Distributed optimization approaches for the integrated problem of real-time railway traffic management and train control

Xiaojie Luan [a,1], Bart De Schutter [b], Ton van den Boom [b], Lingyun Meng [c],
Gabriel Lodewijks [d], Francesco Corman [e]
[a] Section Transport Engineering and Logistics, Delft University of Technology
Mekelweg 2, 2628 CD, Delft, the Netherlands
[1] E-mail: x.luan@tudelft.nl, Tel.: +31 (0) 15 27 87294
[b] Delft Center for Systems and Control, Delft University of Technology
[c] State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University
[d] School of Aviation, Faculty of Science, University of New South Wales
[e] Institute for Transport Planning and Systems (IVT), ETH Zürich

Abstract

This paper introduces distributed optimization approaches, with the aim of improving the computational efficiency of an integrated optimization problem for large-scale railway networks. We first propose three decomposition methods to decompose the whole problem into a number of subproblems, namely a geography-based (GEO), a train-based (TRA), and a time-interval-based (TIN) decomposition respectively. As a result of the decomposition, couplings exist among the subproblems, and the presence of these couplings leads to a non-separable structure of the whole problem. To handle this issue, we further introduce three distributed optimization approaches. An Alternating Direction Method of Multipliers (ADMM) algorithm is developed to solve each subproblem through coordination with the other subproblems in an iterative manner. A priority-rule-based (PR) algorithm is proposed to sequentially and iteratively solve the subproblems in a priority order with respect to the solutions of the other subproblems solved with a higher priority. A Cooperative Distributed Robust Safe But Knowledgeable (CDRSBK) algorithm is presented, where four types of couplings are defined and each subproblem is iteratively solved together with its actively coupled subproblems. Experiments are conducted based on the Dutch railway network to comparatively examine the performance of the three proposed algorithms with the three decomposition methods, in terms of feasibility, computational efficiency, solution quality, and estimated optimality gap. Overall, the combinations GEO-ADMM, TRA-ADMM, and TRA-CDRSBK yield better performance. Based on our findings, a feasible solution can be found quickly by using TRA-ADMM, and then a better solution can be potentially obtained by GEO-ADMM or TRA-CDRSBK at the cost of more CPU time.

Keywords

Distributed optimization, Decomposition, Integration of real-time traffic management and train control, Mixed-integer linear programming (MILP), Large-scale

1 Introduction

Real-time traffic management is of great importance to limit the negative consequences caused by perturbations occurring in real-time railway operations. Train control problem reflects the traffic control by defining speed profiles to let the delayed trains reach the stations at the times specified by the traffic management problem. Due to the real-time nature, a solution is required in a very short computation time for dealing with delayed and canceled train services and for evacuating delayed and stranded passengers as quickly as possible.

The real-time traffic management problem has been studied extensively in the literature, and we refer to the review papers by Cacchiani et al. (2014) and Corman and Meng (2015). There are many optimization approaches available for the railway traffic management problem, using different formulation methods, e.g., the alternative-graph-based method by D'Ariano et al. (2007) and the cumulative flow variable based method by Meng and Zhou (2014), and having different focuses, e.g., considering multiple classes of running traffic in Corman et al. (2011) and integrating train control in Luan et al. (2018). These approaches often lead to large and rather complex optimization problems, especially when considering microscopic details or when integrating traffic management with other problems (e.g., train control problem). They mostly have excellent performance on small-scale cases, where optimality can be achieved in a short computation time. However, when enlarging the scale of the case, the computation time for finding a solution or for proving the optimality of a solution increases exponentially in general.

Distributed optimization approaches have gained a lot of attention to face the need for fast and efficient solutions for problems arising in the context of large-scale networks, such as utility maximization problems. We refer to Nedic and Ozdaglar (2010) and Meinel et al. (2014) for more details. The main idea is to solve the problems either serially or in parallel to jointly minimize a separable objective function, usually subject to coupling constraints that force the different problems to exchange information during the optimization process. In the literature, these approaches have been widely studied in many fields. In transportation systems, they have been explored for controlling road traffic (Findler and Stapp, 1992), for managing air traffic (Wangermann and Stengel, 1996), and for railway traffic (Kersbergen et al., 2016). Kersbergen et al. (2016) focused on the railway traffic management problem with macroscopic details and considered a geography-based decomposition. Lamorgese et al. (2016) proposed a Benders'-like decomposition within a master/slave scheme to address the train dispatching problem. The master and the slave problems correspond to a macroscopic and microscopic representation of the railway.

