Transportation Research Part B 141 (2020) 72–97

Contents lists available at ScienceDirect

Transportation Research Part B

journal homepage: www.elsevier.com/locate/trb

Decomposition and distributed optimization of real-time traffic management for large-scale railway networks

Xiaojie Luan[a,b], Bart De Schutter[c], Lingyun Meng[d,*], Francesco Corman[a]

[a] Institute for Transport Planning and Systems (IVT), ETH Zürich, Stefano-Franscini-Platz 5, 8093 Zürich, Switzerland
[b] Section Transport Engineering and Logistics, Delft University of Technology, 2628 CD Delft, the Netherlands
[c] Delft Center for Systems and Control, Delft University of Technology, 2628 CD Delft, the Netherlands
[d] School of traffic and transportation, Beijing Jiaotong University, 100044, Beijing, China

ARTICLE INFO

Article history:
Received 15 January 2020
Revised 16 July 2020
Accepted 6 September 2020
Available online 17 September 2020

Keywords:
Real-time traffic management
Decomposition
Distributed optimization
Large-scale railway network

ABSTRACT

This paper introduces decomposition and distributed optimization approaches for the real-time railway traffic management problem considering microscopic infrastructure characteristics, aiming at an improved computational efficiency when tackling large-scale railway networks.

Based on the nature of the railway traffic management problem, we consider three decomposition methods, namely a geography-based (GEO) decomposition, a train-based (TRA) decomposition, and a time-interval-based (TIN) decomposition, in order to partition the large railway traffic management optimization problem into several subproblems. In particular, an integer linear programming (ILP) model is developed to generate the optimal GEO solution, with the objectives of minimizing the number of interconnections among regions and of balancing the size of regions. The decomposition creates couplings among the subproblems, in terms of either capacity usage or transit time consistency; therefore the whole problem gets a non-separable structure. To handle the couplings, we introduce three distributed optimization approaches, namely an Alternating Direction Method of Multipliers (ADMM) algorithm, a priority-rule-based (PR) algorithm, and a Cooperative Distributed Robust Safe But Knowledgeable (CDRSBK) algorithm, which operate iteratively.

We test all combinations of the three decomposition methods and the three distributed optimization algorithms on a large-scale railway network in the South-East of the Netherlands, in terms of feasibility, computational efficiency, and optimality. Overall the CDRSBK algorithm with the TRA decomposition performs best, where high-quality (optimal or near-optimal) solutions can be found within 10 s of computation time.

1. Introduction

Real-time traffic management is of great importance to recover train operations from disturbances that occur in real time. In the presence of disturbances, where the pre-defined train timetables become infeasible, train dispatchers are in charge of adjusting the infeasible train timetables by means of taking proper control measures, e.g., re-timing, re-ordering,

* Corresponding author.
E-mail addresses: Xiaojie.Luan@ivt.baug.ethz.ch (X. Luan), B.DeSchutter@tudelft.nl (B. De Schutter), lymeng@bjtu.edu.cn (L. Meng), francesco.corman@ivt.baug.ethz.ch (F. Corman).

https://doi.org/10.1016/j.trb.2020.09.004
0191-2615/

and re-routing, so as to recover from infeasibility and to reduce negative consequences (e.g., delays). This problem is known as the real-time traffic management problem. Due to its real-time nature, a solution is required in a very short computation time for resolving train conflicts as quickly as possible, recovering delays and avoiding cancellations, for the benefit of all passengers.

In railway operations, the real-time traffic management problem is mostly tackled manually by train dispatchers; so the results of their decisions are generally sub-optimal. Effective and efficient algorithms are desired to support the decision making of dispatchers, in order to implement intelligent traffic management when disturbances occur. The real-time railway traffic management problem has been studied extensively. In the literature, many optimization approaches are available for solving the problem to optimality (see Cacchiani et al., 2014). These approaches consider either macroscopic or microscopic infrastructure details. For small-scale instances, they mostly have excellent performance, where optimality can be achieved in a short computation time. For large-scale instances, we can still solve the problem with the macroscopic approaches, as the problem complexity is reduced by accepting a lower precision at the local level (e.g., stations and segments). The microscopic approaches generate solutions with high precision by including all necessary details; however, it is difficult to solve large-scale instances, due to memory and computation time limits. In general, the computation time for finding an optimal solution increases exponentially when enlarging the scale of the problem instances, which limits their application in the practice.

Distributed optimization has gained a lot of attention to face the need of fast and efficient solutions for problems arising in the context of large-scale networks, such as utility maximization problems (e.g., for electric power systems studied by Evangelopoulos et al., 2016; Molzahn et al., 2017). The main idea of distributed optimization is solving the problems either serially or in parallel to jointly minimize a separable objective function, usually subject to complicating constraints that force the subproblems to exchange information during the optimization process (see Palomar and Chiang, 2006, for more details). In the literature, distributed optimization approaches have been widely studied in many fields, e.g., for controlling road traffic (Findler and Stapp, 1992), for managing air traffic (Wangermann and Stengel, 1996), and for rescheduling railway traffic (Kersbergen et al., 2016). Kersbergen et al. (2016) performed a geography-based decomposition and proposed distributed model predictive control method to address the railway traffic management problem for the Dutch national railway network with macroscopic infrastructure characteristics.

This paper extends those ideas for the much more complex case of microscopic characteristics of railway infrastructure, with an improved computational efficiency. Our model considers a time-continuous approach, where routes, orders, and departure and arrival times of the trains are optimized. We can write the problem as a mixed-integer linear program (MILP), as detailed in Appendix A. For the sake of compactness, the MILP problem is represented in a general form:

$$\min_{\lambda} \quad \mathcal{Z}(\lambda) = c^{\top} \cdot \lambda, \tag{1a}$$

$$\text{s.t.} \quad A \cdot \lambda \leq b \tag{1b}$$

with variable $\lambda \in \mathbb{R}^n$, matrix $A \in \mathbb{R}^{m \times n}$, and vectors $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$.

We consider disturbed traffic conditions, i.e., relatively small perturbations, rather than disruptions resulting in blockages and cancellations.

This work focuses on proposing novel models and solution algorithms, to enable a very short computation time for solving the mathematical problem (1), for large real-life cases where decision-making assistance is needed the most. To this end, we introduce three decomposition methods, namely a geography-based (GEO) decomposition, a train-based (TRA) decomposition, and a time-interval-based (TIN) decomposition, which exploit different structures based on either geographic extent, time extent, or individual train services as component of the different subproblems. No matter which decomposition method is used, couplings always exist among subproblems, i.e., when two subproblems are representing the same situation, such as a train crossing regions (GEO), time intervals (TIN), or due to the competitive use of infrastructure by trains (TRA).

These couplings lead to a non-separable structure of the whole optimization problem, which is solved by one of three optimization approaches. An Alternating Direction Method of Multipliers (ADMM) algorithm solves each problem through iterative coordination with the other subproblems. A priority-rule-based (PR) algorithm is introduced to solve subproblems sequentially and iteratively in a priority order (according to delays) with respect to the solutions of the other subproblems with a higher priority that have been solved previously. Finally, a Cooperative Distributed Robust Safe But Knowledgeable (CDRSBK) algorithm defines four types of couplings, considered either as constraints or to be variables in the iterative solution process.

We conduct experiments to determine how the different decomposition and algorithms perform in terms of solution quality and computation efficiency, also compared to a centralized approach. A large-scale railway network in the South-East of the Netherlands with more than 150 trains is our test case. The best combination, i.e., the TRA decomposition and the CDRSBK algorithm, is able to find a high-quality (optimal or near-optimal) solution within just 10 s of computation time.

The remainder of this paper is organized as follows. Section 2 reviews the literature on the real-time railway traffic management problem. Section 3 introduces three decomposition methods, where several subproblems with couplings are obtained. In Section 4, three distributed optimization approaches are developed for handling the couplings among the resulting subproblems. Section 5 discusses the implementation of the proposed decomposition and distributed optimization approaches. Section 6 evaluates the performance by means of experiments on the large-scale Dutch railway network. Finally, the conclusions and suggestions for future research are given in Section 7.

2. Literature review

This section gives an overview of the literature on real-time railway traffic management. We restrict the focus on mathematical optimization problem formulations in Section 2.1 and on decomposition and solution algorithms that have been developed for algorithmically solving the real-time railway traffic management problem in Section 2.2. For more information, we refer to the review papers by Cacchiani et al. (2014), Corman and Meng (2015), and Fang et al. (2015) and to the book by Hansen and Pachl (2014).

2.1. Centralized optimization models of real-time railway traffic management

Mathematical optimization can provide better use of capacity, improved efficiency, reduced infrastructure and operating costs, improved reliability, and more punctuality (Borndörfer et al., 2018). Several approaches based on operations research techniques have been developed to formulate the railway network topology (infrastructure), traffic situation, and traffic constraints, e.g., the alternative-graph based formulation (D'Ariano et al., 2007), the time-continuous formulation (Mannino, 2011), the time-indexed formulation (Meng and Zhou, 2014), and the event-activity based formulation (Schöbel, 2007).

D'Ariano et al. (2007) proposed an alternative-graph based model, which uses a combination of job shop scheduling and alternative graph techniques. In the alternative graph, each node represents an operation, i.e., the traveling of a train through a block section. An arc between two connected nodes represents the passing of a train through two adjacent block sections. Conflicting requests of trains are modeled by pairs of alternative arcs, where processing orders and headways of the conflicting train movements are determined. The alternative-graph based formulation was extended to deal with multiple classes of running traffic (Corman et al., 2011a); to determine solutions to the bi-objective problem of reducing delays while maintaining as many passenger connections as possible (Corman et al., 2012a); to investigate the impact of the levels of detail towards solution quality and computational efficiency (Kecman et al., 2013); and to reschedule high-speed traffic based on a quasi-moving block system, optimizing also speed, braking, and headway aspects (Xu et al., 2017).

Many studies employed the time-continuous formulation to build optimization models. In this formulation, continuous variables are used for representing the times that trains arrive at or depart from the visited block sections (or stations at a macroscopic level). Törnquist and Persson (2007) presented an MIP model that takes into account retiming, reordering, and rerouting of trains with the goal of minimizing the consequences of a single disturbance. Mannino (2011) proposed an MILP model for solving the traffic management problem on single and double track railways. Boccia et al. (2013) presented an MILP model, where train rerouting is described by using binary variables to indicate whether or not a route is assigned to a train. An MILP model was developed by Pellegrini et al. (2014), modeling the infrastructure in terms of electrical circuits to reliably detect the presence of trains on a particular track section. Luan et al. (2017a) developed an MILP model to optimize train orders, routes, and departure and arrival times at passing stations, with additional constraints and the objective of ensuring delay equity among competing trains and operators. Luan et al. (2018) considered a nonlinear programming problem and an MILP problem, with the focus on including speed control into train rescheduling, requiring additional continuous and binary variables to describe train speeds and their influence on train movements.

In the time-indexed formulation, the time horizon is discretized into small time periods, say 1 second, 10 seconds, or 1 minute. The time period size has a large influence on both solution accuracy and computational efficiency: a larger time period size corresponds to a smaller mathematical model/problem and consequently faster computation, but the solution may be sub-optimal or even infeasible. The schedule of a train on a block section (or at a station) is modeled by a set of binary variables, each associated to a feasible departure or arrival time period, to be selected. Capacity constraints can be expressed by simple inequalities but generally they require a larger number of variables and constraints, growing quickly with the time horizon and the number of steps in the time period. The problem easily becomes intractable when solving real-time and large-scale problems, i.e., we cannot directly solve time-indexed models for real-time and large-scale problems. As an alternative, the time-indexed formulation is often transformed into a time-space network based formulation, solved by using for instance shortest path algorithm, where decomposition is usually applied, see e.g., Meng and Zhou (2014). Harrod (2011) proposed an integer programming description based on the time-indexed formulation (hypergraph formulation), which can ensure the operational feasibility of the generated train paths (i.e., the deadlock problem that may happen when considering macroscopic infrastructure details, is addressed). Caimi et al. (2012) proposed a discrete-time control system to manage traffic by retiming and rerouting of trains as well as partial speed profile coordination. Meng and Zhou (2014) proposed an MILP problem extending the time-indexed formulation by cumulative flow binary variables, where the benefits of simultaneous train rerouting and rescheduling were investigated. Their work was further extended by Luan et al. (2017b), where an MILP description was developed with the aim of scheduling trains and planning maintenance tasks at the same time.

The event-activity based formulation was proposed by Schöbel (2007) to tackle the delay management problem, which consists in deciding whether train connections should be maintained or be dropped with the goal of minimizing the total passenger delay. An event-activity network is a graph where train arrival and departure events are all described by nodes, and activities corresponding to waiting or running of trains and transferring of passengers are described by arcs. A solution is obtained by assigning a time to each event, such that the minimal duration needed for performing each activity is satisfied, for instance by an integer programming approach. This event-activity based problem was further extended to address a discrete time/cost trade-off problem of maintaining service quality and reducing passengers' inconve-

nience (Ginkel and Schöbel, 2007); and by including headways and capacity constraints (Schachtebeck and Schöbel, 2010). Dollevoet et al. (2012) presented an event-activity based description to address the problem of rerouting passengers in the delay management process. Besides the variables and constraints used in Schöbel (2007), additional binary variables and constraints are used to model passenger rerouting decisions. Zhan et al. (2015) employed the event-activity network to reschedule the operations when a segment of a high-speed railway was totally blocked without considering rerouting, aiming to minimize the number of canceled and delayed trains. This stream of studies considers only macroscopic infrastructure characteristics; thus they are able to handle relatively large-scale instances. However, with the inclusion of other aspects, e.g., passenger rerouting, the model complexity increases, causing computation burden for large-scale instances.

2.2. Decomposition and solution algorithm of real-time railway traffic management

The approaches introduced in Section 2.1 are centralized optimizations. Their performance decreases for large-scale instances, since computation time and memory requirements increase exponentially as the system grows in size with more trains, tracks, and stations. Moreover, the NP-completeness of the railway scheduling problem has been proved by Strotmann (2008), which implies that it is very unlikely to solve the railway scheduling problem in polynomial time. Decomposition approaches have been proposed to avoid this exponential increase, computing fast a solution of the real-time railway traffic management problem.

In many countries, the large national railway network is partitioned into several local dispatching areas, and railway traffic is hierarchically organized into two decision levels. A lower level manages traffic flows in each individual dispatching area without any knowledge about the traffic flows of other areas; an upper level coordinates the traffic management over several dispatching areas with a global vision of the traffic flows, focusing on the big picture, and taking decisions at the border between areas. Among the heuristic approaches proposed in the literature (see e.g., D'Ariano et al., 2007; Schachtebeck and Schöbel, 2010; Pellegrini et al., 2015; Samà et al., 2017; Van Thielen et al., 2018), some studies investigated decomposition methods or considered the multi-area railway traffic management problem. A two-level approach for rescheduling trains between multiple dispatching areas was proposed by Strotmann (2008). At a lower level, local solutions were computed in each area by greedy heuristic scheduling procedures while, at a higher level, a coordinator was used to check whether neighboring areas have consistent solutions. The coordination procedure imposed train ordering constraints at borders between areas with an iterative approach until a feasible schedule to the global problem was found or the procedure failed in finding a globally feasible schedule. Corman et al. (2010) proposed a distributed system to control trains running in a large-scale railway network divided into dispatching areas. A coordination framework combined microscopic modeling of train movements at the local level with an aggregate view of the situation at the global level. Corman et al. (2012b) studied the problem of coordinating several dispatchers of different dispatching areas with the objective of driving their behavior towards globally optimal solutions, considered as a bi-level optimization problem. New heuristic algorithms were further presented by Corman et al. (2014) for both local dispatching and global coordination. Based on the coordination procedure introduced by Corman et al. (2012b), a pre-processing phase was introduced to increase the chances of reaching a globally feasible solution, by identifying precedences among trains that must be kept in all feasible schedules. Kersbergen et al. (2016) partitioned the railway network into regions and proposed a distributed model predictive control method to address the railway traffic management problem for a large-scale railway network with macroscopic infrastructure characteristics.

