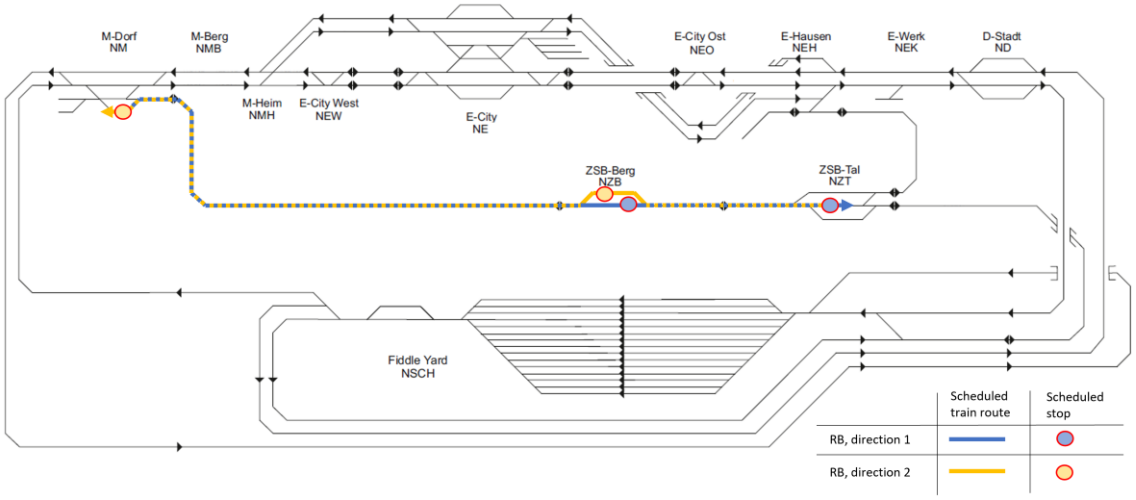


Figure 69: Scheduled train routes of the RB train line

Figure 70 shows the scheduled train routes of the two LDF lines. Both lines enter and exit the dispatching area in different directions. The line LDF 1 enters the dispatching area in the station M-Dorf and is scheduled to run until the station D-Stadt from where it exits the dispatching area again, without any scheduled or operating stop in between. The line LDF 2 enters the dispatching area in the station D-Stadt, runs until M-Dorf and then exits the dispatching area again, also without any scheduled or operating stop along its route.

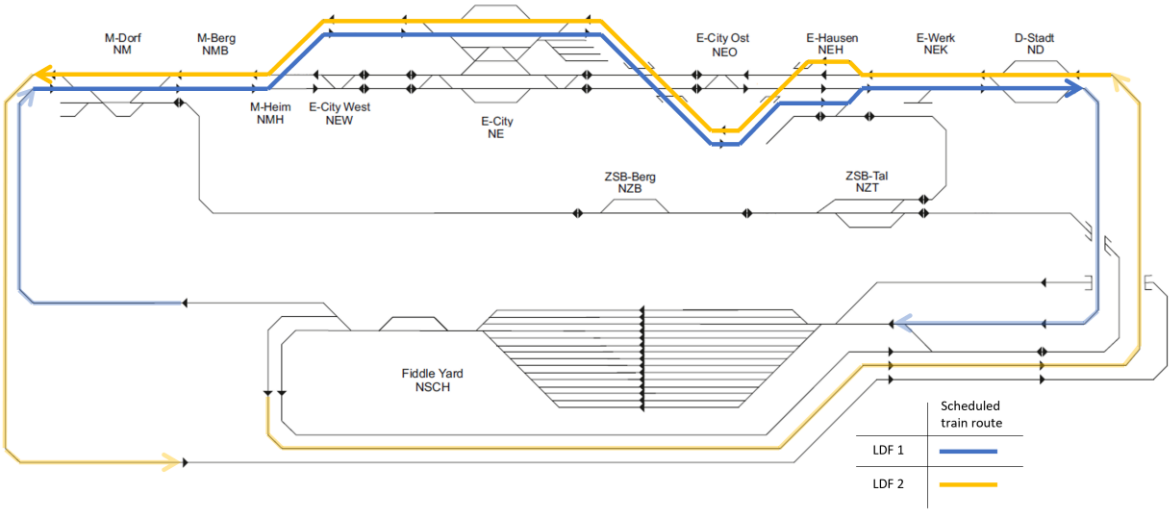


Figure 70: Scheduled train routes of the LDF train lines

The last train lines used in this timetable concept are the RF lines, shown in Figure 71, which both enter and exit the dispatching area and traverse the single-track line in opposite directions. The line RF 1 enters the dispatching area in the station M-Dorf, traverses the station ZSB-Berg where it has an operating stop and exits the dispatching area in the station ZSB-Tal. The line RF 2 enters the dispatching area in ZSB-Tal, traverses ZSB-Berg where it has an operating stop and exits the dispatching area again in M-Dorf.

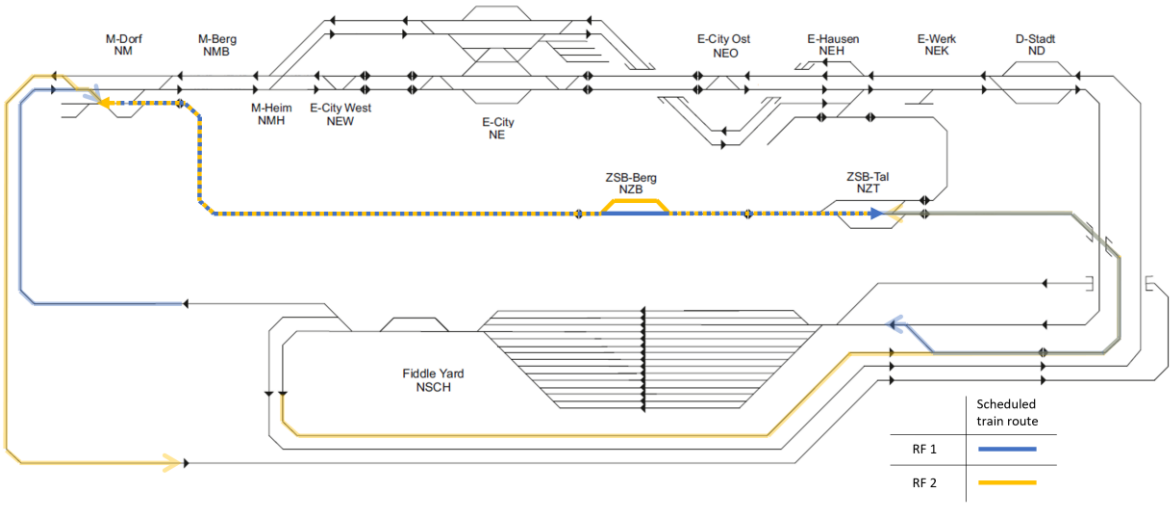


Figure 71: Scheduled train routes of the RF train lines

To be able to include three hours of regular operation in the investigation period, i.e. all train runs which are scheduled within these three hours shall run within this time interval, it is necessary for some train runs to already start before or to terminate after this three-hour time frame. Thus, the complete investigation period for this timetable concept is determined to be four hours, including half an hour of operation before and after the three-hour time frame with regular operation respectively.

High traffic volume and investigation period of four hours (timetable concept 2)

The second timetable concept incorporates a high traffic volume and an investigation period of four hours. Twelve different vehicles are used for this timetable concept: two high-speed passenger trains, one long-distance passenger train, two regional express passenger trains, one local regional passenger train, three long-distance freight trains and three regional freight trains.

In Table 16, the train lines used in timetable concept 2 are summarized. As in timetable concept 1, the frequency of trains turning around within the dispatching area refers to complete circulations. For timetable concept 2, the three train lines RE_2, LDF_2 and RF_2 were added and the number of operated ICE, IC and RB lines was increased to operate more trains simultaneously. In total, an average number of 18 train runs is operated within one hour, which is twice as many as in timetable concept 1.

Table 16: Overview of the train lines in timetable concept 2

Train line	Category	Maximum train speed [km/h]	Train length [m]	Highest compatible train protection system	Entering the dispatching area	Frequency [min]
ICE 1	high-speed long-distance passenger train	300	226	LZB	yes	60
ICE 2	high-speed long-distance passenger train	300	146	LZB	yes	60
IC	long-distance passenger train	160	220	PZB	no	30
RE	regional express passenger train	160	220	PZB	no	60
RE_2	regional express passenger train	160	220	PZB	no	60
RB	local regional passenger train	100	154	PZB	no	30
LDF 1	long-distance freight train	100	196	PZB	yes	120
LDF 2	long-distance freight train	100	242	PZB	yes	120
LDF_2	long-distance freight train	100	250	PZB	no	120
RF 1	regional freight train	100	194	PZB	yes	120
RF 2	regional freight train	100	182	PZB	yes	120
RF_2	regional freight train	100	168	PZB	no	120

The train routes for all train lines also existing in timetable concept 1 were not changed and can be found in Figure 66 to Figure 71.

The scheduled train route for the line RE_2 is the same as the one for the line RE (see Figure 68), however, it starts in the station D-Stadt, hence in the opposite direction first.

The LDF_2 train line represents a long-distance freight train which runs back and forth between the stations D-Stadt and M-Dorf. The scheduled train routes are shown in Figure 72. Scheduled stops are only present in the terminal stations of the two train runs.

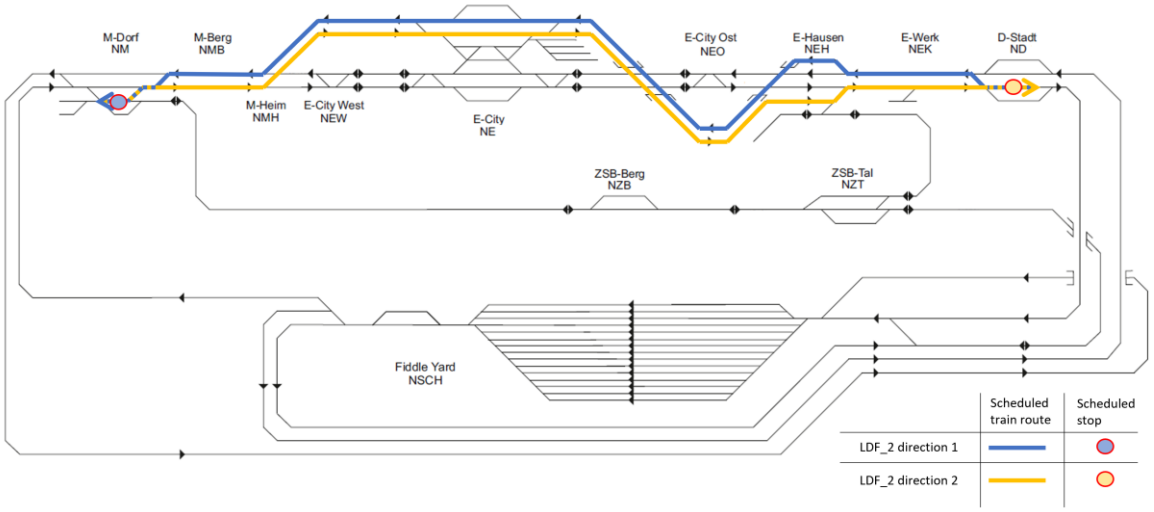


Figure 72: Scheduled train routes of the LDF_2 train line

The RF_2 train line resembles a regional freight train running back and forth on the one-directional line segment between the stations M-Dorf and ZSB-Tal, as shown in Figure 73. For each direction, scheduled stops occur in the terminal stations.

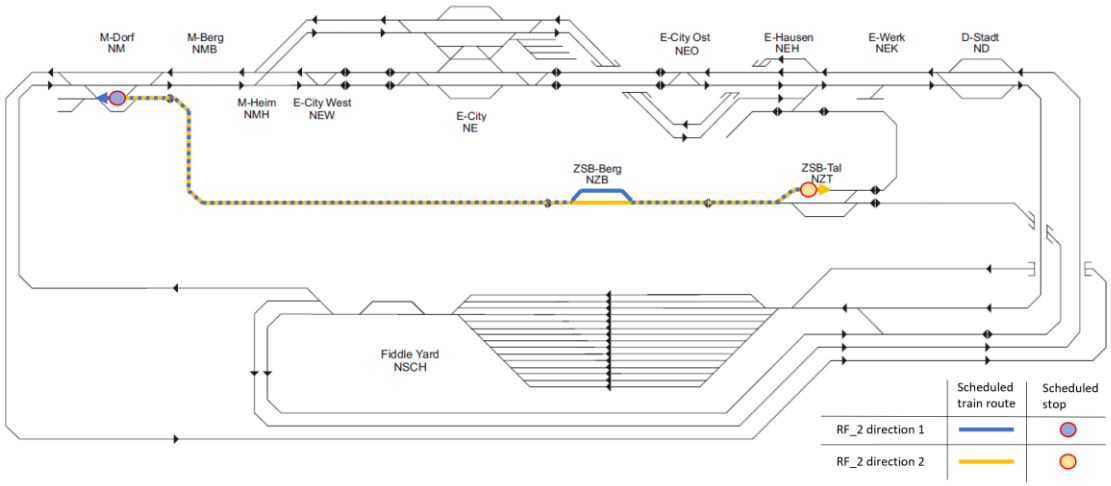


Figure 73: Scheduled train routes of the RF_2 train line

High traffic volume and investigation period of two hours (timetable concept 3)

To be able to perform a higher quantity of test cases, a third timetable concept was defined with an investigation period of two hours. These two hours include one hour of regular

operation, as well as half an hour before and after, such that all train runs can start and terminate. A high traffic volume was chosen for timetable concept 3. A total of ten vehicles are used in this timetable concept, each operating a certain train line: two high-speed passenger trains, one long-distance passenger train, two regional express passenger trains, one local regional passenger train, two long-distance freight trains and two regional freight trains.

Table 17 gives an overview of the train lines present in timetable concept 3. Compared to timetable concept 2, the train lines LDF_2 and RF_2 were removed and the frequency of the other four freight train lines was reduced from 120 minutes to 60 minutes. As in timetable concept 2, an average number of 18 train runs is operated within a time frame of one hour. The traffic density, however, is higher than in timetable concept 2, as more trains are staying inside the dispatching area and thus constantly occupy the infrastructure.

Table 17: Overview of the train lines in timetable concept 3

Train line	Category	Maximum train speed [km/h]	Train length [m]	Highest compatible train protection system	Entering the dispatching area	Frequency [min]
ICE 1	high-speed long-distance passenger train	300	226	LZB	yes	60
ICE 2	high-speed long-distance passenger train	300	146	LZB	yes	60
IC	long-distance passenger train	160	220	PZB	no	30
RE	regional express passenger train	160	220	PZB	no	60
RE_2	regional express passenger train	160	220	PZB	no	60
RB	local regional passenger train	100	154	PZB	no	30
LDF 1	long-distance freight train	100	196	PZB	yes	60
LDF 2	long-distance freight train	100	242	PZB	yes	60
RF 1	regional freight train	100	194	PZB	yes	60
RF 2	regional freight train	100	182	PZB	yes	60

The scheduled train routes for all train lines, except for the LDF lines, were not changed compared to timetable concept 2. The routes of both the LDF 1 and LDF 2 line were changed to run via the crossovers E-City West and E-City Ost, to achieve a higher traffic density along

these segments. Figure 74 shows the scheduled train routes of both LDF lines for timetable concept 3.

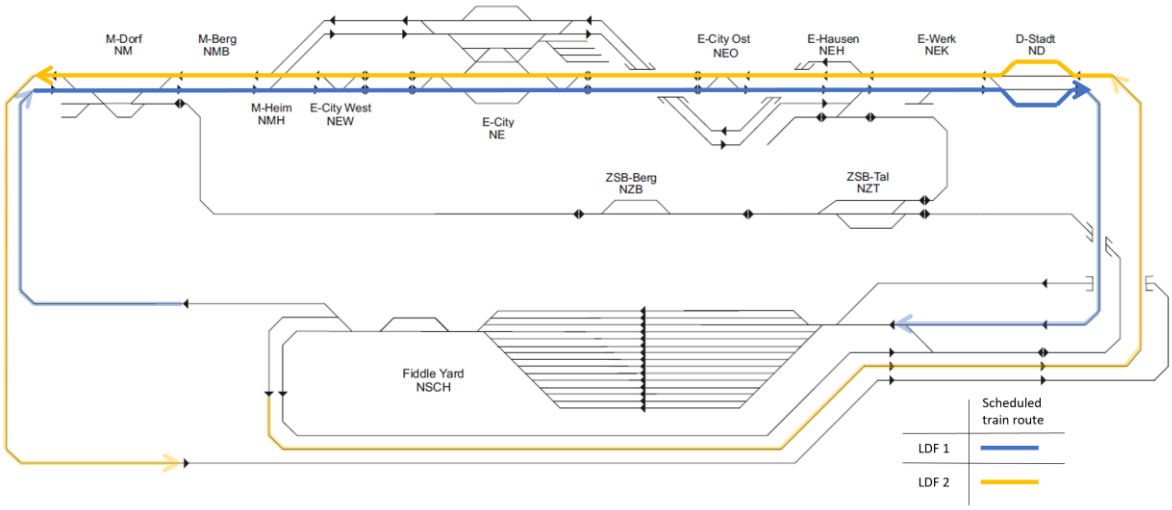


Figure 74: Scheduled train routes of the LDF train lines via the crossovers

For all three timetable concepts, unique train numbers are defined for each train run. For a train run, let n be the number of train runs of the same train line and direction that are scheduled prior to this train run. The index $i = 2 \cdot (n + 1)$ is used at the end of each train number to obtain uniqueness. The format of all train numbers is shown in Table 18. For example, the train number of the first ICE 1 train operated within a timetable is defined as ICE 1/2, since $i = 2 \cdot (n + 1) = 2 \cdot (0 + 1) = 2$. For the IC line, the train number of the second IC train operated in direction 1 is IC 1/4, whereas the train number of the third RB train operated in direction 2 is RB 2/6.

Table 18: Unique train numbers

Train line	Train number (starting direction)	Train number (opposite direction)
ICE 1	ICE $1/i$	-
ICE 2	ICE $2/i$	-
IC	IC $1/i$	IC $2/i$
RE	RE $1/i$	RE $2/i$
RE_2	RE_2 $1/i$	RE_2 $2/i$
RB	RB $1/i$	RB $2/i$
LDF 1	LDF $1/i$	-
LDF 2	LDF $2/i$	-
LDF_2	LDF_2 $1/i$	LDF_2 $2/i$
RF 1	RF $1/i$	-
RF 2	RF $2/i$	-
RF_2	RF_2 $1/i$	RF_2 $2/i$

Generation of delay data

To obtain realistic results, the delay data used in the case study must resemble delays occurring during real operation. As described in section 6.5, delay data are randomly

generated, based on the delay distribution function. The parameters for the delay distribution function used in this case study were taken from the Richtlinie Fahrwegkapazität (Ril 405) by DB Netz AG [152]. For both delays at entry into the dispatching area and primary delays, assumed to originate in a station within the dispatching area during operation, average values for different train classes are given in the Ril 405.

Table 19 shows the parameters for the delay distribution function of delays at entry according to [152]. Note that the average delay $\frac{1}{\lambda}$ only refers to the delay values greater than 0. Access routes are routes leading towards the dispatching area, i.e. the routes connecting the fiddle yard with the dispatching area are access routes. For the case study, a high degree of capacity utilization of the access routes was assumed for all three timetable concepts.

Table 19: Average parameters of the delay distribution function for delays at entry according to [152]

	Degree of capacity utilization of the access route			
Train class	Low		High	
	Probability of delay p_d	Average delay $\frac{1}{\lambda}$ [min]	Probability of delay p_d	Average delay $\frac{1}{\lambda}$ [min]
Long-distance passenger trains	0.50	5.0	0.50	5.0
Regional express passenger trains	0.50	2.0	0.60	4.5
Local regional passenger trains	0.20	1.3	0.25	2.0
Regional freight trains	0.40	20	0.60	10
Long-distance freight trains	0.50	20	0.60	10

Table 20 shows the parameters for the delay distribution function of primary delays according to [152]. These primary delays originate during operation. To implement primary delays, the delays are allocated to scheduled and operating stops. In case a train receives a primary delay at a stop, its dwell time is increased by the respective delay. For timetable concept 1, a low degree of capacity utilization was assumed for all stations within the dispatching area, whereas for timetable concepts 2 and 3 a high degree was assumed.

Table 20: Average parameters of the delay distribution function for primary delays according to [152]

	Degree of capacity utilization in the station			
Train class	Low		High	
	Probability of delay p_d	Average delay $\frac{1}{\lambda}$ [min]	Probability of delay p_d	Average delay $\frac{1}{\lambda}$ [min]
Long-distance passenger trains	0.05	1.0	0.10	2.0
Regional express passenger trains	0.05	0.5	0.10	1.0
Local regional passenger trains	0.05	0.2	0.10	0.5
Freight trains	0.10	5.0	0.10	5.0

Unique delay data were generated for all 20 test cases, based on the above parameters of the delay distribution function. Additionally, certain conflicts between train runs were manually created by adding certain delay values to the delay data. This way, conflict resolution by OptDis and a manual dispatcher could directly be compared for these specific cases which are described below. The input delay data for all test cases can be found in Appendix F.

Manually generated conflict situations

In addition to the randomly generated delay values, four specific conflict situations were created to investigate how OptDis and the experienced human dispatcher react to and resolve certain operational situations. This was achieved by manually changing certain delays in the delay data files prior to operation. All other delay values in the delay data file remained unchanged.

In the first conflict situation, the high-speed passenger train ICE 1/2 enters the dispatching area with a delay of 5 minutes, causing a conflict with the regional express passenger train RE 1/2 which is scheduled to depart in M-Dorf (NM) shortly before ICE 1/2. After the departure of RE 1/2, the long-distance freight train LDF_2 1/2 is scheduled to arrive and stop in M-Dorf on the same track as RE 1/2 is starting from. Figure 75 shows both the originally scheduled blocking time stairways on the left side, as well as the resulting occupation conflicts due to the delayed ICE train, highlighted in red, on the right side. Note that LDF_2 1/2 reaches M-Dorf (NM) coming from M-Heim (NMH), hence running on the opposite track in M-Heim and M-Berg. It will thus not cause any occupation conflicts with ICE 1/2 and RE 1/2 before reaching the entry signal of M-Dorf. This is indicated by the light yellow part of its blocking time stairway.

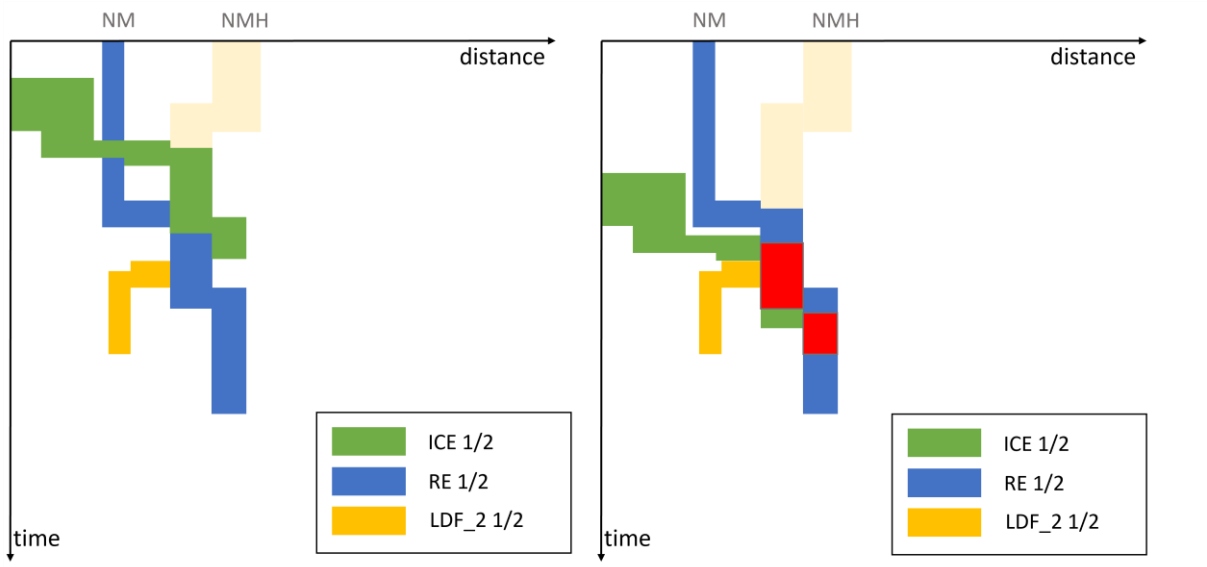


Figure 75: Conflict situation 1: Originally scheduled blocking time stairways (left) and occupation conflicts (red) caused by a delayed ICE train (right)

In the second conflict situation, the regional express passenger train RE_2 1/2 receives a delay of 10 minutes in the station E-City (NE). This results in occupation conflicts with the long-distance freight train LDF 2/2, as well as with the regional freight train RF 2/2 on the target track in M-Dorf (NM). Figure 76 shows the originally scheduled blocking time stairways on the left side and the occupation conflicts arising from the delay which the RE train receives in E-City, highlighted in red, on the right side.

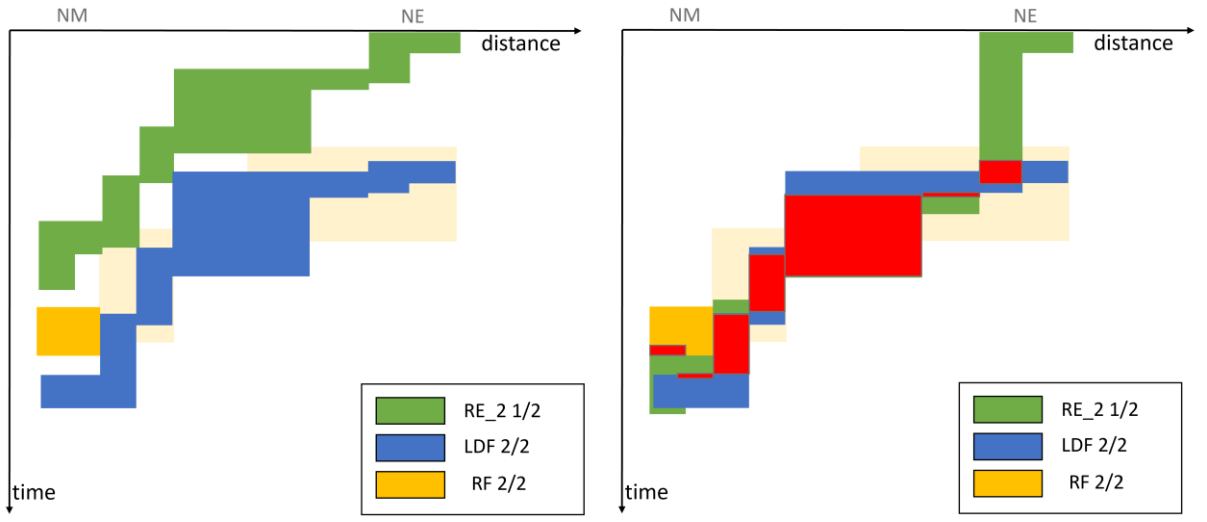


Figure 76: Conflict situation 2: Originally scheduled blocking time stairways (left) and occupation conflicts (red) caused by a delayed RE train (right)

For the third conflict situation, the long-distance freight train LDF_2 2/2 receives a delay of 14 minutes in the station M-Dorf (NM), leading to occupation conflicts with the succeeding long-distance passenger train IC 1/4 between E-City (NE) and D-Stadt (ND). In Figure 77, the originally scheduled blocking time stairways are shown on the left side and the occupation conflicts, caused by the delayed LDF train, are displayed on the right side and are highlighted in red.