Bad computational efficiency is one limitation that (integrated) optimization approaches have for large-scale networks. Overcoming this limitation will promote the application of such optimization approaches in practice. Thus, we aim at improving the computational efficiency of solving such (integrated) optimization problems by using distributed optimization approaches. The optimization problem that we focus on in this paper is a mixed-integer linear programming (MILP) problem, developed in our previous work (Luan et al., 2018), where the traffic-related variables (i.e., a set of times, orders, and routes to be followed by trains) and the train-related variables (i.e., speed trajectories) are optimized simultaneously.

In this paper, we consider three decomposition methods, namely a geography-based (GEO) decomposition, a train-based (TRA) decomposition, and a time-interval-based (TIN) decomposition. The GEO decomposition consists of first partitioning the whole railway network into many elementary block sections and then clustering these block sections into a given number of regions. An integer linear optimization approach is proposed to cluster the block sections with the objective of minimizing the total number of train service interconnections among the regions and of balancing the region sizes. Consequently, several subproblems are obtained, and each region corresponds to one subproblem. For the TRA decomposition, we decompose an F-train problem into F subproblems, and each subproblem includes one individual train only. The TIN decomposition makes a division of the time horizon into equal-interval pieces, and each time-interval piece corresponds to one subproblem, which consists of all events (i.e., train departures and arrivals) that are estimated to happen in this time-interval. No matter which decomposition method is used, couplings always exist among subproblems, and the presence of these couplings leads to a non-separable structure of the whole optimization problem. To handle the issue of the couplings, we introduce three distributed optimization approaches. The first one is an Alternating Direction Method of Multipliers (ADMM) algorithm, where each subproblem is solved through coordination with the other subproblems in an iterative manner. The second one is a priority-rule-based (PR) algorithm, where the subproblems are sequentially and iteratively solved in a priority order (based on train delays) with respect to the solutions of the other subproblems that have been solved with a higher priority. The third one is a Cooperative Distributed Robust Safe But Knowledgeable (CDRSBK) algorithm, where four types of couplings are defined and each subproblem is iteratively solved together with its actively coupling subproblems. Experiments are conducted based on the Dutch railway network to comparatively test the performance of the three proposed algorithms with the three decomposition methods, in terms of feasibility, computational efficiency, solution quality, and estimated optimality gap.

The reminder of this paper is organized as follows. In Section 2, we briefly introduce an MILP problem that we focus on in this paper, which addresses the integrated problem of real-time traffic management and train control. Section 3 introduces three decomposition methods, where a number of subproblems are obtained. In Section 4, three distributed optimization approaches are developed for handling the couplings among the resulting subproblems. Section 5 examines the performance of the proposed algorithms and decomposition methods, through experiments on the Dutch railway network. Finally, the conclusions and suggestions for future research are given in Section 6.

2 An MILP Approach for Addressing the Integration of Traffic Management and Train Control

An MILP approach has been developed in our previous work (Luan et al., 2018) for addressing the integrated problem of real-time traffic management and train control. This MILP approach incorporates the representations of microscopic traffic regulations and train speed trajectories into a single MILP optimization problem of the following form:

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

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

with variable $\lambda \in \mathbb{R}^n$, matrix $A \in \mathbb{R}^{m \times n}$, and vectors $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$. The objective function $Z(\lambda)$ in (1a) minimizes the weighted sum of the total train delay times at all visited stations and the energy consumption of the train movements. The vector λ contains both the traffic-related variables and train-related variables for describing the train movements on block sections, in particular, the arrival times a, departure times d, train orders θ, incoming speeds v^{in}, cruising speeds v^{cru}, outgoing speeds v^{out}, approach time τ^{approach}, and clear time τ^{clear}. In (1b), all constraints (inequalities and equalities) are represented for ensuring the train speed limitations, for enforcing the consistency of train transition times and speeds, for guaranteeing the required dwell times, for determining train blocking times, and for respecting the block section capacities. The MILP problem (1a)-(1b) can be solved by a standard MILP solver, e.g., CPLEX or Gurobi. Interested readers are referred to the optimization problem named P_{TSPO} in Luan et al. (2018) for a more detailed description.