Lamorgese and Mannino (2015) proposed a Benders' decomposition to split the initial problem in two MILP subproblems, i.e., a master and a slave, corresponding to a macroscopic and a microscopic representation of the railway infrastructure, communicating through suitable feasibility cuts in the variables of the master subproblem. The solutions of the master and slave subproblems were embedded into a heuristic iterative mechanism by Lamorgese and Mannino (2015), and an exact approach was further developed by Lamorgese et al. (2016) with performance improvement. Dollevoet et al. (2014) introduced an iterative optimization approach to jointly solve a macroscopic delay management model derived from Dollevoet et al. (2012) and a microscopic train scheduling model derived from D'Ariano et al. (2007). Both the macroscopic and microscopic problems are formulated as MILP problems, and they are solved iteratively.

Brännlund et al. (1998) performed a decomposition of the train timetabling problem into individual trains and used a Lagrangian relaxation method to coordinate the solutions of the resulting single-train subproblems. A similar idea can be found in Meng and Zhou (2014), focusing on the traffic management problem and with the inclusion of train rerouting, solved by a time-dependent shortest-path algorithm.

Rodriguez (2000) has shown that a temporal decomposition can be used to compute suitable solutions in considerably less computation time compared to the global solution of the problem. D'Ariano and Pranzo (2009) decomposed a long time horizon into tractable time intervals and used the ROMA dispatching system to globally solve conflicts on each time interval, coordinating the speed of each train across time intervals. A rolling-horizon solution approach was applied by Zhan et al. (2016) for the real-time rescheduling of trains in case of a partial segment blockage. The uncertainty of disruption was handled by re-optimizing each time new information regarding the disruption was available. Effectively this is a temporal decomposition of the problem, focusing on only the events that occur in the current time horizon with a certain length.

To conclude, when focusing on the decomposition methods, we can classify them into four groups: (1) Geography-based (GEO) decomposition, where a large network is partitioned into several dispatching regions, see e.g., D'Ariano and Pranzo (2009), Corman et al. (2010), and Kersbergen et al. (2016); (2) Train-based (TRA) decomposition, where an initial

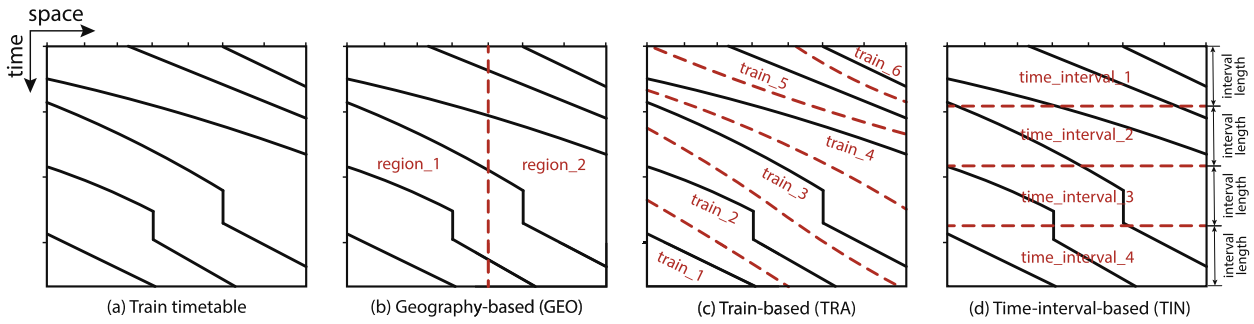


Fig. 1. Illustration of the three decomposition methods in a time-space graph.

large problem is split into several single-train subproblems, see e.g., Brännlund et al. (1998) and Meng and Zhou (2014); (3) Time-interval-based (TIN) decomposition, where a large problem is divided into several subproblems according to a given time-interval size (length of planning horizon), see e.g., Rodriguez (2000); Zhan et al. (2016); (4) Macro-micro (MM) decomposition, where an initial complex problem is decomposed into a master subproblem and a slave subproblem, corresponding to macroscopic and microscopic infrastructure characteristics, see e.g., Lamorgese and Mannino (2015) and Lamorgese et al. (2016). In fact, the MM decomposition is somehow similar to the GEO decomposition, with different interpretations. Both of them manage traffic independently in either local areas or the slave subproblems (station areas) and coordinate the areas/subproblems through a high level controller (a global coordinator or a master subproblem). Table 1 summarizes some relevant studies of the decomposition and solution algorithms for the real-time traffic management problem, in terms of problem description (i.e., the level of infrastructure detail and control measure), decomposition method, solution algorithm (for solving each subproblem), distributed solution algorithm (for coordinating subproblems), and the scale of test case.

2.3. Contributions of the paper

As reviewed in Section, great efforts have been put on generating high-quality (optimal) solutions, seeking for fast computation of a solution. For speeding up the computation, researchers have either accepted low precision (i.e., focusing on macroscopic characteristics only or using a larger time period size in the time-indexed formulation), developed fast solution approaches (for solving the centralized traffic management problem), or performed decomposition and then developed distributed solution algorithms (for addressing the coordination issue). Low precision may cause infeasibility issues when executing the obtained solutions in practice. The fast solution approaches most often improve the computational efficiency for small problems, but still have troubles for solving large-scale problems.

In this paper, we achieve global feasibility and high-quality (often optimal) solutions by (i) reducing the problem scale (decomposition) to several small-scale subproblems that are easier to handle, and (ii) solving them in a coordinated way (distributed optimization) by including key aspects of the different subproblems while solving them individually, in an iterative manner. The main contributions are summarized as follows:

- We tackle the large-scale traffic management problem microscopically by means of decomposition and distributed optimization, avoiding sacrificing solution quality when pursuing fast computation.
- Based on the nature of the traffic management problem, we exploit extensively three decomposition methods (based on geography, trains, and time intervals respectively), which enable us to partition a large-scale problem into several small-scale subproblems, reducing the overall computation burden.
- Three distributed solution algorithms (distributed optimization) are presented, successfully communicating among the subproblems, to solve them in a coordinated way.
- A thorough assessment of the centralized optimization and the proposed decomposition and distributed optimization approaches for managing traffic on a large-scale railway network is presented.
- The overall excellent performance of the TRA decomposition with the use of the CDRSBK algorithm is demonstrated, where high-quality (optimal or near-optimal) solutions can be found within 10 s of computation time.

3. Problem decomposition

This section introduces three decomposition methods for splitting a large traffic management problem into several small subproblems, i.e., the geography-based (GEO), the train-based (TRA), and the time-interval-based (TIN) decomposition, described in Sections 3.1–3.3 respectively. Section 3.4 discusses the consequence of the problem decomposition, i.e., subproblems and couplings (interactions). Fig. 1 conceptually illustrates the three decomposition methods in a time-space graph, where black lines indicate train paths and red dashed lines indicate how a problem is decomposed.

Table 1
Summary of decomposition and solution algorithms.

Publication	Infrastructure detail	Control measure	Decomposition method	Solution algorithm	Distributed solution algorithm	Test case
Strotmann (2008)	micro	Rt, Ro	GEO	G, H (PR)	high-level coordination procedure	Hypothetical examples with up to 16 trains and 73 block sections
Corman et al. (2010)	micro	Rt, Ro, Rr	GEO	B&B, H (FCFS)	set coordination constraint (iterative heuristic)	A network including more than 600 block sections, with up tp 80 trains
Corman et al. (2012b)	micor	Rt, Ro, Rr	GEO	B&B	B&B procedure	A network including more than 1200 block sections, with up tp 154 trains
Corman et al. (2014)	micro	Rt, Ro, Rr	GEO	B&B, H (JGH, AGH, FCFS, etc.)	heuristic coordination, exact coordination	A network consisting of more than 1000 block sections, and up to 205 trains are scheduled in a 90-minute planning horizon
Kersbergen et al. (2016)	macro	Rt, Ro	GEO	MPC	DMPC	The Dutch national railway network with 408 trains scheduled in a 75-minute planning horizon
Lamorgese and Mannino (2015)	macro-micro	Rt, Ro, Rr	MM	H (PR)	RG	Railway lines with up to 17 stations and with up to 54 trains scheduled
Lamorgese et al. (2016)	macro-micro	Rt, Ro, Rr	MM	H (G), B&C	RG, CG (add cuts violated by slave to master)	Railway lines with up to 34 stations, consisting of up to 53 block sections, and up to 130 trains are scheduled
Dollevoet et al. (2014)	macro-micro	Rt, Ro	MM	H, B&B	Iterative procedure	A network comprising 46 stations (macroscopic) and 554 block sections (microscopic), on which 310 trains are scheduled in a 240-minute planning horizon
Brännlund et al. (1998)	macro	-	TRA	SP	LR, H (PR)	A single-track railway network connecting 17 stations, where up to 30 trains are scheduled
Meng and Zhou (2014)	micro	Rt, Ro, Rr	TRA	SP	LR, H (PR)	A network composed of 97 block sections, with up to 40 trains
D'Ariano and Pranzo (2009)	micro	Rt, Ro, Rr	TIN	B&B, H (FCFS)	set coordination constraint (iterative heuristic)	A network consisting of 191 block sections, on which up to 40 trains are scheduled per hour
Zhan et al. (2016)	macro	Rt, Ro, Rr, canceling	TIN	CS	H (RH)	A Chinese high-speed railway line comprising 23 stations divide it into 22 segments, around 180 trains (services) are scheduled.
This paper	micro	Rt, Ro, Rr	GEO, TRA, and TIN	CS	ADMM, H (PR), and CDRSBK	A network consisting of 968 block sections, and a case of 154 trains is tested

* Symbol descriptions for Table 1: Re-timing (Rt); Re-ordering (Ro); Re-routing (Rr); Heuristic (H); Greedy (G); Local search (LS); Job greedy heuristic (JGH); Arc greedy heuristics (AGH); Branch and Bound (B&B); Branch and Cut (B&C); Model predictive control (MPC); Distributed model predictive control (DMPC); Row generation (RG); Column generation (CG); Shortest path algorithm (SP); Lagrangian relaxation (LR); Priority-rule (PR); Rolling horizon (RH); First-Come-First-Serve (FCFS); Model Predictive Control (MPC); Commercial solver (CS); Alternating Direction Method of Multipliers (ADMM); Cooperative Distributed Robust Safe But Knowledgeable (CDRSBK) algorithm.

3.1. Geography-based decomposition

The GEO decomposition partitions the whole railway network into a given number of regions, where each region corresponds to one subproblem. Consider a railway network composed of an ordered set of block sections E and a set of scheduled trains F traversing this network. We could easily partition the whole network into $|E|$ units, by means of a geography-(i.e., block section)-based decomposition; however, this could result in a large number of subproblems. In general, a larger number of subproblems implies more couplings among them, which makes coordination difficult and which may affect the overall performance of the system; therefore, we cluster these elementary block sections into a pre-defined number $|R|$ of regions, where $R = \{1, 2, \ldots, |R|\}$ is the set of regions. Fig. 1(b) illustrates a 2-region example of the GEO decomposition; as shown, the timetable is spatially partitioned.

To distribute $|E|$ different units into $|R|$ groups, there are $|R|^{|E|}$ ways, e.g., up to $2^{20} \approx 10^6$ ways for distributing 20 different units into only 2 groups. Thus, given a pre-defined number of regions, a huge number of the GEO decomposition results are available even for a small-scale network. To obtain the optimal decomposition result, an integer linear programming (ILP) approach is developed, with the objective of minimizing the number of couplings among regions (i.e., the total number of train service interconnections) and balancing the region sizes (i.e., the absolute deviation between the number of block sections contained in an individual region and the average value $|E|/|R|$).

We next describe the ILP model, using the following notations: The ordered set E_f contains the sequence of block sections composing the route of train f, and $|E_f|$ represents the number of block sections along the route of train f. The binary vector β_f indicates whether two consecutive block sections along the route of train f belong to different regions, e.g., if $(\beta_f)_j = 1$, then the j^{th} and $(j+1)^{\text{th}}$ block sections in the ordered set E_f are assigned to different regions, otherwise, $(\beta_f)_j = 0$. The binary vector α_r indicates the block section assignment for region r, e.g., if $(\alpha_r)_i = 1$, then the i^{th} block section in the ordered set E is assigned to region r, otherwise, $(\alpha_r)_i = 0$. The route matrix $B_f \in \mathbb{Z}^{(|E_f|-1)\times|E|}$ indicates that train f traverses a sequence of block sections, e.g., if train f traverses from the 1^{st} block section to the 3^{rd} block section in the ordered set E, then $B_f = \begin{bmatrix} 1 & 0 & -1 & 0 & \ldots \end{bmatrix}$. The integer vector $\mu \in (\mathbb{Z}^+)^{|E|}$ indicates the indices of regions that the block sections $e \in E$ belong to. A subscript of a vector indicates a component of the vector. For instance, the i^{th} component of vector μ is denoted by $(\mu)_i$, indicating the region index to which the i^{th} block section in the ordered set E belongs. We use $\|\cdot\|_1$ to denote the 1-norm.

The objective function is formulated as follows:

$$\min_{\alpha,\beta} \left[\zeta \cdot \left(\sum_{f\in F} \|\beta_f\|_1 \right) + (1-\zeta) \cdot \left(\sum_{r=1}^{|R|} \left| \|\alpha_r\|_1 - \frac{|E|}{|R|} \right| \right) \right], \tag{2a}$$

where the weight $\zeta \in [0, 1]$ is used to balance the importance of the two objectives. The first term serves to minimize the number of train service interconnections among regions, and the second term aims at balancing the region sizes.

We consider four constraints, presented as follows:

$$\frac{\left|\left(B_f \cdot \mu\right)_j\right|}{|R|-1} \le \left(\beta_f\right)_j, \quad \forall f \in F, j \in \left\{1, \ldots, |E_f|-1\right\}, \tag{2b}$$

guarantees that $(\beta_f)_j > 0$ (i.e., $(\beta_f)_j = 1$, as $(\beta_f)_j$ is binary), if the two consecutive block sections along the route of train f are assigned to different regions, i.e., $|(B_f \cdot \mu)_j| > 0$. The constraint

$$(\mu)_i \in \{1, \ldots, |R|\}, \quad \forall i \in \{1, \ldots, |E|\}, \tag{2c}$$

enforces that the indices of the resulting regions cannot exceed the pre-defined number of regions, while

$$(\alpha_r)_i \le 1 - \frac{|\mu_i - r|}{|R|-1}, \quad \forall r \in \{1, \ldots, |R|\}, i \in \{1, \ldots, |E|\}, \tag{2d}$$

$$\text{and } \|\alpha_r\|_1 \ge 1, \quad \forall r \in \{1, \ldots, |R|\}, \tag{2e}$$

are used to avoid empty regions to which no block section is assigned. Specifically, in (2d), if the i^{th} block section in set E is assigned to region r, i.e., $\mu_i = r$, then the binary variable $(\alpha_r)_i = 1$; otherwise, $(\alpha_r)_i = 0$. In (2e), we ensure that at least one block section is assigned to each region. As a result, (2d) and (2e) imply that the number of the resulting regions must equal the given number $|R|$. For the sake of clarity, an example is provided in Appendix B to illustrate the interpretation of the above mathematical formulation.

By applying the ILP approach (2), the block sections are assigned to $|R|$ regions, with the minimum number of couplings among regions and similar region scales, depending on the network layout and the train routes planned in the original timetable. This implies that the optimal GEO decomposition result is delay-independent, i.e., the GEO result is the same for all delay scenarios and can be computed off-line.

By applying the GEO decomposition, a train service may be cut off at the region border, and the legs of the train service are then scheduled separately and locally by region controllers. As a matter of fact, global feasibility requires that the

entry and exit events of a train service at the region border must be consistent, i.e., the time that a train departs from a region must equal the time that the train arrives at the consecutive region. Therefore, for the GEO decomposition, the time transition constraint (A.1g) in Appendix A for trains traversing the block sections at the border is the complicating (or coupling) constraint, which involves variables of different regions (i.e., subproblems) and prevents solving all the subproblems independently.

3.2. Train-based decomposition

The TRA decomposition simply splits an $|F|$-train problem into $|F|$ subproblems, where each subproblem corresponds to a single-train problem, as illustrated in Fig. 1(c). For a given instance, the TRA decomposition result is unique, only depending on the trains involved.