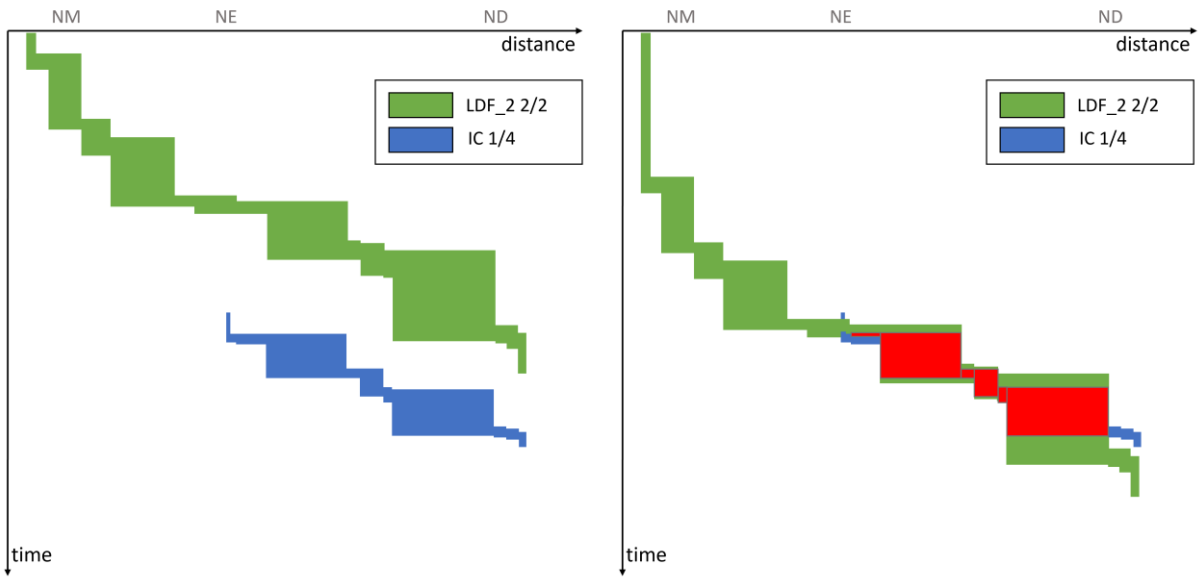


Figure 77: Conflict situation 3: Originally scheduled blocking time stairways (left) and occupation conflicts (red) caused by a delayed LDF train (right)

In the fourth conflict situation that was created manually, the ICE 1/2 train receives a delay of 8 minutes in E-City (NE) which results in an occupation conflict between E-Hausen (NEH) and D-Stadt (ND) with the IC 1/2 train that was originally scheduled to succeed the ICE train. Figure 78 shows the originally scheduled blocking time stairways on the left side and the conflict resulting from the delayed ICE train, highlighted in red, on the right side.

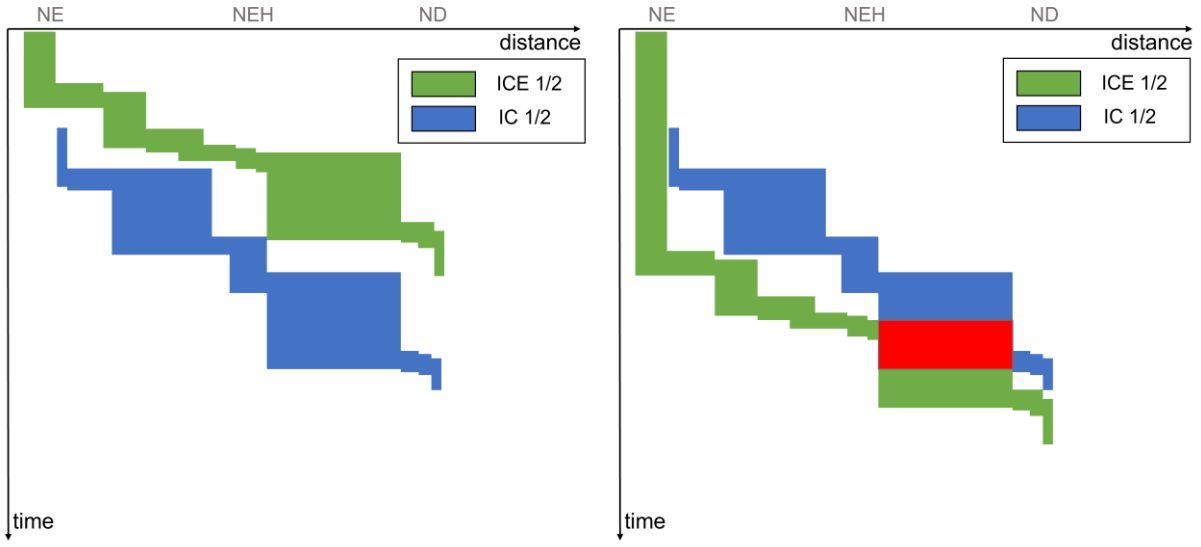


Figure 78: Conflict situation 4: Originally scheduled blocking time stairways (left) and occupation conflict (red) caused by a delayed ICE train (right)

All four manually created conflict situations were incorporated within one of the test cases of timetable concept 2 respectively. It was checked that none of the other randomly generated delays could possibly interfere with these manually constructed conflict situations. In section 9.1.2, the conflict resolution strategies of both OptDis and the experienced human dispatcher are compared.

General settings and parameters

Minimum dwell times are set to 30 seconds for all passenger trains. For freight trains, all stops are considered as operating stops (see 3.2) and therefore have no minimum dwell times. The time interval for the distance calculation timer (see 6.4) is set to 300 milliseconds.

The message “requestDispatching” (see 4.2.1) is sent once every 60 seconds during operation under automatic dispatching. As the computation time required for the optimization within OptDis on the ELVA network is usually smaller than 5 seconds, it will have terminated once a new optimization is requested. The message is sent each time the second count of the system clock (see 6.1) reaches ‘00’. To increase computational performance of the dispatching system, the “synchronizeTime” messages are sent each time the seconds count reaches ‘30’, thus the processing of these two messages will not impede each other. Predicted times of entry into the dispatching area are sent to the dispatching system 30 minutes prior to a train’s time of entry for predictions to be sufficiently reliable.

During the conflict resolution process, OptDis minimizes the overall weighted delay of the train runs. All weights, by which the train runs are prioritized during the optimization, are standardized such that the smallest weight equals 1. The non-standardized weights are determined via the product of the train class factor, the freight train correction factor and the total train mass. Table 21 shows the values that are utilized in the case study. The train class factors can be found in [153]. The freight train correction factor is set to 0.1 for freight trains and to 1 for passenger trains [153]. For the case study, it is assumed that all trains of the same train class have the same total mass. Note that it is also possible for different trains of the same class to have different masses and thus to be weighted differently within OptDis.

Table 21: Weights used by OptDis during conflict resolution

Train class	**ICE**	**IC**	**RE**	**RB**	**LDF**	**RF**
Train class factor	1.1	0.99	0.77	0.61	0.88	0.66
Freight train correction factor	1	1	1	1	0.1	0.1
Total mass [t]	896	536.9	332	128.8	1887	1887
Non-standardized weight	985.6	531.531	255.64	78.568	166.056	124.542
Standardized weight used by OptDis	**12.545**	**6.765**	**3.254**	**1**	**2.114**	**1.585**

9.1.2 Results

This section presents the results obtained from the operation of the test cases defined in 9.1.1 on the Railway Signalling Lab (ELVA) of RWTH Aachen University with the dispatching system OptDis under both automatic and manual conflict resolution.

As the defined timetable concepts consist of a set of train lines which run through the network multiple times as distinct train runs, a train run reaching its terminal station within the dispatching area or the fiddle yard late can cause all future train runs of the same train line to be delayed. The performance of a dispatching strategy is therefore assessed by its ability to reduce already existing delays in the network effectively. In order not to distort the results,

the time difference between the delay at the start of a train run is compared to its delay when arriving at its terminal station. This way, only delays arising within the dispatching area itself are evaluated and all occurring delays only contribute once to the evaluation, instead of multiple times if succeeding train runs cannot start on time due to their preceding train run terminating late. Therefore, the starting delay of a train run starting in the dispatching area is defined as $\max(T_d, t_a + T_{dwell}) - T_d + pd$ where T_d is the originally scheduled departure time of the train run, t_a is the actually achieved arrival time of the preceding train run, T_{dwell} is the minimum dwell time at the turnaround and pd is the primary delay at the starting station, according to the predefined delay data. By defining the starting delay of a train run starting in the dispatching area as described above, the delayed arrival of its preceding train run is not accounted for twice. For the first train run of a train line starting in the dispatching area, the starting delay is pd. For train runs entering the dispatching area, the starting delay is the delay at entry. The final delay for trains terminating in the dispatching area is defined as the deviation between the achieved and the originally scheduled arrival times at the destination station of the train run. For train runs exiting the dispatching area, the final delay is the delay at the last reference element. As an example, consider a train run arriving at its final station with a total delay of 9 minutes. The minimum dwell time for the turnaround is 5 minutes and no additional primary delay occurs. The succeeding train run was originally scheduled to depart 6 minutes after the arrival of the preceding train run and now departs with a delay of 8 minutes, arriving at its final station with a delay of 2 minutes. For the second train run, a delay reduction of 6 minutes is assumed, as the starting delay of 8 minutes is reduced to 2 minutes.

To evaluate the overall relative delay reduction of a dispatching strategy, simply taking the average of the single delay reductions of the test cases will not yield reliable results. Thus, for each train class, the sum of all starting delays and the sum of all final delays over all test cases are considered respectively for the calculation of the overall relative delay reduction. As an example, consider two test cases and assume that in the first test case the overall delay of ICE trains was reduced from 5 minutes to 1 minute and from 2 hours to 1 hour in the second test case. If both relative delay reductions were weighted equally, the result would be distorted. An overall delay reduction from 2:05:00 to 1:01:00 hours is thus considered, yielding an overall relative delay reduction of 51.2 %. In the following, a delay increase will be referred to by a “+” sign and a delay reduction by a “-“ sign.

Timetable concept 1

For each test case of timetable concept 1, Table 22 shows the absolute delay increase or reduction over all train runs of each of the six train classes, as well as the overall weighted delay increase or reduction achieved under automatic dispatching through OptDis. The utilized weights can be found in Table 21.

Table 22: Absolute delay increase / reduction for the 5 test cases of timetable concept 1 under automatic dispatching through OptDis

Timetable concept 1, automatic dispatching					
	Delay increase / reduction over all train runs [h:mm:ss]				
Test case / Train class	1	2	3	4	5
ICE	- 0:09:21	- 0:17:14	- 0:15:17	- 0:13:34	- 0:05:03
IC	0:00:00	- 0:01:57	- 0:00:38	+ 0:18:05	+ 0:00:08
RE	+ 0:00:39	- 0:00:23	+ 0:00:35	+ 0:01:34	+ 0:00:21
RB	- 0:00:48	- 0:01:24	+ 0:02:33	+ 0:02:58	+ 0:01:30
LDF	- 0:16:55	- 0:03:02	- 0:05:15	- 0:07:20	- 0:15:30
RF	- 0:18:13	- 0:26:29	- 0:16:25	- 0:12:06	- 0:22:36
Overall weighted delay	- 3:00:37	- 4:40:25	- 03:48:41	- 1:14:29	- 2:08:24

Table 23 displays the resulting absolute delay increase or reduction obtained through manual dispatching by an experienced human dispatcher, where OptDis serves as a tool for visualization, performs conflict detection and provides a user interface for editing and transmitting dispatching measures.

Table 23: Absolute delay increase / reduction for the 5 test cases of timetable concept 1 under manual dispatching by an experienced human dispatcher

Timetable concept 1, manual dispatching					
	Delay increase / reduction over all train runs [h:mm:ss]				
Test case / Train class	1	2	3	4	5
ICE	- 0:09:28	- 0:16:32	- 0:12:52	- 00:14:50	- 0:05:49
IC	0:00:00	- 0:01:57	- 0:00:54	+ 0:02:11	0:00:00
RE	- 0:00:58	- 0:01:18	- 0:00:04	+ 0:05:01	+ 0:00:48
RB	+ 0:04:17	- 0:01:31	- 0:01:22	- 0:01:02	+ 0:02:34
LDF	+ 0:09:59	- 0:03:47	- 0:03:45	+ 0:02:57	+ 0:18:04
RF	- 0:02:56	- 0:21:04	- 0:08:14	- 0:13:11	- 0:21:26
Overall weighted delay	- 1:41:10	- 4:24:42	- 03:10:04	- 2:50:41	- 1:03:35

Table 24 shows the total starting and final delays over all train runs and test cases of timetable concept 1, obtained through implementation of automatic and manual dispatching respectively. As explained above, this allows for far more significant results, as distortions caused by very small or large reductions in one of the test cases are prevented. For each train class, this sum was computed separately. Additionally, the overall weighted starting and final delay was determined by using the weights shown in Table 21.

Table 24: Total starting and final delays over all test cases of timetable concept 1

Train class	Total starting delay OptDis [h:mm:ss]	Total final delay OptDis [h:mm:ss]	Total starting delay manual dispatching [h:mm:ss]	Total final delay manual dispatching [h:mm:ss]
ICE	1:19:41	0:19:12	1:19:41	0:20:10
IC	0:09:28	0:25:06	0:04:05	0:03:25
RE	0:17:30	0:20:16	0:51:30	0:54:59
RB	0:07:55	0:12:44	0:05:39	0:08:34
LDF	2:05:39	1:17:37	2:05:39	2:29:07
RF	3:05:31	1:30:42	3:06:31	1:59:40
Overall weighted delay	28:08:10	13:17:11	29:21:40	16:08:29

As explained above, the delay increase or reduction resulting from the implementation of a certain dispatching strategy can only be reasonably interpreted if the ratios between the overall starting delays and the respective overall final delays are considered, hence which ratio of the starting delays could be reduced by the respective strategy.

The relative delay reductions, distinguished by train classes, and the overall average weighted delay reduction are given in Figure 79 for both automatic dispatching through OptDis and manual dispatching.

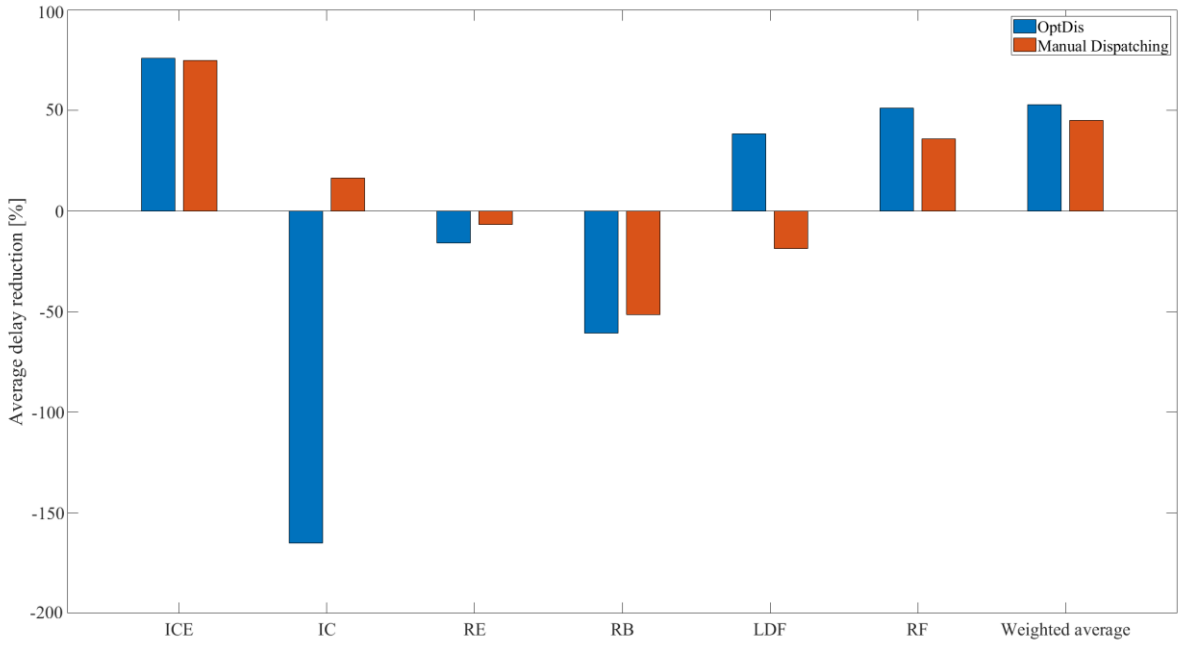


Figure 79: Comparison of average delay reductions [%] for timetable concept 1 under automatic dispatching through OptDis and manual dispatching by an experienced human dispatcher

For timetable concept 1, no significant advantage of automatic dispatching through OptDis can be observed over manual dispatching. While OptDis managed to reduce slightly more delay for the ICE high-speed trains and freight trains, more delay was reduced for the IC, RE and RB passenger trains with manual dispatching. The manual dispatcher reported that their

strategy included the prevention of further knock-on delays to any of the ICE, IC and RE trains, as delays of trains in these train classes are weighted highest. This often included adding additional stops for one of the LDF trains to enable overtaking by an ICE, IC or RE train. However, OptDis only enables such an overtaking, in case of a lower resulting overall weighted delay. Hence, if letting an LDF freight train run before an ICE, IC or RE train yields a lower overall weighted delay, this option is preferred by OptDis. The average delay increase for IC trains by OptDis is about 160 %. This result is primarily caused by test case 4 where automatic dispatching through OptDis yields an absolute delay increase of about 18 minutes, whereas through manual dispatching, a delay increase of merely 2 minutes is obtained. As the delay reduction or increase of IC trains in the other four test cases do not differ significantly between the two dispatching strategies, the result of test case 4 thus contributes most to the overall result. The manual dispatcher reports that for timetable concept 1 with low traffic volume, conflict resolution was usually not complex and could mostly by achieved by local adjustments of the conflicting train trajectories. Overall, for timetable concept 1, OptDis achieved a 7 % higher weighted delay reduction than the experienced human dispatcher, thus no significant improvement through OptDis could be observed.

Timetable concept 2

Table 25 shows the absolute increase or reduction of the delays for each of the five test cases of timetable concept 2 under automatic dispatching.

Table 25: Absolute delay increase / reduction for the 5 test cases of timetable concept 2 under automatic dispatching through OptDis

Timetable concept 2, automatic dispatching					
	Delay increase / reduction over all train runs [h:mm:ss]				
Test case / Train class	1	2	3	4	5
ICE	- 0:09:24	- 0:19:06	- 0:09:27	- 0:04:59	- 0:15:46
IC	- 0:03:39	- 0:05:21	+ 0:03:11	+ 0:24:35	+ 0:05:11
RE	+ 0:29:40	- 0:01:01	- 0:10:43	+ 0:50:03	+ 0:30:39
RB	- 0:30:09	- 0:19:15	- 0:10:50	- 0:31:47	- 0:24:53
LDF	- 0:37:42	- 0:01:42	+ 0:40:38	+ 0:15:27	+ 0:01:38
RF	- 0:02:41	- 0:12:50	+ 0:02:13	- 0:04:47	- 0:01:51
Overall weighted delay	- 2:40:11	- 5:22:18	- 0:53:18	+ 4:19:57	- 1:27:21

Similarly, Table 26 shows the absolute delay increase or reduction obtained through manual dispatching by an experienced human dispatcher.

Table 26: Absolute delay increase / reduction for the 5 test cases of timetable concept 2 under manual dispatching by an experienced human dispatcher

Timetable concept 2, manual dispatching					
	Delay increase / reduction over all train runs [h:mm:ss]				
Test case / Train class	1	2	3	4	5
ICE	+ 0:16:27	- 0:12:55	+ 0:16:51	+ 0:24:18	+ 0:16:45
IC	+ 0:01:00	- 0:03:49	+ 0:01:10	- 0:01:52	- 0:01:02
RE	+ 0:10:18	- 0:00:49	+ 0:26:41	+ 0:09:41	+ 0:01:43
RB	- 0:29:54	- 0:25:29	- 0:29:41	- 0:31:23	- 0:26:16
LDF	+ 0:40:12	+ 0:45:22	+ 1:15:07	+ 0:56:33	+ 0:27:45
RF	- 0:02:36	+ 0:05:51	+ 0:06:30	-0:06:57	- 0:07:23
Overall weighted delay	+ 4:57:36	- 1:50:49	+ 7:25:31	+ 6:40:52	+ 3:49:25

The total starting and final delays for timetable concept 2 under both automatic dispatching through OptDis and manual dispatching are displayed in Table 27.

Table 27: Total starting and final delays over all test cases of timetable concept 2

Train class	**Total starting delay OptDis [h:mm:ss]**	**Total final delay OptDis [h:mm:ss]**	**Total starting delay manual dispatching [h:mm:ss]**	**Total final delay manual dispatching [h:mm:ss]**
ICE	1:55:13	0:52:46	2:35:57	3:11:32
IC	2:11:29	2:28:28	1:45:23	1:36:11
RE	1:19:28	2:27:12	3:10:20	4:13:27
RB	6:31:41	3:53:38	4:43:40	2:20:42
LDF	3:05:00	3:02:23	4:09:23	6:56:28
RF	1:35:54	1:26:24	2:33:57	2:29:17
Overall weighted delay	61:00:00	54:39:30	81:34:00	102:36:40

The corresponding relative delay reductions are visualized in Figure 80.

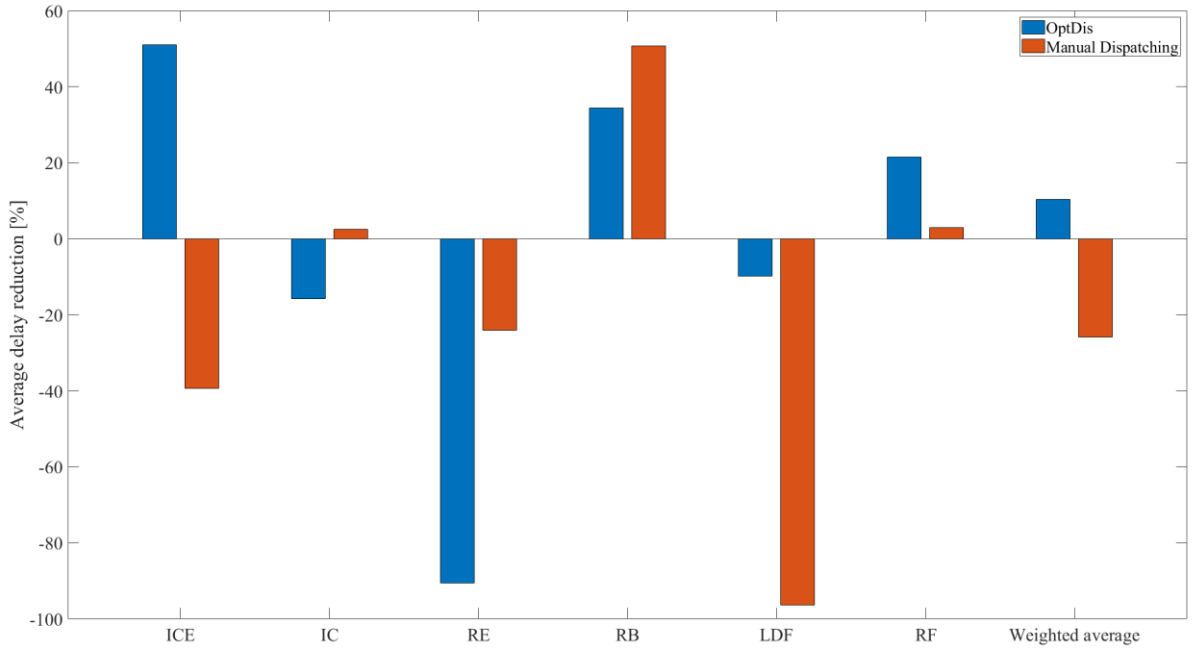


Figure 80: Comparison of average delay reductions [%] for timetable concept 2 under automatic dispatching through OptDis and manual dispatching by an experienced human dispatcher

For timetable concept 2, automatic dispatching through OptDis generated significantly higher delay reductions for the ICE high-speed trains and for the long-distance freight trains of about 89 % and 87 % more respectively, compared to manual dispatching. A significantly higher delay reduction was achieved through manual dispatching for the RE passenger trains of about 67 % more, compared to automatic dispatching. The human dispatcher reports that in some cases they reacted too late and missed the opportunity to reschedule an ICE train to run before an IC or RE train of lower priority. Thus, for the IC and RE trains, more delay could be reduced or less additional delay increase was obtained, while resulting in an additional delay increase for the ICE trains, yielding higher overall weighted delays. On the one-directional line segment between the station M-Dorf and ZSB-Tal on which the local regional passenger RB trains and the regional freight RF trains run, the overall weighted delay reduction is slightly higher for automatic dispatching through OptDis. The human dispatcher mainly focused on the busy bi-directional line segments between the stations M-Dorf and D-Stadt where more complex conflict situations occurred and thus less attention was paid to the one-directional line segment. Overall, for timetable concept 2, OptDis achieved a 35 % higher weighted delay reduction than the experienced human dispatcher.

Timetable concept 3

For timetable concept 3, Table 28 gives an overview of the obtained absolute increase or reduction of the delays under automatic dispatching through OptDis for each of the ten test cases.

Table 28: Absolute delay increase / reduction for the 10 test cases of timetable concept 3 under automatic dispatching through OptDis

Timetable concept 3, automatic dispatching					
	Delay increase / reduction over all train runs [h:mm:ss]				
Test case / Train class	1	2	3	4	5
ICE	- 0:11:15	- 0:02:50	- 0:05:58	- 0:06:42	- 0:00:34
IC	+ 0:00:14	+ 0:01:38	+ 0:01:18	- 0:03:53	- 0:04:02
RE	+ 0:10:02	- 0:04:24	- 0:00:03	- 0:00:32	- 0:00:35
RB	+ 0:16:20	+ 0:26:08	+ 0:05:49	+ 0:00:24	+ 0:11:52
LDF	+ 0:01:24	- 0:00:49	- 0:28:30	- 00:03:27	- 0:10:50
RF	- 0:01:54	- 0:23:20	- 0:38:05	- 0:06:24	- 0:20:50
Overall weighted delay	- 1:30:37	- 0:51:23	- 3:01:01	- 2:09:06	- 1:20:21
Test case / Train class	6	7	8	9	10
ICE	+ 0:00:48	- 0:08:34	- 0:11:42	- 0:06:57	- 0:05:33
IC	- 0:01:43	+ 0:01:51	- 0:00:41	- 0:00:05	- 0:05:14
RE	+ 0:00:20	+ 0:01:18	+ 0:01:28	- 0:00:13	- 0:02:12
RB	+ 0:32:44	+ 0:04:53	+ 0:29:01	+ 0:08:59	+ 0:00:50
LDF	- 0:08:19	- 0:06:57	- 0:13:51	- 0:04:19	- 0:06:15
RF	- 0:04:37	- 0:13:21	- 0:00:50	+ 0:07:33	- 0:00:10
Overall weighted delay	+ 0:07:21	- 2:01:42	-2:28:13	- 1:16:38	- 2:04:50

An overview of the absolute increase or reduction of the delays under manual dispatching is given in Table 29 for all ten test cases of timetable concept 3.