3 Problem Decomposition

Three decomposition methods, i.e., the geography-based (GEO), the train-based (TRA), and the time-interval-based (TIN) decomposition, are described in Sections 3.1-3.3 respectively. Section 3.4 discusses the decomposition result, i.e., subproblems and couplings. Figure 1 comparatively illustrates the three decomposition methods in a time-space graph, where black lines indicate train paths and red dashed lines indicate boundaries of subproblems.

3.1 Geography-Based Decomposition

The GEO decomposition partitions the whole railway network into a given number of regions. Consider a railway network composed of a 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 with couplings. 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 $R = \{1, 2, ..., |R|\}$ is the set of regions. Figure 1(b) illustrates a 2-region example of the geography-based decomposition; as shown, the timetable is split in the dimension of space.

To distribute $|E|$ different units into $|R|$ groups, there are $|R|^{|E|}$ ways, e.g., up to 10^6 ways for distributing 20 units into 2 groups only. Thus, in our case, a huge number of the GEO decomposition results are available. To obtain the optimal decomposition result, an integer linear programming (ILP) approach is proposed in Appendix B, 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 $|E|/|R|$).

For the GEO decomposition with a pre-defined number of regions, there are two impact

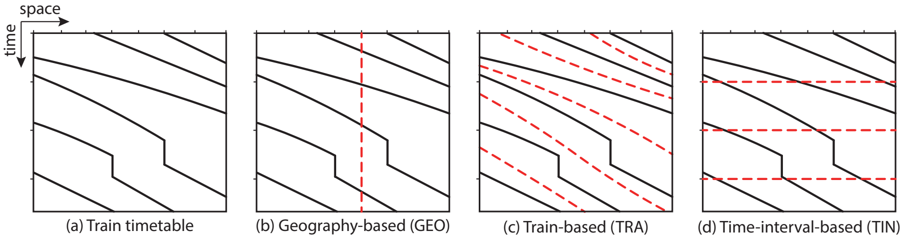


Figure 1: Illustration of the three decomposition methods in a time-space graph

factors: the network layout and the train routes planned in the original timetable. This implies that the optimal decomposition result is same for all delay cases.

When applying the GEO decomposition, some trains may traverse from one region to another region. The time and speed that a train leaves one region should equal the time and speed that the train arrives at the other region. Therefore, the time and speed transition constraints are the complicating constraints for the GEO decomposition, which cause the couplings among regions (i.e., subproblems). The time and speed transition constraints of the MILP problem (1) are formulated in (15a)-(15b) of Appendix A.

3.2 Train-Based Decomposition

The TRA decomposition simply splits a $|F|$-train problem into $|F|$ subproblems, and each subproblem corresponds to a 1-train problem, as illustrated in Figure 1(c). Thus, for a given instance, only one decomposition result is available. The only impact factor of the TRA decomposition is the involved trains. Brännlund et al. (1998) used such train-based decomposition for addressing train timetabling problem by using Lagrangian relaxation.

When applying the TRA decomposition, each train is independently scheduled in each subproblem, so that trains may use the same infrastructure at the same time, resulting in conflicts. Therefore, the capacity constraint is the complicating constraint for the TRA decomposition. The capacity constraint is formulated in (15c)-(15d) of Appendix A.

3.3 Time-Interval-Based Decomposition

The time-interval-based (TIN) decomposition makes a division of a train timetable in the dimension of time, based on a given size of time interval, as illustrated in Figure 1. The TIN decomposition is implemented with consideration of disruptions (delays). We independently schedule all trains by taking disruptions into account, generating an infeasible timetable, where train conflicts exist. With this infeasible timetable, we estimate the times that all events (e.g., train departure and arrival) may happen. Each event is then assigned to one time interval based on its estimated happen time. As a result, the subproblem of each time interval includes all events that are estimated to happen in this time interval. The TIN decomposition result mainly depends on the given size of time interval and the estimated train schedule, which can be different in delay cases.

One train service consists of a set of events indicating the departures and arrivals of the train on block sections. When applying the TIN decomposition, these events may be split into more than one time intervals. Thus, same to the GEO decomposition (where trains may traverse from region to region), the time and speed when a train leaves a time interval should be consistent with those when the train enters the next time interval, i.e., the time and speed transition constraints are complicating constraints, as formulated in (15a)-(15b) of Appendix A. Moreover, as the TIN decomposition is based on an estimated infeasible timetable, an event assigned to time interval t maybe further scheduled to the next time interval $t+1$, causing conflicts with the events in time interval $t+1$. Therefore, the capacity constraint in (15c)-(15d) is also a complicating constraint for the TIN decomposition.