When applying the TRA decomposition, each train is scheduled independently in each subproblem. It turns out that trains may use the same infrastructure at the same time, resulting in conflicts and hence global infeasibility. In this case, capacity constraint (as formulated in (A.1l)-(A.1m) of Appendix A) for forcing that each block section can only be occupied by one train at any time is the complicating constraint. A train order variable (i.e., $\theta_{f,f',i,j}$ in Appendix A) couples two trains/subproblems to indicate the sequence of using a block section; however, it is not decisive (i.e., it is not a complicating variable), as the problem is still non-separable by fixing the train order variables.

3.3. Time-interval-based decomposition

Based on a given length of the time interval, the TIN decomposition divides a long time horizon into tractable intervals with an equal length, as illustrated in Fig. 1(d). We consider train delays when performing the TIN decomposition. More specifically, each train is first scheduled independently by taking disturbances into account, in order to generate an initial timetable where train conflicts may exist. With this infeasible timetable, we estimate the times that all events (e.g., train departures and arrivals) may occur. Each event is then assigned to one time interval based on its estimated occurrence time. As a result, the subproblem of each time interval includes all events that are estimated to occur in this time interval. The TIN decomposition result mainly depends on the given length of the time interval and the estimated train schedule, which can be different in delay scenarios.

Each train service consists of a set of departure and arrival events for the train on block sections along its route. When applying the TIN decomposition, these events may be assigned to different time intervals. Thus, similar to the GEO decomposition (where trains may traverse from one region to another), the time that a train leaves a time interval should be equal to the time that the train enters the consecutive time interval. The time transition constraint (A.1g) is therefore the complicating constraint for the TIN decomposition. Moreover, as the TIN decomposition is performed based on an estimated (possibly) infeasible timetable, an event assigned to time interval t may be further scheduled to the next time interval $t+1$, causing conflicts with the events in time interval $t+1$. Therefore, the capacity constraint in (A.1l)-(A.1m) is also the complicating constraint for the TIN decomposition.

3.4. Subproblems and couplings

Let us denote S as the set of the $|S|$ subproblems resulting from the decomposition, e.g., $|S| = |R|$ for the GEO decomposition and $|S| = |F|$ for the TRA decomposition. No matter which decomposition method is used, we can always divide the constraints of the MILP problem (1) into two categories, i.e., local constraints and complicating constraints. As we have explained, a complicating constraint is associated with at least two subproblems, so that it results in a non-separable structure of the overall optimization problem. A local constraint is related to a single subproblem only. We thus rewrite (1b) into a general form of the following local and complicating constraints:

$$A^{\text{loc}} \cdot \lambda \leq b^{\text{loc}} \tag{3a}$$

$$A^{\text{cpl}} \cdot \lambda \leq b^{\text{cpl}} \tag{3b}$$

with matrices $A^{\text{loc}} \in \mathbb{R}^{m_1 \times n}$ and $A^{\text{cpl}} \in \mathbb{R}^{m_2 \times n}$ and vectors $b^{\text{loc}} \in \mathbb{R}^{m_1}$ and $b^{\text{cpl}} \in \mathbb{R}^{m_2}$.

For the GEO decomposition, (3b) includes the time transition constraint (A.1g) for trains traversing the border block sections. For the TRA decomposition, (3b) includes the capacity constraint (A.1l)-(A.1m). The coupling constraint (3b) of the TIN decomposition includes both (A.1g) and (A.1l)-(A.1m). Let us denote $Q_p = \left\{q_{p,1}, q_{p,2}, \ldots, q_{p,m_p}\right\}$ as the set of m_p subproblems that have couplings with subproblem p. The subproblem $p \in S$ of the MILP problem with objective (1a) and constraints (3a)-(3b) is formulated as

$$\min_{\lambda_p} \quad \mathcal{Z}_p(\lambda_p) = c_p^\top \cdot \lambda_p \tag{4a}$$

$$\text{s.t.} \quad A_p^{\text{loc}} \cdot \lambda_p \leq b_p^{\text{loc}} \tag{4b}$$

$$A_{p,q}^{\text{cpl}} \cdot \lambda_p + A_{q,p}^{\text{cpl}} \cdot \lambda_q \leq b_{p,q}^{\text{cpl}}, \ \forall q \in Q_p \tag{4c}$$

where $A^{cpl}_{p,q}$ and $A^{cpl}_{q,p}$ are selection matrices for selecting the coupling variables between subproblems p and q.

Since each coupling constraint in (4c) includes the variables λ_p and λ_q of two subproblems p and q, we cannot explicitly add them to any individual subproblem. Instead we can determine and exchange values of the complicating variables among subproblems in an iterative way. The train(s) of one subproblem p can obtain an agreement through iterations that inform the train(s) of its coupled subproblems $q \in Q_p$ about what subproblem p prefers the values of coupling variables to be. To achieve this agreement, for a single subproblem p, we have to compute the optimal coupling variables (inputs) for its coupled subproblems $q \in Q_p$ as well, rather than only focusing on computing optimal local variables. Moreover, for its coupled subproblems $q \in Q_p$, we need to compute both the optimal local variables and coupling variables (outputs). Through exchanging these desired coupling variables, the values of these outputs and inputs should converge to each other, and a set of local inputs that is overall feasible should be found. Distributed optimization approaches are developed for reaching this agreement in Section 4.

4. Distributed optimization approaches

This section introduces three distributed optimization approaches to address the coupling issues among subproblems, namely the Alternating Direction Method of Multipliers (ADMM) algorithm, the priority-rule-based (PR) algorithm, and the Cooperative Distributed Robust Safe But Knowledgeable (CDRSBK) algorithm, presented in Section 4.1–4.3 respectively. Section 4.4 uses a simple example of the TRA decomposition to explain the scheduling strategy of the three algorithms. A key challenge in distributed optimization algorithms is to ensure that the solution generated for a single subproblem leads to feasible solutions that satisfy the complicating constraints with other subproblems.

4.1. Alternating direction method of multipliers algorithm

The alternating direction method of multipliers (ADMM) algorithm (Boyd et al., 2011) solves problems of the following form:

$$\min_{z_1,z_2} \quad \mathcal{F}(z_1) + \mathcal{G}(z_2) \tag{5a}$$

$$\text{s.t.} \quad A \cdot z_1 + B \cdot z_2 = b, \tag{5b}$$

with variables $z_1 \in \mathbb{R}^n$ and $z_2 \in \mathbb{R}^m$, matrices $A \in \mathbb{R}^{p\times n}$ and $B \in \mathbb{R}^{p\times m}$, and vector $b \in \mathbb{R}^p$. Assume that the variables z_1 and z_2 can be split into two parts, with the objective function separable across this splitting. We can then form the augmented Lagrangian relaxation as

$$\mathcal{L}_\rho(z_1, z_2, y) = \mathcal{F}(z_1) + \mathcal{G}(z_2) + y^\top (A \cdot z_1 + B \cdot z_2 - b) + \frac{\rho}{2} \cdot \|A \cdot z_1 + B \cdot z_2 - b\|_2^2, \tag{6}$$

where y is the dual variable (Lagrangian multiplier), the parameter $\rho > 0$ indicates the penalty multiplier, and $\| \cdot \|_2$ denotes the Euclidean norm. The augmented Lagrangian function is optimized by minimizing over z_1 and z_2 alternately or sequentially and then evaluating the resulting equality constraint residual. By applying the dual ascent method, the ADMM algorithm consists of the following iterations:

$$z_1^{i+1} := \arg\min_{z_1} \mathcal{L}_\rho(z_1, z_2^i, y^i), \tag{7a}$$

$$z_2^{i+1} := \arg\min_{z_2} \mathcal{L}_\rho(z_1^{i+1}, z_2, y^i), \tag{7b}$$

$$y^{i+1} := y^i + \rho \cdot (A \cdot z_1^{i+1} + B \cdot z_2^{i+1} - b) \tag{7c}$$

where i is the iteration counter. In the ADMM algorithm, the variables z_1 and z_2 are updated in an alternating or sequential fashion, which accounts for the term alternating direction.

The ADMM algorithm can obviously deal with linear equality constraints, but it can also handle linear inequality constraints. The latter can be reduced to linear equality constraints by replacing constraints of the form $A \cdot x \le b$ by $A \cdot x + s = b$, adding the slack variable s to the set of optimization variables, and setting $\mathcal{Z}(s) = 0$, if $s \ge 0$, otherwise, setting $\mathcal{Z}(s) = \infty$. Alternatively, we can also work with an equivalent reformulation of problem (4), where we replace the complicating constraint (4c) by

$$\mathcal{C}_p(\lambda_p, \lambda_q) = 0 \tag{8}$$

by letting $\mathcal{C}_p(\lambda_p, \lambda_q) = \max\left\{0, A^{cpl}_{p,q} \cdot \lambda_p + A^{cpl}_{q,p} \cdot \lambda_q - b^{cpl}_{p,q}\right\}$ with component-wise maximum. In such a way, we can transform the inequality constraints into equality constraints.

Now we can apply the ADMM algorithm, and the augmented Lagrangian formulation of the MILP problem (1) can be described as follows:

$$\mathcal{L}_\rho = \sum\nolimits_{p\in S}\left[\mathcal{Z}_p(\lambda_p) + \sum\nolimits_{q\in Q_p}\left[y_{p,q}^\top \cdot \mathcal{C}_p(\lambda_p, \lambda_q) + \frac{\rho}{2} \cdot ||\mathcal{C}_p(\lambda_p, \lambda_q)||_2^2\right]\right] \tag{9}$$

The iterations to compute the solution of the MILP problem (1) based on the augmented Lagrangian formulation (9) include quadratic terms; therefore, the function $\mathcal{L}_\rho$ cannot directly be distributed over subproblems. Inspired by Negenborn et al. (2008), for handling this non-separable issue, the function $\mathcal{L}_\rho$ in (9) can be approximated by solving $|S|$ separate problems of the form

$$\min_{\lambda_p} \ \mathcal{Z}_p(\lambda_p) + \sum_{q \in Q_p} \mathcal{J}_p(\lambda_p, \lambda_q, y_{p,q}) \tag{10}$$

subject to (4b) for the train movements of the single subproblem p, where the additional term $\mathcal{J}_p(\cdot)$ deals with coupling variables.

We now define the term $\mathcal{J}_p(\cdot)$ by using a serial implementation. We apply a block coordinate descent approach (Beltran Royoa and Heredia, 2002; Negenborn et al., 2008). The approach minimizes the quadratic term directly in a serial manner. One subproblem after another minimizes its local and coupling variables while the variables of the other subproblems stay fixed. At iteration i, let us denote $\widehat{Q}_p^i \subseteq Q_p$ as the set of those subproblems that are coupled with subproblem p and have been solved before solving subproblem p.

The serial implementation uses the information from both the current iteration i and the previous iteration $i-1$. With the information $\bar{\lambda}_q = \lambda_q^{(i)}$ computed in the current iteration i for subproblems $q \in \widehat{Q}_p^i$ and the information $\bar{\lambda}_q = \lambda_q^{(i-1)}$ obtained in the previous iteration $i-1$ for the other subproblems $q \in Q_p \backslash \widehat{Q}_p^i$, we can solve (10) for subproblem p by using the following function:

$$\mathcal{J}_p(\lambda_p, \bar{\lambda}_q, y_{p,q}) = y_{p,q}^\top \cdot \mathcal{C}_p(\lambda_p, \bar{\lambda}_q) + \frac{\rho}{2} \cdot ||\mathcal{C}_p(\lambda_p, \bar{\lambda}_q)||_2^2 \tag{11}$$

The second term of (11) penalizes the deviation from the coupling variables that were computed for the subproblems solved before subproblem p in the current iteration i and for the other subproblems during the previous iteration $i-1$.

We can also adopt a parallel implementation for the ADMM algorithm. To do this, we only use the information obtained in the previous iteration, i.e., we define $\bar{\lambda}_q = \lambda_q^{(i-1)}, \forall q \in Q_p$ for (11) at iteration i.

The solution procedure of the ADMM algorithm is described as follows:

The solution procedure of the ADMM Algorithm.

Initialization: Set the iteration counter $i := 1$, the penalty multiplier $\rho := \rho^{(0)}$, the Lagrange multipliers $y^{(0)} := 0$, and the solution set $\Theta_{\text{sol}} := \{\bar{\lambda}_p | p \in S\}$ to be empty. Denote the maximum number of iterations as $I^{\max}$. The penalty multiplier ρ is updated as a function of the iteration index, denoted by $\varrho(\cdot)$.

1: **for** iteration $i := 1, 2, \ldots, I^{\max}$ **do**
2: Randomly generate the orders of subproblems, denoted as $P_{\text{order}}^{(i)}$. - - - - -[*for serial*: lines 3–6]- - - - -
3: **for** subproblem $j := 1, 2, \ldots, |S|$ **do**
4: Solve subproblem $p := P_{\text{order}}^{(i)}(j)$, consisting of objective function (10) and constraint (4b), by taking the available solutions in Θ_{sol} for all $q \in \widehat{Q}_p^i$ into account.
5: Denote the obtained solution of subproblem p as $\lambda_p^{(i)}$, and update the latest solution set Θ_{sol} by adding or setting $\bar{\lambda}_p := \lambda_p^{(i)}$.
6: **end for** -[*for parallel*: lines 7–8]- -
7: Solve all subproblems $p \in S$ in parallel, consisting of objective function (10) and constraint (4b), by taking the solutions in Θ_{sol} for all $q \in \widehat{Q}_p^i$ into account (which is obtained in the previous iteration).
8: Denote the newly obtained solution of each subproblem $p \in S$ as $\lambda_p^{(i)}$, and update the solution set Θ_{sol} by adding or setting $\bar{\lambda}_p := \lambda_p^{(i)}$ for all $p \in S$. -
9: Update the Lagrange multipliers by $y_{p,q}^{(i)} := y_{p,q}^{(i-1)} + \rho \cdot \mathcal{C}_p(\lambda_p^{(i)}, \lambda_q^{(i)})$ for all $p \in S$ and $q \in Q_p$.
10: Update the penalty multiplier by $\rho := \rho^{(i)}$, where $\rho^{(i)}$ is obtained by the predefined function of $\varrho(i)$.
11: Break the iterations if the deviation of the coupling variables at the current iteration step~i is less than the expected gap ϵ, i.e., $\|\mathcal{C}_p\|_\infty \le \epsilon$ for all $p \in S$, where ϵ is a small positive scalar and $\|\cdot\|_\infty$ denotes the infinity norm[1].
12: **end for**

---

[1] The infinity norm (also known as the maximum norm) of a vector $x = [\,x_1 \quad \ldots \quad x_n\,]^\top$ is denoted as $\|x\|_\infty$ and is defined as $\|x\|_\infty = \max\{|x_1|, \ldots, |x_n|\}$.

By applying the ADMM algorithm, we solve the subproblems $p \in S$ in an iterative manner, with respect to the local constraint (4b) of a single subproblem p and taking the solutions of all coupled subproblems (i.e., the variable $\bar{\lambda}_q$ for $q \in Q_p$ obtained in either the current iteration or the previous iteration) into account. In (9), only the local objective $\mathcal{Z}_p$ for a single subproblem p is minimized, not the global objective $\sum_{p\in S} \mathcal{Z}_p$.

In order to further improve the performance of the ADMM algorithm, we can add a cost-to-go function $Z_p^{\text{ctg}}(\lambda_p)$ into the objective function of each subproblem, which provides an estimation of the train movement to its destination. Then, the objective function (10) for subproblem $p \in S$ can be rewritten as follows:

$$\min_{\lambda_p} \ \mathcal{Z}_p(\lambda_p) + \mathcal{Z}_p^{\text{ctg}}(\lambda_p) + \sum\nolimits_{q\in Q_p} \mathcal{J}_p(\lambda_p, \lambda_q, y_{p,q}). \tag{12}$$

For instance, with the GEO decomposition, we can define the cost-to-go function as the deviation between the actual and planned departure time from the block section where a train crosses a region border. Thus, a timetable given detailed information about train departure/arrival times from/at block sections is needed.

The ADMM algorithm incorporates the complicating constraint (4c) into the objective function and strives to make the information consistent among subproblems (i.e., each subproblem takes the information of the other subproblems into account) in an iterative manner. However, convergence cannot be guaranteed for non-convex optimization problems, so that a feasible solution may not be available. Therefore, we also need to explore other distributed optimization approaches.