Table 29: Absolute delay increase / reduction for the 10 test cases of timetable concept 3 under manual dispatching by an experienced human dispatcher

Timetable concept 3, manual dispatching					
	Delay increase / reduction over all train runs [h:mm:ss]				
Test case / Train class	1	2	3	4	5
ICE	- 0:10:12	+ 0:03:16	+ 0:03:41	- 0:09:14	+ 0:03:48
IC	+ 0:00:44	+ 0:10:56	+ 0:07:40	- 0:03:45	- 0:04:32
RE	+ 0:03:14	+ 0:01:02	+ 0:10:53	- 0:00:32	- 0:01:22
RB	+ 0:00:48	+ 0:08:43	+ 0:11:18	+ 0:00:06	+ 0:01:02
LDF	+ 0:16:11	+ 0:42:47	+ 0:00:18	+ 0:12:13	+ 1:21:37
RF	- 0:01:39	- 0:04:03	+ 0:10:52	- 0:06:19	- 0:17:41
Overall weighted delay	- 1:20:05	+ 3:31:03	+ 2:42:39	- 2:07:01	+ 2:38:06
Test case / Train class	6	7	8	9	10
ICE	+ 0:00:03	- 0:09:00	- 0:11:01	- 0:06:35	- 0:03:34
IC	- 0:00:51	+ 0:00:31	+ 0:00:45	+ 0:00:48	- 0:01:50
RE	+ 0:00:16	+ 0:01:39	+ 0:07:22	- 0:00:14	+ 0:02:57
RB	+ 0:26:36	+ 0:14:04	+ 0:00:44	+ 0:04:40	+ 0:01:10
LDF	- 0:08:06	- 0:06:57	+ 0:15:39	+ 0:02:16	+ 0:00:25
RF	- 0:17:46	- 0:08:31	- 0:16:57	+ 0:10:40	- 0:00:08
Overall weighted delay	- 0:22:56	- 1:58:10	- 1:42:12	- 0:51:34	- 0:45:43

Table 30 shows the total starting and final delays obtained by the two dispatching strategies over all train runs and all test cases of timetable concept 3.

Table 30: Total starting and final delays over all test cases of timetable concept 3

Train class	**Total starting delay OptDis [h:mm:ss]**	**Total final delay OptDis [h:mm:ss]**	**Total starting delay manual dispatching [h:mm:ss]**	**Total final delay manual dispatching [h:mm:ss]**
ICE	2:34:56	1:35:39	2:34:56	1:56:08
IC	0:31:57	0:21:20	0:45:51	0:56:17
RE	1:01:09	1:06:18	1:01:56	1:27:11
RB	4:13:49	6:30:49	1:22:32	2:31:43
LDF	3:27:24	2:05:31	3:27:24	6:03:47
RF	4:17:02	2:35:04	4:17:02	3:25:30
Overall weighted delay	57:38:30	41:02:00	56:23:40	56:07:50

The resulting relative delay reductions are displayed in Figure 81.

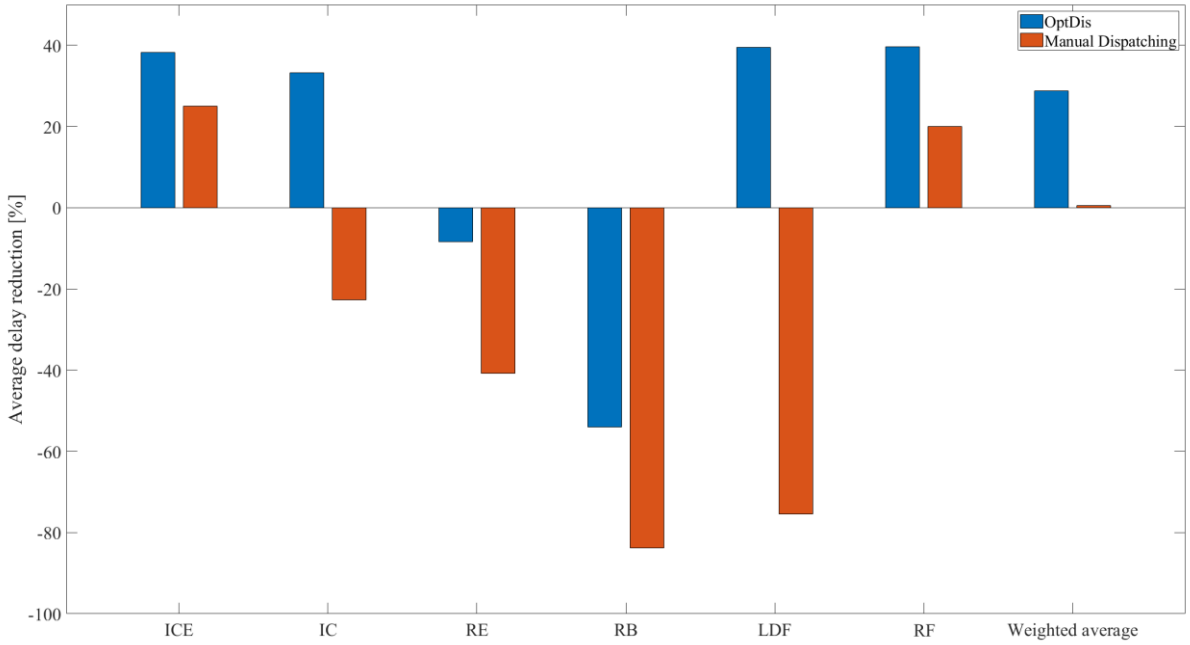


Figure 81: Comparison of average delay reductions [%] for timetable concept 3 under automatic dispatching through OptDis and manual dispatching by an experienced human dispatcher

For timetable concept 3, significantly higher delay reductions were obtained under automatic dispatching through OptDis for all train classes. The traffic volume operated in the test cases of timetable concept 3 is the highest among all three timetable concepts and thus the complexity of the operation and the resulting occupation conflicts is increased immensely, compared to timetable concepts 1 and 2. Again, the human dispatcher reports that they often were not aware of upcoming conflicts until it was already too late for them to intervene. Additionally, most attention was paid to the bi-directional line segments, especially to the ICE trains with highest priority. Thus, trains with less priority or the trains running along the one-directional line segment were often neglected during busy operational situations.

Manually generated conflict situations

For the four conflict situations which were manually generated (see 9.1.1), the conflict resolution approaches of both the dispatching system OptDis and the experienced human dispatcher are evaluated and compared.

In the first conflict situation, the delayed ICE 1/2 train causes occupation conflicts with the RE 1/2 train that is scheduled to start in direct succession of the ICE 1/2 train. Both the dispatching system OptDis and the experienced human dispatcher solved this conflict by extending the dwell time of the RE 1/2 train in M-Dorf (NM), letting the ICE 1/2 train run first and letting the RE 1/2 train follow such that future occupation conflicts were avoided, i.e. avoiding overlaps in the two trains' blocking time stairways. As the starting track of the RE 1/2 train coincides with the target track of the LDF_2 1/2 train in M-Dorf, this conflict resolution approach causes another occupation conflict between the RE 1/2 train and the LDF_2 1/2 train. OptDis solved this conflict by bending the trajectory of the LDF_2 1/2 train, delaying its

arrival in M-Dorf and thus preventing the occupation conflict with the RE 1/2 train on its target track. The human dispatcher did not adjust the trajectory of the LDF_2 1/2 train, resulting in an unscheduled stop of the freight train in front of the entry signal in M-Dorf. In the examined test case, both strategies yield equally good results, however, in general the resolution approach by OptDis is preferrable, as it prevents the LDF_2 1/2 train from blocking the section of the infrastructure leading from the entry signal in M-Dorf towards the tracks of M-Dorf, e.g. for trains coming from the crossover E-City West. Additionally, from an energy-saving perspective, bending should be preferred over running at full speed and stopping in front of block signals unnecessarily. Figure 82 shows the resulting blocking time stairways of the conflict resolution approach.

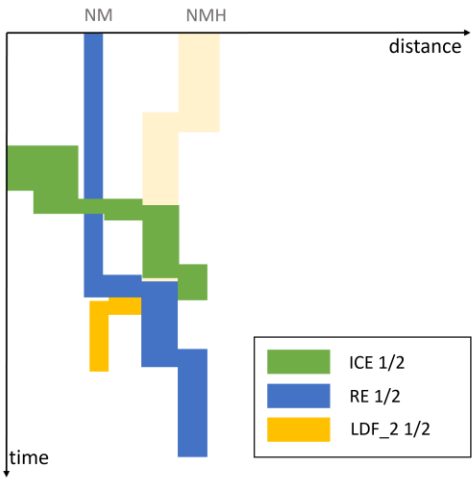


Figure 82: Conflict resolution for conflict situation 1

In the second conflict situation, the delay of the RE_2 1/2 train in E-City (NE) causes occupation conflicts with the LDF 2/2 train, as well as with the RF 2/2 train on the target track in M-Dorf (NM). The conflict resolution approaches of the dispatching system OptDis and the experienced human dispatcher differ from each other. Both OptDis and the human dispatcher bent the trajectory of the delayed RE_2 1/2 train by using all available running time reserves. OptDis bent the trajectory of the LDF 2/2 train such that it runs at a lower speed to prevent the occupation conflict with the RE_2 1/2 train. The human dispatcher, on the other hand, added an additional stop in E-City for the LDF 2/2 train. Additionally, the human dispatcher bent the trajectory of the RF 2/2 train to pass through M-Dorf 2 minutes early and thus preventing the occupation conflict on the target track of the RE_2 1/2 train. From a punctuality point of view, both conflict resolution approaches yield equally good results. Again, from an energy-saving perspective, the unnecessary stop of the LDF 2/2 train and the additional acceleration of the RF 2/2 train generate higher consumptions, however, energy-saving was not an objective in this part of the case study. Both approaches are visualized in Figure 83.

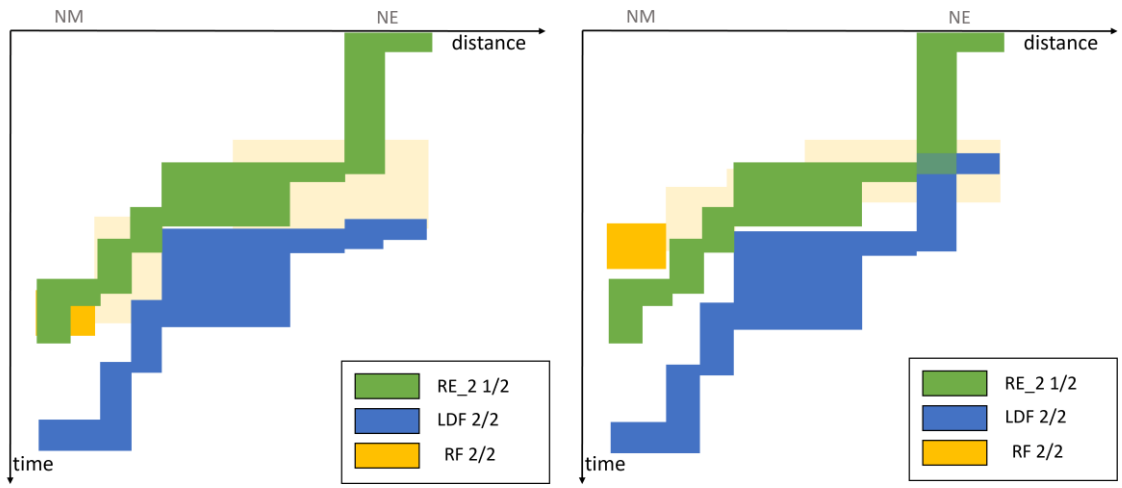


Figure 83: Conflict resolution of OptDis (left) and the experienced human dispatcher (right) for conflict situation 2

In the third conflict situation, the delay which the LDF_2 2/2 train receives in M-Dorf (NM) leads to occupation conflicts with the succeeding IC 1/4 train. The resolution approach applied by the dispatching system OptDis involves the adding of an additional stop for the LDF_2 2/2 train in the station E-City (NE) and letting the IC 1/4 train run first. The experienced human dispatcher reports that they became aware of the situation too late, such that the adding of an additional stop for the LDF_2 2/2 train was no longer possible, resulting in the freight train with lower priority running before the long-distance passenger train with higher priority. The long-distance passenger train IC 1/4 receives an additional delay of 4 minutes. The resulting blocking time stairways are displayed in Figure 84.

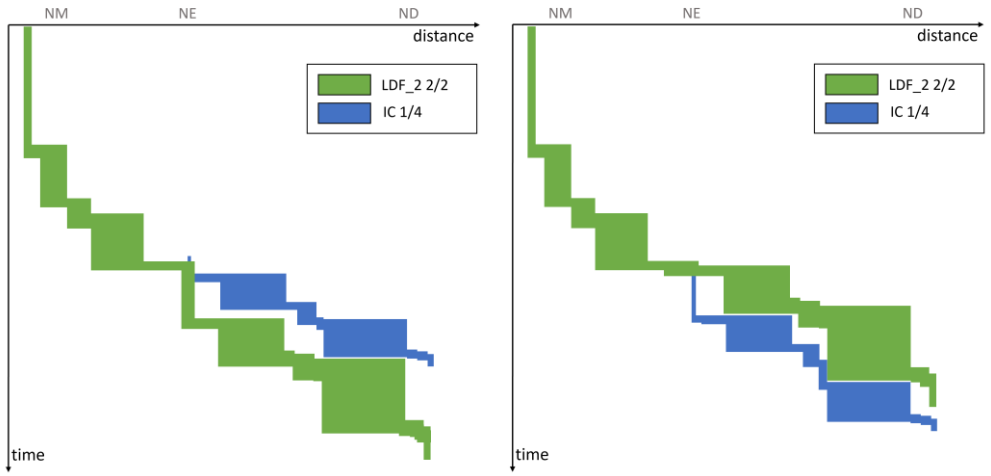


Figure 84: Conflict resolution of OptDis (left) and the experienced human dispatcher (right) for conflict situation 3

In the fourth conflict situation, the delay of the ICE 1/2 train in E-City (NE) results in an occupation conflict with the IC 1/2 train between E-Hausen (NEH) and D-Stadt (ND). Both the dispatching system OptDis and the experienced human dispatcher resolved this conflict by adding an additional stop for the IC 1/2 train in the station E-Hausen (NEH) and bending the trajectory of the ICE 1/2 train such that all running time reserves are fully used. The blocking time stairways resulting from this conflict resolution approach are shown in Figure 85.

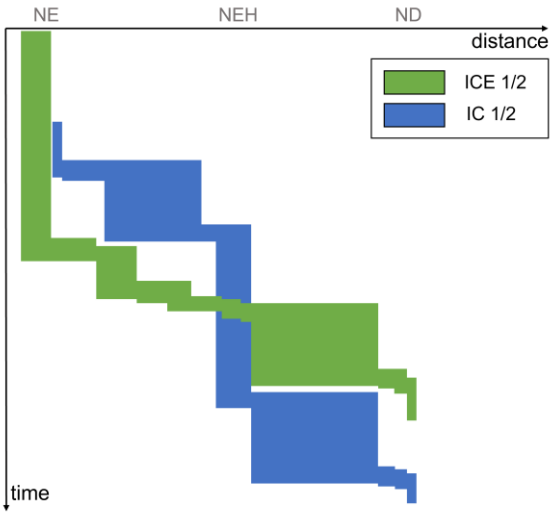


Figure 85: Conflict resolution for conflict situation 4

The four manually constructed conflict situations show that, from a punctuality point of view, conflict resolution strategies are equally good among automatic and manual dispatching. However, as seen in the evaluation of conflict situation 3, it is essential for the dispatcher to react to conflicts early, as otherwise intervention might not be possible anymore. The preferred conflict resolution strategy by OptDis is bending of train trajectories, whereas the human dispatcher preferably added additional stops in stations to solve future occupation conflicts and enable overtaking by trains with a higher priority. Especially in stations with few tracks, the adding of additional stops leads to a longer occupation of the tracks inside the station which could prevent other trains from arriving on time. Additionally, from an energy-saving perspective, bending is also preferable over the adding of additional stops. However, bending is only possible if new recommended speeds or arrival times can either be communicated to the train drivers who then adjust the running speeds of the trains or if trains are operated automatically.

Discussion of the results

In total, 20 test cases were defined which were operated under both automatic dispatching through OptDis and manual dispatching by an experienced human dispatcher. Three timetable concepts were defined for the case study. The first concept includes a low traffic volume and an examination period of four hours, whereas the second and third concept are both operated under a high traffic volume for four and two hours respectively. For the first and second timetable concept, five test cases were defined each and for the third concept, ten test cases were defined. By defining explicit test cases, the operational situations could be precisely reconstructed, enabling the reliable evaluation and comparison of the two approaches. The results were presented at the beginning of this section, while putting special emphasis on the weighted relative delay increase or reduction achieved by the two dispatching strategies.

The case study showed that the dispatching system OptDis can precisely calculate expected weighted delays and the experienced human dispatcher decides rather intuitively which train shall run first on a line segment or be preferred. Hence, OptDis usually obtains results which

are optimized holistically on a global level, whereas the human dispatcher rather optimizes operational processes locally. For low traffic volume, automatic dispatching through OptDis does not provide a significant advantage over manual dispatching by an experienced human dispatcher. For high traffic volumes, the complexity of conflicts increases and usually more than two train runs are affected. The results show that in such complex operational situations, automatic dispatching through OptDis yields great benefits over manual dispatching. Conflicts are solved holistically, taking all train runs into account and combining conflict resolution of all occurring conflicts. The resulting delay reduction can be precisely computed during the optimization process. Especially during busy operational situations with several conflicts requiring attention, the human dispatcher reports that they often recognized and reacted to conflicts too late and thus conflict resolution was often restricted or not possible anymore. Additionally, the human dispatcher reports that during major disruptions, they primarily focused on the bi-directional line segments, as in particular the high-speed trains with the greatest priority run on these segments and the one-directional line segment was often neglected by the human dispatcher.

Currently, short turning or the early termination or cancellation of train runs are not considered during dispatching through OptDis. Additionally, rerouting via operating stations not contained in the originally scheduled train path, or the passing of operating stations in another order, is currently not possible within OptDis. In certain operational situations, these dispatching measures are crucial in order to maintain a smooth operation. Thus, for OptDis to be applied during real-world operations, the available dispatching measures should be extended.

In general, the case study showed that the recommendation for an application of an automated dispatching system strongly depends on the network structure and the traffic volume operated on the network. Especially for high traffic volumes and complex operational situations, the results suggest an advantage of the application of an automated dispatching system over manual dispatching.

9.2 Evaluation of the Proposed Energy-Saving Driving Method

This section introduces the test case defined for the validation and evaluation of the energy-saving driving method (see chapter 7) in section 9.2.1. Section 9.2.2 briefly describes how the resulting energy consumption was calculated. Then, in section 9.2.3, the resulting effects of the proposed method on the energy consumption are presented and discussed.

9.2.1 Description of the Test Case

For the case study used to evaluate the energy-saving driving method presented in chapter 7, the line segment from D-Stadt to M-Dorf via the two crossovers was chosen. A high-speed long-distance passenger train (ICE) is scheduled to depart in E-Hausen and a regional express passenger train (RE) is scheduled to depart in D-Stadt. Figure 86 shows the train routes of both the ICE and RE train for the test case.

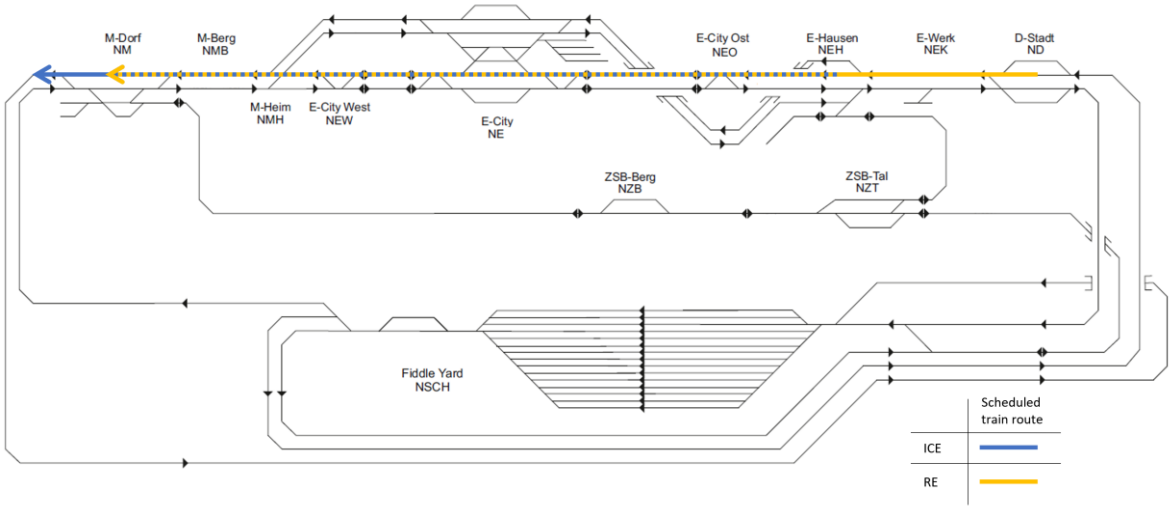


Figure 86: Train routes of the ICE and RE trains

Due to a delay of the ICE train, occupation conflicts between the two trains' trajectories occur, preventing the RE train from running smoothly at its designated speed, thus resulting in frequent unnecessary deceleration and acceleration processes of the RE train in front of block signals in case energy optimization is not performed. Figure 87 shows the scheduled blocking time stairways of both the ICE and RE trains and the resulting occupation conflicts which are caused by the delayed ICE train.

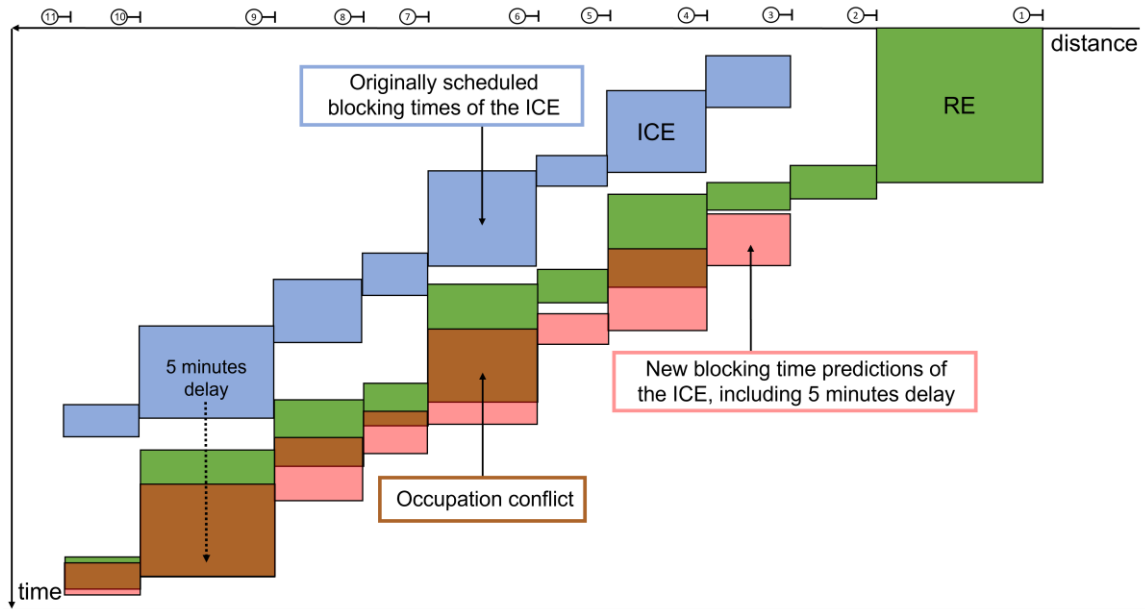


Figure 87: Blocking time stairways of the ICE and RE trains and occupation conflict resulting from the delayed ICE train

In Figure 88, all block sections along the line segment for the test case, as well as their respective distances, are displayed. Note that the distances in the figure are not according to scale.

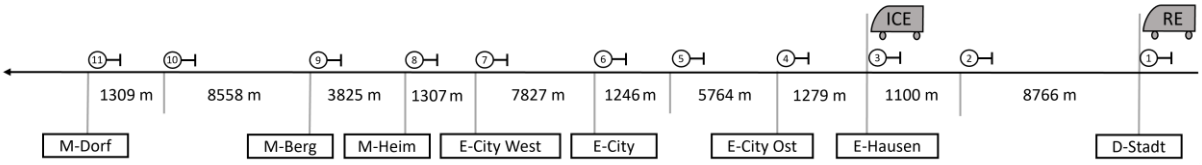


Figure 88: Line setup for the test case of the energy-saving driving method

OptDis was used to predict the block clearing times of the delayed ICE train. As OptDis does not yet provide an interface to automatically send predicted block clearing times, these times were exported into a CSV file prior to operation. Note that this is possible in the test case, as the delay of 5 minutes is predefined prior to operation. During real operation, exporting of predicted block clearing times before operation cannot be performed, thus the central traffic management system must automatically provide this information and update it whenever predictions change. For the test case, the earliest recommended passing times for the block signals were set to the respective block clearing times by the preceding ICE train. Note that for simplicity no overhead for the route setup time, the signal watching time and the approach time was considered, as route setting can be performed instantaneously and neither do human drivers have to recognize signal aspects, nor are there any pre-signals located on the ELVA infrastructure. However, during real operation these overheads would have to be considered. Latest possible passing times for the block signals were not defined, as no train directly succeeds the RE train.

During real operation, information on recommended signal passing times will not be reliable for the far future. Thus, to resemble realistic conditions in the test case, prediction horizons of 2, 5 and 8 minutes prior to the respective earliest recommended signal passing times were

assumed at which the central traffic management system provides the recommendations to the trains. The resulting energy consumptions were compared to operation without a central traffic management system, i.e. without rescheduling and energy-saving recommendations.

The energy-saving driving method described in chapter 7 was tested on the ELVA railway lab for the test case described above. The method was implemented within the timetable module of the ELVA control software (see 6.5), as this module already provides functionalities to compute target speeds which meet a given arrival time at some reference point. The optimized cruising speed is then forwarded to the train control module (see 6.4) which adjusts the train's speed profile accordingly, similar to a new maximum recommended speed due to a dispatching measure.