3.4 Subproblems and Couplings

Let us denote S as the set of the $|S|$ resulting subproblems, e.g., $|S|$ = $|R|$ for the GEO 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. A local constraint is only related to a single subproblem, so that it leads to a separable structure of an optimization problem. A complicating constraint is associated with at least two subproblems, so that it results in a non-separable structure. We thus rewrite (1b) into a general form of the following local and complicating constraints:

$$A^{\text{loc}} \cdot \lambda \leq b^{\text{loc}} \tag{2a}$$

$$A^{\text{cpl}} \cdot \lambda \leq b^{\text{cpl}} \tag{2b}$$

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 $b^{\text{cpl}} \in \mathbb{R}^{m_2}$. A detailed explanation of the complicating constraints of the MILP problem (1) is given in Appendix A. Let us denote $Q_p = \{q_1, q_2, ..., q_{m_p}\}$ as the set of m_p subproblems that have couplings with subproblem p. The subproblem $p \in S$ of the MILP problem (1) is formulated as

$$\min_{\lambda_p} \mathcal{Z}_p(\lambda_p) = c_p^\top \cdot \lambda_p \tag{3a}$$

$$\text{s.t. } A_p^{\text{loc}} \cdot \lambda_p \leq b_p^{\text{loc}} \tag{3b}$$

$$A_{p,q}^{\text{cpl}} \cdot \lambda_p + A_{q,p}^{\text{cpl}} \cdot \lambda_q \leq b_{p,q}^{\text{cpl}}, \ \forall q \in Q_p \tag{3c}$$

where $A_{p,q}^{\text{cpl}}$ and $A_{q,p}^{\text{cpl}}$ are selection matrices for selecting the coupling variables between subproblems p and q. Since each coupling constraint in (3c) 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 coupling 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 coupling 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 coupling subproblems $q \in Q_p$ as well, rather than only focusing on computing optimal local variables. Moreover, for its coupling 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 optimal 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 issue of couplings 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 Sections 4.1-4.3 respectively. 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 (see e.g., Boyd et al., 2011) solves problems in the following form:

$$\min_{x,z} \quad f(x) + g(z) \tag{4a}$$

$$\text{s.t.} \quad A \cdot x + B \cdot z = b, \tag{4b}$$

with variables $x \in \mathbb{R}^n$ and $z \in \mathbb{R}^m$, 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 x and z can be split into two parts, with the objective function separable across this splitting. We can then form the augmented Lagrangian relaxation as

$$L_\rho(x, z, y) = f(x) + g(z) + y^\top (A \cdot x + B \cdot z - b) + \tfrac{\rho}{2} \cdot \|A \cdot x + B \cdot z - b\|_2^2, \tag{5}$$

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 x and z 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:

$$x^{i+1} := \arg\min_x L_\rho(x, z^i, y^i), \tag{6a}$$

$$z^{i+1} := \arg\min_z L_\rho(x^{i+1}, z, y^i), \tag{6b}$$

$$y^{i+1} := y^i + \rho(A \cdot x^{i+1} + B \cdot z^{i+1} - b) \tag{6c}$$

where i is the iteration counter. In the ADMM algorithm, the variables x and z 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 $A \cdot x \leq b$ by $A \cdot x + s = b$, adding the slack variable s to the set of optimization variables, and setting $\mathcal{Z}(s) = 0$, if $s \geq 0$, otherwise, setting $\mathcal{Z}(s) = \infty$. Alternatively, we can also work with an equivalent reformulation of problem (3), where we replace the complicating constraint (3c) by

$$\mathcal{C}_p(\lambda_p, \lambda_q) = 0 \tag{7}$$

where $\mathcal{C}_p(\lambda_p, \lambda_q) = \max\left\{0, A_{p,q}^{\text{cpl}} \cdot \lambda_p + A_{q,p}^{\text{cpl}} \cdot \lambda_q - b_{p,q}^{\text{cpl}}\right\}$ 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:

$$L_\rho = \sum\nolimits_{p \in S} \left[\mathcal{Z}_p(\lambda_p) + \sum\nolimits_{q \in Q_p} \left[y_{p,q}^\top \cdot \mathcal{C}_p(\lambda_p, \lambda_q) + \frac{\rho}{2} \cdot ||\mathcal{C}_p(\lambda_p, \lambda_q)||_2^2 \right] \right] \tag{8}$$

The iterations to compute the solution of the MILP problem (1) based on the augmented Lagrangian formulation (8) include quadratic terms; therefore, the function cannot directly be distributed over subproblems. Inspired by Negenborn et al. (2008), for handling this non-separable issue, the function (8) can be approximated by solving $|S|$ separate problems of the form

$$\min_{\lambda_p} \; \mathcal{Z}_p(\lambda_p) + \sum\nolimits_{q \in Q_p} \mathcal{J}_p(\lambda_q, y_{p,q}) \tag{9}$$

subject to (3b) for the train movements of 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 coupling subproblems (of subproblem p) that have been solved before solving subproblem p.