4.2. Priority-rule-based algorithm

This section introduces the priority-rule-based (PR) algorithm. The main idea of the PR algorithm is to optimize train schedules of subproblems in a sequential manner according to problem priorities, with respect to the solutions of the other subproblems that have already been solved in the current iteration. The problem priorities are determined by the train delay times of the subproblems, e.g., we solve the subproblem with the largest delay time first. The PR algorithm rewrites (4c) for the subproblem $p \in S$ as follows:

$$A_{p,q}^{\text{cpl}} \cdot \lambda_p + A_{q,p}^{\text{cpl}} \cdot \bar{\lambda}_q \le b_{p,q}^{\text{cpl}}, \quad \forall q \in Q_p \tag{13}$$

with the solution $\bar{\lambda}_q = \lambda_q^{(i)}$ computed in the current iteration i for all subproblems $q \in \widehat{Q}_p^i$.

The solution procedure of the PR algorithm is described as follows:

The solution procedure of the PR Algorithm.

Initialization: Set the iteration counter $i := 1$, the local upper bound $o_{\text{UB}}^{(0)} := M$, and the global upper bound $O_{\text{UB}}^{(0)} := M$, where M is a sufficiently large positive number. Initialize the problem priorities $P_{\text{prior}}^{(0)}$ arbitrarily. Denote the maximum number of iterations as $I^{\max}$.

1: **for** iteration $i := 1, 2, \ldots, I^{\max}$ **do**
2: Sort subproblems in set S in a descending order by problem priorities $P_{\text{prior}}^{(i-1)}$, denoted as $P_{\text{order}}^{(i)}$.
3: Set the solution set Θ_{sol} to be empty.
4: **for** subproblem $j := 1, 2, \ldots, |S|$ **do**
5: Solve subproblem $p := P_{\text{order}}^{(i)}(j)$, including objective function (4a) and constraints (4b) and (13), with respect to the available solutions in Θ_{sol} for all $q \in \widehat{Q}_p^i$.
6: Denote the obtained solution of subproblem p as $\lambda_p^{(i)}$, and update the solution set Θ_{sol} by adding $\bar{\lambda}_p := \lambda_p^{(i)}$.
7: **end for**
8: Compute the local upper bound $o_{\text{UB}}^{(i)}$, and update the global upper bound by

$$O_{\text{UB}}^{(i)} := \begin{cases} o_{\text{UB}}^{(i)}, & \text{if } i = 1 \text{ or } \mathrm{O}_{\text{UB}}^{(\mathrm{i}-1)} > \mathrm{o}_{\text{UB}}^{(\mathrm{i})} \\ O_{\text{UB}}^{(i-1)}, & \text{otherwise} \end{cases}$$

9: Update the problem priorities $P_{\text{prior}}^{(i)}$ by the train delay times of the subproblems.
10: Break the iterations if the global upper bounds are not improved for a given number of iterations κ, i.e., $O_{\text{UB}}^{(i)} = O_{\text{UB}}^{(i-\kappa)}$.
11: **end for**

In the PR algorithm, we solve each subproblem $p \in S$ in a sequential manner according to the priorities of the subproblems, with respect to the local constraint (4b) and the outputs $\bar{\lambda}_q$ of the coupled subproblems $q \in Q_p$ in (13). Similar to the ADMM algorithm, only the local objective $\mathcal{Z}_p$ is minimized when solving subproblem p, rather than the global objective $\sum_{p\in S} \mathcal{Z}_p$ for all subproblems. Constraint (13) ensures that the coupling variables of subproblem p satisfy those of its coupled subproblems $q \in Q_p$ obtained in the current iteration. For the first solved subproblem in each iteration, the complicating constraint (13) is relaxed, as $\widehat{Q}_p^1 = \varnothing, \ \forall p \in S$.

As the PR algorithm solves each subproblem with full respect to the solution outputs of the other subproblems, the presence of the hard constraint (13) is most likely to cause infeasibility, especially for those subproblems at the end of the

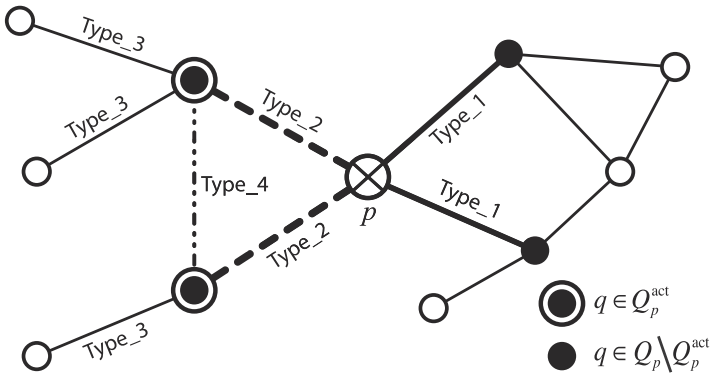


Fig. 2. Four types of couplings defined in the CDRSBK algorithm.

queue in each iteration. This issue does not arise for the TRA decomposition, where we assume an infinite planning horizon so that trains can always be feasibly scheduled at the cost of large delays. We next introduce the third algorithm, where the feasibility issue is somehow addressed.

4.3. Cooperative distributed robust safe but knowledgeable algorithm

The third algorithm considered is called the Cooperative Distributed Robust Safe But Knowledgeable (CDRSBK) algorithm. It was developed by Kuwata and How (2011) to address trajectory planning problems. Inspired by Kuwata and How (2011), the CDRSBK algorithm introduced here is an extended and improved version for coping with the railway traffic management problem. In the CDRSBK algorithm, four types of couplings among subproblems are defined for a subproblem $p \in S$, as illustrated in Fig. 2. Each circle in Fig. 2 indicates a subproblem, which represents a region, a train, and a time interval for the GEO, TRA, and TIN decomposition respectively.

Type_1 indicates a non-active coupling between subproblem $p \in S$ and its neighboring subproblem; Type_2 indicates an active coupling between subproblem p and its neighboring subproblem; Type_3 indicates the coupling between the actively coupled neighbors of subproblem p and their neighbors; and Type_4 indicates the coupling between two actively coupled neighbors of subproblem p. Let us denote Q_p as the set of all coupled neighbors of subproblem p and denote Q_p^{act} as the set of subproblem p's neighbors that have an active coupling with subproblem p. The interpretation of active and non-active couplings can be different for different decomposition methods. We discuss the details regarding their implementations in Section 5.1.

By applying the CDRSBK algorithm, the MILP problem (4a)-(4c) can be reformulated for subproblem $p \in S$ as

$$\min_{\lambda_p, \xi_q} \quad \mathcal{Z}_p(\lambda_p) + \sum_{q \in Q_p^{\text{act}}} \mathcal{Z}_q(\bar{\lambda}_q + T_q \cdot \xi_q) \tag{14a}$$

$$\text{s.t.} \quad A_p^{\text{loc}} \cdot \lambda_p \le b_p^{\text{loc}} \tag{14b}$$

$$A_q^{\text{loc}} \cdot (\bar{\lambda}_q + T_q \cdot \xi_q) \le b_q^{\text{loc}}, \quad \forall q \in Q_p^{\text{act}} \tag{14c}$$

$$A_{p,q}^{\text{cpl}} \cdot \lambda_p + A_{q,p}^{\text{cpl}} \cdot \bar{\lambda}_q \le b_{p,q}^{\text{cpl}}, \quad \forall q \in Q_p \backslash Q_p^{\text{act}} \tag{14d}$$

$$A_{p,q}^{\text{cpl}} \cdot \lambda_p + A_{q,p}^{\text{cpl}} \cdot (\bar{\lambda}_q + T_q \cdot \xi_q) \le b_{p,q}^{\text{cpl}}, \quad \forall q \in Q_p^{\text{act}} \tag{14e}$$

$$A_{o,q}^{\text{cpl}} \cdot \bar{\lambda}_o + A_{q,o}^{\text{cpl}} \cdot (\bar{\lambda}_q + T_q \cdot \xi_q) \le b_{o,q}^{\text{cpl}}, \quad \forall o \in Q_q \backslash Q_p^{\text{act}}, q \in Q_p^{\text{act}} \tag{14f}$$

$$A_{q_1,q_2}^{\text{cpl}} \cdot (\bar{\lambda}_{q_1} + T_{q_1} \cdot \xi_{q_1}) + A_{q_2,q_1}^{\text{cpl}} \cdot (\bar{\lambda}_{q_2} + T_{q_2} \cdot \xi_{q_2}) \le b_{q_1,q_2}^{\text{cpl}}, \quad \forall q_1, q_2 \in Q_p^{\text{act}}, q_2 \in Q_{q_1}, q_1 \in Q_{q_2} \tag{14g}$$

In (14a), the objective function of both subproblem p and its actively coupled subproblems $q \in Q_p^{\text{act}}$ are included. Constraints (14b)-(14c) represent the local constraints of subproblem p and its actively coupled subproblems $q \in Q_p^{\text{act}}$ respectively. In (14d)-(14g), coupling constraint (4c) is rewritten for the four types of couplings respectively. When solving subproblem p, besides the local variable λ_p, the variable ξ_q is also optimized for its actively coupled subproblems $q \in Q_p^{\text{act}}$ based

on the communicated solution $\bar{\lambda}_q$ as follows:

$$\lambda_q = \bar{\lambda}_q + T_q \cdot \xi_q \tag{15}$$

parameterized with a matrix T_q, which enforces that the variable ξ_q can change only the rows corresponding to the active complicating constraints. This can also be interpreted as allowing a change for the constraint that has a non-zero Lagrange multiplier. In (14a), the objectives of a single subproblem p and its actively coupled neighbors $q \in Q_p^{\text{act}}$ are both minimized, so that we can explore solutions with a global vision. Moreover, allowing the variable changes of actively coupled subproblems greatly resolve the issue of infeasibility.

The solution procedure of the CDRSBK algorithm is described as follows:

The solution procedure of the CDRSBK Algorithm.

Initialization: Set the iteration counter $i := 1$, the local upper bound $o_{\text{UB}}^{(1)} := M$, and the global upper bound $O_{\text{UB}}^{(1)} := M$, and the solution set $\Theta_{\text{sol}} := \{\bar{\lambda}_p | p \in S\}$ to be empty. Denote the maximum number of iterations as $I^{\max}$.

1: **for** iteration $i := 1, 2, \ldots, I^{\max}$ **do**
2: Randomly generate the orders of subproblems, denoted as $P_{\text{order}}^{(i)}$.
3: **for** subproblem $j := 1, 2, \ldots, |S|$ **do**
4: Solve subproblem $p := P_{\text{order}}^{(i)}(j)$ and its actively coupled subproblems $q \in Q_p^{\text{act}}$, consisting of objective function (14a) and constraints (14b)-(14g), by taking the available solutions in set Θ_{sol} for all $l \in (Q_p \backslash Q_p^{\text{act}}) \cup (Q_q \backslash Q_p^{\text{act}})$ into account.
5: Denote the obtained solutions of subproblem p and its actively coupled subproblems $q \in Q_p^{\text{act}}$ as $\lambda_p^{(i)}$ and $\lambda_q^{(i)}$ (which is obtained by (15)) respectively, and update the latest solution set Θ_{sol} by adding or setting $\bar{\lambda}_p := \lambda_p^{(i)}$ and $\bar{\lambda}_q := \lambda_q^{(i)}$ for all $q \in Q_p^{\text{act}}$.
6: **end for**
7: Compute the local upper bound $o_{\text{UB}}^{(i)}$, and update the global upper bound by

$$O_{\text{UB}}^{(i)} := \begin{cases} o_{\text{UB}}^{(i)}, & \text{if } i = 1 \text{ or } \mathrm{O}_{\text{UB}}^{(\mathrm{i}-1)} > \mathrm{o}_{\text{UB}}^{(\mathrm{i})} \\ O_{\text{UB}}^{(i-1)}, & \text{otherwise} \end{cases}$$

8: Break the iterations if the global upper bounds are not improved for a given number of iterations κ, i.e., $O_{\text{UB}}^{(i)} = O_{\text{UB}}^{(i-\kappa)}$.
9: **end for**

In each iteration, the CDRSBK algorithm solves each subproblem with additional objectives and complicating constraints that include the changeable (local) variables of its actively coupled subproblems $q \in Q_p^{\text{act}}$. If the variables of its actively coupled subproblems are unchangeable, i.e., $\lambda_q = \bar{\lambda}_q$ when ξ_q has no impact on the variables, the complicating constraints are automatically satisfied and could be omitted.

4.4. Algorithm strategy

Let us consider a simple example of the TRA decomposition to understand the scheduling strategies of the three algorithms, presented in the time-space graphs of Fig. 3. As shown in Fig. 3(a), there are 4 trains scheduled independently, and a conflict appears between train_1 and train_2. We assume that the train paths in black are fixed, and the focus is on scheduling train_2.

When a conflict is detected, the ADMM algorithm resolves the conflict by penalizing it in the objective with the Lagrangian and penalty multipliers. As long as the conflict exists, the corresponding Lagrangian multiplier increases continuously, causing the increase of the penalty value (for the corresponding conflict). When the penalty of the conflict is large enough, an alternative solution (better, with a smaller penalty value) will be found; thus, the conflict can be resolved. However, resolving one conflict may lead to some other conflicts, see Fig. 3(b), where train_2 is rescheduled to avoid the conflict

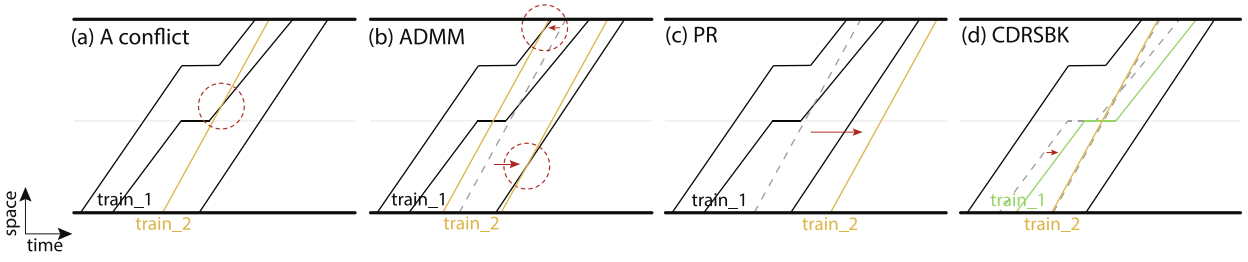


Fig. 3. Algorithm strategy: an example of the TRA decomposition on time-space graph.

with train_1, influenced by the large penalty of the conflict; however, train_2 may still conflict with the other two trains. Some iterations and updates of multipliers will be needed to resolve all conflicts and to generate a conflict-free timetable.

The PR algorithm searches for each delayed train the earliest arrival possibility at stations, based on the infrastructure use of some other trains. Therefore, as shown in Fig. 3(c), train_2 is rescheduled to the end of the timetable, subject to the other three fixed train paths. As such, the feasibility of the system is guaranteed, but trains with lower priorities (in the trail of the queue) may undertake large delays, e.g., train_2 faces a large delay.

The CDRSBK algorithm recognizes the couplings among trains that have a conflict to be active. When rescheduling one train, some variables of its actively coupled train(s) can also be changed. Therefore, as we can see in Fig. 3(d), the path of train_1 is optimized together with that of train_2. With a slight change of train_1, a solution better than the PR solution is found for train_2, as well as for the overall system (less total delays). The CDRSBK strategy can apparently gather the trains involved in each conflict into a group and to schedule each group of train(s) independently, until a feasible solution is found. For the TRA decomposition, the CDRSBK algorithm somehow reflects the actual behavior of human dispatchers who detect "potential" conflicts and prevent them according to their experience (differently, we resolve the conflicts by means of optimization).

5. Implementation of the proposed decomposition methods and distributed algorithms

We consider a complete combination of the three decomposition methods and three distributed algorithms, resulting in 9 algorithmic configurations. In this section, we discuss their implementation. Some interpretations are given with regards to how to define the active and non-active couplings used by the CDRSBK algorithm, followed by some tips and suggestions for achieving feasibility.