All speeds used during operation, as well as the time interval the speeds are run at, are recorded into a file to allow for the evaluation of the resulting energy consumption (see 9.2.2).

9.2.2 Calculation of the Energy Consumption

To evaluate the performance of the proposed energy-saving driving method on the presented test cases, the energy consumptions must be determined. The energy consumption of the physical model vehicles on the ELVA railway lab does not correspond to the energy consumption of real trains, as it is about constant regardless of the current speed or driving mode. Thus, a model to calculate the energy consumption needs to be defined and assumptions on running characteristics must be made.

Let M be the total mass of the train in kg, let $\eta_p(v)$ and $\eta_b(v)$ be the functions describing the powering and braking efficiency at speed v and let $R_r(v)$ be the running resistance of the train in Newton. Based on [154], the power consumption is modeled.

During acceleration, the power consumption is given by

$$p(t) = \frac{1}{\eta_p(v)} \cdot p_m(t)$$

and during braking by

$$p(t) = \eta_b(v) \cdot p_m(t)$$

where the mechanical power consumption $p_m(t)$ is given by

$$p_m(t) = M \cdot v(t) \cdot (a(t) + R_r(v(t))).$$

The variable t in these functions refers to the elapsed running time and $v(t)$ $[km/h]$ and $a(t)$ [m/s^2] are the current speed and current acceleration at time t respectively. For the RE train in the case study, the mass and running resistance of the train are assumed to equal the parameters of the train series BR 193 [155]. The mass is set to $M = 90000$ kg and the running resistance is set to

$$R_r(v) = 9.81 \cdot 10^{-3}\left((1.34 + 0.02071\, v) + 2.909 \cdot 10^{-6} v^2\right) \text{[N]}$$

where v is given in the unit km/h. Further, the powering efficiency is set as

$$\eta_p(v) = \frac{v + 0.1}{0.0006\, v^2 + v + 10}$$

and the braking efficiency is assumed as

$$\eta_b(v) = \frac{\eta_p(v)}{2}.$$

The total energy consumption E is calculated by integrating the power consumption $p(t)$ over the total running time T_r of the investigated speed profile:

$$E = \int_0^{T_r} p(t)\, dt.$$

During operation on the ELVA, the speeds of the trains are stored in CSV files. Each time the speed of a train changes, the new speed together with a time stamp is added to the file. The interpolation between these speed points enables the reconstruction of continuous speed profiles with intervals of constant acceleration between the speed points, which are then used to compute the corresponding energy consumptions.

9.2.3 Results

The results obtained for the different prediction horizons are compared to operation without recommendations by a central traffic management system.

The recommended time corridors for the passing of block signals become available at a certain time prior to the earliest recommended signal passing time, depending on the prediction horizon. Figure 89 shows the speed profiles of the RE train, which follows the delayed ICE train, for the different prediction horizons. If no rescheduling or energy optimization is performed, the train has to decelerate and accelerate frequently in front of block signals showing “stop”. In front of the block signal of the operating station E-City Ost, the RE train has to stop completely. For a prediction horizon of 2 minutes, i.e. the recommended time corridor becomes available 2 minutes before the earliest recommended signal passing time, less deceleration and acceleration processes are required, as new adjusted cruising speeds are available. For a prediction horizon of 5 minutes, unnecessary acceleration and subsequent deceleration occurs only until the operating station E-Hausen and the speed deviations are significantly reduced, compared to a prediction horizon of 2 minutes. For a prediction horizon of 8 minutes, no unnecessary deceleration or acceleration occurs. By applying cruising at constant intermediate speeds, the speed profile is optimized such that the train is not required to decelerate until reaching its maximum permitted speed of 160 km/h. Deceleration is thus only required at the destination signal, in front of which the train is scheduled to stop. Note that during real operation, deceleration directly following acceleration or acceleration directly following deceleration should be avoided.

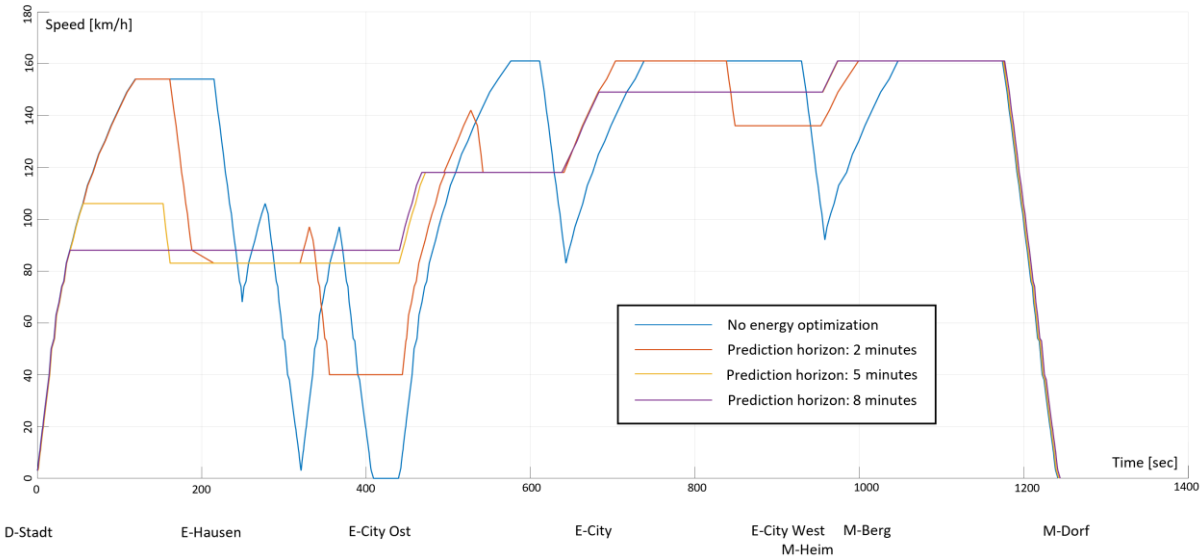


Figure 89: Speed profiles for different prediction horizons

Table 31 shows the resulting energy consumptions of the four speed profiles. A clear correlation can be seen between the prediction horizon and the energy consumption. Larger prediction horizons enable the early adjustment of speed profiles for energy optimization and thus the energy consumption decreases. However, as predictions are not reliable in the far future, prediction horizons should not be chosen too large and predictions should be constantly adjusted by the central traffic management system once new information are available. Based on the case study, prediction horizons of at least 8 minutes seem reasonable. Note that depending on the traffic volume during operation and the network topology, e.g. the length of the block sections, different prediction horizons might have to be applied to generate equally good results.

Table 31: Energy consumptions of the test cases

Test case	Energy consumption [kWh]
No energy optimization	93.7
Prediction horizon: 2 minutes	65.9
Prediction horizon: 5 minutes	52.7
Prediction horizon: 8 minutes	45.9

Discussion of the results

The case study showed that longer prediction horizons yield better results, as predictions are available early and can be considered for speed profile optimization at an early stage, thus further preventing unnecessary deceleration and acceleration processes. For real-world operation, however, predictions will not always be reliable and can change over time, especially in highly frequented network areas and when prediction horizons are very large. Further research is required to evaluate optimal lengths for prediction horizons, taking the network structure and the operated traffic volume into account.

For the case study, the earliest recommended signal passing times were set to the block clearing times of the preceding train. For an application in practice, an overhead for the route

setup time, the signal watching time and the approach time must be added. An additional overhead is also recommended, as otherwise small deviations of a few seconds from the predicted block clearing time of a preceding train might cause a succeeding train to brake unnecessarily in front of the block signal and significantly increase the overall energy consumption. The overhead should not be chosen too large, however, as this decreases the potential for energy optimization. Hence, a tradeoff must be made when choosing the prediction horizon to achieve both a good energy-saving potential and robustness of the speed profile.

For all three prediction horizons, extremely high reductions of the total energy consumption by up to 51 % could be achieved. Note that during real operation, such effects will not always be observed. The test case was constructed such that the time interval at which the RE train follows the ICE train is sufficiently small for the method to yield energy-saving benefits at all. If only few trains are operated on a network, such situations are less likely to occur and no optimization might be necessary. Additionally, the block sections on the ELVA network have different sizes. Thus, if a short block section follows a long one, the energy-saving potential is higher, as the preceding train will take a longer period of time to clear the long section, such that the train's cruising speed through the short section will be reduced to save energy. However, if all block sections have about the same size or in case moving block is used, the energy-saving effect is diminished. How often situations in which the proposed method yields such high saving potentials occur during real operation, thus strongly depends on the infrastructure, in particular on the distribution of the block sections, and on the timetable concept and capacity usage of the network.

10 Conclusion and Outlook

There still is a large gap between state-of-the-art dispatching methods proposed in the scientific literature and the use of dispatching systems for real world applications. Dispatching algorithms and strategies require intensive testing before they can be utilized during real train operation. New approaches are first tested during case studies in computer simulations. Prior to their commissioning, newly developed systems need to be adapted during extensive feedback loops, in order to obtain practical usability and acceptance.

Testing of new dispatching strategies in railway labs seems to be a promising alternative to performing the first practical tests directly on the real field. Previously performed tests of train dispatching systems in railway labs provided only decision support to human dispatchers through dispatching recommendations and operation was not performed completely automated. The research aim of this work was to design and develop a realistic testing environment for evaluations of real-time dispatching strategies during fully automated train operation in a railway lab. Furthermore, the aim was to assess the practicability of conducting part of the testing process in a railway lab and to investigate the possible benefits of an automated dispatching system over manual dispatching. These research aims were achieved by establishing the novel testing environment described in this thesis and proving its validity during a case study comparing the performance of an automated dispatching system and manual dispatching by an experienced human dispatcher.

A fundamental railway lab control software was extended and a communication interface to connect a dispatching system was established. The validity and correctness of the developed systems and their applicability for real-time operation were confirmed during extensive tests on the Railway Signalling Lab ELVA of RWTH Aachen University, while connecting the dispatching system OptDis integrated in the software LUKS®.

The results of the case study show that human dispatchers decide mostly intuitively, based on their experience, whereas an automated dispatching system such as OptDis applies a holistic mathematical programming approach which minimizes the overall weighted delay of all affected trains. Especially for very complex operational situations, where several trains are involved in the occurring conflicts, the results suggest that automatic dispatching provides significant benefits over manual dispatching, proving the general feasibility of a future application of an automated dispatching system.

Furthermore, a novel method for energy-saving driving of single trains was introduced. By integrating the proposed energy-saving driving method after the dispatching process, additional benefits regarding the energy consumption could be obtained. The proposed energy-saving driving method is particularly suitable for operation under GoA3 or GoA4, i.e. driverless and unattended train operation. However, in case of non-automated train operation, the acceptance of train drivers needs to be considered. Furthermore, as trains do not run precisely as predicted, very small deviations can cause a succeeding train to decelerate unnecessarily, diminishing the energy-saving effect. Hence, a tradeoff between the energy-saving potential and the robustness of the solution has to be made. Additionally, the length of the prediction horizons for the time corridors which the method is based on should be further investigated. Longer prediction horizons enable the algorithm to react early and

possibly yield greater energy-saving potentials. However, prediction horizons should not be too long, as the recommendations will be unreliable in the far future and already performed optimizations might have to be revoked as new information become available. This results in higher computational complexity and possible confusion for train drivers if speed recommendations are changed too frequently in case of non-driverless operation. Thus, to achieve practical applicability of the proposed energy-saving driving method, further research is necessary.

Despite the possibilities the newly developed testing environment provides, testing in a railway lab does not completely replace real field tests. The size of the infrastructure in a physical railway lab is limited and tests in larger network areas are essential, in particular for the evaluation of computation times. Furthermore, data communication between the various systems might not always be stable during real operation and specific local directives might not yet be digitized. So far, these factors cannot be reasonably accounted for when testing in a railway lab. Based on the results of this thesis, testing for the general applicability and practical usability of newly developed dispatching systems in railway labs is strongly recommended prior to performing the first tests on the real field, which nonetheless remain crucial during the development and testing process.

References

[1] F. Weymann and N. Nießen, "Verbesserung der Disposition des Eisenbahnbetriebs durch innovative Optimierungsverfahren," *ZEVRail Glasers Annalen,* no. 139, pp. 1-2, 2015.

[2] F. Weymann, "Qualität von Heuristiken in der Disposition des Eisenbahnbetriebs," RWTH Aachen, Dissertation, 2011.

[3] D. Janecek and F. Weymann, "LUKS - Analysis of lines and junctions," in *Proceedings of the 12th world conference on transport research (WCTR)*, 2010.

[4] D. Janecek and F. Weymann, "LUKS - integriertes Werkzeug zur Leistungsuntersuchung von Eisenbahnknoten und –strecken," *Eisenbahntechnische Rundschau,* no. 59, pp. 25-32, 2010.

[5] A. Liebhold, F. Weymann and N. Nießen, "Praxisnahe Testung von Dispositionsalgorithmen in einem Eisenbahnlabor," *IRSA 2021: Tagungsband, Proceedings: 3rd International Railway Symposium Aachen,* pp. 147-164, 2021.

[6] A. Liebhold and N. Nießen, "Testung von Dispositionsalgorithmen an der ELVA Aachen," *Deine Bahn,* no. 10, pp. 54-56, 2021.

[7] A. Liebhold, S. Miyoshi, N. Nießen and T. Koseki, "Onboard train speed optimization for energy saving using the prediction of block clearing times under real-time rescheduling," *Journal of Rail Transport Planning & Management,* vol. 26, 2023.

[8] I. Hansen and J. Pachl, "Railway timetable and traffic: analysis, modelling, simulation," *Eurailpress,* 2008.

[9] A. Kuckelberg, D. Janecek and N. Nießen, "Grundlagen zur Simulation der Fahrplanerstellung und Betriebsabwicklung," *Eisenbahntechnische Rundschau,* vol. 62, pp. 50-55, 2013.

[10] Y. Zhu and R. Goverde, "Integrated timetable rescheduling and passenger reassignment during railway disruptions," *Transportation Research Part B: Methodological,* vol. 140, pp. 282-314, 2020.

[11] G. Feng, P. Xu, D. Cui, X. Dai, H. Liu and Q. Zhang, "Multi-stage timetable rescheduling for high-speed railways: a dynamic programming approach with adaptive state generation," *Complex & Intelligent Systems,* vol. 7, no. 3, 2021.

[12] T. Klug, "Freight Train Routing," in *Handbook of Optimization in the Railway Industry*, Springer International Publishing, 2018, p. 73–92.

[13] DB Netz AG, "DB Richtlinie 420," 2010.

[14] R. Batley, J. Dargay and M. Wardman, "The impact of lateness and reliability on passenger rail demand," *Transportation Research Part E: Logistics and Transportation Review,* vol. 47, pp. 61-72, 2011.

[15] J. Parbo, O. Nielsen and C. Prato, "Passenger Perspectives in Railway Timetabling: A Literature Review," *Transport Reviews,* vol. 36, pp. 500-526, 2016.

[16] N. Ghaemi, O. Cats and R. Goverde, "Railway disruption management challenges and possible solution directions," *Public Transport,* vol. 9, p. 343–364, 2017.

[17] A. D'Ariano and M. Pranzo, "An Advanced Real-Time Train Dispatching System for Minimizing the Propagation of Delays in a Dispatching Area Under Severe Disturbances," *Networks and Spatial Economics,* vol. 9, p. 63–84, 2009.

[18] A. D'Ariano, "Innovative Decision Support System for Railway Traffic Control," *IEEE Intelligent Transportation Systems Magazine,* vol. 1, no. 4, pp. 8-16, 2009.

[19] J. Törnquist and J. Persson, "N-tracked railway traffic re-scheduling during disturbances," *Transportation Research Part B: Methodological,* vol. 41, no. 3, pp. 342-362, 2007.

[20] Y. Zhu and R. Goverde, "Dynamic and robust timetable rescheduling for uncertain railway disruptions," *Journal of Rail Transport Planning & Management,* vol. 15, 2020.

[21] A. Schöbel, "A Model for the Delay Management Problem based on Mixed-Integer-Programming," *Electronic Notes in Theoretical Computer Science,* vol. 50, no. 1, pp. 1-10, 2001.

[22] S. Kanai, K. Shiina, S. Harada and N. Tomii, "An optimal delay management algorithm from passengers' viewpoints considering the whole railway network," *Journal of Rail Transport Planning & Management,* vol. 1, no. 1, pp. 25-37, 2011.

[23] S. Binder, M. Maknoon, S. Sharif Azadeh and M. Bierlaire, "Passenger-centric timetable rescheduling: A user equilibrium approach," *Transportation Research Part C: Emerging Technologies,* vol. 132, 2021.

[24] T. Dollevoet, D. Huisman, M. Schmidt and A. Schöbel, "Delay Management with Rerouting of Passengers," *Transportation Science,* vol. 46, no. 1, pp. 74-89, 2011.

[25] B. Sharma, P. Pellegrini, J. Rodriguez and N. Chaudhary, "A review of passenger-oriented railway rescheduling approaches," *European Transport Research Review,* vol. 15, 2023.

[26] A. Higgins, E. Kozan and L. Ferreira, "Optimal scheduling of trains on a single line track," *Transportation Research Part B: Methodological,* vol. 30, no. 2, pp. 147-161, 1996.

[27] E. Gafarov, A. Dolgui and A. Lazarev, "Two-station single-track railway scheduling problem with trains of equal speed," *Computers & Industrial Engineering,* vol. 85, pp. 260-267, 2015.

[28] M. Tamannaei, M. Saffarzadeh, A. Jamili and S. Seyedabrishami, "A double-track train rescheduling for incident conditions: Optimisation model and decomposition method," *International Journal of Operational Research,* vol. 26, no. 1, pp. 62-87, 2016.

[29] I. Louwerse and D. Huisman, "Adjusting a railway timetable in case of partial or complete blockades," *European Journal of Operational Research,* vol. 235, no. 3, pp. 583-593, 2014.

[30] S. Zhan, L. Kroon, L. Veelenturf and J. Wagenaar, "Real-time high-speed train rescheduling in case of a complete blockage," *Transportation Research Part B: Methodological,* vol. 78, pp. 182-201, 2015.

[31] R. Wang, Q. Zhang, X. Dai, Z. Yuan, T. Zhang, S. Ding and Y. Jin, "An efficient evolutionary algorithm for high-speed train rescheduling under a partial station blockage," *Applied Soft Computing,* vol. 145, 2023.

[32] L. Chen, F. Schmid, M. Dasigi and B. Ning, "Real-Time Train Rescheduling in Junction Areas," *Proceedings of The Institution of Mechanical Engineers Part F - Journal of Rail and Rapid Transit,* 2010.

[33] V. Cacchiani, D. Huisman, M. Kidd, L. Kroon, P. Toth, L. Veelenturf and J. Wagenaar, "An overview of recovery models and algorithms for real-time railway rescheduling," *Transportation Research Part B: Methodological,* vol. 63, pp. 15-37, 2014.

[34] W. Fang, S. Yang and X. Yao, "A Survey on Problem Models and Solution Approaches to Rescheduling in Railway Networks," *IEEE Transactions on Intelligent Transportation Systems,* vol. 16, no. 6, pp. 2997-3016, 2015.

[35] Y. Zhu, H. Wang and R. Goverde, "Reinforcement Learning in Railway Timetable Rescheduling," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, Rhodes, Greece, 2020.

[36] D. Šemrov, R. Marsetič, M. Žura, L. Todorovski and A. Srdic, "Reinforcement learning approach for train rescheduling on a single-track railway," *Transportation Research Part B: Methodological,* vol. 86, pp. 250-267, 2016.

[37] J. Böttcher, "Künstliche Intelligenz und mathematische Optimierung für eine automatische Disposition," *Deine Bahn,* no. 11, pp. 10-15, 2019.

[38] SBB Swiss Federal Railways, "RCS-DISPO – das Dispositionssystem für alle Fälle.," 2018. [Online]. Available: https://bahninfrastruktur.sbb.ch/de/produkte-dienstleistungen/bahninformatiksysteme/bahnbetrieb/rcs.html. [Accessed 19. 01. 2024].

[39] SBB Swiss Federal Railways, "The swiss way to capacity optimization for Traffic Management.," 2017.

[40] K. Smith, "DB Networks buys SBB traffic management system," International Railway Journal, 2015.

[41] T. Schnick, J. Sindl and F. Falley, "Das Programm PRISMA," *Deine Bahn,* no. 10, pp. 20-25, 2021.

[42] A. Kuckelberg and F. Bednarz, "Network-wide infrastructure data for operations control centres – the PRISMA project," *Signal + Draht,* vol. 113, no. 1, pp. 28-34, 2021.

[43] G. Thiemt, M. Kant and S. Breu, "LeiDis-N Network Dispatching at the Network Management Centre of DB AG at Frankfurt," *Signal + Draht,* vol. 95, no. 6, pp. 14-19, 2003.

[44] Thales Transportation Systems, "ARAMIS Die Produktfamilie für Leitzentralen," 2016. [Online]. Available: https://docplayer.org/173490674-Aramis-die-produktfamilie-fuer-leitzentralen-thales-transportation-systems.html. [Accessed 19. 01. 2024].

[45] K. Sasaki, "New Shinkansen Operation Management and Control Systems for JR-East Shinkansen Network," *Reliability Engineering Association of Japan,* pp. 34-42, 2000.

[46] P. Noury, "ICONIS: the window for URBALIS controlled automatic METRO," in *Computers in Railways X*, Southampton, WIT Press, 2006, pp. 431-440.

[47] C. Mannino and A. Mascis, "Optimal Real-Time Traffic Control in Metro Stations," *Operations Research,* vol. 57, no. 4, pp. 1026-1039, 2009.

[48] M. Montigel, "Operations Control System in the Lotschberg Base Tunnel," *European Rail Technology Review,* vol. 49, pp. 42-44, 2009.

[49] M. Giuliari, F. Pellegrini and S. Savio, "Moving block and traffic management in railway applications: the EU project COMBINE," in *Proc. of the 7th International Conference COMPRAIL 2000*, Bologna, Italy, 2000.

[50] M. Giannettoni and S. Savio, "Traffic Management In Moving Block Railway Systems: The Results Of The EU Project COMBINE," in *Computers in Railways VIII*, Southampton, WIT Press, 2002, pp. 953-962.

[51] M. Mazzarello and E. Ottaviani, "A traffic management system for real-time traffic optimisation in railways," *Transportation Research Part B: Methodological,* vol. 41, no. 2, pp. 246-274, 2007.

[52] R. Rückert, M. Lemnian, C. Blendinger, S. Rechner and M. Müller-Hannemann, "PANDA: a software tool for improved train dispatching with focus on passenger flows," *Public Transport,* vol. 9, no. 1, pp. 307-324, 2017.

[53] A. Berger, C. Blaar, A. Gebhardt, M. Müller-Hannemann and M. Schnee, "Passenger Flow-Oriented Train Disposition," in *European Symposium on Algorithms - ESA 2011*, Saarbrücken, 2011.

[54] P. Pellegrini, G. Marliere, R. Pesenti and J. Rodriguez, "RECIFE-MILP: An effective MILP-based heuristic for the real-time railway traffic management problem," *IEEE Transactions on Intelligent Transportation Systems,* vol. 16, no. 5, pp. 2609 - 2619, 2015.

[55] P. Pellegrini, G. Marlière and J. Rodriguez, "A detailed analysis of the actual impact of real-time railway traffic management optimization," *Journal of Rail Transport Planning & Management,* vol. 6, no. 1, pp. 13-31, 2016.

[56] E. Quaglietta, "A simulation-based approach for the optimal design of signalling block layout in railway networks," *Simulation Modelling Practice and Theory,* vol. 46, pp. 4-24, 2014.

[57] E. Quaglietta, F. Corman and R. Goverde, "Stability analysis of railway dispatching plans in a stochastic and dynamic environment," *Journal of Rail Transport Planning & Management,* vol. 3, no. 4, pp. 137-149, 2013.

[58] A. Radtke and D. Hauptmann, "Automated planning of timetables in large railway networks using a microscopic data basis and railway simulation techniques," *WIT Transactions on The Built Environment,* vol. 74, 2004.

[59] A. Radtke and J. Bendfeldt, "Handling of Railway Operation Problems with RailSys," in *Proceedings of the 5th World Congress on Rail Research (WCRR 2001)*, Cologne, Germany, 2001.

[60] D. Huerlimann and A. Nash, OpenTrack - Simulation of Railway Networks, User Manual Version 1.3, ETH Zürich, Institute for Transportation Planning and Systems, 2003.

[61] A. Nash and D. Huerlimann, "Railroad Simulation Using OpenTrack," in *Computers in Railways IX*, Southampton, WIT Press, 2004, pp. 45-54.

[62] I. Pänke and A. Klimmt, "Vernetzung von Technologie und Anwenderwissen," *Deine Bahn,* no. 3, pp. 44-47, 2012.

[63] M. Hoffmann and J. Böttcher, "Entwicklung und Einführung der Zuglaufregelung," *Deine Bahn,* no. 10, pp. 6-13, 2018.

[64] H. Richta, M. Rittner and S. Große, "Automatische Dispositionsunterstützung mit ADA-PMB," *Deine Bahn,* no. 10, pp. 16-19, 2022.

[65] H. Richta, M. Rittner and S. Große, "Automatische Dispositionsassistenz im Knoten Frankfurt/Main gestartet," *Deine Bahn,* no. 2, p. 4, 2023.

[66] L. Lamorgese, C. Mannino and M. Piacentini, "Optimal Train Dispatching by Benders'-Like Reformulation," *Transportation Science,* vol. 50, no. 3, 2016.

[67] R. Borndörfer, T. Klug, L. Lamorgese, C. Mannino, M. Reuther and T. Schlechte, "Recent success stories on integrated optimization of railway systems," *Transportation Research Part C Emerging Technologies,* vol. 74, no. 2, pp. 196-211, 2017.

[68] J. Jacobs, N. Nießen and A. Kaldenbach, "DB Training nutzt die ELVA der RWTH Aachen zu Ausbildungszwecken," *Deine Bahn,* no. 7, pp. 40-43, 2022.

[69] J. Friedrich and P. Schneider, "Didaktischer Mehrwert von Eisenbahnbetriebsfeldern," *Deine Bahn,* no. 11, pp. 18-22, 2016.

[70] J. Trinckauf, "Das neue Eisenbahnlabor der Technischen Universität Dresden," *Signal + Draht,* vol. 92, no. 12, pp. 34-35, 2000.

[71] A. Stelzer and C. Streitzig, "Simulationsmöglichkeiten im Eisenbahnbetriebsfeld Darmstadt," *Signal + Draht,* vol. 103, no. 7, pp. 30-34, 2011.

[72] F. Jeker, Das Eisenbahnbetriebslabor der ETH Zürich, vdf Hochschulverlag, 2022.

[73] S. Heller and T. Schaer, "DisKon – Disposition und Konfliktlösungsmanagement der DB AG," *Der Eisenbahningenieur,* no. 9, 2004.