The serial implementation uses the information from both the current iteration i and the last 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 last iteration $i - 1$

for the other subproblems $q \in Q_p \backslash \widehat{Q_p^i}$, we can solve (9) for subproblem p by using the following function:

$$\mathcal{J}_p(\bar{\lambda}, y_{p,q}) = y_{p,q}^{\top} \cdot \mathcal{C}_p(\lambda_p, \bar{\lambda}_q) + \frac{\rho}{2} \cdot ||\mathcal{C}_p(\lambda_p, \bar{\lambda}_q)||_2^2 \tag{10}$$

The second term of (10) penalizes the deviation from the coupling variable iterates that were computed for the subproblems before subproblem p in the current iteration i and by the other subproblems during the last iteration $i-1$.

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 $\rho := 1$, the Lagrange multipliers $y^{(0)} := 0$, and all elements in the latest solution set $\mathcal{S}_{\text{sol}} := \{\bar{\lambda}_p | p \in S\}$ to be empty. Denote the maximum number of iterations as $I^{\max}$.

```
for iteration $i := 1, 2, ..., I^{\max}$ do
    Randomly generate the orders of subproblems, denoted as $P_{\text{order}}^{(i)}$.
    for subproblem $j := 1, 2, ..., |S|$ do
        Solve subproblem $p := P_{\text{order}}^{(i)}(j)$, consisting of objective function (9) and constraint (3b), by taking the available solutions in $\mathcal{S}_{\text{sol}}$ for all $q \in \widehat{Q_p^i}$ into account.
        Denote the obtained solution of subproblem $p$ as $\lambda_p^{(i)}$, and update the latest solution set $\mathcal{S}_{\text{sol}}$ by adding or setting $\bar{\lambda}_p := \lambda_p^{(i)}$.
    end for
    Update the Lagrange multipliers by $y_{p,q}^{(i)} := y_{p,q}^{(i-1)} + \rho \cdot \mathcal{C}_p(\lambda_p^{(i-1)}, \lambda_q^{(i-1)})$ for all $p \in S$ and $q \in Q_p$.
    Break the iterations if the difference of the coupling variables at the current iteration step $i$ is less than the expected gap $\epsilon$, i.e., $\|\mathcal{C}\|_\infty \leq \epsilon$, where $\epsilon$ is a small positive scalar and $\|\cdot\|_\infty$ denotes the infinity norm.
end for
```

By applying the ADMM algorithm, we solve the subproblems $p \in S$ in an iterative manner, with respect to the local constraint (3b) of a single subproblem p and taking the solutions of all coupling subproblems (i.e., the variable $\bar{\lambda}_q$ for $q \in Q_p$ obtained in either the current iteration or the last iteration) into account. In (8), only the local objective $\mathcal{Z}_p$ for a single subproblem p is minimized, not the global objective $\sum_{p \in S} \mathcal{Z}_p$ for all subproblems.

In order to further improve the performance of the ADMM algorithm, we can consider 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 running to its destination. Then, the objective function (9) for subproblem $p \in S$ can be rewritten as follows:

$$\min_{\lambda_p} \ \mathcal{Z}_p(\lambda_p) + \mathcal{Z}_p^{\text{ctg}}(\lambda_p) + \sum\nolimits_{q \in Q_p} \mathcal{J}_p(\lambda_q, y_{p,q}) \tag{11}$$

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 leaves a region. Thus, an original timetable with more details is needed, where the departure and arrival times are given not only for stations but also for block sections.

4.2 Priority-Rule-Based Algorithm

The ADMM algorithm incorporates the complicating constraint (3c) 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 need to explore other distributed optimization approaches. We next introduce a priority-rule-based (PR) algorithm.

The main idea of the PR algorithm is to optimize train schedules of the 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. Note that the result could be different even with the same problem priorities, as multiple optimal solutions may exist for each subproblem. These different optimal solutions with the same objective value for one subproblem could result in different objective values for the other subproblems.