5.1. Active and non-active couplings of the CDRSBK algorithm

We now explain how to define the active and non-active couplings of the CDRSBK algorithm for different decomposition methods.

- For the GEO decomposition
  If two regions are connected by tracks, i.e., they are neighbors, then we consider that a coupling exists between the two subproblems (regions). A coupling between two subproblems is considered to be active (Type_2) if there is any train traverse between the two regions of the two subproblems; otherwise, the coupling is recognized as non-active coupling (Type_1). Regarding Type_3 and Type_4 couplings, we follow their general definitions, i.e., the couplings between an actively coupled neighbor and its coupled neighbors are considered as Type_3 coupling and the coupling between two actively coupled neighbors is labeled as Type_4.
- For the TRA decomposition
  If two trains use the same infrastructure (block sections), then we consider that a coupling exists between the two subproblems of these two trains. If a conflict exists between these two trains, then their coupling is recognized as an active coupling; otherwise, their coupling is considered to be non-active. For coupling Type_3 and Type_4, we follow their general definitions.
  In the TRA decomposition, we often have many trains that use the same infrastructure; but conflicts may never happen among some of them, e.g., a train scheduled in the early morning has little chance to have a conflict with another train scheduled in the late afternoon. Thus, to further reduce the problem complexity for large-scale networks, we provide two more options for defining couplings of different types. We denote the option described above as Opt_1. The difference between Opt_1 and Opt_2 lies in the definition of coupling Type_3: in Opt_2, we label the couplings between an actively coupled neighbor and its *actively* coupled neighbor as Type_3. Based on Opt_2, we further discard all Type_1 couplings, which results in Opt_3, i.e., when and only when a conflict occurs between two trains, a coupling will appear between them and be recognized as active coupling (Type_2). An illustrative example is provided in Appendix B to graphically explain these three options.
- For the TIN decomposition
  Due to the nature of the TIN decomposition, the relation among subproblems is relatively simple in this case. Couplings exist only between two consecutive subproblems (i.e., two subproblems of two consecutive time intervals t and $t+1$). All existing couplings are recognized as active couplings (Type_2). As a result, according to the general definition of the four types of couplings introduced in Section 4.3, the couplings between a consecutive subproblem and its consecutive subproblem are considered as Type_3 (e.g., for subproblem t, a Type_3 coupling exists between subproblems $t+1$ and $t+2$). Moreover, Type_1 and Type_4 couplings do not appear.

Remark (Feasibility achievement)**.**

- It is essential to ensure that train orders in the subproblems are feasible, in order to avoid unnecessary iterations and to achieve fast convergence. To do this, we keep a consistency of the train orders that are interrelated, e.g., if two trains cannot overtake on a sequence of block sections, then the train orders of these two trains on these block sections are interrelated and must be the same.

- When implementing the TIN decomposition with the CDRSBK algorithm, we suggest to solve subproblems by following the order of time horizon (i.e., in a sequence of time intervals $t = 1, 2, 3, \ldots$.) in the first iteration, in order to generate a feasible solution.
- An inappropriate choice of big-M in (A.1) may lead to computing errors and hence cause infeasibility.
- The CDRSBK algorithm requires precise computation (regarding feasibility tolerance and integral tolerance); otherwise, the computing errors may lead to infeasibility.
- For the TRA decomposition, a feasible solution can always be found by the PR or CDRSBK algorithm in an infinite planning horizon. The ADMM algorithm can also find feasible solutions under the TRA decomposition, which is achieved by appropriately setting the penalty multiplier.

6. Case study

6.1. Set-up

We consider a railway network in the South-East of the Netherlands that spans 10 dispatching areas of the Dutch railway network, as shown in Fig. 4. The network includes the 4 major stations of Utrecht Central, Arnhem, Nijmegen, and Den Bosch ('s-Hertogenbosch), plus other 40 minor stations. The two main traffic directions are served by the line between Utrecht and Arnhem (towards Germany) and the line between Utrecht and Den Bosch (from Amsterdam towards Eindhoven and the southern part of the country). The network comprises a combination of single and double tracks of different lengths, described by 891 nodes and 968 links. There are 46 links that trains can use in both directions, and the other links are practically used in a single direction by the current timetable. A case of 154 trains scheduled on this network is considered as test bed, including both local and intercity services. We consider only local rerouting and fix train global routes. Moreover, we consider 10 delay scenarios randomly generated by following a 3-parameter Weibull distribution (see Corman et al., 2011b). Additionally, we report the details of the instance and of the experimental results in the online repository (Research Collection ETH Zurich).

For the GEO decomposition, we partition the network shown in Fig. 4 into 3, 5, 7, and 9 regions, and for the TIN decomposition, we set the time interval length to be 5 and 10 mins. For the ADMM algorithm, the penalty multiplier ρ is considered to be a function of the iteration index. We consider 4 options for updating the penalty multiplier, i.e., $\varrho^{\text{small}}(\cdot)$, $\varrho^{\text{large}}(\cdot)$, $\varrho^{\text{linear}}(\cdot)$, and $\varrho^{\text{exp}}(\cdot)$, respectively indicating a small constant multiplier function, a large constant multiplier

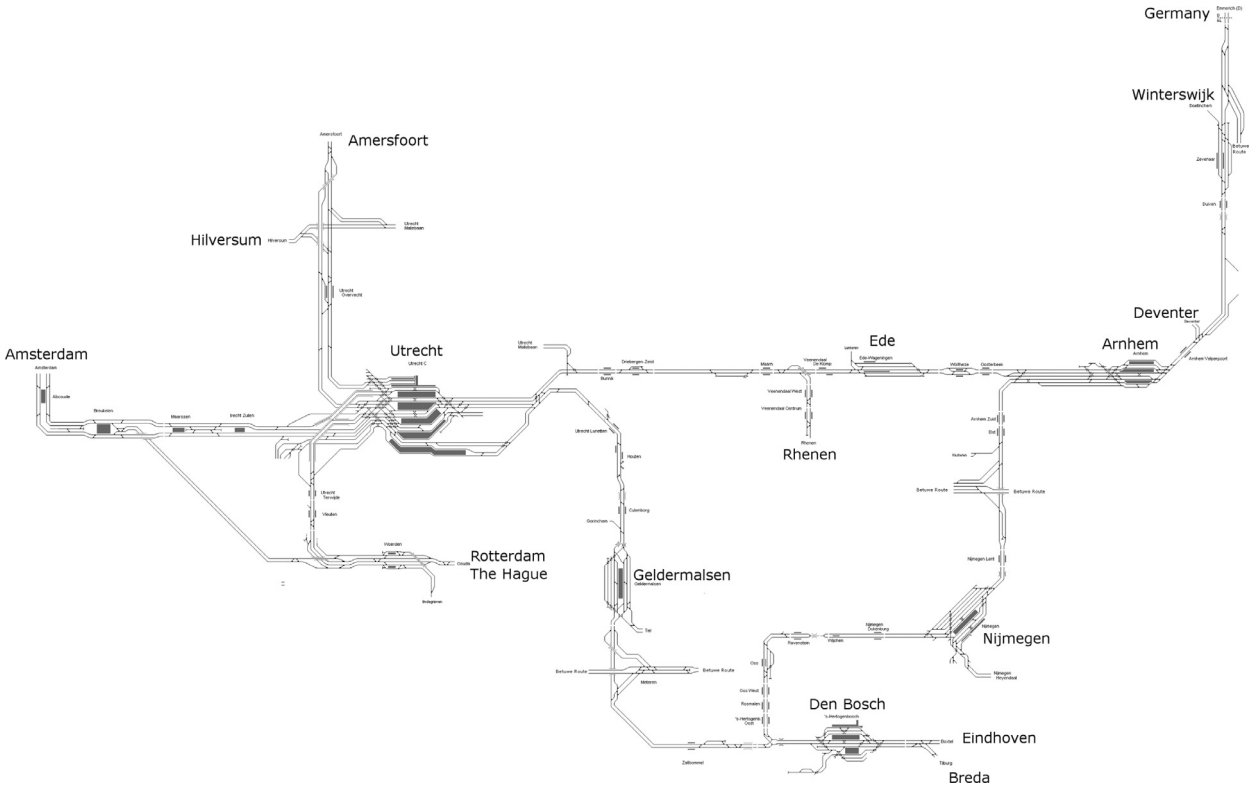


Fig. 4. A large-scale railway network.

function, a linearly increasing function, and an exponentially increasing function. These functions are set to be in the range of [1,10]. This means that we let $\varrho^{\text{small}}(\cdot) = 1$ (i.e., $\rho^{(0)} = \rho^{(\infty)} = 1$) and $\varrho^{\text{large}}(\cdot) = 10$ (i.e., $\rho^{(0)} = \rho^{(\infty)} = 10$) for all iterations, and respectively let the function $\varrho^{\text{linear}}(\cdot)$ and $\varrho^{\text{exp}}(\cdot)$ increase linearly and exponentially from 1 to 10 as a function of the iteration index, i.e., $\rho^{(0)} = 1$ and $\rho^{(\infty)} = 10$. We here consider only the serial implementation of the ADMM algorithm and leave its parallel implementation for future research. Moreover, we follow Opt_3 to define active and non-active couplings for the CDRSBK algorithm in the case of the TRA decomposition, because it performs much better than Opt_1 and Opt_2 according to our tests on a small instance.

We adopt the CPLEX solver version 12.9.0 implemented in the MATLAB (R2019a) to solve the MILP problems. The experiments are performed on a computer with an Intel® Core™ i7 @ 1.80 GHz processor and 16GB RAM.

6.2. Preliminaries

We first show the GEO decomposition results in Section 6.2.1, followed by the lower bounds of the decomposition in Section 6.2.2.

6.2.1. The GEO decomposition results

We apply the ILP model (2) to decompose the network of Fig. 4 into 3, 5, 7, and 9 regions. Fig. 5 illustrates the results of the railway network decomposition. In each subfigure of Fig. 5, a digit with a circle indicates a region, containing a number of block sections. Each directed solid/dashed line indicate an active/non-active coupling, with a digit to show the number of train service interconnections for the corresponding pair of regions.

In the late sections, all distributed computation results of the GEO decomposition are obtained based on the network decomposition shown in Fig. 5.

6.2.2. Lower bounds

We now discuss lower bounds to the problems, generated by ignoring the couplings among subproblems, so that the subproblems are fully independent and can be solved separately. The lower bound (LB) for the centralized (CEN) approach is 7586.55, which is also the optimal solution. The distributed approaches have different values, depending on the approach and the amount of subproblems. The TIN decomposition results in the loosest LB, at only 1654.00 (respectively 2797.80) for a 5 min time interval (respectively 10 min). The TRA decomposition yields a LB of 5873.50. For the GEO decomposition, a smaller number of regions generally leads to a tighter LB, i.e., from 3 to 5, 7 and 9 regions, the following LBs are computed: 7125.00, 6313.60, 5886.35, and 5338.00.

The number of the relaxed time transition constraints for the GEO decomposition is 38, 140, 220, and 258 in the case of 3, 5, 7, and 9 regions respectively. For the TRA decomposition, there are up to 22,498 capacity constraints relaxed, even though not all are decisive (i.e., removing them will not affect the optimal solution). However, we cannot identify those decisive capacity constraints in advance, and they all need to be included to guarantee a conflict-free timetable. The CDRSBK

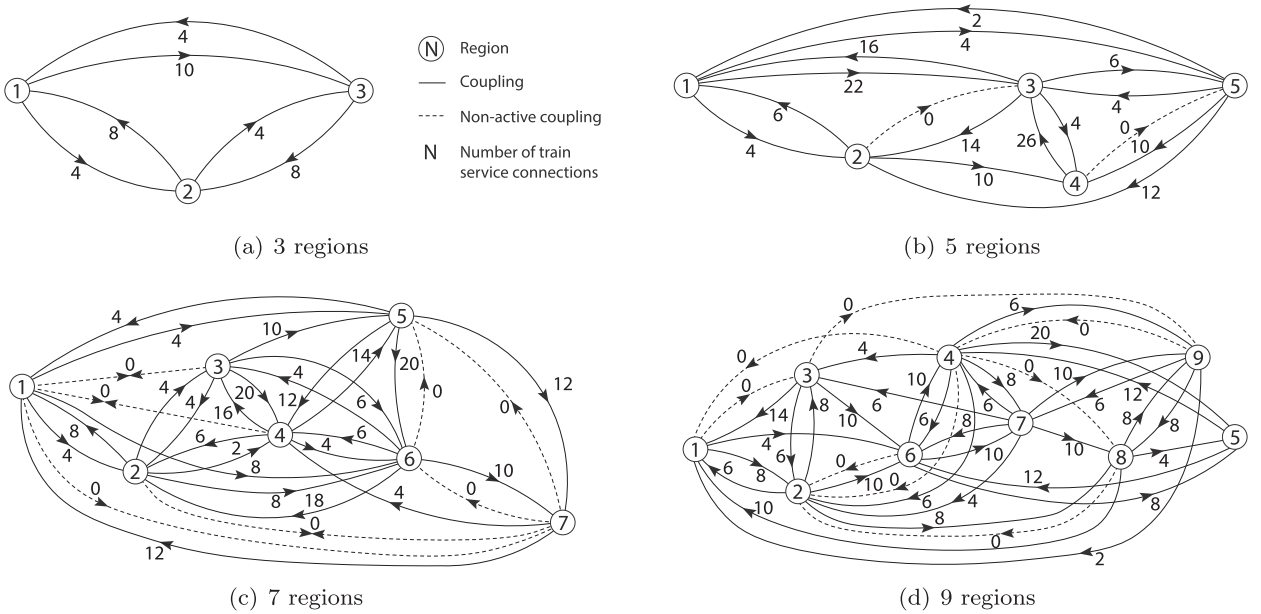


Fig. 5. Railway network decomposition (GEO decomposition) in the cases of 3, 5, 7, and 9 regions.

algorithm implicitly considers this fact for the TRA decomposition, as it supposes that all capacity constraints are not decisive, and iteratively including only the capacity constraints for those trains with detected conflicts (i.e., active couplings).

The TIN decomposition takes train primary delays into account; so the numbers of relaxed constraints are different in different delay scenarios, with an average of 23,008 or 22,787 constraints for respectively a 5 or 10 min time interval. Such a large number identifies the capacity and time transition constraints relaxed when computing the LB. It is therefore understandable why the LB of the TIN decomposition is so loose.

6.3. Distributed computation results

We now present the distributed computation results. In Section 6.3.1, we show the average results of the randomly generated 10 delay scenarios, from the viewpoints of feasibility, computational efficiency, and optimality gap, in order to compare the overall performance of the proposed decomposition and distributed solution algorithms, and with the centralized (CEN) results as well. Section 6.3.2 reports the number of changed variables and solution updates along the iterations. Section 6.3.3 presents the trade-off between the feasibility of solving subproblems independently (without any coordination) and the optimality gap of the distributed optimization results.

6.3.1. Overall performance: Feasibility, computational efficiency, and optimality

The overall performance of the decomposition and distributed algorithms is examined in terms of feasibility, computational efficiency, and optimality. Fig. 6(a) reports the optimality gap between the best feasible solution and the CEN solution, and Fig. 6(b) shows CENthe computational efficiency, measured by the cumulative computation time for finding the best feasible solution. We use colors and marks: a darker green color implies a smaller optimality gap and a shorter computation time as well; also, more plus marks are labeled to indicate better performance. In the figures, the feasibility information is delivered by the indicated black blocks with “infea” marks, which means no feasible solution is found. We report here the average of the 10 delay scenarios generated randomly.

From the feasibility perspective, the ADMM and CDRSBK algorithm can find feasible solutions for all cases of decomposition and delay scenarios. The PR algorithm can only find feasible solutions for the cases of the TRA decomposition and the GEO decomposition with 3 regions.