[74] A. Schöbel, J. Jacobs, N. Bissantz, S. Güttler, S. Kurby, S. Scholl and T. Schaer, "DisKon – Laborversion eines flexiblen, modularen und automatischen Dispositionsassistenzsystems," *Eisenbahntechnische Rundschau,* no. 12, pp. 809-821, 2005.

[75] T. Schaer, N. Bissantz, S. Güttler, J. Jacobs, S. Kurby, A. Schöbel and S. Scholl, "DisKon - Disposition und Konfliktlösungsmanagement für die beste Bahn," in *Grenzenloser Verkehr in einem grenzenlosen Europa : Verkehrswissenschaftliche Tage*, Dresden, 2005.

[76] A. Kuckelberg, S. Kurby, U. Steinborn and M. Bär, "DisKon-Tests im Eisenbahnbetriebslabor der TU Dresden," *Deine Bahn,* no. 1, pp. 15-18, 2008.

[77] S. Dietsch, M. Huth, H. Meier, A. Naumann and E. Schöne, "Evaluation einer Dispositionsoberfläche im Dresdener Eisenbahnbetriebslabor," *Signal + Draht,* vol. 107, no. 4, pp. 37-42, 2015.

[78] B. Thomas-Friedrich, P. Schneider, H. Herholz and J. Grippenkoven, "Measuring Rail Signaller Workload in a highly realistic simulated environment," in *Sixth International Human Factors Rail Conference*, 2017.

[79] TU Berlin, "EBuEf: Eisenbahn-Betriebs- und Experimentierfeld Berlin," 2022. [Online]. Available: https://www.ebuef.de/einsatz/forschung/. [Accessed 19. 01. 2024].

[80] T. Schnick, A. Wolters and A. Stelzer, "Visualisierung von Anschlusskonflikten für die Disposition," *Deine Bahn,* no. 6, pp. 26-29, 2013.

[81] Eisenbahnbetriebslabor Schweiz, "EBL – Schweiz," 2023. [Online]. Available: https://ebl-schweiz.ch/forschung/. [Accessed 19. 01. 2024].

[82] J. Pachl, "Virtuelles Eisenbahnbetriebslabor der TU Braunschweig," *Deine Bahn,* no. 1, pp. 32-37, 2013.

[83] K. Ichikawa, "Application of Optimization Theory for Bounded State Variable Problems to the Operation of Train," *Bulletin of JSME,* vol. 11, no. 47, pp. 857-865, 1968.

[84] M. Miyatake and H. Ko, "Optimization of train speed profile for minimum energy consumption," *IEEJ Transactions on Electrical and Electronic Engineering,* vol. 5, p. 263–269, 2010.

[85] P. Howlett, P. Pudney and X. Vu, "Local energy minimization in optimal train control," *Automatica,* vol. 45, no. 11, pp. 2692-2698, 2009.

[86] R. Chevrier, P. Pellegrini and J. Rodriguez, "Energy saving in railway timetabling: A bi-objective evolutionary approach for computing alternative running times," *Transportation Research Part C: Emerging Technologies,* vol. 37, pp. 20-41, 2013.

[87] P. Wang and R. Goverde, "Multi-train trajectory optimization for energy-efficient timetabling," *European Journal of Operational Research,* vol. 272, no. 2, pp. 621-635, 2019.

[88] G. Scheepmaker, P. Pudney, A. Albrecht, R. Goverde and P. Howlett, "Optimal running time supplement distribution in train schedules for energy-efficient train control," *Journal of Rail Transport Planning & Management,* vol. 14, 2020.

[89] G. Scheepmaker and R. Goverde, "The interplay between energy-efficient train control and scheduled running time supplements," *Journal of Rail Transport Planning & Management,* vol. 5, no. 4, pp. 225-239, 2015.

[90] A. Xu, B. Jia, X. Li, M. Li and A. Ghiasi, "An integrated micro-macro approach for high-speed railway energy-efficient timetabling problem," *Transportation Research Part C: Emerging Technologies,* vol. 112, pp. 88-115, 2020.

[91] Y. Huang, L. Yang, T. Tang, Z. Gao and F. Cao, "Joint train scheduling optimization with service quality and energy efficiency in urban rail transit networks," *Energy,* vol. 138, pp. 1124-1147, 2017.

[92] M. Miyatake, R. Kuwahara and S. Nakasa, "A Simple Adjustment Of Runtimes Between Stations For Saving Traction Energy By Means Of Mathematical Programming," *WIT Transactions on The Built Environment,* vol. 127, pp. 451-459, 2012.

[93] S. Watanabe and T. Koseki, "Energy-saving train scheduling diagram for automatically operated electric railway," *Journal of Rail Transport Planning & Management,* vol. 5, no. 3, pp. 183-193, 2015.

[94] T. Graffagnino, R. Schäfer, M. Tuchschmid and M. Weibel, "Energy Savings with Enhanced Static Tmetable Information for Train Driver," *Linköping Electron. Conf. Proc.,* vol. 69, no. 23, pp. 340-349, 2019.

[95] J. Liao, F. Zhang, S. Zhang and C. Gong, "A Real-Time Train Timetable Rescheduling Method Based on Deep Learning for Metro Systems Energy Optimization under Random Disturbances," *Journal of Advanced Transportation,* pp. 1-14, 2020.

[96] J. Yin, T. Tang, Z. Gao and B. Ran, "Energy-efficient metro train rescheduling with uncertain time-variant passenger demands: An approximate dynamic programming approach," *Transporation Research Part B: Methodological,* vol. 91, pp. 178-210, 2016.

[97] Z. Hou, H. Dong, S. Gao, G. Nicholson, L. Chen and C. Roberts, "Energy-Saving Metro Train Timetable Rescheduling Model Considering ATO Profiles and Dynamic Passenger Flow," *IEEE Transactions on Intelligent Transportation Systems,* vol. 20, no. 7, pp. 2774-2785, 2019.

[98] Y. Guo and C. Zhang, "Near real-time timetabling for metro system energy optimization considering passenger flow and random delays," *Journal of Rail Transport Planning & Management,* vol. 21, 2022.

[99] W. Zhou, Y. Huang, L. Deng and L. Qin, "Collaborative optimization of energy-efficient train schedule and train circulation plan for urban rail," *Energy,* vol. 263 part A, 2023.

[100] S. Zhan, P. Wang, S. Wong and S. Lo, "Energy-efficient high-speed train rescheduling during a major disruption," *Transportation Research Part E: Logistics and Transportation Review,* vol. 157, 2022.

[101] A. Toletti, V. De Martinis and U. Weidmann, "Energy savings in mixed rail traffic rescheduling: an RCG approach," *IEEE 19th International Conference on Intelligent Transportation Systems (ITSC),* pp. 2430-2435, 2016.

[102] A. Toletti, V. De Martinis and U. Weidmann, "Enhancing energy efficiency in railway operation through RCG-based rescheduling," *2017 IEEE International Conference on Environment and Electrical Engineering and 2017 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe),* pp. 1-6, 2016.

[103] F. Naldini, P. Pellegrini and J. Rodriguez, "Real-Time Optimization of Energy Consumption in Railway Networks," *Transportation Research Procedia,* vol. 62, pp. 35-42, 2022.

[104] M. Lagos, "CATO offers energy savings," *Railway Gazette International,* no. 167(5), 2011.

[105] L. Yang, L. Leander and P. Leander, "Achieving energy-efficiency and on-time performance with Driver Advisory Systems," *IEEE International Conference on Intelligent Rail Transportation Proceedings,* pp. 13-18, 2013.

[106] A. Galapitage, A. Albrecht, P. Pudney, X. Vu and P. Zhou, "Optimal real-time junction scheduling for trains with connected driver advice systems," *Journal of Rail Transport Planning & Management,* vol. 8, no. 1, pp. 29-41, 2018.

[107] P. Wang, R. Goverde and J. van Luipen, "A connected driver advisory system framework for merging freight trains," *Transportation Research Part C: Emerging Technologies,* vol. 105, pp. 203-221, 2019.

[108] J. Förster, M. Kümmling, M. Olesch, P. Reinhart, K. Vandoorne and T. Vogel, "ETCS-Bremskurven im Spiegel der Praxis," *Der Eisenbahningenieur,* no. 6, pp. 45-50, 2023.

[109] S. Iwnicki, Handbook of Railway Vehicle Dynamics, CRC Press, 2006.

[110] T. Schank, "A Fast Algorithm for Computing the Running-Time of Trains by Infinitesimal Calculus," 2011.

[111] W. Davis, "The Tractive Resistance of Electric Locomotives and Cars," *General Electr. Rev.,* vol. 29, pp. 685-708, 1926.

[112] W. Schwanhäußer, "Der zusätzliche Luftwiderstand der Eisenbahn im Tunnel," in *Tagungsband IRSA 2017*, Aachen, 2017, pp. 204-217.

[113] J. Pachl, Systemtechnik des Schienenverkehrs, Stuttgart: Teubner, 1999.

[114] Protopapadakis, Bemerkungen über die zur Berechnung des Krümmungswiderstandes angewendeten Formeln, 1937.

[115] R. Hanker, Eisenbahnoberbau - Die Grundlagen des Gleisbaues, Wien: Springer-Verlag, 1952.

[116] S. Zain, Techniques of classical mechanics: from Lagrangian to Newtonian mechanics, Bristol, England: Institute of Physics (Great Britain), 2019.

[117] H. Krugmann, Lauf der Schienenfahrzeuge im Gleis: Eine Einführung, Oldenbourg Verlag: München, 1982.

[118] D. Wende, "Fahrdynamik des Schienenverkehrs," Stuttgart, Vieweg+Teubner, 2003.

[119] I. Hansen and J. Pachl, Railway Timetabling & Operations, Hamburg: Eurailpress, 2014.

[120] A. Stelzer, Automatisierte Konfliktbewertung und -lösung für die Anschlussdisposition im (Schienen-)Personenverkehr, Darmstadt: Schriftenreihe des Instituts für Verkehr, Fachgebiet Bahnsysteme und Bahntechnik, 2016.

[121] W. Müller, Eisenbahnanlagen und Fahrdynamik: Zweiter Band Bahnlinie und Fahrdynamik der Zugförderung, Berlin, Heidelberg: Springer Berlin Heidelberg, 1953.

[122] O. Happel, "Sperrzeiten als Grundlage der Fahrplankonstruktion," *Eisenbahn-technische Rundschau (ETR),* no. 2, pp. 79-90, 1959.

[123] J. Pachl, Railway Signalling Principles, Braunschweig, 2021.

[124] J. Pachl, Das Sperrzeitmodell in der Fahrplankonstruktion, Springer Vieweg, 2015.

[125] DB Systems GmbH, "IT solutions for timetable construction/representation: RUT-K – computer-aided train-path management EBuLa – electronic timetable sheet and list of TSRs," 2005.

[126] Hacon Ingenieurgesellschaft mbH, "TIMETABLE MANAGEMENT - Effective Network-wide Train Path Planning & Capacity Management," [Online]. Available: https://www.hacon.de/fileadmin/user_upload/Portfolio/Factsheets/TPS/TPS.plan_english.pdf. [Accessed 19. 01. 2024].

[127] J. Pachl, "Timetable design principles," *Railway Timetable & Traffic,* pp. 9-42, 2008.

[128] A. Radtke, "Infrastructure Modelling," in *Railway Timetabling & Operations: Analysis, Modelling, Optimisation, Simulation, Performance Evaluation*, Eurailpress, 2014, p. 47–63.

[129] R. Diestel, Graph Theory, Heidelberg: Springer Berlin, 2018.

[130] J. Bang-Jensen and G. Gutin, Digraphs: Theory, Algorithms and Applications (Second Edition), Berlin: Springer-Verlag, 2009.

[131] N. Bešinović, R. Goverde and E. Quaglietta, "Microscopic Models and Network Transformations for Automated Railway Traffic Planning," *Computer-Aided Civil and Infrastructure Engineering,* vol. 32, no. 2, pp. 89-106, 2016.

[132] S. De Fabris, G. Longo, G. Medeossi and R. Pesenti, "Automatic generation of railway timetables based on a mesoscopic infrastructure model," *Journal of Rail Transport Planning and Management,* vol. 4, no. 1-2, pp. 2-13, 2014.

[133] A. Gille, M. Klemenz and T. Siefer, "Applying multiscaling analysis to detect capacity resources in railway networks," in *Computers in Railways XI*, vol. 103, Southampton, WIT Press, 2008, pp. 595-604.

[134] V. G. Cerf and R. E. Kahn, "A Protocol for Packet Network Intercommunication," *IEEE Trans. Commun.,* no. 22, pp. 637-648, 1974.

[135] J. Kurose and K. Ross, Computer Networking: A Top-Down Approach, Pearson Education Limited, 2012.

[136] International Union of Railways, "ETCS Implementation Handbook," 2008. [Online]. Available: https://uic.org/cdrom/2011/05_ERTMS_training2011/docs/ETCS_handbookf.pdf. [Accessed 19. 01. 2024].

[137] Eisenbahn-Bau- und Betriebsordnung (EBO), § 4 Begriffserklärungen, "Bundesministerium der Justiz," [Online]. Available: https://www.gesetze-im-internet.de/ebo/__4.html. [Accessed 19. 01. 2024].

[138] M. Farsi, K. Ratcliff and M. Barbosa, "An Overview of Controller Area Network," *Computing and Control Engineering Journal,* no. 10, pp. 113-120, 1999.

[139] J. Scalise, "How Track Circuits detect and protect trains," 11 2014. [Online]. Available: https://www.railwaysignalling.eu/wp-content/uploads/2014/11/How-track-circuits-detect-and-protect-trains.pdf. [Accessed 19. 01. 2024].

[140] E. Hofstetter and K. Haas, "Axle counter for railroad installations," *U.S. Patent US3015725A,* 06 1960.

[141] World Wide Web Consortium, "Extensible Markup Language (XML) 1.0 (Fifth Edition)," 26 11 2008. [Online]. Available: https://www.w3.org/TR/2008/REC-xml-20081126/. [Accessed 19. 01. 2024].

[142] O. Brünger, Konzeption einer Rechnerunterstützung für die Feinkonstruktion von Eisenbahnfahrplänen, RWTH Aachen: Dissertation, 1995.

[143] J. Postel, "User Datagram Protocol," Internet Engineering Task Force, 1980.

[144] D. Kozen, "Depth-First and Breadth-First Search," in *The Design and Analysis of Algorithms. Texts and Monographs in Computer Science*, New York, Springer, 1992, p. 19–24.

[145] The Qt Company, "QT Documentation - QTimer Class," [Online]. Available: https://doc.qt.io/qt-6/qtimer.html. [Accessed 19. 01. 2024].

[146] W. Schwanhäußer, "Die Bemessung der Pufferzeiten im Fahrplangefüge der Eisenbahn," RWTH Aachen, Dissertation, 1974.

[147] M. Matsumoto and T. Nishimura, "Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator," *Association for Computing Machinery,* p. 3–30, 1998.

[148] B. Stroustrup, The C++ programming language, Addison-Wesley, 1995.

[149] T. Koseki, S. Watanabe, T. Miura, T. Mizuma and E. Isobe, "Technical challenges to realize energy-efficient linear metros in Japan," in *11th International Symposium on Linear Drives for Industry Applications (LDIA)*, Osaka, Japan, 2017.

[150] Roco, "Z21 User Manual," Roco, [Online]. Available: https://www.z21.eu/en/downloads/manuals. [Accessed 19. 01. 2024].

[151] Gurobi Optimization, LCC, "Gurobi Optimizer Reference Manual," 2023. [Online]. Available: https://www.gurobi.com/wp-content/plugins/hd_documentations/documentation/9.0/refman.pdf. [Accessed 19. 01. 2024].

[152] DB Netz AG, *Richtlinie Fahrwegkapazität (Ril 405),* 405.0204A03, pp. 1-2, 2009.

[153] VIA Consulting & Development GmbH, LUKS Handbuch. Version 3.0 (not published), 2020.

[154] V. Doan, S. Watanabe and T. Koseki, "The design of an optimal running curve for train operation based on a novel parameterization method aiming to minimize the total energy consumption," *WIT Transactions on The Built Environment ,* vol. 135, pp. 175-190, 2014.

[155] F. Sauthoff, Die Bewegungswiderstände der Eisenbahnwagen unter besonderer Berücksichtigung der neueren Versuche der Deutschen Reichsbahn, VDI-Verlag, 1933.

[156] A. Nash, D. Huerlimann, J. Schuette and V. Krauss, "RailML – A standard data interface for railroad applications," *WIT Transactions on the Built Environment,* vol. 74, 2004.

List of Figures

List of Tables

Appendix A: Microscopic Infrastructure Data

Figure A.1 shows a graphic representation of the microscopic infrastructure model of an example operating station created with the software tool LUKS® [3, 4].

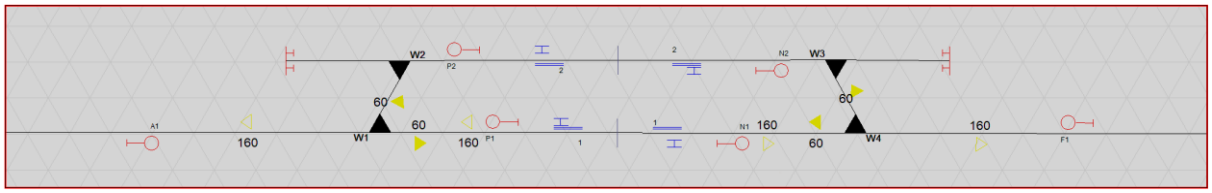


Figure A.1: Graphic representation of the microscopic infrastructure model of an example operating station

To exchange infrastructure data between different systems, a mutual interface is required. One possibility is the use of an XML file interface. An example extract of such an XML format is given below for the operating station shown in Figure A.1. The data was encoded using LUKS® which uses the German Spurplan infrastructure graph format [142]. The tags were translated into English for better understandability, as the generated files contained German tags. The operating station is subdivided into sections which correspond to the inter-switch segments.

```
<Operating_station>
    <Name>OS_1</Name>
    <Sections>
        <Section>
            <Line_number>0</Line_number>
            <Node>
                <End_of_track>
                    <ID>2</ID>
                    <Kilometrage>-3,549+0</Kilometrage>
                    <Name>2600-1</Name>
                </End_of_track>
                <Main_signal_ascending_kilometrage>
                    <ID>4</ID>
                    <Kilometrage>-2,544+0</Kilometrage>
                    <Name>A1</Name>
                </Main_signal_ascending_kilometrage>
                <Permitted_speed_descending_kilometrage>
                    <ID>7</ID>
                    <Kilometrage>-2,344+0</Kilometrage>
                    <Speed Unit="km/h">160</Speed>
                    <Valid_from>Here</Valid_from>
                    <Train_protection_system>All</Train_protection_system>
                </Permitted_speed_descending_kilometrage>
                <Switch_beginning>
                    <ID>9</ID>
                    <Kilometrage>-2,294+0</Kilometrage>
                    <Name>W1</Name>
                    <Straight_path Type="Switch_straight_path" ID="17">W1</Straight_path>
                    <Diverging_path Type="Switch_diverging_path_right" ID="12">W1</Diverging_path>
                </Switch_beginning>
```

```
    </Node>
</Section>
<Section>
    <Line_number>0</Line_number>
    <Node>
        <Bumper>
            <ID>10</ID>
            <Kilometrage>-2,324+0</Kilometrage>
            <Name>GE-W4</Name>
        </Bumper>
        <Switch_diverging_path_right>
            <ID>11</ID>
            <Kilometrage>-2,234+0</Kilometrage>
            <Partner Type="Switch_beginning" ID="36">W2</Partner>
        </Switch_diverging_path_right>
    </Node>
</Section>
<Section>
    <Line_number>0</Line_number>
    <Node>
        <Switch_diverging_path_right>
            <ID>12</ID>
            <Kilometrage>-2,294+0</Kilometrage>
            <Partner Type="Switch_beginning" ID="9">W1</Partner>
        </Switch_diverging_path_right>
        <Permitted_speed_ascending_kilometrage>
            <ID>13</ID>
            <Kilometrage>-2,293+0</Kilometrage>
            <Speed Unit="km/h">60</Speed>
            <Valid_from>Signal</Valid_from>
            <Train_protection_system>All</Train_protection_system>
        </Permitted_speed_ascending_kilometrage>
        <Switch_straight_path>
            <ID>16</ID>
            <Kilometrage>-2,234+0</Kilometrage>
            <Partner Type="Switch_beginning" ID="36">W2</Partner>
        </Switch_straight_path>
    </Node>
</Section>
<Section>
    <Line_number>0</Line_number>
    <Node>
        <Switch_straight_path>
            <ID>17</ID>
            <Kilometrage>-2,294+0</Kilometrage>
            <Partner Type="Switch_beginning" ID="9">W1</Partner>
        </Switch_straight_path>
        <Permitted_speed_ascending_kilometrage>
            <ID>18</ID>
            <Kilometrage>-2,293+0</Kilometrage>
```

```
    <Speed Unit="km/h">60</Speed>
    <Valid_from>Signal</Valid_from>
    <Train_protection_system>All</Train_protection_system>
</Permitted_speed_ascending_kilometrage>
<Permitted_speed_descending_kilometrage>
    <ID>21</ID>
    <Kilometrage>-2,144+0</Kilometrage>
    <Speed Unit="km/h">160</Speed>
    <Valid_from>Here</Valid_from>
    <Train_protection_system>All</Train_protection_system>
</Permitted_speed_descending_kilometrage>
<Main_signal_descending_kilometrage>
    <ID>23</ID>
    <Kilometrage>-2,144+0</Kilometrage>
    <Name>P1</Name>
</Main_signal_descending_kilometrage>
<Stopping_position_passenger_train_right_descending_kilometrage>
    <ID>25</ID>
    <Kilometrage>-2,024+0</Kilometrage>
    <Name>1</Name>
    <Length Unit="m">420</Length>
</Stopping_position_passenger_train_right_descending_kilometrage>
<Reference_point>
    <ID>26</ID>
    <Kilometrage>-1,819+0</Kilometrage>
    <Name>2</Name>
</Reference_point>
<Stopping_position_passenger_train_left_ascending_kilometrage>
    <ID>27</ID>
    <Kilometrage>-1,604+0</Kilometrage>
    <Name>1</Name>
    <Length Unit="m">420</Length>
</Stopping_position_passenger_train_left_ascending_kilometrage>
<Main_signal_ascending_kilometrage>
    <ID>29</ID>
    <Kilometrage>-1,484+0</Kilometrage>
    <Name>N1</Name>
</Main_signal_ascending_kilometrage>
<Permitted_speed_ascending_kilometrage>
    <ID>31</ID>
    <Kilometrage>-1,484+0</Kilometrage>
    <Speed Unit="km/h">160</Speed>
    <Valid_from>Here</Valid_from>
    <Train_protection_system>All</Train_protection_system>
</Permitted_speed_ascending_kilometrage>
<Permitted_speed_descending_kilometrage>
    <ID>34</ID>
    <Kilometrage>-1,335+0</Kilometrage>
    <Speed Unit="km/h">60</Speed>
    <Valid_from>Signal</Valid_from>
```

```
            <Train_protection_system>All</Train_protection_system>
        </Permitted_speed_descending_kilometrage>
        <Switch_straight_path>
            <ID>35</ID>
            <Kilometrage>-1,334+0</Kilometrage>
            <Partner Type="Switch_beginning" ID="56">W4</Partner>
        </Switch_straight_path>
    </Node>
</Section>
<Section>
    <Line_number>0</Line_number>
    <Node>
        <Switch_beginning>
            <ID>36</ID>
            <Kilometrage>-2,234+0</Kilometrage>
            <Name>W2</Name>
            <Straight_path Type="Switch_straight_path" ID="16">W2</Straight_path>
            <Diverging_path Type="Switch_diverging_path_right" ID="11">W2</Diverging_path>
        </Switch_beginning>
        <Main_signal_descending_kilometrage>
            <ID>39</ID>
            <Kilometrage>-2,144+0</Kilometrage>
            <Name>P2</Name>
        </Main_signal_descending_kilometrage>
        <Stopping_position_passenger_train_left_descending_kilometrage>
            <ID>41</ID>
            <Kilometrage>-2,024+0</Kilometrage>
            <Name>2</Name>
            <Length Unit="m">420</Length>
        </Stopping_position_passenger_train_left_descending_kilometrage>
        <Reference_point>
            <ID>42</ID>
            <Kilometrage>-1,819+0</Kilometrage>
            <Name>4</Name>
        </Reference_point>
        <Stopping_position_passenger_train_right_ascending_kilometrage>
            <ID>43</ID>
            <Kilometrage>-1,604+0</Kilometrage>
            <Name>2</Name>
            <Length Unit="m">420</Length>
        </Stopping_position_passenger_train_right_ascending_kilometrage>
        <Main_signal_ascending_kilometrage>
            <ID>45</ID>
            <Kilometrage>-1,484+0</Kilometrage>
            <Name>N2</Name>
        </Main_signal_ascending_kilometrage>
        <Switch_beginning>
            <ID>48</ID>
            <Kilometrage>-1,394+0</Kilometrage>
            <Name>W3</Name>
```

```
            <Straight_path Type="Switch_straight_path" ID="51">W3</Straight_path>
            <Diverging_path Type="Switch_diverging_path_left" ID="49">W3</Diverging_path>
        </Switch_beginning>
    </Node>
</Section>
<Section>
    <Line_number>0</Line_number>
    <Node>
        <Switch_diverging_path_left>
            <ID>49</ID>
            <Kilometrage>-1,394+0</Kilometrage>
            <Partner Type="Switch_beginning" ID="48">W3</Partner>
        </Switch_diverging_path_left>
        <Bumper>
            <ID>50</ID>
            <Kilometrage>-1,304+0</Kilometrage>
        </Bumper>
    </Node>
</Section>
<Section>
    <Line_number>0</Line_number>
    <Node>
        <Switch_straight_path>
            <ID>51</ID>
            <Kilometrage>-1,394+0</Kilometrage>
            <Partner Type="Switch_beginning" ID="48">W3</Partner>
        </Switch_straight_path>
        <Permitted_speed_descending_kilometrage>
            <ID>54</ID>
            <Kilometrage>-1,335+0</Kilometrage>
            <Speed Unit="km/h">60</Speed>
            <Valid_from>Signal</Valid_from>
            <Train_protection_system>All</Train_protection_system>
        </Permitted_speed_descending_kilometrage>
        <Switch_diverging_path_left>
            <ID>55</ID>
            <Kilometrage>-1,334+0</Kilometrage>
            <Partner Type="Switch_beginning" ID="56">W4</Partner>
        </Switch_diverging_path_left>
    </Node>
</Section>
<Section>
    <Line_number>0</Line_number>
    <Node>
        <Switch_beginning>
            <ID>56</ID>
            <Kilometrage>-1,334+0</Kilometrage>
            <Name>W4</Name>
            <Straight_path Type="Switch_straight_path" ID="35">W4</Straight_path>
            <Diverging_path Type="Switch_diverging_path_left" ID="55">W4</Diverging_path>
```

```
                </Switch_beginning>
                <Permitted_speed_ascending_kilometrage>
                    <ID>58</ID>
                    <Kilometrage>-1,284+0</Kilometrage>
                    <Speed Unit="km/h">160</Speed>
                    <Valid_from>Here</Valid_from>
                    <Train_protection_system>All</Train_protection_system>
                </Permitted_speed_ascending_kilometrage>
                <Main_signal_descending_kilometrage>
                    <ID>61</ID>
                    <Kilometrage>-1,084+0</Kilometrage>
                    <Name>F1</Name>
                </Main_signal_descending_kilometrage>
                <End_of_track>
                    <ID>67</ID>
                    <Kilometrage>0,001+0</Kilometrage>
                    <Name>GE_2_0-1</Name>
                </End_of_track>
            </Node>
        </Section>
    </Sections>
</Operating_station>
```

Appendix B: Train Data

An XML file format was defined to allow for the manual definition of train data which are provided as input to the train control software module (see sections 5.3.5 and 6.4). An example of such a format is given below. The address is a unique integer that is used to identify a physical vehicle. E.g., for the setting of speed levels, the address of the vehicle is needed. The initial operating station and signal are the starting position the physical vehicle is positioned in front of at the beginning of operation and the initial running direction indicates whether the vehicle will initially run forward or backward. The protection system denotes the highest compatible protection system for the vehicle, i.e. a vehicle compatible with LZB can also run on segments equipped with PZB. Trains are also grouped into different train classes such that different speed profiles can be assigned to the vehicles.

```
<Trains>
    <Train
        Address="5"
        Train_number="RF 1"
        Initial_operating_station="NSCH"
        Initial_signal="ZR5"
        Initial_running_direction="Forward"
        Maximum_velocity_kmh="100"
        Protection_system="PZB"
        Train_class="Local_freight_train"
        Number_of_axes="12"
        Length_m="194"/>
    <Train
        Address="7"
        Train_number="ICE 2"
        Initial_operating_station="NSCH"
        Initial_signal="ZR7"
        Initial_running_direction="Forward"
        Maximum_velocity_kmh="300"
        Protection_system="LZB"
        Train_class="High_speed_passenger_train"
        Number_of_axes="6"
        Length_m="146"/>
    <Train
        Address="9"
        Train_number="IC 1"
        Initial_operating_station="NE"
        Initial_signal="N4"
        Initial_running_direction="Forward"
        Maximum_velocity_kmh="160"
        Protection_system="PZB"
        Train_class="Long_distance_passenger_train"
        Number_of_axes="10"
        Length_m="220"/>
</Trains>
```

Appendix C: Timetable Data

To transmit timetable data between different software systems, a mutual encoding standard is essential. One possible way to encode railway timetable data is via the timetable scheme defined by RailML [156]. The following extract shows an example of a timetable encoded via this scheme. The timetable data was generated using LUKS® [3, 4]. For better comprehensibility, parts of the data which are not relevant for the railway lab control software described in chapter 6 were removed, which is indicated by three dots.