By applying the PR algorithm, the complicating constraint (3c) for the subproblem $p \in S$ can be rewritten as follows:

$$A^{\mathrm{cpl}}_{p,q} \cdot \lambda_p + A^{\mathrm{cpl}}_{q,p} \cdot \bar{\lambda}_q \leq b^{\mathrm{cpl}}_{p,q},\ \forall q \in Q_p \tag{12}$$

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

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 $o^{(0)}_{\mathrm{UB}} := M$, and the global upper bound $O^{(0)}_{\mathrm{UB}} := M$, where M is a sufficient large positive number. Initialize the problem priorities $P^{(0)}_{\mathrm{prior}}$ arbitrarily. Denote the maximum number of iterations as $I^{\max}$.

```
for iteration $i := 1, 2, ..., I^{\max}$ do
    Sort subproblems in set $S$ in a descending order by their problem priorities $P^{(i-1)}_{\mathrm{prior}}$, denoted as $P^{(i)}_{\mathrm{order}}$.
    Set the solution set $\mathcal{S}_{\mathrm{sol}} := \{\bar{\lambda}_p | p \in S\}$ to be empty.
    for subproblem $j := 1, 2, ..., |S|$ do
        Solve subproblem $p := P^{(i)}_{\mathrm{order}}(j)$, including objective function (3a) and constraints (3b) and (12), with respect to the available solutions in $\mathcal{S}_{\mathrm{sol}}$ for all $q \in \widehat{Q^i_p}$.
        Denote the obtained solution of subproblem $p$ as $\lambda^{(i)}_p$, and update the solution set $\mathcal{S}_{\mathrm{sol}}$ by adding $\bar{\lambda}_p := \lambda^{(i)}_p$.
    end for
    Compute the local upper bound $o^{(i)}_{\mathrm{UB}}$, and update the global upper bound by
        $O^{(i)}_{\mathrm{UB}} := \begin{cases} o^{(i)}_{\mathrm{UB}}, & \text{if } O^{(i-1)}_{\mathrm{UB}} > o^{(i)}_{\mathrm{UB}} \\ O^{(i-1)}_{\mathrm{UB}}, & \text{otherwise} \end{cases}$
    Update the problem priorities $P^{(i)}_{\mathrm{prior}}$ by the train delay times of the subproblems.
    Break the iterations if the global upper bounds are not improved for a given number of iterations $\kappa$, i.e., $O^{(i)}_{\mathrm{UB}} = O^{(i-\kappa)}_{\mathrm{UB}}$.
end for
```

In the priority-rule-based 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 (3b) and the outputs $\bar{\lambda}_q$ of the coupling subproblems $q \in Q_p$ in (12). Similar to the ADMM algorithm, only the local objective Z_p is minimized when solving subproblem p, rather than the global objective $\sum_{p\in R} Z_p$ for all subproblems. Constraint (12) ensures that the coupling variables of subproblem p satisfy those of its coupling subproblems $q \in Q_p$ obtained in the current iteration. For the first solved subproblem in each iteration, the complicating constraint (12) is relaxed.

4.3 Cooperative Distributed Robust Safe but Knowledgeable Algorithm

The third algorithm considered in this paper is the Cooperative distributed robust safe but knowledgeable (CDRSBK) algorithm, introduced by Kuwata and How (2011) to address trajectory planning problems. In the CDRSBK algorithm, four types of couplings among subproblems are defined for a subproblem $p \in S$, as illustrated in Figure 2.

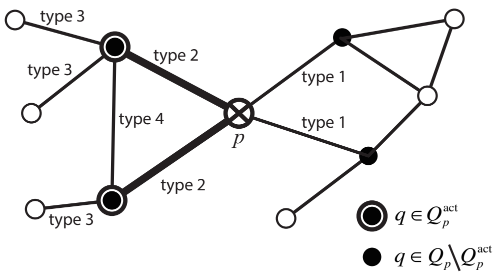


Figure 2: Four types of couplings defined in the CDRSBK algorithm

Type_1 indicates a non-active coupling between subproblem $p \in S$ and its neighbor; Type_2 indicates an active coupling between subproblem p and its neighbor; Type_3 indicates the coupling between the active coupling neighbors of subproblem p and their neighbors; and Type_4 indicates the coupling between two active coupling neighbors of subproblem p. Let us denote Q_p as the set of all coupling 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 4.4.