Let us look at the optimality gap in Fig. 6(a), which reflects the solution quality of the best feasible solution obtained. As shown, the quality of the CDRSBK solution is outstanding, reaching the optimum or being near-optimal for most cases. The ADMM algorithm performs differently, depending on the decomposition method and penalty multiplier ρ used. When looking horizontally, the ADMM solution quality is fine for the GEO decomposition with a smaller number of regions, and it gets worse when increasing the number of regions under consideration. For the TIN decomposition, the ADMM solution

(a) Optimality gap

Algorithm	Setting	Decomposition: CEN -	GEO: 3	GEO: 5	GEO: 7	GEO: 9	TRA -	TIN: 5	TIN: 10
CEN	-	+++							
ADMM	ϱ^{small}		++	+	++	+	+	–	–
	ϱ^{large}		++	+	+	–	–	–	–
	ϱ^{linear}		++	+	+	+	–	–	–
	ϱ^{exp}		++	++	++	+	+	–	–
PR	-		++	infea	infea	infea	–	infea	infea
CDRSBK	-		+++	+++	+++	+++	+++	++	+++

0% 100%

+++ best – worse

(b) Computational efficiency

Algorithm	Setting	Decomposition: CEN -	GEO: 3	GEO: 5	GEO: 7	GEO: 9	TRA -	TIN: 5	TIN: 10
CEN	-	–							
ADMM	ϱ^{small}		++	+	+	–	+	–	–
	ϱ^{large}		++	++	++	++	++	–	–
	ϱ^{linear}		++	+	+	+	+	–	–
	ϱ^{exp}		++	+	+	+	+	–	–
PR	-		++	infea	infea	infea	++	infea	infea
CDRSBK	-		++	+	++	++	+++	+++	+++

0 s 300 s 1200 s

+++ best – worse

Fig. 6. Overall performance in terms of feasibility, computational efficiency, and optimality.

quality becomes much worse. When looking vertically, the small constant multiplier ϱ^{small} or the exponentially increasing multiplier ϱ^{exp} helps to find a better solution for the ADMM algorithm, and considering a large constant multiplier ϱ^{large} results in a larger optimality gap. It is also observed from our experimental results that considering the cost-to-go function (12) can significantly improve the ADMM solution quality, achieving up to 67% improvement for a single delay scenario and to 38% on average of the 10 delay scenarios. For the sake of compactness, Fig. 6(a) does not present the results without the cost-to-go function.

When focusing on the computational efficiency shown in Fig. 6(b), the CDRSBK algorithm finds the best feasible solution most efficiently for the TRA and the TIN decomposition. The ADMM algorithm performs the worst for the TIN decomposition, consuming long computation times. Considering a large constant multiplier ϱ^{large} or a smaller number of regions makes the ADMM algorithm converge faster, and hence yields a better computational efficiency.

Considering the CEN solution as a benchmark, we summarize the overall performance of the proposed decomposition and distributed algorithms as follows:

(1) The CEN solution has a high quality (optimum) achieved at the expense of a longer computation time.
(2) The ADMM algorithm has improved computational efficiency, at the cost of reduced solution quality, in the cases of the GEO and the TRA decomposition. The ADMM algorithm performs worst in both solution quality and computational efficiency for the TIN decomposition.
(3) The PR algorithm fails in finding feasible solutions for the GEO and TIN decomposition, but it can find a feasible solution quickly in the case of the TRA decomposition with a guarantee for feasibility but not for solution quality.
(4) The CDRSBK algorithm generates optimal or near-optimal solutions most efficiently, where computational efficiency is improved without sacrificing the solution quality (or with only a little sacrifice), in comparison with the CEN solution. Among the three decomposition methods, the TRA decomposition performs best in terms of both optimality and computational efficiency, as well as feasibility.

In addition, we can observe that the performance of a decomposition method can be different in different algorithms (and vice versa). The algorithm is of comparable importance to the decomposition. The best collocation enables us to play with their advantages and promote the overall performance.

6.3.2. Number of changed variables and updates of feasible solutions

In Fig. 7(a)-(c), the number of changed variables in the solving process is shown as a function of the iteration, for a representative case of the ADMM, PR, and CDRSBK algorithm respectively. The black solid line with circles indicates the number of changed time variables (including train departure times and arrival times) at iterations, and the green solid line with triangles indicates the number of changed train order variables at iterations. The purple dashed line and the red dashed line with dots respectively report the incumbent and the best feasible solution found.

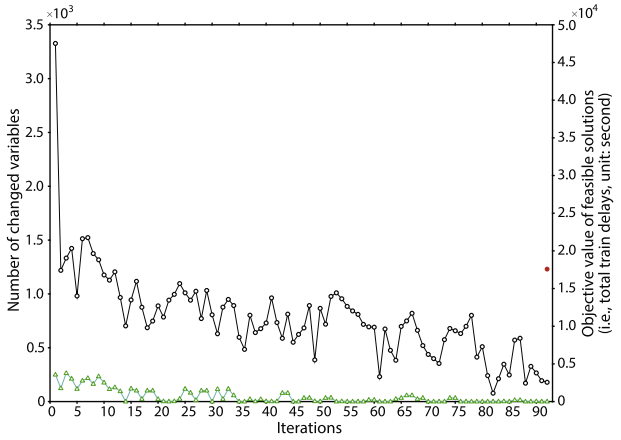
The ADMM algorithm in Fig. 7(a) shows a steady reduction of the number of changed time variables from the first iteration onwards. Similarly, train orders change heavily at the first iterations; then they decreases towards almost no change at the later iterations. The ADMM solving process is terminated when all coupling constraints hold, i.e., when a feasible solution is found. Therefore, for the ADMM algorithm, only one feasible solution is indicated, corresponding to the last iteration in Fig. 7(a).

The PR algorithm with the TRA decomposition in Fig. 7(b) reports instead irregular variations for both changed time and train order variables. This is due to the iterative behavior, ignoring the solution of the last iteration, but focusing only on the solutions at the current iteration. The procedure has a feasible solution at each iteration, but with no guarantee of quality improvement (in fact with an erratic behavior).

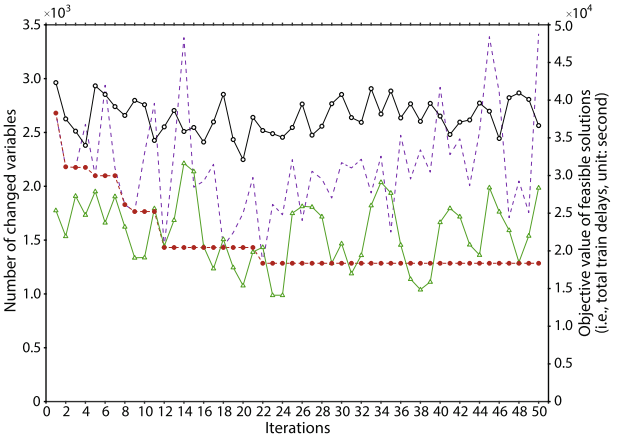
For the CDRSBK algorithm, a 5-min time interval case of the TIN decomposition (with 14 subproblems) is illustrated in Fig. 7(c). The numbers of changed time variables and train order variables are large at the first iterations and then both decrease sharply, to a constant, non-zero value. Meanwhile, train order variables do not change anymore. In fact, once a feasible solution is found, the CDRSBK algorithm generates new solutions by improving the obtained feasible solution. Each new feasible solution found guarantees improvement; therefore, the purple dashed line and the red dashed line overlap. The variations of the time variables after 3 iterations, which are not connected to any improvement of solution quality, hint at a multiplicity of optimal solutions for each subproblem, we accept one optimal solution out of the solution pool, which may be different from iteration to iteration but with the same objective value.

6.3.3. Feasibility and optimality gap

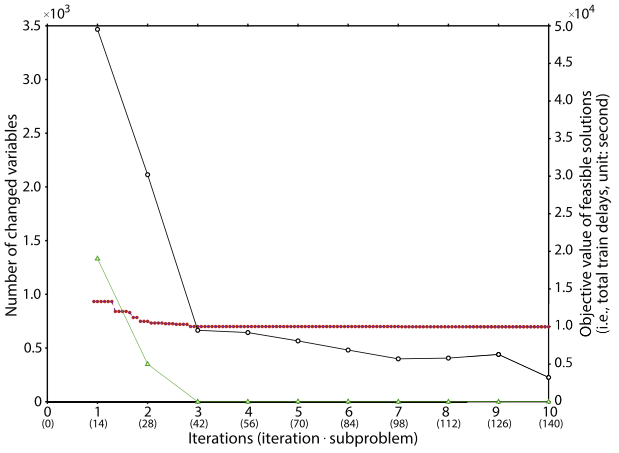
Fig. 8 is divided into three parts for the GEO, TRA, and TIN decomposition respectively, and it presents the trade-off between the feasibility of the solution without coordination of subproblems (i.e., the LB solution) and the optimality gap of the distributed optimization results. The feasibility of a LB solution is measured by the percentage of the constraints that are satisfied in the LB solution, indicated by the dashed line with triangles in red, represented on the y-axis on the right-hand side of the plot. The optimality gap is measured by the distance of the obtained feasible solution from the centralized (CEN) solution, indicated by circles, pluses, and squares for the ADMM, PR, and CDRSBK algorithm respectively, represented on the left-hand-side y-axis. Some plus symbols are missing because the PR algorithm does not find any feasible solution for these cases. For the centralized computation, there is no coupling (and thus no complicating constraint), and the problem is



(a) ADMM algorithm



(b) PR algorithm



(c) CDRSBK algorithm

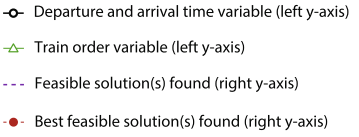


Fig. 7. Change of variables and feasible solution(s).

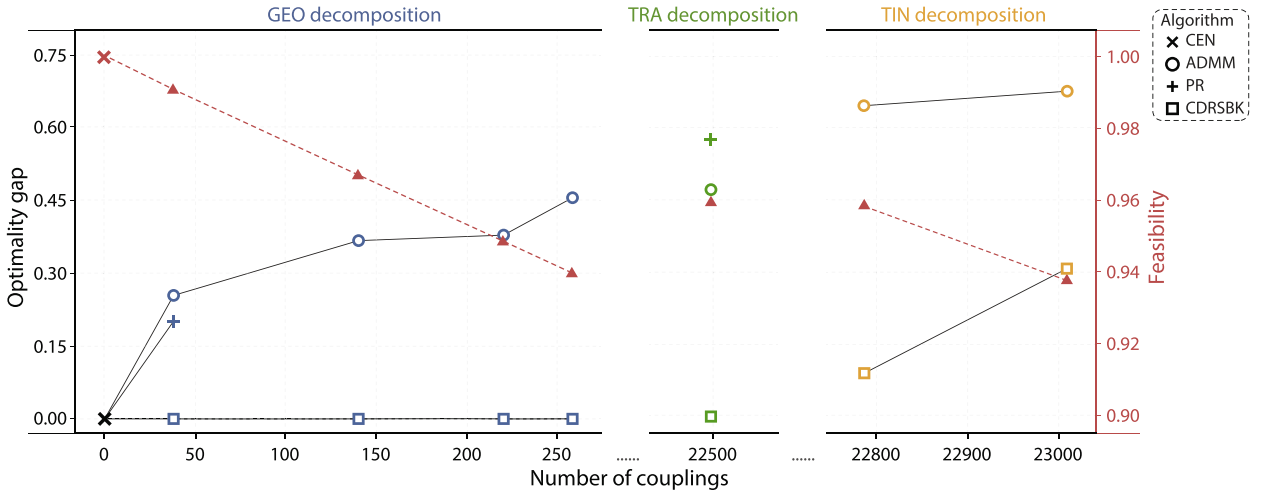


Fig. 8. Feasibility of lower bound and optimality gap of distributed optimization.

solved to optimality. So the feasibility is true (has a value of 1), and the optimality gap is zero. Fig. 8 reports the average of the 10 delay scenarios.

Overall, for a specific decomposition method, the feasibility of independently solving subproblems decreases with the increasing number of coupling constraints (i.e., the relaxed constraints) on the x-axis. For the TRA decomposition, although a large number of coupling constraints are relaxed, only a small number of them are not satisfied in the LB solution (which is reflected by the red triangle that goes up in the column of the TRA decomposition, comparing with the GEO results), revealing again the observation and discussion in Section 6.2.2. When focusing on the results of the ADMM algorithm, we can observe that the optimality gap of the best feasible solution becomes larger when relaxing more constraints, as generally recognized. Let us look at the results of the CDRSBK algorithm. For the GEO and TRA decomposition, the optimality gap does not increase with the number of coupling constraints, staying (close to) zero. In the case of the TIN decomposition, the CDRSBK optimality gap increases, but not as much as that of the ADMM algorithm. This highlights the outstanding performance of the CDRSBK algorithm.

7. Conclusions

We have introduced decomposition and distributed optimization approaches, aiming at improving the computational efficiency of solving the real-time traffic management optimization problem for large-scale railway networks. We have considered decomposing the whole optimization problem into several tractable subproblems, based on geography (GEO), trains (TRA), and time intervals (TIN). Three distributed optimization approaches have been proposed for coordinating subproblems, namely the Alternating Direction Method of Multipliers (ADMM) algorithm, the priority-rule-based (PR) algorithm, and the Cooperative Distributed Robust Safe But Knowledgeable (CDRSBK) algorithm. Their performance has been examined on a large real-life case, in terms of feasibility, computational efficiency, and optimality.

According to the experimental results, we conclude that the CDRSBK algorithm applied to the case of the TRA decomposition overall yields the best performance: a high-quality (optimal or near-optimal) solution can be found within 10 s of computation time, with a guarantee of finding feasible solution(s). This proves the potential of decomposition and distributed optimization approaches in the real-time traffic management problem.

The other approaches also have merits. The ADMM algorithm is able to generate feasible solutions, albeit their quality is far from optimality, and requires a lot of iterations and a long computation time. The PR algorithm fails in finding a feasible solution for the GEO decomposition with larger numbers of regions and for the TIN decomposition. When used with the TRA decomposition, the PR algorithm is extremely fast (2 s) in finding a first feasible solution, although with a bad solution quality, up to 60% optimality gap.

The improvement in computation speed and quality is an enabler of further relevant extensions of the problem, such as considering individual passenger flows and their nonlinear decision process (Corman, 2020); nonlinear variations of train speed (Luan et al., 2018; Xu et al., 2017); including strategic issues like line planning, rolling stock circulation, maintenance (Luan et al., 2017b; Schöbel, 2017). Finally, a fast, effective algorithm is a strong prerequisite for any real-life implementation (Mannino, 2011; Corman et al., 2018).

Currently, we let the CDRSBK algorithm search by itself the first feasible solution and improve the found feasible solution iteratively. We could also give an initial feasible solution (e.g., using the solution quickly generated by the PR algorithm in the case of the TRA decomposition) to the CDRSBK algorithm. We could then let the CDRSBK algorithm improve the given feasible solution, to assess a computational efficiency gain (warm start), and explore whether the CDRSBK algorithm can escape local minima, and find the global optimum. This leads to one direction of the future research. In addition, we can explore how to make an appropriate choice of the penalty multiplier for the ADMM algorithm, in order to improve the convergence speed as well as to make the performance of the ADMM algorithm less dependent on the given initial value of the penalty multiplier. We will also investigate the parallel implementation of the ADMM algorithm, where an improvement of the ADMM algorithm in computational efficiency can be expected. Moreover, as an extension of the TIN decomposition, we could consider a closed-loop control of the traffic to cope with stochastic disturbances and uncertain information and perform an experimental analysis of the closed-loop behavior (see Corman et al., 2018).

CRediT authorship contribution statement

Xiaojie Luan: Methodology, Software, Validation, Investigation, Writing - original draft. **Bart De Schutter:** Conceptualization, Resources, Writing - review & editing, Supervision. **Lingyun Meng:** Investigation, Writing - review & editing, Supervision. **Francesco Corman:** Methodology, Resources, Writing - review & editing, Supervision.

Acknowledgments

The authors gratefully acknowledge the support provided by the national key research project “Railroad Comprehensive Efficiency and Service Level Improvement Technology with High Speed Railway Network (Grant No. 2018YFB1201403)” in China, Science & Technology Development Plan of China Railway Corporation (2020F019) and Beijing Municipal Natural Science Foundation (L181007). The work of the first author is also supported by China Scholarship Council under Grant 201507090058.

Appendix A. An MILP approach for addressing the real-time traffic management problem

This section introduces an MILP model based on the time-continuous formulation method for addressing the railway traffic management problem, where train routes, orders, and departure and arrival times are not considered static, nor decided based on departure times at stations, but instead optimized based on the current delay and traffic situation.