The train number RB/2/1 stores the two train runs RB 1/2 and RB 2/2. Information on whether a train run breaks into or breaks out of the examination area are stored via the properties “breakin“ and “breakout“. The sequence of service points gives the sequence of operating stations a train run passes in their chronological order. For each operating station, its name (“servicePoint“ and “longAppellation“), the scheduled train route (“trackSystem“), arrival and departure times and the minimum dwell time (“minStopTime“) in case of a scheduled stop are given. Note that in case of no scheduled or operating stop, only the departure time will be given.

```
<railml>
    <timetable version="1.00">
        ...
        <train trainID="1" trainNumber="RB/2/1" kind="RB 19.3" trainStatus="approved">
            ...
            <fineConstruction entryIndexTrassenID="1">
                <constructionTrain>
                    <trainNumber>
                        <mainNumber>RB 1/2</mainNumber>
                        <subNumber>0</subNumber>
                        ...
                    </trainNumber>
                    ...
                    <sequence>
                        <breakin>false</breakin>
                        <breakout>false</breakout>
                        <sequenceServicePoints>
                            <sequenceServicePoint>
                                <servicePoint>NM</servicePoint>
                                <longAppellation>M-Dorf</longAppellation>
                                <trackSystem>NSCH_WA(1) - 4 - NZB(0)</trackSystem>
                                <arrival>
                                    <time>06:06:00.00</time>
                                </arrival>
                                <departure>
                                    <time>06:11:00.00</time>
                                </departure>
                                ...
                                <minStopTime>PT5M</minStopTime> <!---5 minutes---->
                            </sequenceServicePoint>
                            <sequenceServicePoint>
```

```
                    <servicePoint>NZB</servicePoint>
                    <longAppellation>ZSB-Berg</longAppellation>
                    <trackSystem>NM(0) - 2 - NZT(0)</trackSystem>
                    <departure>
                        <time>06:16:50.08</time>
                    </departure>
                </sequenceServicePoint>
                <sequenceServicePoint>
                    <servicePoint>NZT</servicePoint>
                    <longAppellation>ZSB-Tal</longAppellation>
                    <trackSystem>NZB(0) - 3 - NSCH_WE(0)</trackSystem>
                    <arrival>
                        <time>06:23:13.65</time>
                    </arrival>
                    <departure>
                        <time>06:24:02.65</time>
                    </departure>
                    ...
                    <minStopTime>PT30S</minStopTime> <!---30 seconds---->

                </sequenceServicePoint>
            </sequenceServicePoints>
            ...
        </sequence>
    </constructionTrain>
    <constructionTrain>
        <trainNumber>
            <mainNumber>RB 2/2</mainNumber>
            <subNumber>0</subNumber>
            ...
        </trainNumber>
        ...
        <sequence>
            <breakin>false</breakin>
            <breakout>false</breakout>
            <sequenceServicePoints>
                <sequenceServicePoint>
                    <servicePoint>NZT</servicePoint>
                    <longAppellation>ZSB-Tal</longAppellation>
                    <trackSystem>NSCH_WE(0) - 3 - NZB(0)</trackSystem>
                    <arrival>
                        <time>06:24:02.00</time>
                    </arrival>
                    <departure>
                        <time>06:24:51.00</time>
                    </departure>
                    ...
                    <minStopTime>PT30S</minStopTime> <!---30 seconds---->
                </sequenceServicePoint>
                <sequenceServicePoint>
```

```
                        <servicePoint>NZB</servicePoint>
                        <longAppellation>ZSB-Berg</longAppellation>
                        <trackSystem>NZT(0) - 1 - NM(0)</trackSystem>
                        <departure>
                            <time>06:31:47.68</time>
                        </departure>
                    </sequenceServicePoint>
                    <sequenceServicePoint>
                        <servicePoint>NM</servicePoint>
                        <longAppellation>M-Dorf</longAppellation>
                        <trackSystem>NZB(0) - 4 - NSCH_WE(2)</trackSystem>
                        <arrival>
                            <time>06:37:55.63</time>
                        </arrival>
                        <departure>
                            <time>06:38:55.63</time>
                        </departure>
                        ...
                        <minStopTime>PT30S</minStopTime> <!---30 seconds---->
                    </sequenceServicePoint>
                </sequenceServicePoints>
                ...
            </sequence>
        </constructionTrain>
    </fineConstruction>
</train>
</timetable>
</railml>
```

Appendix D: Route Setting Plan

The automatic route setting (see 6.6) uses a route setting plan. For the route setting plan, an XML file interface was defined which is manually edited prior to operation. An extract of a route setting plan for a single operating station “NEH” is shown below to illustrate this format. The route setting plan contains a collection of route setting rules. Each rule contains the operating station and name of the corresponding associated signal, the distance from the triggering axis counting circuit until the signal and whether there is a stopping position in front of the signal. The operating station and name of the triggering axis counting circuit are given and optionally the operating station and name of a condition signal (see section 6.6 for details).

```
<Route_setting_plan>

    <Route_setting_rule>
        <Associated_signal Operating_station="NEH" Name="A" Distance_m="6547" Stopping_position="No"/>
        <Triggering_axis_counting_circuit Operating_station="NE" Name="FMA_14B201"/>
        <Condition_signal Operating_station="NEO" Name="A"/>
    </Route_setting_rule>

    <Route_setting_rule>
        <Associated_signal Operating_station="NEH" Name="A" Distance_m="6547" Stopping_position="No"/>
        <Triggering_axis_counting_circuit Operating_station="NEO" Name="FMA_15B102"/>
        <Condition_signal Operating_station="NEO" Name="AA"/>
    </Route_setting_rule>

    <Route_setting_rule>
        <Associated_signal Operating_station="NEH" Name="N2" Distance_m="7343" Stopping_position="Yes"/>
        <Triggering_axis_counting_circuit Operating_station="NE" Name="FMA_14B301"/>
        <Condition_signal Operating_station="NEH" Name="B"/>
    </Route_setting_rule>

    <Route_setting_rule>
        <Associated_signal Operating_station="NEH" Name="N3" Distance_m="1772" Stopping_position="Yes"/>
        <Triggering_axis_counting_circuit Operating_station="NEO" Name="FMA_15B201"/>
        <Condition_signal Operating_station="NEH" Name="A"/>
    </Route_setting_rule>

    <Route_setting_rule>
        <Associated_signal Operating_station="NEH" Name="N3" Distance_m="7343" Stopping_position="Yes"/>
        <Triggering_axis_counting_circuit Operating_station="NE" Name="FMA_14B301"/>
        <Condition_signal Operating_station="NEH" Name="B"/>
    </Route_setting_rule>

    <Route_setting_rule>
        <Associated_signal Operating_station="NEH" Name="N2" Distance_m="1772" Stopping_position="Yes"/>
        <Triggering_axis_counting_circuit Operating_station="NEO" Name="FMA_15B201"/>
        <Condition_signal Operating_station="NEH" Name="A"/>
    </Route_setting_rule>
```

```
    <Route_setting_rule>
        <Associated_signal Operating_station="NEH" Name="N5" Distance_m="7483" Stopping_position="Yes"/>
        <Triggering_axis_counting_circuit Operating_station="NE" Name="FMA_14B301"/>
        <Condition_signal Operating_station="NEH" Name="B"/>
    </Route_setting_rule>

    <Route_setting_rule>
        <Associated_signal Operating_station="NEH" Name="N5" Distance_m="1912" Stopping_position="Yes"/>
        <Triggering_axis_counting_circuit Operating_station="NEO" Name="FMA_15B201"/>
        <Condition_signal Operating_station="NEH" Name="A"/>
    </Route_setting_rule>

    <Route_setting_rule>
        <Associated_signal Operating_station="NEH" Name="B" Distance_m="6195" Stopping_position="No"/>
        <Triggering_axis_counting_circuit Operating_station="NE" Name="FMA_14B301"/>
    </Route_setting_rule>

    <Route_setting_rule>
        <Associated_signal Operating_station="NEH" Name="F" Distance_m="8239" Stopping_position="No"/>
        <Triggering_axis_counting_circuit Operating_station="ND" Name="FMA_18B102"/>
    </Route_setting_rule>

    <Route_setting_rule>
        <Associated_signal Operating_station="NEH" Name="P4" Distance_m="9339" Stopping_position="Yes"/>
        <Triggering_axis_counting_circuit Operating_station="ND" Name="FMA_18B102"/>
        <Condition_signal Operating_station="NEH" Name="F"/>
    </Route_setting_rule>

    <Route_setting_rule>
        <Associated_signal Operating_station="NEH" Name="P5" Distance_m="9339" Stopping_position="Yes"/>
        <Triggering_axis_counting_circuit Operating_station="ND" Name="FMA_18B102"/>
        <Condition_signal Operating_station="NEH" Name="F"/>
    </Route_setting_rule>

</Route_setting_plan>
```

Appendix E: Timetable Concepts for the Case Study

The following tables show the timetables for all train runs of timetable concepts 1, 2 and 3 respectively. For each operating station, the arrival and departure times at the reference element and the scheduled track number are given. Note that if the arrival time equals the departure time, the train is not scheduled to stop in this operating station. If no arrival time is given in an operating station, the train run starts in the operating station and if no departure time is given, the train run terminates in the operating station.

Timetable Concept 1

Table E.1: Timetable Concept 1 - ICE 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	06:15:55	06:15:55	2
M-Berg	06:20:22	06:20:22	1
M-Heim	06:22:02	06:22:02	1
E-City West	06:22:36	06:22:36	1
E-City	06:26:52	06:27:53	8
E-City Ost	06:31:22	06:31:22	1
E-Hausen	06:32:15	06:32:15	3
E-Werk	06:32:59	06:32:59	1
D-Stadt	06:37:47	06:38:17	1

Table E.2: Timetable Concept 1 - ICE 1/4

Operating station	Arrival time	Departure time	Track
M-Dorf	08:15:55	08:15:55	2
M-Berg	08:20:22	08:20:22	1
M-Heim	08:22:02	08:22:02	1
E-City West	08:22:36	08:22:36	1
E-City	08:26:52	08:27:53	8
E-City Ost	08:31:22	08:31:22	1
E-Hausen	08:32:15	08:32:15	3
E-Werk	08:32:59	08:32:59	1
D-Stadt	08:37:47	08:38:17	1

Table E.3: Timetable Concept 1 - ICE 2/2

Operating station	Arrival time	Departure time	Track
D-Stadt	06:27:19	06:27:49	3
E-Werk	06:32:25	06:32:25	2
E-Hausen	06:33:09	06:33:09	4
E-City Ost	06:34:00	06:34:00	2
E-City	06:37:42	06:38:12	5
E-City West	06:42:12	06:42:12	2
M-Heim	06:42:44	06:42:44	2
M-Berg	06:44:24	06:44:24	2
M-Dorf	06:48:52	06:48:52	1

Table E.4: Timetable Concept 1 - ICE 2/4

Operating station	Arrival time	Departure time	Track
D-Stadt	08:27:19	08:27:49	3
E-Werk	08:32:25	08:32:25	2
E-Hausen	08:33:09	08:33:09	4
E-City Ost	08:34:00	08:34:00	2
E-City	08:37:42	08:38:12	5
E-City West	08:42:12	08:42:12	2
M-Heim	08:42:44	08:42:44	2
M-Berg	08:44:24	08:44:24	2
M-Dorf	08:48:52	08:48:52	1

Table E.5: Timetable Concept 1 - IC 1/2

Operating station	Arrival time	Departure time	Track
E-City	-	06:47:52	4
E-Hausen	06:54:14	06:54:44	2
E-Werk	06:56:40	06:56:40	1
D-Stadt	07:01:41	-	1

Table E.6: Timetable Concept 1 - IC 2/2

Operating station	Arrival time	Departure time	Track
D-Stadt	-	07:02:41	1
E-Werk	07:08:01	07:08:01	2
E-Hausen	07:10:54	07:11:24	5
E-City	07:18:46	-	4

Table E.7: Timetable Concept 1 - IC 1/4

Operating station	Arrival time	Departure time	Track
E-City	-	07:47:52	4
E-Hausen	07:54:14	07:54:44	2
E-Werk	07:56:40	07:56:40	1
D-Stadt	08:01:41	-	1

Table E.8: Timetable Concept 1 – IC 2/4

Operating station	Arrival time	Departure time	Track
D-Stadt	-	08:02:41	1
E-Werk	08:08:01	08:08:01	2
E-Hausen	08:10:54	08:11:24	5
E-City	08:18:46	-	4

Table E.9: Timetable Concept 1 - IC 1/6

Operating station	Arrival time	Departure time	Track
E-City	-	08:47:52	4
E-Hausen	08:54:14	08:54:44	2
E-Werk	08:56:40	08:56:40	1
D-Stadt	09:01:41	-	1

Table E.10: Timetable Concept 1 - IC 2/6

Operating station	Arrival time	Departure time	Track
D-Stadt	-	09:02:41	1
E-Werk	09:08:01	09:08:01	2
E-Hausen	09:10:54	09:11:24	5
E-City	09:18:46	-	4

Table E.11: Timetable Concept 1 - RE 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	-	06:23:02	3
M-Berg	06:29:09	06:29:09	1
M-Heim	06:31:26	06:31:26	1
E-City	06:37:51	06:38:21	3
E-Hausen	06:44:23	06:44:53	2
E-Werk	06:46:40	06:46:40	1
D-Stadt	06:51:37	-	1

Table E.12: Timetable Concept 1 - RE 2/2

Operating station	Arrival time	Departure time	Track
D-Stadt	-	06:52:42	1
E-Werk	06:57:49	06:57:49	2
E-Hausen	07:00:42	07:01:12	5
E-City	07:07:45	07:08:15	2
M-Heim	07:14:01	07:14:01	2
M-Berg	07:16:34	07:16:34	2
M-Dorf	07:23:21	-	3

Table E.13: Timetable Concept 1 - RE 1/4

Operating station	Arrival time	Departure time	Track
M-Dorf	-	07:24:21	3
M-Berg	07:29:59	07:29:59	1
M-Heim	07:32:16	07:32:16	1
E-City	07:38:40	07:39:10	3
E-Hausen	07:45:00	07:45:30	2
E-Werk	07:47:17	07:47:17	1
D-Stadt	07:52:13	-	1

Table E.14: Timetable Concept 1 - RE 2/4

Operating station	Arrival time	Departure time	Track
D-Stadt	-	07:53:13	1
E-Werk	07:58:20	07:58:20	2
E-Hausen	08:01:13	08:01:43	5
E-City	08:08:16	08:08:46	2
M-Heim	08:14:32	08:14:32	2
M-Berg	08:17:05	08:17:05	2
M-Dorf	08:23:52	-	3

Table E.15: Timetable Concept 1 - RE 1/6

Operating station	Arrival time	Departure time	Track
M-Dorf	-	08:24:52	3
M-Berg	08:30:30	08:30:30	1
M-Heim	08:32:46	08:32:46	1
E-City	08:39:11	08:39:41	3
E-Hausen	08:45:03	08:45:33	2
E-Werk	08:47:20	08:47:20	1
D-Stadt	08:52:17	-	1

Table E.16: Timetable Concept 1 - RE 2/6

Operating station	Arrival time	Departure time	Track
D-Stadt	-	08:53:17	1
E-Werk	08:58:23	08:58:23	2
E-Hausen	09:01:16	09:01:46	5
E-City	09:08:20	09:08:50	2
M-Heim	09:14:36	09:14:36	2
M-Berg	09:17:09	09:17:09	2
M-Dorf	09:23:56	-	3

Table E.17: Timetable Concept 1 - RE 1/8

Operating station	Arrival time	Departure time	Track
M-Dorf	-	09:24:56	3
M-Berg	09:30:33	09:30:33	1
M-Heim	09:32:50	09:32:50	1
E-City	09:39:14	09:39:44	3
E-Hausen	09:45:47	09:46:17	2
E-Werk	09:48:04	09:48:04	1
D-Stadt	09:53:00	-	1

Table E.18: Timetable Concept 1 - RE 2/8

Operating station	Arrival time	Departure time	Track
D-Stadt	-	09:54:00	1
E-Werk	09:59:07	09:59:07	2
E-Hausen	10:02:00	10:02:30	5
E-City	10:09:03	10:09:33	2
M-Heim	10:15:19	10:15:19	2
M-Berg	10:17:53	10:17:53	2
M-Dorf	10:24:39	-	3

Table E.19: Timetable Concept 1 - RB 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	-	06:23:10	4
ZSB-Berg	06:29:34	06:30:04	2
ZSB-Tal	06:37:29	-	2

Table E.20: Timetable Concept 1 - RB 2/2

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	06:38:29	2
ZSB-Berg	06:46:07	06:46:37	1
M-Dorf	06:53:59	-	4

Table E.21: Timetable Concept 1 - RB 1/4

Operating station	Arrival time	Departure time	Track
M-Dorf	-	07:23:53	4
ZSB-Berg	07:30:18	07:30:48	2
ZSB-Tal	07:38:13	-	2

Table E.22: Timetable Concept 1 - RB 2/4

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	07:39:13	2
ZSB-Berg	07:46:51	07:47:55	1
M-Dorf	07:55:17	-	4

Table E.23: Timetable Concept 1 - RB 1/6

Operating station	Arrival time	Departure time	Track
M-Dorf	-	08:23:10	4
ZSB-Berg	08:29:34	08:30:04	2
ZSB-Tal	08:37:29	-	2

Table E.24: Timetable Concept 1 - RB 2/6

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	08:38:29	2
ZSB-Berg	08:46:07	08:46:37	1
M-Dorf	08:53:59	-	4

Table E. 25: Timetable Concept 1 - RB 1/8

Operating station	Arrival time	Departure time	Track
M-Dorf	-	09:23:55	4
ZSB-Berg	09:30:20	09:30:50	2
ZSB-Tal	09:38:15	-	2

Table E.26: Timetable Concept 1 - RB 2/8

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	10:08:43	2
ZSB-Berg	10:16:21	10:16:21	1
M-Dorf	10:24:13	-	4

Table E.27: Timetable Concept 1 - LDF 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	06:48:16	06:48:16	2
M-Berg	06:55:20	06:55:20	1
M-Heim	06:58:29	06:58:29	1
E-City	07:06:02	07:06:02	3
E-Hausen	07:13:15	07:13:15	3
E-Werk	07:15:54	07:15:54	1
D-Stadt	07:22:40	07:22:40	1

Table E.28: Timetable Concept 1 - LDF 1/4

Operating station	Arrival time	Departure time	Track
M-Dorf	08:48:16	08:48:16	2
M-Berg	08:55:20	08:55:20	1
M-Heim	08:58:29	08:58:29	1
E-City	09:06:02	09:06:02	3
E-Hausen	09:13:15	09:13:15	3
E-Werk	09:15:54	09:15:54	1
D-Stadt	09:22:40	09:22:40	1

Table E.29: Timetable Concept 1 - LDF 2/2

Operating station	Arrival time	Departure time	Track
D-Stadt	07:37:17	07:37:17	3
E-Werk	07:43:33	07:43:33	2
E-Hausen	07:45:54	07:45:54	5
E-City	07:53:48	07:53:48	2
M-Heim	08:00:25	08:00:25	2
M-Berg	08:04:21	08:04:21	2
M-Dorf	08:11:25	08:11:25	1

Table E.30: Timetable Concept 1 - LDF 2/4

Operating station	Arrival time	Departure time	Track
D-Stadt	09:37:18	09:37:18	3
E-Werk	09:43:34	09:43:34	2
E-Hausen	09:45:55	09:45:55	5
E-City	09:53:49	09:53:49	2
M-Heim	10:00:26	10:00:26	2
M-Berg	10:04:22	10:04:22	2
M-Dorf	10:11:26	10:11:26	1

Table E.31: Timetable Concept 1 - RF 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	07:38:52	07:38:52	3
ZSB-Berg	07:47:21	07:47:21	2
ZSB-Tal	07:55:02	07:55:02	2

Table E.32: Timetable Concept 1 - RF 1/4

Operating station	Arrival time	Departure time	Track
M-Dorf	09:35:56	09:35:56	3
ZSB-Berg	09:43:24	09:43:24	2
ZSB-Tal	09:51:16	09:53:16	3

Table E.33: Timetable Concept 1 - RF 2/2

Operating station	Arrival time	Departure time	Track
ZSB-Tal	06:46:21	06:46:21	2
ZSB-Berg	06:54:12	06:58:25	1
M-Dorf	07:06:18	07:06:18	3

Table E.34: Timetable Concept 1 - RF 2/4

Operating station	Arrival time	Departure time	Track
ZSB-Tal	08:43:22	08:43:22	2
ZSB-Berg	08:54:12	08:58:25	1
M-Dorf	09:06:18	09:06:18	3

Timetable Concept 2

Table E.35: Timetable Concept 2 - ICE 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	06:25:05	06:25:05	2
M-Berg	06:29:33	06:29:33	1
M-Heim	06:31:13	06:31:13	1
E-City West	06:31:47	06:31:47	1
E-City	06:36:02	06:36:32	8
E-City Ost	06:40:54	06:40:54	1
E-Hausen	06:32:15	06:32:15	3
E-Werk	06:41:38	06:41:38	1
D-Stadt	06:46:26	06:46:56	1

Table E.36: Timetable Concept 2 - ICE 1/4

Operating station	Arrival time	Departure time	Track
M-Dorf	07:26:06	07:26:06	2
M-Berg	07:30:34	07:30:34	1
M-Heim	07:32:13	07:32:13	1
E-City West	07:32:47	07:32:47	1
E-City	07:37:03	07:37:33	8
E-City Ost	07:41:01	07:41:01	1
E-Hausen	07:41:54	07:41:54	3
E-Werk	07:42:38	07:42:38	1
D-Stadt	07:47:26	07:47:56	1

Table E.37: Timetable Concept 2 - ICE 1/6

Operating station	Arrival time	Departure time	Track
M-Dorf	08:26:44	08:26:44	2
M-Berg	08:31:12	08:31:12	1
M-Heim	08:32:51	08:32:51	1
E-City West	08:33:25	08:33:25	1
E-City	08:37:41	08:38:11	8
E-City Ost	08:41:39	08:41:39	1
E-Hausen	08:42:32	08:42:32	3
E-Werk	08:43:16	08:43:16	1
D-Stadt	08:48:04	08:48:34	1

Table E.38: Timetable Concept 2 - ICE 1/8

Operating station	Arrival time	Departure time	Track
M-Dorf	09:27:06	09:27:06	2
M-Berg	09:31:34	09:31:34	1
M-Heim	09:33:14	09:33:14	1
E-City West	09:33:48	09:33:48	1
E-City	09:38:03	09:39:57	8
E-City Ost	09:43:25	09:43:25	1
E-Hausen	09:44:18	09:44:18	3
E-Werk	09:45:03	09:45:03	1
D-Stadt	09:49:50	09:50:20	1

Table E.39: Timetable Concept 2 - ICE 2/2

Operating station	Arrival time	Departure time	Track
D-Stadt	06:20:51	06:21:21	3
E-Werk	06:25:57	06:25:57	2
E-Hausen	06:26:41	06:26:41	4
E-City Ost	06:27:32	06:27:32	2
E-City	06:31:13	06:31:43	5
E-City West	06:35:44	06:35:44	2
M-Heim	06:36:16	06:36:16	2
M-Berg	06:37:56	06:37:56	2
M-Dorf	06:42:24	06:42:24	1

Table E.40: Timetable Concept 2 - ICE 2/4

Operating station	Arrival time	Departure time	Track
D-Stadt	07:21:17	07:21:47	3
E-Werk	07:26:23	07:26:23	2
E-Hausen	07:27:07	07:27:07	4
E-City Ost	07:27:58	07:27:58	2
E-City	07:31:39	07:32:47	5
E-City West	07:36:47	07:36:47	2
M-Heim	07:37:19	07:37:19	2
M-Berg	07:38:59	07:38:59	2
M-Dorf	07:43:27	07:43:27	1