By applying the CDRSBK algorithm, the subproblem $p \in S$ of the MILP problem (3a)-(3c) can be reformulated as

$$\min_{\lambda_p,\xi_q} \quad \mathcal{Z}_p(\lambda_p) + \sum\nolimits_{q\in Q_p^{\text{act}}} \mathcal{Z}_q(\bar{\lambda}_q + T_q \cdot \xi_q) \tag{13a}$$

$$\text{s.t.} \quad A_p \cdot \lambda_p \leq b_p^{\text{loc}} \tag{13b}$$

$$A_q \cdot (\bar{\lambda}_q + T_q \cdot \xi_q) \leq b_q^{\text{loc}}, \ \forall q \in Q_p^{\text{act}} \tag{13c}$$

$$A_{p,q}^{\text{cpl}} \cdot \lambda_p + A_{q,p}^{\text{cpl}} \cdot \bar{\lambda}_q \leq b_{p,q}^{\text{cpl}}, \ \forall q \in Q_p \backslash Q_p^{\text{act}} \tag{13d}$$

$$A_{p,q}^{\text{cpl}} \cdot \lambda_p + A_{q,p}^{\text{cpl}} \cdot (\bar{\lambda}_q + T_q \cdot \xi_q) \leq b_{p,q}^{\text{cpl}}, \ \forall q \in Q_p^{\text{act}} \tag{13e}$$

$$A_{o,q}^{\text{cpl}} \cdot \bar{\lambda}_o + A_{q,o}^{\text{cpl}} \cdot (\bar{\lambda}_q + T_q \cdot \xi_q) \leq b_{o,q}^{\text{cpl}}, \ \forall o \in Q_q \backslash Q_p^{\text{act}}, q \in Q_p^{\text{act}} \tag{13f}$$

$$\begin{aligned} A_{q_1,q_2}^{\text{cpl}} \cdot (\bar{\lambda}_{q_1} + T_{q_1} \cdot \xi_{q_1}) + A_{q_2,q_1}^{\text{cpl}} \cdot (\bar{\lambda}_{q_2} + T_{q_2} \cdot \xi_{q_2}) \leq b_{q_1,q_2}^{\text{cpl}}, \\ \forall q_1, q_2 \in Q_p^{\text{act}}, q_2 \in Q_{q_1}, q_1 \in Q_{q_2} \end{aligned} \tag{13g}$$

In (13a), the objective function of both subproblem p and its actively coupled subproblems $q \in Q_p^{\text{act}}$ are included. Constraints (13b)-(13c) represent the local constraints of subproblem p and its actively coupled subproblems $q \in Q_p^{\text{act}}$ respectively. In (13d)-(13g), coupling constraints (3c) are rewritten for the four types of couplings among subproblems respectively. When solving subproblem p, besides the local variable λ_p, the variable ξ_q is also optimized for its actively coupled subproblems $q \in Q_p^{\text{act}}$ on the communicated solution $\bar{\lambda}_q$, as follows:

$$\lambda_q = \bar{\lambda}_q + T_q \cdot \xi_q \tag{14}$$

parameterized with a matrix T_q, which is formed to allow the variable ξ_q to change only the rows corresponding to the active complicating constraints. This can be also interpreted as allowing a change for the constraint that has a non-zero Lagrange multiplier. In (13a), the objectives of a single subproblem p and its actively coupled neighbors $q \in Q_p^{\text{act}}$ are both minimized.