The following assumptions are made: (1) each train is represented by a virtual dot for simplicity; (2) for a double-track segment between two stations, each track is described as a sequence of directional block sections; (3) for a single-track segment, the track is modeled as a series of bi-directional block sections, and the block sections in different directions are considered to be identical; (4) only one train is permitted on a block section at any given time; (5) interlocking is not considered but can be included by introducing incompatible sets of block sections, i.e., if one block section is occupied by a train, then the block sections in its incompatible set are also blocked for other trains to use. More details of the model assumptions can be found in Luan et al. (2017a). Table 2 lists the sets, subscripts, input parameters, and decision variables used by the MILP model.

The objective is to minimize the sum over all trains of the delay times (including early delays) at all visited stations, i.e., minimizing the deviation from the planned train timetable:

$$\min\ Z = \sum_{f\in F}\sum_{(i,j)\in E_f^{\text{stop}}} \left| d_{f,i,j} - w_{f,i,j} - D_{f,i,j} \right|, \tag{A.1a}$$

The objective function (A.1a) includes the absolute value operator. This can be handled directly by using the "absolute value function" under the environment of the ILOG CPLEX Optimization Studio. Alternatively, we can reformulate (A.1a) as $\min\ Z = \sum_{f\in F}\sum_{(i,j)\in E_f^{\text{stop}}} z_{f,i,j}$ while adding the constraints of $z_{f,i,j} \geq d_{f,i,j} - w_{f,i,j} - D_{f,i,j}$ and $z_{f,i,j} \geq -d_{f,i,j} + w_{f,i,j} + D_{f,i,j}$. The nature of the optimization problem remains linear.

The following three constraints:

$$\sum_{j:(\psi_f,j)\in E_f} x_{f,\psi_f,j} = 1, \qquad \forall f \in F \tag{A.1b}$$

$$\sum_{i:(i,j)\in E_f} x_{f,i,j} = \sum_{k:(j,k)\in E_f} x_{f,j,k}, \qquad \forall f \in F,\ j \in V\backslash\{\psi_f, \sigma_f\} \tag{A.1c}$$

Table 2
Sets, subscripts, input parameters, and decision variables.

Symbol	Description
	Subscripts and sets
F	set of trains, $\|F\|$ is the number of trains
V	set of nodes, $\|V\|$ is the number of nodes
E	set of links (i.e., block sections), $E \subseteq V \times V$, $\|E\|$ is the number of links
f	train index, $f \in F$
i, j, k	node index, $i, j, k \in V$
e	link index, denoted by $e = (i, j) \in E$
E_f	set of links that train f may use, $E_f \subseteq E$
E_f^{stop}	set of links in which train f should stop, $E_f^{\text{stop}} \subseteq E_f$, $\|E_f^{\text{stop}}\|$ is the number of stops of train f
	Input parameters
ψ_f	origin node of train f
σ_f	destination node of train f
ϕ_f^{pri}	primary delay time of train f at its origin node
ϕ_f	planned departure time of train f at its origin node
u_f	direction of train f
$D_{f,i,j}$	planned arrival time of train f on link (i, j), $(i, j) \in E_f^{\text{stop}}$
$\tau_{f,i,j}^{\min}$	free flow running time of train f to drive through link (i, j)
$w_{f,i,j}^{\min}$	minimum dwell time of train f on link (i, j)
$w_{f,i,j}^{\max}$	maximum dwell time of train f on link (i, j)
$g_{f,i,j}$	safety time interval between occupancy of link (i, j) and arrival of train f, including setup time, sight and reaction time, and approach time
$h_{f,i,j}$	safety time interval between departure of train f and release of link (i, j), including clearing time and release time
M	a sufficiently large positive number
	Decision variables
$a_{f,i,j}$	arrival time of train f at link (i, j)
$d_{f,i,j}$	departure time of train f from link (i, j)
$w_{f,i,j}$	dwell time of train f on link (i, j)
$x_{f,i,j}$	binary route selection variable: $x_{f,i,j} = 1$ if train f uses link (i, j), and otherwise $x_{f,i,j} = 0$
$\theta_{f,f',i,j}$	binary train ordering variables: $\theta_{f,f',i,j} = 1$ if train f' arrives at link (i, j) after train f, and otherwise $\theta_{f,f',i,j} = 0$

$$\sum_{j:(i,\sigma_f)\in E_f} x_{f,i,\sigma_f} = 1, \qquad \forall f \in F \tag{A.1d}$$

ensure the movement of a train on the network, from its origin node, via the intermediate stops, and to its destination node respectively.

The following constraint ensures that trains do not leave their origins before the earliest departure time,

$$a_{f,\psi_f,j} \geq \left(\phi_f + \phi_f^{\text{pri}}\right) \cdot x_{f,\psi_f,j}, \quad \forall f \in F, \left(\psi_f, j\right) \in E_f, \tag{A.1e}$$

i.e., the sum of the planned departure time and the primary delay time.

To force the transition of a train within a link, i.e., the train departure time from a link is greater (later) than its arrival time at the same link, the following constraint is proposed:

$$d_{f,i,j} \geq a_{f,i,j}, \quad \forall f \in F, (i,j) \in E_f. \tag{A.1f}$$

If two adjacent links (i, j) and (j, k) are consecutively used by train f, then we should ensure that the departure time of train f from link (i, j) equals its arrival time at link (j, k), formulated as follows:

$$\sum_{i:(i,j)\in E_f} d_{f,i,j} = \sum_{k:(j,k)\in E_f} a_{f,j,k}, \quad \forall f \in F, j \in V \backslash \left\{\psi_f, \sigma_f\right\} \tag{A.1g}$$

The train dwell time constraint

$$w_{f,i,j}^{\min} \cdot x_{f,i,j} \leq w_{f,i,j} \leq w_{f,i,j}^{\max} \cdot x_{f,i,j}, \quad \forall f \in F, (i,j) \in E_f \tag{A.1h}$$

guarantees the required minimum and maximum dwell times at stations. The minimum dwell time is the time required to complete the processes of passengers boarding and alighting, goods loading and unloading, etc. The maximum dwell time is used to avoid unallowed dwell events of trains. If a train is allowed to stop on a link, then the corresponding maximum dwell time is set to be sufficiently large; if a train is not allowed to stop on some particular links, then the corresponding maximum dwell times are set to be zero.

The train travel time constraint

$$d_{f,i,j} - w_{f,i,j} - a_{f,i,j} \geq \tau_{f,i,j}^{\min} \cdot x_{f,i,j}, \quad \forall f \in F, (i,j) \in E_f \tag{A.1i}$$

enforces the required train free-flow running time.

The train order variables $\theta_{f,f',i,j}$ and the route selection variables $x_{f,i,j}$ are linked by the following two constraints:

$$\theta_{f,f',i,j} + \theta_{f',f,i,j} \geq x_{f,i,j} + x_{f',i,j} - 1, \quad \forall f, f' \in F, f \neq f', (i,j) \in E_f \cap E_{f'}, \tag{A.1j}$$

$$\theta_{f,f',i,j} + \theta_{f',f,i,j} \leq 3 - x_{f,i,j} - x_{f',i,j}, \quad \forall f, f' \in F, f \neq f', (i,j) \in E_f \cap E_{f'}. \tag{A.1k}$$

Specifically, constraints (A.1j)-(A.1k) make sure that if and only if both trains f and f' traverse the link (i, j), i.e., $x_{f,i,j} = x_{f',i,j} = 1$, then both the inequalities reduce to an equality: $\theta_{f,f',i,j} + \theta_{f',f,i,j} = 1$. This equality indicates that either train f' arrives at link (i, j) after train f or train f arrives at link (i, j) after train f'.

With the following two constraints,

$$a_{f',i,j} - g_{f',i,j} + \left[3 - x_{f,i,j} - x_{f',i,j} - \theta_{f,f',i,j}\right] \cdot M \geq d_{f,i,j} + h_{f,i,j}, \quad \forall f \in F, f' \in F, f \neq f', u_f = u_{f'}, (i,j) \in E_f \cap E_{f'} \tag{A.1l}$$

$$a_{f',j,i} - g_{f',j,i} + \left[3 - x_{f,i,j} - x_{f',i,j} - \theta_{f,f',i,j}\right] \cdot M \geq d_{f,i,j} + h_{f,i,j}, \quad \forall f \in F, f' \in F, f \neq f', u_f \neq u_{f'}, (i,j) \in E_f, (j,i) \in E_{f'} \tag{A.1m}$$

we ensure that any pair of trains using one link in the same or different direction respectively are conflict-free. If two trains are running on the same link, the successive train can only access to the link after the link is released for the proceeding train.

The MILP problem (A.1) can be solved by a standard MILP solver, e.g., CPLEX or Gurobi.

Appendix B. An illustrative example

In this appendix, we use a small instance to explain the proposed decomposition methods and algorithms. As illustrated in Fig. 9, the instance includes 4 trains following the pre-defined routes, i.e., train f_1: $e_1 \to e_2 \to e_4$, train f_2 and f_3: $e_1 \to e_3 \to e_5$, and train f_4: $e_3 \to e_5$.

We now illustratively explain the formulation of the ILP problem proposed in Section 3.1. We can write the set of block sections as $E = \{e_1, e_2, e_3, e_4, e_5\}$. The route matrix B_{f_1} and the variable vector β_{f_1} for train f_1 and the variable vector μ for block sections can be expressed as

$$B_{f_1} = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \end{bmatrix}, \ \beta_{f_1} = \begin{bmatrix} (\beta_{f_1})_1 \\ (\beta_{f_1})_2 \end{bmatrix}, \text{ and } \mu = \begin{bmatrix} \mu_1 & \mu_2 & \mu_3 & \mu_4 & \mu_5 \end{bmatrix}^\top.$$

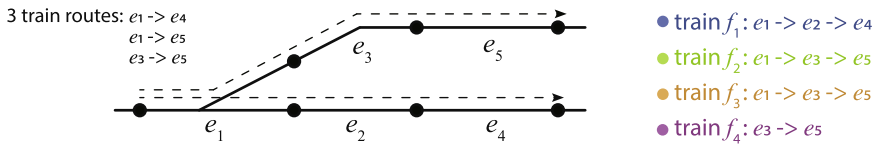


Fig. 9. A small instance.

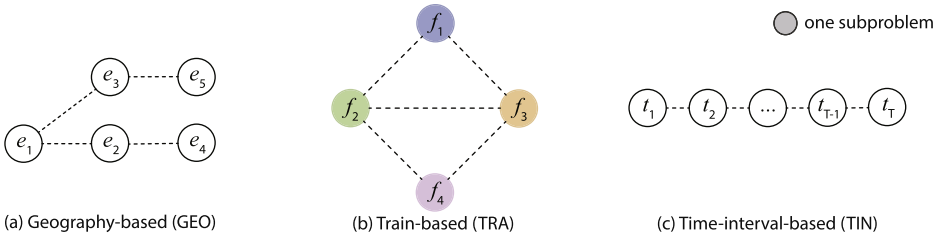


Fig. 10. Subproblems and couplings.

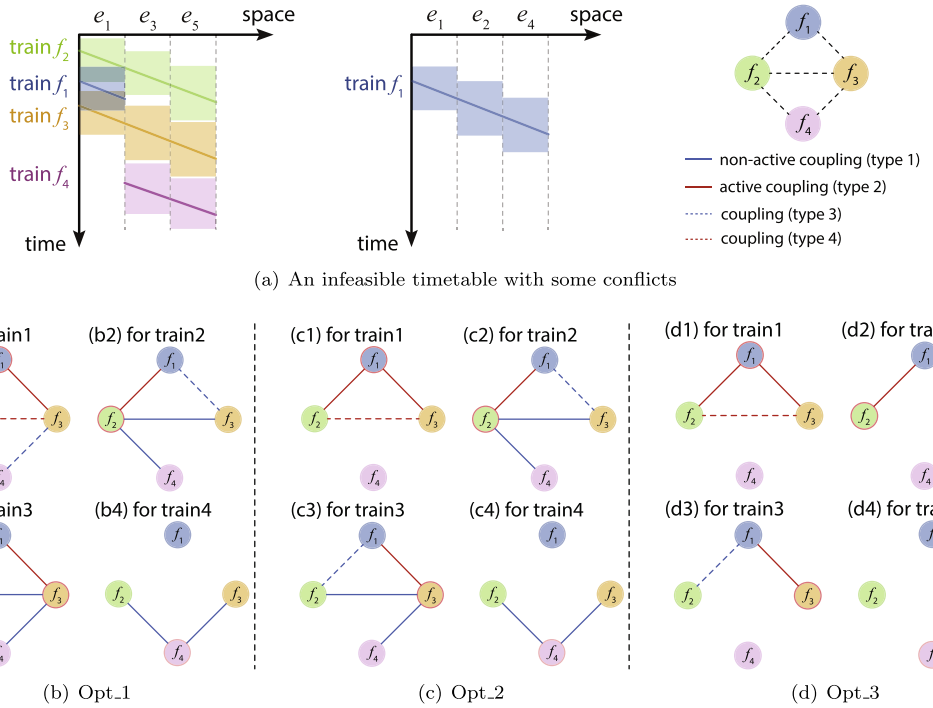
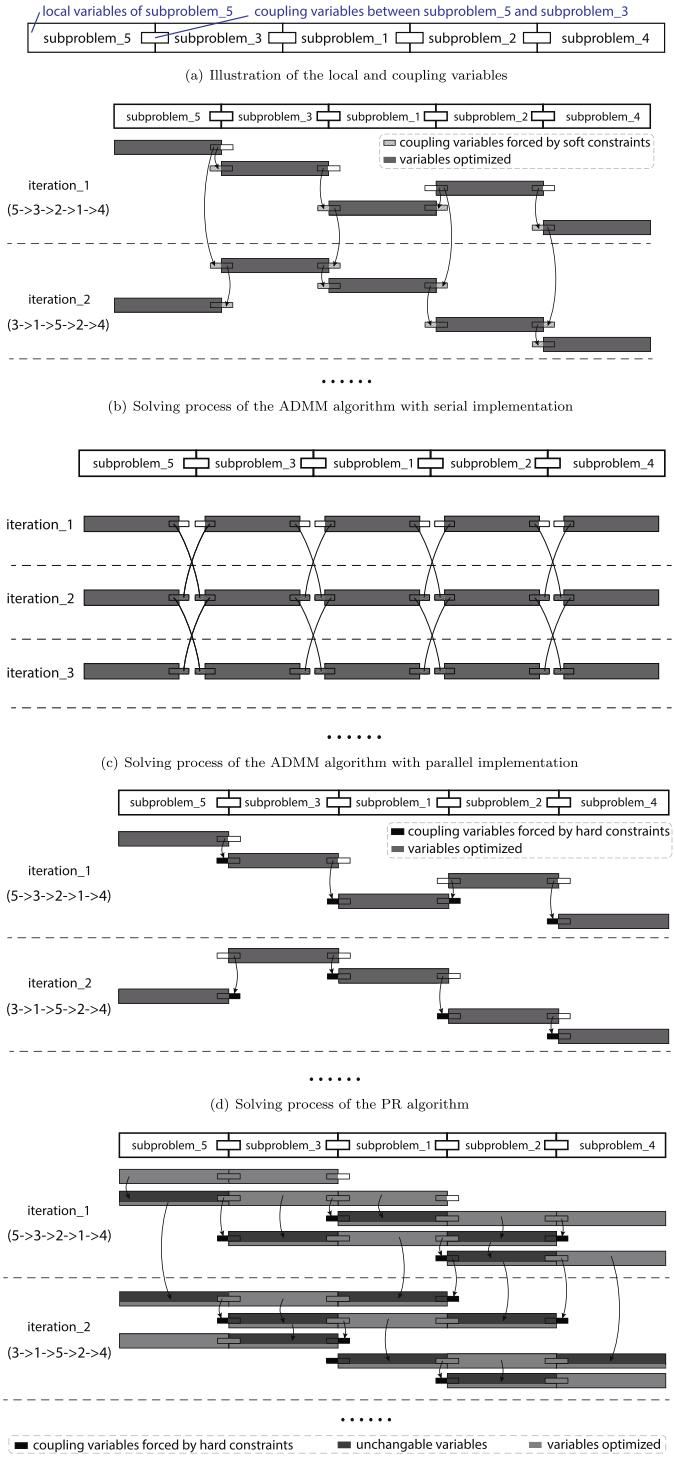


Fig. 11. Three options of the CDRSBK algorithm with the TRA decomposition.