Table E.41: Timetable Concept 2 - ICE 2/6

Operating station	Arrival time	Departure time	Track
D-Stadt	08:20:42	08:21:12	3
E-Werk	08:25:48	08:25:48	2
E-Hausen	08:26:32	08:26:32	4
E-City Ost	08:27:23	08:27:23	2
E-City	08:31:04	08:36:54	5
E-City West	08:40:55	08:40:55	2
M-Heim	08:41:27	08:41:27	2
M-Berg	08:43:07	08:43:07	2
M-Dorf	08:47:34	08:47:34	1

Table E.42: Timetable Concept 2 - ICE 2/8

Operating station	Arrival time	Departure time	Track
D-Stadt	09:22:42	09:23:12	3
E-Werk	09:27:48	09:27:48	2
E-Hausen	09:28:32	09:28:32	4
E-City Ost	09:29:23	09:29:23	2
E-City	09:33:04	09:33:34	5
E-City West	09:37:35	09:37:35	2
M-Heim	09:38:07	09:38:07	2
M-Berg	09:39:47	09:39:47	2
M-Dorf	09:44:15	09:44:15	1

Table E.43: Timetable Concept 2 - IC 1/2

Operating station	Arrival time	Departure time	Track
E-City	-	06:40:51	4
E-Hausen	06:47:13	06:47:43	2
E-Werk	06:49:39	06:49:39	1
D-Stadt	06:54:40	-	1

Table E.44: Timetable Concept 2 - IC 2/2

Operating station	Arrival time	Departure time	Track
D-Stadt	-	06:55:40	1
E-Werk	07:00:59	07:00:59	2
E-Hausen	07:03:53	07:04:23	5
E-City	07:11:45	-	4

Table E.45: Timetable Concept 2 - IC 1/4

Operating station	Arrival time	Departure time	Track
E-City	-	07:12:45	4
E-Hausen	07:19:07	07:19:37	2
E-Werk	07:21:32	07:21:32	1
D-Stadt	07:26:34	-	1

Table E.46: Timetable Concept 2 - IC 2/4

Operating station	Arrival time	Departure time	Track
D-Stadt	-	07:27:34	1
E-Werk	07:32:53	07:32:53	2
E-Hausen	07:35:47	07:36:17	5
E-City	07:43:22	-	4

Table E.47: Timetable Concept 2 - IC 1/6

Operating station	Arrival time	Departure time	Track
E-City	-	07:44:22	4
E-Hausen	07:50:09	07:50:39	2
E-Werk	07:52:35	07:52:35	1
D-Stadt	07:57:37	-	1

Table E.48: Timetable Concept 2 - IC 2/6

Operating station	Arrival time	Departure time	Track
D-Stadt	-	07:58:37	1
E-Werk	08:03:56	08:03:56	2
E-Hausen	08:06:49	08:07:19	5
E-City	08:13:27	-	4

Table E.49: Timetable Concept 2 - IC 1/8

Operating station	Arrival time	Departure time	Track
E-City	-	08:14:27	4
E-Hausen	08:20:01	08:20:31	2
E-Werk	08:22:27	08:22:27	1
D-Stadt	08:27:28	-	1

Table E.50: Timetable Concept 2 - IC 2/8

Operating station	Arrival time	Departure time	Track
D-Stadt	-	08:28:28	1
E-Werk	08:33:47	08:33:47	2
E-Hausen	08:36:41	08:37:11	5
E-City	08:43:42	-	4

Table E.51: Timetable Concept 2 - IC 1/10

Operating station	Arrival time	Departure time	Track
E-City	-	08:44:42	4
E-Hausen	08:51:03	08:51:33	2
E-Werk	08:53:29	08:53:29	1
D-Stadt	08:58:31	-	1

Table E.52: Timetable Concept 2 - IC 2/10

Operating station	Arrival time	Departure time	Track
D-Stadt	-	08:59:31	1
E-Werk	09:04:50	09:04:50	2
E-Hausen	09:07:43	09:08:13	5
E-City	09:14:34	-	4

Table E.53: Timetable Concept 2 - IC 1/12

Operating station	Arrival time	Departure time	Track
E-City	-	09:15:35	4
E-Hausen	09:20:53	09:21:23	2
E-Werk	09:23:18	09:23:18	1
D-Stadt	09:28:20	-	1

Table E.54: Timetable Concept 2 - IC 2/12

Operating station	Arrival time	Departure time	Track
D-Stadt	-	09:31:37	1
E-Werk	09:36:56	09:36:56	2
E-Hausen	09:39:49	09:40:19	5
E-City	09:47:41	-	4

Table E.55: Timetable Concept 2 - RE 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	-	06:30:23	3
M-Berg	06:39:26	06:39:26	1
M-Heim	06:43:06	06:43:06	1
E-City	06:53:25	06:53:55	3
E-Hausen	06:59:58	07:00:28	2
E-Werk	07:02:15	07:02:15	1
D-Stadt	07:07:12	-	1

Table E.56: Timetable Concept 2 - RE 2/2

Operating station	Arrival time	Departure time	Track
D-Stadt	-	07:08:11	1
E-Werk	07:13:18	07:13:18	2
E-Hausen	07:16:11	07:16:41	5
E-City	07:22:09	07:22:39	2
M-Heim	07:28:25	07:28:25	2
M-Berg	07:30:58	07:30:58	2
M-Dorf	07:37:45	-	3

Table E.57: Timetable Concept 2 - RE 1/4

Operating station	Arrival time	Departure time	Track
M-Dorf	-	07:38:45	3
M-Berg	07:44:22	07:44:22	1
M-Heim	07:46:39	07:46:39	1
E-City	07:53:03	07:53:33	3
E-Hausen	07:59:36	08:00:06	2
E-Werk	08:01:53	08:01:53	1
D-Stadt	08:06:50	-	1

Table E.58: Timetable Concept 2 - RE 2/4

Operating station	Arrival time	Departure time	Track
D-Stadt	-	08:14:19	1
E-Werk	08:19:12	08:19:12	2
E-Hausen	08:21:59	08:22:29	5
E-City	08:28:30	08:29:01	2
M-Heim	08:34:46	08:34:46	2
M-Berg	08:37:20	08:37:20	2
M-Dorf	08:44:06	-	3

Table E.59: Timetable Concept 2 - RE 1/6

Operating station	Arrival time	Departure time	Track
M-Dorf	-	08:45:07	3
M-Berg	08:50:44	08:50:44	1
M-Heim	08:53:01	08:53:01	1
E-City	08:59:26	08:59:56	3
E-Hausen	09:04:58	09:05:28	2
E-Werk	09:07:07	09:07:07	1
D-Stadt	09:11:40	-	1

Table E.60: Timetable Concept 2 - RE 2/6

Operating station	Arrival time	Departure time	Track
D-Stadt	-	09:12:40	1
E-Werk	09:17:26	09:17:26	2
E-Hausen	09:20:08	09:20:38	5
E-City	09:26:06	09:26:36	2
M-Heim	09:31:58	09:31:58	2
M-Berg	09:34:21	09:34:21	2
M-Dorf	09:40:39	-	3

Table E.61: Timetable Concept 2 - RE_2 1/2

Operating station	Arrival time	Departure time	Track
D-Stadt	-	06:29:54	4
E-Werk	06:35:01	06:35:01	2
E-Hausen	06:37:54	06:38:24	5
E-City	06:45:17	06:45:51	2
M-Heim	06:51:36	06:51:36	2
M-Berg	06:54:10	06:54:10	2
M-Dorf	07:00:56	-	3

Table E.62: Timetable Concept 2 - RE_2 2/2

Operating station	Arrival time	Departure time	Track
M-Dorf	-	07:01:57	3
M-Berg	07:07:34	07:07:34	1
M-Heim	07:09:51	07:09:51	1
E-City	07:09:51	07:18:44	3
E-Hausen	07:24:46	07:26:13	2
E-Werk	07:28:00	07:28:00	1
D-Stadt	07:32:57	-	1

Table E.63: Timetable Concept 2 - RE_2 1/4

Operating station	Arrival time	Departure time	Track
D-Stadt	-	07:34:54	1
E-Werk	07:40:00	07:40:00	2
E-Hausen	07:42:53	07:43:23	5
E-City	07:48:51	07:49:21	2
M-Heim	07:55:07	07:55:07	2
M-Berg	07:57:40	07:57:40	2
M-Dorf	08:04:27	-	3

Table E.64: Timetable Concept 2 - RE_2 2/4

Operating station	Arrival time	Departure time	Track
M-Dorf	-	08:05:27	3
M-Berg	08:11:04	08:11:04	1
M-Heim	08:13:21	08:13:21	1
E-City	08:19:45	08:20:15	3
E-Hausen	08:26:18	08:27:08	2
E-Werk	08:28:55	08:28:55	1
D-Stadt	08:33:51	-	1

Table E.65: Timetable Concept 2 - RE_2 1/6

Operating station	Arrival time	Departure time	Track
D-Stadt	-	08:35:48	1
E-Werk	08:40:12	08:40:12	2
E-Hausen	08:42:42	08:43:12	5
E-City	08:48:40	08:49:10	2
M-Heim	08:53:58	08:53:58	2
M-Berg	08:56:06	08:56:06	2
M-Dorf	09:01:44	-	3

Table E.66: Timetable Concept 2 - RE_2 2/6

Operating station	Arrival time	Departure time	Track
M-Dorf	-	09:02:44	3
M-Berg	09:08:21	09:08:21	1
M-Heim	09:10:38	09:10:38	1
E-City	09:17:02	09:20:38	3
E-Hausen	09:26:41	09:27:58	2
E-Werk	09:29:28	09:29:28	1
D-Stadt	09:33:35	-	1

Table E.67: Timetable Concept 2 - RB 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	-	06:21:31	4
ZSB-Berg	06:27:55	06:28:25	2
ZSB-Tal	06:35:51	-	2

Table E.68: Timetable Concept 2 - RB 2/2

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	06:36:50	2
ZSB-Berg	06:44:28	06:44:58	1
M-Dorf	06:52:20	-	4

Table E.69: Timetable Concept 2 - RB 1/4

Operating station	Arrival time	Departure time	Track
M-Dorf	-	06:53:20	4
ZSB-Berg	06:59:44	07:06:44	2
ZSB-Tal	07:14:10	-	2

Table E.70: Timetable Concept 2 - RB 2/4

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	07:15:10	2
ZSB-Berg	07:22:47	07:23:45	1
M-Dorf	07:29:54	-	4

Table E.71: Timetable Concept 2 - RB 1/6

Operating station	Arrival time	Departure time	Track
M-Dorf	-	07:30:54	4
ZSB-Berg	07:37:18	07:38:29	2
ZSB-Tal	07:44:40	-	2

Table E.72: Timetable Concept 2 - RB 2/6

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	07:45:39	2
ZSB-Berg	07:52:07	07:53:13	1
M-Dorf	08:00:35	-	4

Table E.73: Timetable Concept 2 - RB 1/8

Operating station	Arrival time	Departure time	Track
M-Dorf	-	08:01:35	4
ZSB-Berg	08:07:59	08:08:29	2
ZSB-Tal	08:15:54	-	2

Table E.74: Timetable Concept 2 - RB 2/8

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	08:16:53	2
ZSB-Berg	08:24:31	08:26:08	1
M-Dorf	08:33:30	-	4

Table E.75: Timetable Concept 2 - RB 1/10

Operating station	Arrival time	Departure time	Track
M-Dorf	-	08:34:30	4
ZSB-Berg	08:40:54	08:41:24	2
ZSB-Tal	08:48:49	-	2

Table E.76: Timetable Concept 2 - RB 2/10

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	08:48:49	2
ZSB-Berg	08:56:27	08:59:05	1
M-Dorf	09:06:27	-	4

Table E.77: Timetable Concept 2 - RB 1/12

Operating station	Arrival time	Departure time	Track
M-Dorf	-	09:07:27	4
ZSB-Berg	09:13:51	09:14:21	2
ZSB-Tal	09:21:46	-	2

Table E.78: Timetable Concept 2 - RB 2/12

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	09:23:25	2
ZSB-Berg	09:31:03	09:31:33	1
M-Dorf	09:38:55	-	4

Table E.79: Timetable Concept 2 - LDF 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	07:42:56	07:45:08	2
M-Berg	07:54:14	07:54:14	1
M-Heim	07:57:22	07:57:22	1
E-City	08:04:51	08:04:51	3
E-Hausen	08:10:56	08:10:56	2
E-Werk	08:13:24	08:13:24	1
D-Stadt	08:20:08	08:20:08	1

Table E.80: Timetable Concept 2 - LDF 1/4

Operating station	Arrival time	Departure time	Track
M-Dorf	09:10:40	09:10:40	2
M-Berg	09:17:44	09:17:44	1
M-Heim	09:20:54	09:20:54	1
E-City	09:28:27	09:28:27	3
E-Hausen	09:35:40	09:35:40	3
E-Werk	09:37:52	09:37:52	1
D-Stadt	09:43:31	09:43:31	1

Table E.81: Timetable Concept 2 - LDF 2/2

Operating station	Arrival time	Departure time	Track
D-Stadt	06:37:37	06:37:37	3
E-Werk	06:43:54	06:43:54	2
E-Hausen	06:46:14	06:46:14	5
E-City	06:54:08	06:54:08	2
M-Heim	07:00:46	07:00:46	2
M-Berg	07:04:41	07:04:41	2
M-Dorf	07:11:46	07:11:46	1

Table E.82: Timetable Concept 2 - LDF 2/4

Operating station	Arrival time	Departure time	Track
D-Stadt	08:06:42	08:06:42	3
E-Werk	08:12:15	08:12:15	2
E-Hausen	08:14:20	08:14:20	5
E-City	08:21:21	08:21:21	2
M-Heim	08:27:13	08:27:13	2
M-Berg	08:30:41	08:30:41	2
M-Dorf	08:36:58	08:36:58	1

Table E.83: Timetable Concept 2 - LDF_2 1/2

Operating station	Arrival time	Departure time	Track
E-City	-	06:14:29	1
M-Heim	06:23:11	06:23:11	2
M-Berg	06:27:02	06:27:02	2
M-Dorf	06:36:22	-	3

Table E.84: Timetable Concept 2 - LDF_2 2/2

Operating station	Arrival time	Departure time	Track
M-Dorf	-	06:42:10	3
M-Berg	06:51:22	06:51:22	1
M-Heim	06:54:32	06:54:32	1
E-City	07:02:00	07:02:00	3
E-Hausen	07:09:13	07:09:13	3
E-Werk	07:11:49	07:11:49	1
D-Stadt	07:20:38	-	2

Table E.85: Timetable Concept 2 - LDF_2 1/4

Operating station	Arrival time	Departure time	Track
D-Stadt	-	07:44:59	2
E-Werk	07:53:31	07:53:31	2
E-Hausen	07:55:53	07:55:53	5
E-City	08:03:50	08:03:50	2
M-Heim	08:10:32	08:10:32	2
M-Berg	08:14:26	08:14:26	2
M-Dorf	08:23:53	-	3

Table E.86: Timetable Concept 2 - LDF_2 2/4

Operating station	Arrival time	Departure time	Track
M-Dorf	-	08:31:58	3
M-Berg	08:41:08	08:41:08	1
M-Heim	08:44:17	08:44:17	1
E-City	08:51:44	08:51:44	3
E-Hausen	08:59:52	09:11:25	3
E-Werk	09:14:17	09:14:17	1
D-Stadt	09:23:05	-	2

Table E.87: Timetable Concept 2 - RF 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	08:18:03	08:18:03	3
ZSB-Berg	08:25:35	08:25:35	2
ZSB-Tal	08:32:21	08:32:21	2

Table E.88: Timetable Concept 2 - RF 2/2

Operating station	Arrival time	Departure time	Track
ZSB-Tal	06:47:49	06:47:49	2
ZSB-Berg	06:55:40	06:59:50	1
M-Dorf	07:07:43	07:07:43	3

Table E.89: Timetable Concept 2 - RF 2/4

Operating station	Arrival time	Departure time	Track
ZSB-Tal	08:57:41	08:57:41	2
ZSB-Berg	09:05:32	09:13:31	1
M-Dorf	09:21:24	09:21:24	3

Table E.90: Timetable Concept 2 - RF_2 1/2

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	06:55:44	1
ZSB-Berg	07:05:53	07:07:53	1
M-Dorf	07:16:47	-	3

Table E.91: Timetable Concept 2 - RF_2 2/2

Operating station	Arrival time	Departure time	Track
M-Dorf	-	07:16:49	3
ZSB-Berg	07:23:12	07:23:12	2
ZSB-Tal	07:29:45	-	1

Table E.92: Timetable Concept 2 - RF_2 1/4

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	07:29:47	1
ZSB-Berg	07:37:55	07:37:55	1
M-Dorf	07:44:59	-	4

Table E.93: Timetable Concept 2 - RF_2 2/4

Operating station	Arrival time	Departure time	Track
M-Dorf	-	07:45:01	3
ZSB-Berg	07:52:39	07:52:39	2
ZSB-Tal	08:00:31	-	1

Timetable Concept 3

Table E.94: Timetable Concept 3 - ICE 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	06:02:35	06:02:35	2
M-Berg	06:06:29	06:06:29	1
M-Heim	06:07:56	06:07:56	1
E-City West	06:08:26	06:08:26	1
E-City	06:12:10	06:16:55	8
E-City Ost	06:19:57	06:19:57	1
E-Hausen	06:20:41	06:20:41	3
E-Werk	06:21:18	06:21:18	1
D-Stadt	06:25:18	06:25:48	1

Table E.95: Timetable Concept 3 - ICE 1/4

Operating station	Arrival time	Departure time	Track
M-Dorf	07:03:17	07:03:17	2
M-Berg	07:07:11	07:07:11	1
M-Heim	07:08:38	07:08:38	1
E-City West	07:09:08	07:09:08	1
E-City	07:12:52	07:18:34	8
E-City Ost	07:21:36	07:21:36	1
E-Hausen	07:22:20	07:22:20	3
E-Werk	07:22:57	07:22:57	1
D-Stadt	07:26:57	07:27:27	1

Table E.96: Timetable Concept 3 - ICE 2/2

Operating station	Arrival time	Departure time	Track
D-Stadt	06:09:08	06:11:08	3
E-Werk	06:15:10	06:15:10	2
E-Hausen	06:15:48	06:15:48	4
E-City Ost	06:16:33	06:16:33	2
E-City	06:19:47	06:21:47	5
E-City West	06:25:17	06:25:17	2
M-Heim	06:25:45	06:25:45	2
M-Berg	06:27:13	06:27:13	2
M-Dorf	06:31:07	06:31:07	1

Table E.97: Timetable Concept 3 - ICE 2/4

Operating station	Arrival time	Departure time	Track
D-Stadt	07:10:29	07:12:29	3
E-Werk	07:16:31	07:16:31	2
E-Hausen	07:17:10	07:17:10	4
E-City Ost	07:17:54	07:17:54	2
E-City	07:21:08	07:23:08	5
E-City West	07:26:38	07:26:38	2
M-Heim	07:27:07	07:27:07	2
M-Berg	07:28:34	07:28:34	2
M-Dorf	07:32:28	07:32:28	1

Table E.98: Timetable Concept 3 - IC 1/2

Operating station	Arrival time	Departure time	Track
E-City	-	06:01:30	4
E-Hausen	06:07:04	06:07:34	2
E-Werk	06:09:15	06:09:15	1
D-Stadt	06:14:50	-	2

Table E.99: Timetable Concept 3 - IC 2/2

Operating station	Arrival time	Departure time	Track
D-Stadt	-	06:16:30	2
E-Werk	06:21:09	06:21:09	2
E-Hausen	06:23:41	06:24:11	5
E-City	06:30:37	-	4

Table E.100: Timetable Concept 3 - IC 1/4

Operating station	Arrival time	Departure time	Track
E-City	-	06:32:30	4
E-Hausen	06:38:04	06:38:34	2
E-Werk	06:40:15	06:40:15	1
D-Stadt	06:45:50	-	2

Table E.101: Timetable Concept 3 - IC 2/4

Operating station	Arrival time	Departure time	Track
D-Stadt	-	06:47:30	2
E-Werk	06:52:09	06:52:09	2
E-Hausen	06:54:33	06:55:03	5
E-City	07:01:12	-	4

Table E.102: Timetable Concept 3 - IC 1/6

Operating station	Arrival time	Departure time	Track
E-City	-	07:02:34	4
E-Hausen	07:08:08	07:08:38	2
E-Werk	07:10:19	07:10:19	1
D-Stadt	07:15:54	-	2

Table E.103: Timetable Concept 3 - IC 2/6

Operating station	Arrival time	Departure time	Track
D-Stadt	-	07:18:30	2
E-Werk	07:23:09	07:23:09	2
E-Hausen	07:25:41	07:26:11	5
E-City	07:32:37	-	4

Table E.104: Timetable Concept 3 - IC 1/8

Operating station	Arrival time	Departure time	Track
E-City	-	07:34:30	4
E-Hausen	07:40:04	07:40:34	2
E-Werk	07:42:15	07:42:15	1
D-Stadt	07:47:50	-	2

Table E.105: Timetable Concept 3 - IC 2/8

Operating station	Arrival time	Departure time	Track
D-Stadt	-	07:49:30	2
E-Werk	07:54:09	07:54:09	2
E-Hausen	07:56:41	07:57:11	5
E-City	08:03:37	-	4

Table E.106: Timetable Concept 3 - RE 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	-	06:07:37	3
M-Berg	06:12:32	06:12:32	1
M-Heim	06:14:32	06:14:32	1
E-City	06:20:08	06:20:38	3
E-Hausen	06:25:55	06:26:25	2
E-Werk	06:27:59	06:27:59	1
D-Stadt	06:32:19	-	1

Table E.107: Timetable Concept 3 - RE 2/2

Operating station	Arrival time	Departure time	Track
D-Stadt	-	06:37:43	1
E-Werk	06:42:11	06:42:11	2
E-Hausen	06:44:42	06:45:12	5
E-City	06:50:56	06:54:00	2
M-Heim	06:59:03	06:59:03	2
M-Berg	07:01:17	07:01:17	2
M-Dorf	07:07:13	-	3

Table E.108: Timetable Concept 3 - RE 1/4

Operating station	Arrival time	Departure time	Track
M-Dorf	-	07:11:33	3
M-Berg	07:16:28	07:16:28	1
M-Heim	07:18:27	07:18:27	1
E-City	07:24:04	07:24:34	3
E-Hausen	07:29:51	07:30:21	2
E-Werk	07:31:55	07:31:55	1
D-Stadt	07:36:14	-	1

Table E.109: Timetable Concept 3 - RE 2/4

Operating station	Arrival time	Departure time	Track
D-Stadt	-	07:40:17	1
E-Werk	07:44:45	07:44:45	2
E-Hausen	07:47:16	07:47:46	5
E-City	07:53:30	-	2

Table E.110: Timetable Concept 3 - RE_2 1/2

Operating station	Arrival time	Departure time	Track
D-Stadt	-	06:26:00	4
E-Werk	06:30:28	06:30:28	2
E-Hausen	06:32:52	06:33:22	5
E-City	06:38:50	06:39:20	2
M-Heim	06:44:08	06:44:08	2
M-Berg	06:46:16	06:46:16	2
M-Dorf	06:51:55	-	3

Table E.111: Timetable Concept 3 - RE_2 2/2

Operating station	Arrival time	Departure time	Track
M-Dorf	-	06:52:55	3
M-Berg	06:57:36	06:57:36	1
M-Heim	06:59:30	06:59:30	1
E-City	07:04:50	07:08:05	3
E-Hausen	07:13:08	07:15:31	2
E-Werk	07:17:02	07:17:02	1
D-Stadt	07:22:17	-	2

Table E.112: Timetable Concept 3 - RE_2 1/4

Operating station	Arrival time	Departure time	Track
D-Stadt	-	07:33:47	2
E-Werk	07:38:15	07:38:15	2
E-Hausen	07:40:46	07:41:16	5
E-City	07:47:00	07:47:30	2
M-Heim	07:52:33	07:52:33	2
M-Berg	07:54:47	07:54:47	2
M-Dorf	08:00:43	-	3

Table E.113: Timetable Concept 3 - RB 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	-	06:11:00	4
ZSB-Berg	06:16:20	06:16:50	2
ZSB-Tal	06:23:13	-	3

Table E.114: Timetable Concept 3 - RB 2/2

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	06:24:51	3
ZSB-Berg	06:31:17	06:31:47	1
M-Dorf	06:37:55	-	4

Table E.115: Timetable Concept 3 - RB 1/4

Operating station	Arrival time	Departure time	Track
M-Dorf	-	06:39:56	4
ZSB-Berg	06:45:16	06:45:46	2
ZSB-Tal	06:52:09	-	3

Table E.116: Timetable Concept 3 - RB 2/4

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	06:56:09	3
ZSB-Berg	07:02:35	07:06:20	1
M-Dorf	07:12:28	-	4

Table E.117: Timetable Concept 3 - RB 1/6

Operating station	Arrival time	Departure time	Track
M-Dorf	-	07:13:48	4
ZSB-Berg	07:19:08	07:19:38	2
ZSB-Tal	07:26:01	-	3

Table E.118: Timetable Concept 3 - RB 2/6

Operating station	Arrival time	Departure time	Track
ZSB-Tal	-	07:27:21	3
ZSB-Berg	07:33:47	07:34:17	1
M-Dorf	07:40:25	-	4

Table E.119: Timetable Concept 3 - RB 1/8

Operating station	Arrival time	Departure time	Track
M-Dorf	-	07:41:43	4
ZSB-Berg	07:47:03	07:47:33	2
ZSB-Tal	07:53:56	-	3

Table E.120: Timetable Concept 3 - LDF 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	06:25:54	06:25:54	2
M-Berg	06:32:05	06:32:05	1
M-Heim	06:34:23	06:34:23	1
E-City West	06:35:13	06:35:13	1
E-City	06:41:33	06:43:33	7
E-City Ost	06:49:50	06:49:50	1
E-Hausen	06:51:04	06:51:04	3
E-Werk	06:52:06	06:52:06	1
D-Stadt	06:57:35	06:57:35	1

Table E.121: Timetable Concept 3 - LDF 1/4

Operating station	Arrival time	Departure time	Track
M-Dorf	07:25:54	07:25:54	2
M-Berg	07:32:05	07:32:05	1
M-Heim	07:34:23	07:34:23	1
E-City West	07:35:13	07:35:13	1
E-City	07:41:33	07:43:33	7
E-City Ost	07:49:50	07:49:50	1
E-Hausen	07:51:04	07:51:04	3
E-Werk	07:52:06	07:52:06	1
D-Stadt	07:57:35	07:57:35	1

Table E.122: Timetable Concept 3 - LDF 2/2

Operating station	Arrival time	Departure time	Track
D-Stadt	06:44:35	06:54:06	4
E-Werk	07:01:08	07:01:08	2
E-Hausen	07:02:06	07:02:06	4
E-City Ost	07:03:16	07:03:16	2
E-City	07:07:20	07:07:20	6
E-City West	07:12:34	07:12:34	2
M-Heim	07:13:21	07:13:21	2
M-Berg	07:15:33	07:15:33	2
M-Dorf	07:21:27	07:21:27	1

Table E.123: Timetable Concept 3 - RF 1/2

Operating station	Arrival time	Departure time	Track
M-Dorf	06:58:56	06:58:56	3
ZSB-Berg	07:05:31	07:05:31	2
ZSB-Tal	07:11:27	07:11:27	2

Table E.124: Timetable Concept 3 - RF 2/2

Operating station	Arrival time	Departure time	Track
ZSB-Tal	06:15:38	06:34:50	2
ZSB-Berg	06:43:43	06:47:06	1
M-Dorf	06:54:48	06:54:48	4

Table E.125: Timetable Concept 3 - RF 2/4

Operating station	Arrival time	Departure time	Track
ZSB-Tal	07:19:57	07:36:26	2
ZSB-Berg	07:45:20	07:47:59	1
M-Dorf	07:55:41	07:55:41	4

Appendix F: Input Delay Data for the Case Study

The following tables show the input delays used in the case study. Primary delays are assumed to occur at scheduled and operating stops only. If a train run does not enter the dispatching area or does not stop in an operating station, the respective cell does not contain a delay. For visualization purposes, the train runs are divided into two tables for each test case. The first table contains the train runs along the station E-City and the second table contains the train runs along the one-directional line segment via the stations ZSB-Berg and ZSB-Tal.