Consider the consecutive block sections e_1 and e_2 in the route of train f_1; constraint (2b) results in the inequality $\frac{|\mu_1-\mu_2|}{|R|-1} \leq (\beta_{f_1})_1$. The left-hand side of the inequality is strictly in the range of [0,1]. If the two block sections are assigned to the same region, i.e., $\mu_1 = \mu_2$, then we will have $(\beta_{f_1})_1 = 0$ (as we are solving a minimization problem). If block sections e_1 and e_2 belong to different regions, i.e., $\mu_1 \neq \mu_2$, then we will have $(\beta_{f_1})_1 = 1$, as the left-hand side of the inequality is in the range of (0,1] and β_{f_1} is a binary vector. Constraints (2d)-(2e) are used to avoid the solutions like $\mu = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}^\top$.

We now illustrate the three decomposition methods. Let us assume $|R| = 5$, i.e., we consider 5 regions and each region contains only one block section, and let us denote T as the number of subproblems for the TIN decomposition. By applying the three proposed decomposition methods, the resulting subproblems and (primary) couplings are shown in Fig. 10. As illustrated, the GEO decomposition results in 5 subproblems, corresponding to the 5 block sections respectively; the TRA decomposition leads to 4 subproblems, i.e., the 4 trains; and the TIN decomposition gives T subproblems that are connected in a sequence of time horizon.



(a) Illustration of the local and coupling variables

(b) Solving process of the ADMM algorithm with serial implementation

(c) Solving process of the ADMM algorithm with parallel implementation

(d) Solving process of the PR algorithm

(e) Solving process of the CDRSBK algorithm

Fig. 12. Illustration of the three algorithms, considering the GEO decomposition.

We now interpret the three options of defining the four types of couplings in the CDRSBK algorithm with the TRA decomposition, as discussed in Section 5.1. Let us assume an infeasible timetable of Fig. 11(a), which can be generated by independently scheduling trains one-by-one without coordination. The three options are illustrated in Fig. 11(b)-Fig. 11(d) respectively. Let us now focus on scheduling train f_1 (i.e., solving subproblem f_1).

For Opt_1, let us first focus on solving subproblem f_1, as shown in Fig. 11(b1). The coupling between train f_1 and f_2 is considered to be active (Type_2), because a conflict between train f_1 and train f_2 appears in the timetable of Fig. 11(a). Both f_2 and f_3 are actively coupled subproblems of f_1; so a Type_4 coupling exists between f_2 and f_3. Train f_1 and train f_4 use completely different block sections; therefore no coupling exists between them. Train f_4 uses some block sections that are also used by train f_2 and f_3, but no conflict is detected. So their couplings are recognized as a Type 3 coupling when focusing on solving subproblem f_1. Train f_2 uses the same block sections as all the other trains, but it has a conflict with train f_1 only; therefore, when scheduling train f_2 as shown in Fig. 11(b2), the coupling between f_2 and f_1 is considered to be Type_2, and the coupling between f_2 and f_3 (and f_4) is recognized as Type_1.

For Opt_2, we identify a coupling to be Type_3 only if the coupling is recognized as an active coupling when solving some other subproblems. As shown in Fig. 11(c1), the Type_3 coupling between f_2 and f_4 is removed in comparison with Fig. 11(b1), because the coupling between f_2 and f_4 is recognized to be non-active when solving subproblem f_2 or f_4 as shown in Fig. 11(c2) and (c4). In Fig. 11(c2), the Type_3 coupling between f_1 and f_3 stays because this coupling is recognized to be active when solving subproblem f_1 or f_3 as shown in Fig. 11(c1) and (c3).

For Opt_3, we consider no coupling if there is no conflict. A Type_1 (non-active) coupling describes that no conflict occurs between the corresponding two subproblems. Therefore, as illustrated in Fig. 11(d), we remove all Type_1 couplings based on the coupling architecture of Opt_2 given in Fig. 11(c). Type_3 and Type_4 couplings are still defined in a way that is same as Opt_1 and Opt_2.

We next explain the three proposed distributed algorithms by considering the GEO decomposition as an example. Fig. 10(a), which shows 5 subproblems and their couplings of the GEO decomposition, is re-drawn as Fig. 12(a) to express the local and coupling variables. Fig. 12(a) is used to graphically explain the solution process of the three algorithms.

Fig. 12(b) and Fig. 12(c) illustrate the solving process of the ADMM algorithm with serial and parallel implementation respectively, based on the small instance. For the serial implementation, we randomly generate orders of subproblems in each iteration and solve each subproblem according to the orders through coordination with other neighboring subproblems. As shown, we first solve subproblem_5, then the obtained solution of subproblem_5 is given as a soft constraint to solve subproblem_3. Subproblem_1 is solved based on the solutions of both subproblem_2 and subproblem_3, as soft constraints as well. In each iteration, we always consider the latest solution obtained, e.g., when solving subproblem_1 in iteration_2, we use the solution of subproblem_2 obtained in the last iteration (i.e., iteration_1), as subproblem_2 has not been solved in iteration_2, and we use solution of subproblem_3 obtained in iteration_2, as it has been solved in current iteration_2. For the parallel implementation, as shown in Fig. 12(c), we only consider the solution obtained in the last iteration, rather than the latest solution (updated in the current iteration).

The solving process of the PR algorithm is illustrated in Fig. 12(d) for the small instance. As shown, subproblem_5 is first solved at iteration_1. After solving subproblem_5, the solution of subproblem_5 is given as a hard constraint, indicated by a black block, for solving subproblem_3. As we only respect the solutions obtained at current iteration, there is no explicit interaction between iterations. The only interaction between iterations is that the priority of subproblems in an iteration is determined based on the solution (delays of subproblems) obtained at the last iteration.

Fig. 12(e) illustrates the solving process of the CDRSBK algorithm. In this case, all couplings between two neighboring subproblems are considered to be active (Type_2). As shown, in iteration_1, with the focus on subproblem_5, we first solve subproblem_5 and subproblem_3. Then focusing on subproblem_3, we solve subproblem_5, 3, and 1, but only part of variables in subproblem_5 can be changed. Dark gray indicates unchangeable variables, coming from the last solution obtained for the corresponding subproblem, and light gray indicates changeable variables. When addressing subproblem_2, some variables in subproblem_1 are unchangeable, including the coupling variables related to subproblem_3. Therefore, the coupling between subproblem_1 and subproblem_3 will also be satisfied as a hard constraint when solving subproblem_2.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.trb.2020.09.004

References

Beltran Royoa, C., Heredia, F.J., 2002. Unit commitment by augmented lagrangian relaxation: testing two decomposition approaches. J. Optim. Theory Appl. 112 (2), 295–314.
Boccia, M., Mannino, C., Vasilyev, I., 2013. The dispatching problem on multitrack territories: heuristic approaches based on mixed integer linear programming. Networks 62 (4), 315–326.
Borndörfer, R., Klug, T., Lamorgese, L., Mannino, C., Reuther, M., Schlechte, T., 2018. Handbook of Optimization in the Railway Industry. Springer.
Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundat. Trend. Mach. Learn. 3 (1), 1–122.
Brännlund, U., Lindberg, P.O., Nou, A., Nilsson, J.-E., 1998. Railway timetabling using lagrangian relaxation. Transp. Sci. 32 (4), 358–369.
Cacchiani, V., Huisman, D., Kidd, M., Kroon, L., Toth, P., Veelenturf, L., Wagenaar, J., 2014. An overview of recovery models and algorithms for real-time railway rescheduling. Transp. Res. Part B 63, 15–37.
Caimi, G., Fuchsberger, M., Laumanns, M., Lüthi, M., 2012. A model predictive control approach for discrete-time rescheduling in complex central railway station areas. Comput. Oper. Res. 39 (11), 2578–2593.
Corman, F., 2020. Interactions and equilibrium between rescheduling train traffic and routing passengers in microscopic delay management: a game theoretical study. Transp. Sci..
Corman, F., D'Ariano, A., Hansen, I.A., Pacciarelli, D., 2011. Optimal multi-class rescheduling of railway traffic. J. Rail Transp. Plann. Manage. 1 (1), 14–24.

Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2012. Bi-objective conflict detection and resolution in railway traffic management. Transp. Res. Part C 20 (1), 79–94.
Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2012. Optimal inter-area coordination of train rescheduling decisions. Transp. Res. Part E 48 (1), 71–88.
Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2014. Dispatching and coordination in multi-area railway traffic management. Comput. Oper. Res. 44, 146–160.
Corman, F., D'Ariano, A., Pranzo, M., Hansen, I.A., 2011. Effectiveness of dynamic reordering and rerouting of trains in a complicated and densely occupied station area. Transp. Plan. Technol. 34 (4), 341–362.
Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2010. Centralized versus distributed systems to reschedule trains in two dispatching areas. Publ. Transp. 2 (3), 219–247.
Corman, F., Meng, L., 2015. A review of online dynamic models and algorithms for railway traffic management. IEEE Trans. Intell. Transp. Syst. 16 (3), 1274–1284.
Corman, F., Quaglietta, E., Goverde, R.M., 2018. Automated real-time railway traffic control: an experimental analysis of reliability, resilience and robustness. Transp. Plan. Technol. 41 (4), 421–447.
D'Ariano, A., Pacciarelli, D., Pranzo, M., 2007. A branch and bound algorithm for scheduling trains in a railway network. Eur. J. Oper. Res. 183 (2), 643–657.
Dollevoet, T., Corman, F., D'Ariano, A., Huisman, D., 2014. An iterative optimization framework for delay management and train scheduling. Flexible Serv. Manuf. J. 26 (4), 490–515.
Dollevoet, T., Huisman, D., Schmidt, M., Schöbel, A., 2012. Delay management with rerouting of passengers. Transp. Sci. 46 (1), 74–89.
D'Ariano, A., Pranzo, M., 2009. An advanced real-time train dispatching system for minimizing the propagation of delays in a dispatching area under severe disturbances. Netw. Spatial Econ. 9 (1), 63–84.
Evangelopoulos, V.A., Georgilakis, P.S., Hatziargyriou, N.D., 2016. Optimal operation of smart distribution networks: areview of models, methods and future research. Electr. Power Syst. Res. 140, 95–106.
Fang, W., Yang, S., Yao, X., 2015. A survey on problem models and solution approaches to rescheduling in railway networks. IEEE Trans. Intell. Transp. Syst. 16 (6), 2997–3016.
Findler, N.V., Stapp, J., 1992. Distributed approach to optimized control of street traffic signals. J. Transp. Eng. 118 (1), 99–110.
Ginkel, A., Schöbel, A., 2007. To wait or not to wait? the bicriteria delay management problem in public transportation. Transp. Sci. 41 (4), 527–538.
Hansen, I.A., Pachl, J., 2014. Railway Timetabling & Operations: Analysis, Modelling, Optimisation, Simulation, Performance evaluation. Eurailpress, Hamburg, Germany.
Harrod, S., 2011. Modeling network transition constraints with hypergraphs. Transp. Sci. 45 (1), 81–97.
Kecman, P., Corman, F., D'Ariano, A., Goverde, R.M., 2013. Rescheduling models for railway traffic management in large-scale networks. Pub. Transp. 5 (1–2), 95–123.
Kersbergen, B., van den Boom, T., De Schutter, B., 2016. Distributed model predictive control for railway traffic management. Transp. Res. Part C 68, 462–489.
Kuwata, Y., How, J.P., 2011. Cooperative distributed robust trajectory optimization using receding horizon milp. IEEE Trans. Control Syst. Technol. 19 (2), 423–431.
Lamorgese, L., Mannino, C., 2015. An exact decomposition approach for the real-time train dispatching problem. Oper. Res. 63 (1), 48–64.
Lamorgese, L., Mannino, C., Piacentini, M., 2016. Optimal train dispatching by benders-like reformulation. Transp. Sci. 50 (3), 910–925.
Luan, X., Corman, F., Meng, L., 2017. Non-discriminatory train dispatching in a rail transport market with multiple competing and collaborative train operating companies. Transp. Res. Part C 80, 148–174.
Luan, X., Miao, J., Meng, L., Corman, F., Lodewijks, G., 2017. Integrated optimization on train scheduling and preventive maintenance time slots planning. Transp. Res. Part C 80, 329–359.
Luan, X., Wang, Y., De Schutter, B., Meng, L., Lodewijks, G., Corman, F., 2018. Integration of real-time traffic management and train control for rail networks - part 1: optimization problems and solution approaches. Transp. Res. Part B 115, 41–71.
Mannino, C., 2011. Real-time traffic control in railway systems. In: Proceedings of the 11th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.
Meng, L., Zhou, X., 2014. Simultaneous train rerouting and rescheduling on an n-track network: a model reformulation with network-based cumulative flow variables. Transp. Res. Part B 67, 208–234.
Molzahn, D.K., Dörfler, F., Sandberg, H., Low, S.H., Chakrabarti, S., Baldick, R., Lavaei, J., 2017. A survey of distributed optimization and control algorithms for electric power systems. IEEE Trans. Smart Grid 8 (6), 2941–2962.
Negenborn, R.R., De Schutter, B., Hellendoorn, J., 2008. Multi-agent model predictive control for transportation networks: serial versus parallel schemes. Eng. Appl. Artif. Intell. 21 (3), 353–366.
Palomar, D.P., Chiang, M., 2006. A tutorial on decomposition methods for network utility maximization. IEEE J. Sel. Areas Commun. 24 (8), 1439–1451.
Pellegrini, P., Marlière, G., Pesenti, R., Rodriguez, J., 2015. RECIFE-MILP: An effective MILP-based heuristic for the real-time railway traffic management problem. IEEE Trans. Intell. Transp. Syst. 16 (5), 2609–2619.
Pellegrini, P., Marlière, G., Rodriguez, J., 2014. Optimal train routing and scheduling for managing traffic perturbations in complex junctions. Transp. Res. Part B 59, 58–80.
Research Collection ETH Zurich, 2020. Instance details and experimental results. https://www.research-collection.ethz.ch/handle/20.500.11850/426430.
Rodriguez, J., 2000. Empirical study of a railway traffic management constraint programming model. WIT Trans. Built Environ. 50.
Samà, M., Corman, F., Pacciarelli, D., et al., 2017. A variable neighbourhood search for fast train scheduling and routing during disturbed railway traffic situations. Comput. Oper. Res. 78, 480–499.
Schachtebeck, M., Schöbel, A., 2010. To wait or not to wait-and who goes first? delay management with priority decisions. Transp. Sci. 44 (3), 307–321.
Schöbel, A., 2007. Integer Programming Approaches for Solving the Delay Management Problem. Springer.
Schöbel, A., 2017. An eigenmodel for iterative line planning, timetabling and vehicle scheduling in public transportation. Transp. Res. Part C 74, 348–365.
Strotmann, C., 2008. Railway scheduling problems and their decomposition. Universität Osnabrück.
Törnquist, J., Persson, J.A., 2007. N-Tracked railway traffic re-scheduling during disturbances. Transp. Res. Part B 41 (3), 342–362.
Van Thielen, S., Corman, F., Vansteenwegen, P., 2018. Considering a dynamic impact zone for real-time railway traffic management. Transp Res. Part B 111, 39–59.
Wangermann, J.P., Stengel, R.F., 1996. Distributed optimization and principled negotiation for advanced air traffic management. In: Proceedings of the IEEE International Symposium on Intelligent Control, pp. 156–161.
Xu, P., Corman, F., Peng, Q., Luan, X., 2017. A train rescheduling model integrating speed management during disruptions of high-speed traffic under a quasi-moving block system. Transp. Res. Part B 104, 638–666.
Zhan, S., Kroon, L.G., Veelenturf, L.P., Wagenaar, J.C., 2015. Real-time high-speed train rescheduling in case of a complete blockage. Transp. Res. Part B 78, 182–201.
Zhan, S., Kroon, L.G., Zhao, J., Peng, Q., 2016. A rolling horizon approach to the high speed train rescheduling problem in case of a partial segment blockage. Transp. Res. Part E 95, 32–61.