Timetable Concept 1

Test case 1

Table F.1: Input delays for Timetable Concept 1 - test case 1 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	1.95	-	0	-	0
ICE 1/4	0	-	0	-	0
ICE 2/2	1.66	-	0	-	0
ICE 2/4	3.54	-	0	-	2.43
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	0	0	-
RE 2/4	-	-	0	0	0
RE 1/6	-	0	0	0	-
RE 2/6	-	-	0	0	0
RE 1/8	-	0.97	0	0	-
RE 2/8	-	-	0	0	0
LDF 1/2	18.45	-	-	-	-
LDF 1/4	27.12	-	-	-	-
LDF 2/2	0	-	-	-	-
LDF 2/4	0	-	-	-	-

Table F.2: Input delays for Timetable Concept 1 - test case 1 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0.23
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RB 2/8	-	-	1.49	0
RF 1/2	51.09	-	-	0
RF 1/4	11.03	-	-	0
RF 2/2	0	-	3.11	-
RF 2/4	0	-	0	-

Test case 2

Table F.3: Input delays for Timetable Concept 1 - test case 2 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	0	-	0	-	0
ICE 1/4	5.58	-	6.55	-	0
ICE 2/2	4.23	-	0	-	0
ICE 2/4	1.93	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	1.95
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	0	0	-
RE 2/4	-	-	0	0	0
RE 1/6	-	0	0	0	-
RE 2/6	-	-	0	0	0
RE 1/8	-	0	0	0	-
RE 2/8	-	-	0	0	0
LDF 1/2	3.61	-	-	-	-
LDF 1/4	0	-	-	-	-
LDF 2/2	0.16	-	-	-	-
LDF 2/4	0	-	-	-	-

Table F.4: Input delays for Timetable Concept 1 - test case 2 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0
RB 1/4	-	1.31	0	-
RB 2/4	-	-	1.52	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RB 2/8	-	-	0	0
RF 1/2	0	-	-	0
RF 1/4	8.32	-	-	0
RF 2/2	13.80	-	0	-
RF 2/4	12.53	-	0	-

Test case 3

Table F.5: Input delays for Timetable Concept 1 - test case 3 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	3.82	-	1.98	-	0
ICE 1/4	5.49	-	0	-	0
ICE 2/2	0	-	0	-	0
ICE 2/4	5.32	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0.90
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	0	0.40	-
RE 2/4	-	-	0.71	0	0
RE 1/6	-	0	0	0	-
RE 2/6	-	-	0	0	0.05
RE 1/8	-	0	0	0	-
RE 2/8	-	-	0	0	0
LDF 1/2	0	-	-	-	-
LDF 1/4	0	-	-	-	-
LDF 2/2	0	-	-	-	-
LDF 2/4	14.50	-	-	-	-

Table F.6: Input delays for Timetable Concept 1 - test case 3 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0.40	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RB 2/8	-	-	0	0.96
RF 1/2	10.42	-	-	0
RF 1/4	2.63	-	-	0
RF 2/2	7.09	-	0	-
RF 2/4	0	-	0	-

Test case 4

Table F.7: Input delays for Timetable Concept 1 - test case 4 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	11.18	-	0	-	0
ICE 1/4	13.45	-	0	-	0
ICE 2/2	3.13	-	0	-	0
ICE 2/4	1.41	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	0.15	0	-
RE 2/4	-	-	0	0	0
RE 1/6	-	0	0.89	0	-
RE 2/6	-	-	0	0	0
RE 1/8	-	0	0	0	-
RE 2/8	-	-	0	0	0
LDF 1/2	0	-	-	-	-
LDF 1/4	5.03	-	-	-	-
LDF 2/2	17.56	-	-	-	-
LDF 2/4	4.18	-	-	-	-

Table F.8: Input delays for Timetable Concept 1 - test case 4 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0.86	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0.18	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RB 2/8	-	-	0	0
RF 1/2	2.13	-	-	0
RF 1/4	19.03	-	-	0
RF 2/2	2.66	-	0	-
RF 2/4	1.69	-	0	-

Test case 5

Table F.9: Input delays for Timetable Concept 1 - test case 5 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	0.88	-	0	-	0
ICE 1/4	4.02	-	0	-	0
ICE 2/2	0	-	0	-	0
ICE 2/4	1.13	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	0	0	-
RE 2/4	-	-	0	0	0
RE 1/6	-	0	0	0	-
RE 2/6	-	-	0	0	0
RE 1/8	-	0	0	0	-
RE 2/8	-	-	0	0	0
LDF 1/2	27.16	-	-	-	-
LDF 1/4	1.20	-	-	-	-
LDF 2/2	3.41	-	-	-	-
LDF 2/4	2.73	-	-	-	-

Table F.10: Input delays for Timetable Concept 1 - test case 5 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RB 2/8	-	-	0	0
RF 1/2	6.29	-	-	0
RF 1/4	19.62	-	-	0
RF 2/2	3.98	-	0	-
RF 2/4	10.88	-	0	-

Timetable Concept 2

Test case 1

Table F.11: Input delays for Timetable Concept 2 - test case 1 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	0	-	0	-	2.08
ICE 1/4	0	-	1.33	-	0
ICE 1/6	4.90	-	0	-	0
ICE 1/8	2.33	-	0	-	0
ICE 2/2	0	-	0	-	0
ICE 2/4	0	-	0	-	0
ICE 2/6	4.92	-	0	-	0
ICE 2/8	0	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	6.08	0	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	1.05
IC 1/8	-	-	0	0	-
IC 2/8	-	-	-	0	0
IC 1/10	-	-	0	0	-
IC 2/10	-	-	-	3.32	0
IC 1/12	-	-	0	0	-
IC 2/12	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	0	0	-
RE 2/4	-	-	0	0.30	0
RE 1/6	-	0	0	0	-
RE 2/6	-	-	0.29	0	0
RE_2 1/2	-	-	0	0	0
RE_2 2/2	-	1.81	0	0	-
RE_2 1/4	-	-	0	0	0
RE_2 2/4	-	0	0	0	-
RE_2 1/6	-	-	0	0	0.31
RE_2 2/6	-	0	0	0	-
LDF 1/2	12.64	-	-	-	-
LDF 1/4	25.52	-	-	-	-
LDF 2/2	0	-	-	-	-
LDF 2/4	6.47	-	-	-	-
LDF_2 1/2	-	-	0	-	-
LDF_2 2/2	-	7.17	-	-	-
LDF_2 1/4	-	-	-	-	0
LDF_2 2/4	-	6.02	-	3.85	-

Table F.12: Input delays for Timetable Concept 2 - test case 1 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0.10	-
RB 2/8	-	-	0	0.06
RB 1/10	-	0	0	-
RB 2/10	-	-	0	0
RB 1/12	-	0	0	-
RB 2/12	-	-	0	0
RF 1/2	1.79	-	-	0
RF 2/2	0.91	-	0	-
RF 2/4	0	-	0	-
RF_2 1/2	-	-	0	0
RF_2 2/2	-	0	0	-
RF_2 1/4	-	-	0	0
RF_2 2/4	-	0	0	-

Test case 2

Table F.13: Input delays for Timetable Concept 2 - test case 2 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	5	-	0	-	0
ICE 1/4	6.40	-	0	-	0
ICE 1/6	0	-	0	-	0
ICE 1/8	0	-	0	-	0
ICE 2/2	2.18	-	0	-	0
ICE 2/4	0	-	5.14	-	0
ICE 2/6	0	-	0	-	0
ICE 2/8	2.81	-	0	-	0
IC 1/2	-	-	0	1.28	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	0.15	0	-
IC 2/6	-	-	-	0	0.01
IC 1/8	-	-	0	0	-
IC 2/8	-	-	-	0	0
IC 1/10	-	-	2.21	0	-
IC 2/10	-	-	-	0	0
IC 1/12	-	-	0	2.33	-
IC 2/12	-	-	-	0	0
RE 1/2	-	0	0	0.87	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	0	0	-
RE 2/4	-	-	0	0	0
RE 1/6	-	0	0	0	-
RE 2/6	-	-	1.50	0.13	0
RE_2 1/2	-	-	0	0	0
RE_2 2/2	-	0	0	0	-
RE_2 1/4	-	-	0	0	0
RE_2 2/4	-	0	0	0	-
RE_2 1/6	-	-	0	0	0
RE_2 2/6	-	0	0	0	-
LDF 1/2	0	-	-	-	-
LDF 1/4	0	-	-	-	-
LDF 2/2	1.45	-	-	-	-
LDF 2/4	3.28	-	-	-	-
LDF_2 1/2	-	-	0	-	-
LDF_2 2/2	-	0	-	-	-
LDF_2 1/4	-	-	-	-	0
LDF_2 2/4	-	0	-	0	-

Table F.14: Input delays for Timetable Concept 2 - test case 2 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RB 2/8	-	-	0	0
RB 1/10	-	0	0	-
RB 2/10	-	-	0	0
RB 1/12	-	0	0	-
RB 2/12	-	-	0	0
RF 1/2	0	-	-	0
RF 2/2	9.26	-	0	-
RF 2/4	14.03	-	0	-
RF_2 1/2	-	-	0	0
RF_2 2/2	-	0	0	-
RF_2 1/4	-	-	0	0
RF_2 2/4	-	0	0	-

Test case 3

Table F.15: Input delays for Timetable Concept 2 - test case 3 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	0	-	0	-	0
ICE 1/4	0	-	0	-	0
ICE 1/6	19.88	-	0	-	0
ICE 1/8	0	-	0	-	0
ICE 2/2	0	-	0	-	0
ICE 2/4	1.91	-	0	-	0
ICE 2/6	3.50	-	0	-	0
ICE 2/8	2.89	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	1.82	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	0	3.39	-
IC 2/6	-	-	-	0	0
IC 1/8	-	-	0	0	-
IC 2/8	-	-	-	0	0
IC 1/10	-	-	0	0	-
IC 2/10	-	-	-	0	0
IC 1/12	-	-	0	0	-
IC 2/12	-	-	-	0	0
RE 1/2	-	0	0	3.44	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	0	0	-
RE 2/4	-	-	0	0	0.14
RE 1/6	-	0	0	0	-
RE 2/6	-	-	0	0	0
RE_2 1/2	-	-	10	0	0
RE_2 2/2	-	0	0	0	-
RE_2 1/4	-	-	0	0	0
RE_2 2/4	-	0	0	0	-
RE_2 1/6	-	-	0	0	0
RE_2 2/6	-	0	0	0.05	-
LDF 1/2	2.44	-	-	-	-
LDF 1/4	3.10	-	-	-	-
LDF 2/2	5.47	-	-	-	-
LDF 2/4	0	-	-	-	-
LDF_2 1/2	-	-	0	-	-
LDF_2 2/2	-	0	-	-	-
LDF_2 1/4	-	-	-	-	0
LDF_2 2/4	-	0	-	0	-

Table F.16: Input delays for Timetable Concept 2 - test case 3 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0.98
RB 1/6	-	0.78	0.56	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RB 2/8	-	-	0	0
RB 1/10	-	0	0	-
RB 2/10	-	-	0	0
RB 1/12	-	0	0	-
RB 2/12	-	-	0	0
RF 1/2	0.56	-	-	0
RF 2/2	0	-	0	-
RF 2/4	0	-	0	-
RF_2 1/2	-	-	0	0
RF_2 2/2	-	0	0	-
RF_2 1/4	-	-	0	0
RF_2 2/4	-	0	0	-

Test case 4

Table F.17: Input delays for Timetable Concept 2 - test case 4 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	0	-	0	-	0
ICE 1/4	0	-	0	-	0
ICE 1/6	0.49	-	0	-	0.07
ICE 1/8	1.65	-	0	-	0
ICE 2/2	0.08	-	0	-	0
ICE 2/4	0	-	0	-	0
ICE 2/6	13.68	-	0	-	0
ICE 2/8	0	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	0	2.57	-
IC 2/6	-	-	-	0	0.03
IC 1/8	-	-	0	0	-
IC 2/8	-	-	-	0.71	0
IC 1/10	-	-	0	0	-
IC 2/10	-	-	-	0	0
IC 1/12	-	-	0	0	-
IC 2/12	-	-	-	0	1.56
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	1.23	1.05
RE 1/4	-	0	0	0	-
RE 2/4	-	-	0	0	0
RE 1/6	-	0	0	0	-
RE 2/6	-	-	0	0	0
RE_2 1/2	-	-	0	0	0
RE_2 2/2	-	0	0	0	-
RE_2 1/4	-	-	0.09	0	0
RE_2 2/4	-	0	0	0	-
RE_2 1/6	-	-	0.12	0	0
RE_2 2/6	-	0	0	0	-
LDF 1/2	25.22	-	-	-	-
LDF 1/4	24.47	-	-	-	-
LDF 2/2	9.46	-	-	-	-
LDF 2/4	4.60	-	-	-	-
LDF_2 1/2	-	-	0	-	-
LDF_2 2/2	-	14	-	-	-
LDF_2 1/4	-	-	-	-	0
LDF_2 2/4	-	0	-	0	-

Table F.18: Input delays for Timetable Concept 2 - test case 4 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0.66	0.22
RB 1/8	-	0	0	-
RB 2/8	-	-	0	0
RB 1/10	-	0	0.25	-
RB 2/10	-	-	0	0.74
RB 1/12	-	0	0	-
RB 2/12	-	-	0	0
RF 1/2	4.83	-	-	0
RF 2/2	0	-	0	-
RF 2/4	3.84	-	0	-
RF_2 1/2	-	-	0	0
RF_2 2/2	-	0	0	-
RF_2 1/4	-	-	0	0
RF_2 2/4	-	0	0	-

Test case 5

Table F.19: Input delays for Timetable Concept 2 - test case 5 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	0	-	8	-	0
ICE 1/4	3.01	-	0	-	0
ICE 1/6	0	-	0	-	0
ICE 1/8	0	-	0	-	4.10
ICE 2/2	0	-	0.15	-	0
ICE 2/4	0	-	0	-	0
ICE 2/6	18.71	-	0	-	0
ICE 2/8	0	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	1.58
IC 1/8	-	-	0	0	-
IC 2/8	-	-	-	0	0
IC 1/10	-	-	0	0	-
IC 2/10	-	-	-	0	0
IC 1/12	-	-	0	0	-
IC 2/12	-	-	-	1.67	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0.52	0
RE 1/4	-	0	0	0	-
RE 2/4	-	-	0	0	0
RE 1/6	-	0	0	0	-
RE 2/6	-	-	0	0	0
RE_2 1/2	-	-	0	0.44	0
RE_2 2/2	-	0	0	0	-
RE_2 1/4	-	-	0	0	0
RE_2 2/4	-	0	0	0	-
RE_2 1/6	-	-	0	0	0
RE_2 2/6	-	0	0	0	-
LDF 1/2	0	-	-	-	-
LDF 1/4	29.84	-	-	-	-
LDF 2/2	0	-	-	-	-
LDF 2/4	0	-	-	-	-
LDF_2 1/2	-	-	0	-	-
LDF_2 2/2	-	0	-	-	-
LDF_2 1/4	-	-	-	-	0
LDF_2 2/4	-	0	-	0	-

Table F.20: Input delays for Timetable Concept 2 - test case 5 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RB 2/8	-	-	0	0
RB 1/10	-	0	0	-
RB 2/10	-	-	0	0
RB 1/12	-	0	0	-
RB 2/12	-	-	0	0
RF 1/2	40.26	-	-	0
RF 2/2	8.50	-	0	-
RF 2/4	1.02	-	0	-
RF_2 1/2	-	-	0	0
RF_2 2/2	-	0	0	-
RF_2 1/4	-	-	0	0
RF_2 2/4	-	0	0	-

Timetable Concept 3

Test case 1

Table F.21: Input delays for Timetable Concept 3 - test case 1 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	31.06	-	0	-	0
ICE 1/4	7.60	-	0	-	0
ICE 2/2	0	-	0	-	0
ICE 2/4	4.30	-	1.00	-	0
IC 1/2	-	-	27.74	0	-
IC 2/2	-	-	-	0.15	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	1.90	0	-
IC 2/6	-	-	-	0	0
IC 1/8	-	-	0	0	-
IC 2/8	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	0	0	-
RE 2/4	-	-	-	0	0
RE_2 1/2	-	-	0	0	0
RE_2 2/2	-	12	0	0	-
RE_2 1/4	-	-	0	0	0
LDF 1/2	0	-	0	-	-
LDF 1/4	27.74	-	0	-	-
LDF 2/2	0	-	-	-	0

Table F.22: Input delays for Timetable Concept 3 - test case 1 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0.20	-
RB 2/4	-	-	0	0.08
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RF 1/2	3.71	-	-	0
RF 2/2	0	-	0	0
RF 2/4	0	-	0	0

Test case 2

Table F.23: Input delays for Timetable Concept 3 - test case 2 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	3.67	-	0	-	0
ICE 1/4	0.44	-	0	-	0
ICE 2/2	0	-	0	-	0
ICE 2/4	0	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	1.20
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	0
IC 1/8	-	-	0	0	-
IC 2/8	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	2.00
RE 1/4	-	9.40	0	0	-
RE 2/4	-	-	-	0	0
RE_2 1/2	-	-	0	1.71	0
RE_2 2/2	-	0	0	11.55	-
RE_2 1/4	-	-	0	0	0
LDF 1/2	0.27	-	0	-	-
LDF 1/4	4.74	-	0	-	-
LDF 2/2	0	-	-	-	0

Table F.24: Input delays for Timetable Concept 3 - test case 2 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RF 1/2	14.96	-	-	0
RF 2/2	9.88	-	0	0
RF 2/4	12.83	-	0	0

Test case 3

Table F.25: Input delays for Timetable Concept 3 - test case 3 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	3.35	-	0	-	0
ICE 1/4	2.69	-	0	-	0
ICE 2/2	0	-	0	-	0
ICE 2/4	11.39	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0.86
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	0
IC 1/8	-	-	0	0	-
IC 2/8	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	1.05	0	-
RE 2/4	-	-	-	0.27	0
RE_2 1/2	-	-	0	0	0
RE_2 2/2	-	0	0	0	-
RE_2 1/4	-	-	0	0	0
LDF 1/2	7.91	-	0	-	-
LDF 1/4	23.20	-	0.96	-	-
LDF 2/2	29.26	-	-	-	0

Table F.26: Input delays for Timetable Concept 3 - test case 3 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RF 1/2	5.77	-	-	0
RF 2/2	32.06	-	0	0
RF 2/4	9.08	-	0	0

Test case 4

Table F.27: Input delays for Timetable Concept 3 - test case 4 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	0	-	0.48	-	0
ICE 1/4	0	-	0	-	3.61
ICE 2/2	2.55	-	0	-	0
ICE 2/4	18.51	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	3.59	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0.68
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	0
IC 1/8	-	-	0.60	0	-
IC 2/8	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	0	0	-
RE 2/4	-	-	-	0	0
RE_2 1/2	-	-	0	0	0
RE_2 2/2	-	0	0	0	-
RE_2 1/4	-	-	0	0	0.71
LDF 1/2	2.99	-	0	-	-
LDF 1/4	0	-	0	-	-
LDF 2/2	2.00	-	-	-	0

Table F.28: Input delays for Timetable Concept 3 - test case 4 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RF 1/2	0.66	-	-	0
RF 2/2	4.48	-	2.26	0
RF 2/4	2.56	-	0	0

Test case 5

Table F.29: Input delays for Timetable Concept 3 - test case 5 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	0	-	0	-	0
ICE 1/4	0	-	1.07	-	0
ICE 2/2	0	-	0	-	0
ICE 2/4	0	-	0	-	0
IC 1/2	-	-	0	2.65	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0.33
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	0
IC 1/8	-	-	0	0	-
IC 2/8	-	-	-	3.23	0
RE 1/2	-	0	0.48	0	-
RE 2/2	-	-	0	0.25	0
RE 1/4	-	0	0	0	-
RE 2/4	-	-	-	0	1.83
RE_2 1/2	-	-	0	0	0
RE_2 2/2	-	0	0	0	-
RE_2 1/4	-	-	0	0	0
LDF 1/2	16.04	-	0	-	-
LDF 1/4	0	-	3.70	-	-
LDF 2/2	0	-	-	-	3.12

Table F.30: Input delays for Timetable Concept 3 - test case 5 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0.26	0.84
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0.18
RB 1/6	-	0	0.05	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RF 1/2	26.96	-	-	0
RF 2/2	16.13	-	0	0
RF 2/4	0	-	0	6.54

Test case 6

Table F.31: Input delays for Timetable Concept 3 - test case 6 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	0	-	0	-	0
ICE 1/4	0	-	0	-	0
ICE 2/2	0	-	0	-	0
ICE 2/4	0	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0.19	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	0
IC 1/8	-	-	0	2.70	-
IC 2/8	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	0	0	-
RE 2/4	-	-	-	0	0
RE_2 1/2	-	-	0	0	0
RE_2 2/2	-	0	0	0	-
RE_2 1/4	-	-	0	0	0
LDF 1/2	1.27	-	0	-	-
LDF 1/4	0	-	0	-	-
LDF 2/2	9.46	-	-	-	0

Table F.32: Input delays for Timetable Concept 3 - test case 6 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0.32	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RF 1/2	20.12	-	-	0
RF 2/2	31.56	-	0	0
RF 2/4	0	-	0	0

Test case 7

Table F.33: Input delays for Timetable Concept 3 - test case 7 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	3.45	-	0	-	0
ICE 1/4	0	-	0	-	2.96
ICE 2/2	0	-	0	-	0
ICE 2/4	4.06	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	0
IC 1/8	-	-	0	0	-
IC 2/8	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	0	0	-
RE 2/4	-	-	-	0	0
RE_2 1/2	-	-	0.25	0	0
RE_2 2/2	-	0	0	2.49	-
RE_2 1/4	-	-	0	0	0
LDF 1/2	0.24	-	0	-	-
LDF 1/4	0	-	0	-	-
LDF 2/2	6.72	-	-	-	0

Table F.34: Input delays for Timetable Concept 3 - test case 7 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0.78	0
RB 1/4	-	0.02	0	-
RB 2/4	-	-	0	0
RB 1/6	-	0.24	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0	-
RF 1/2	6.22	-	-	0
RF 2/2	0	-	0	0
RF 2/4	11.92	-	0	0

Test case 8

Table F.35: Input delays for Timetable Concept 3 - test case 8 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	5.03	-	0	-	0
ICE 1/4	3.27	-	0	-	0
ICE 2/2	0.01	-	0	-	0
ICE 2/4	3.59	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0.05	0
IC 1/8	-	-	0	0	-
IC 2/8	-	-	-	1.14	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0	0	0	-
RE 2/4	-	-	-	0	0
RE_2 1/2	-	-	0	1.23	0
RE_2 2/2	-	0	0	0	-
RE_2 1/4	-	-	0	0	0
LDF 1/2	9.88	-	0	-	-
LDF 1/4	0	-	0	-	-
LDF 2/2	10.58	-	-	-	0

Table F.36: Input delays for Timetable Concept 3 - test case 8 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0.48	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0
RB 1/6	-	0.03	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0.05	-
RF 1/2	0	-	-	0
RF 2/2	19.23	-	0	0
RF 2/4	0	-	0	0

Test case 9

Table F.37: Input delays for Timetable Concept 3 - test case 9 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	33.30	-	0	-	0
ICE 1/4	2.00	-	0	-	0
ICE 2/2	0	-	0	-	0
ICE 2/4	0	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	0	0
IC 1/4	-	-	0	1.71	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	0
IC 1/8	-	-	0	0	-
IC 2/8	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0.49
RE 1/4	-	0	0	0	-
RE 2/4	-	-	-	0	0
RE_2 1/2	-	-	0	0	0
RE_2 2/2	-	0	0	0	-
RE_2 1/4	-	-	0	0	0
LDF 1/2	0	-	0	-	-
LDF 1/4	8.49	-	0	-	-
LDF 2/2	1.24	-	-	-	0

Table F.38: Input delays for Timetable Concept 3 - test case 9 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0.49	0	-
RB 2/2	-	-	0.20	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0.28	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0	0
RB 1/8	-	0	0.20	-
RF 1/2	21.86	-	-	0
RF 2/2	0.13	-	0	0
RF 2/4	0	-	0	0

Test case 10

Table F.39: Input delays for Timetable Concept 3 - test case 10 (I)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay E-City [min]	Primary delay E-Hausen [min]	Primary delay D-Stadt [min]
ICE 1/2	2.02	-	0	-	0
ICE 1/4	2.86	-	0	-	0
ICE 2/2	0.67	-	0	-	0
ICE 2/4	0	-	0	-	0
IC 1/2	-	-	0	0	-
IC 2/2	-	-	-	5.73	0
IC 1/4	-	-	0	0	-
IC 2/4	-	-	-	0	0
IC 1/6	-	-	0	0	-
IC 2/6	-	-	-	0	0
IC 1/8	-	-	0	0	-
IC 2/8	-	-	-	0	0
RE 1/2	-	0	0	0	-
RE 2/2	-	-	0	0	0
RE 1/4	-	0.69	2.03	3.23	-
RE 2/4	-	-	-	0	0
RE_2 1/2	-	-	0	0	0
RE_2 2/2	-	0	0	0	-
RE_2 1/4	-	-	0	0	0
LDF 1/2	0	-	0	-	-
LDF 1/4	37.60	-	0	-	-
LDF 2/2	0	-	-	-	0

Table F.40: Input delays for Timetable Concept 3 - test case 10 (II)

Train number	Delay at entry [min]	Primary delay M-Dorf [min]	Primary delay ZSB-Berg [min]	Primary delay ZSB-Tal [min]
RB 1/2	-	0	0	-
RB 2/2	-	-	0	0
RB 1/4	-	0	0	-
RB 2/4	-	-	0	0
RB 1/6	-	0	0	-
RB 2/6	-	-	0.63	0
RB 1/8	-	0	0	-
RF 1/2	0.36	-	-	0
RF 2/2	0	-	0	0
RF 2/4	0	-	0	0