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 all elements in the latest solution set $\mathcal{S}_{\text{sol}} := \{\bar{\lambda}_p | p \in S\}$ to be empty. Denote the maximum number of iterations as $I^{\max}$.

```
for iteration $i := 1, 2, ..., I^{\max}$ do
    Randomly generate the orders of subproblems, denoted as $P_{\text{order}}^{(i)}$.
    for subproblem $j := 1, 2, ..., |S|$ do
        Solve subproblem $p := P_{\text{order}}^{(i)}(j)$ and its actively coupling subproblems $q \in Q_p^{\text{act}}$, consisting of objective function (13a) and constraints (13b)-(13g), by taking the available solutions in set $\mathcal{S}_{\text{sol}}$ for all $o \in (Q_p \backslash Q_p^{\text{act}}) \cup (Q_q \backslash Q_p^{\text{act}})$ into account.
        Denote the obtained solutions of subproblem $p$ and its actively coupling subproblems $q \in Q_p^{\text{act}}$ as $\lambda_p^{(i)}$ and $\lambda_q^{(i)}$ (which is obtained by (14)) respectively, and update the latest solution set $\mathcal{S}_{\text{sol}}$ by adding or setting $\bar{\lambda}_p := \lambda_p^{(i)}$ and $\bar{\lambda}_q := \lambda_q^{(i)}$ for all $q \in Q_p^{\text{act}}$.
    end for
    Compute the local upper bound $o_{\text{UB}}^{(i)}$, and update the global upper bound by
        $O_{\text{UB}}^{(i)} := \begin{cases} o_{\text{UB}}^{(i)}, & \text{if } O_{\text{UB}}^{(i-1)} > o_{\text{UB}}^{(i)} \\ O_{\text{UB}}^{(i-1)}, & \text{otherwise} \end{cases}$
    Break the iterations if the global upper bounds are not improved for a given number of iterations $\kappa$, i.e., $O_{\text{UB}}^{(i)} = O_{\text{UB}}^{(i-\kappa)}$.
end for
```

In each iteration, the CDRSBK algorithm actually solves each subproblem, with additional objectives and coupling 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 coupling constraints are automatically satisfied and could be omitted.

4.4 Remarks on the Implementation of the Decomposition Methods and Algorithms

Here we give some remarks for the implementation of the proposed decomposition methods and algorithms, e.g., interpreting the active and non-active couplings in the CDRSBK algorithm for different decomposition methods and giving some tips for achieving feasibility.

Remark 1 (Train orders in the ADMM algorithm with the GEO decomposition and the TIN decomposition)**.** It is essential to ensure that train orders in 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 same.

Remark 2 (The CDRSBK algorithm & 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 of these two 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_2). For coupling Type_3 and Type_4, we follow their general definitions, i.e., the couplings between an active coupling neighbor and its coupling neighbors are labeled as Type_3 coupling and the coupling between two active coupling neighbor is labeled as Type_4.

Remark 3 (The CDRSBK algorithm & the TRA decomposition)**.** If two trains use the same infrastructure (block section), 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 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 coupling Type_1 and Type_3. We denote the option described above as Opt_1. The difference between Opt_1 and Opt_2 is in the definition of coupling Type_3: in Opt_2, we label the couplings between an active coupling neighbor and its *active* coupling neighbor as Type_3. Based on Opt_2, we discard all Type_1 couplings, which results in Opt_3, i.e., when and only when a conflict happens between two trains, a coupling exists between them and is recognized as active coupling (Type_2). However, we still have Type_3 and Type_4 couplings in Opt_3 by following their general definitions. An illustrative example is provided in Appendix C to graphically explain these three options.

Remark 4 (The CDRSBK algorithm & 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 $t+1$) and are all recognized as active couplings (Type_2). As a result, according to the general definition of the four types of couplings, 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 $t+2$), and Type_1 and Type_4 couplings do not exist. Moreover, for guaranteeing a feasible solution in the first iteration, solving subproblems in a time sequence (i.e., for time intervals $t = 1, 2, 3....$ in sequence) is recommended.

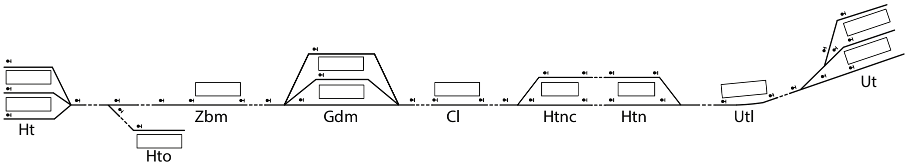


Figure 3: An experimental railway network

5 Case Study

5.1 Set-Up

We consider a line of the Dutch railway network, connecting Utrecht (Ut) to Den Bosch (Ht), of about 50 km length, with 9 stations, as shown in Figure 3. The network comprises 42 nodes and 40 cells. We consider one hour of heterogeneous traffic with 15 trains. Moreover, we considered different numbers of regions for the GEO decomposition, ranging from 2 to 6, and we consider 4 time intervals for the TIN decomposition, i.e., 300s, 600s, 900s, and 1200s. In the result presentation, we present the average result of 15 delay cases with randomly generated primary delays. The maximum number of iterations is set to 200, 100, and 30 for the ADMM, PR, and CDRSBK algorithm respectively. A larger number is set for the ADMM algorithm because it needs some iterations to converge, and a smaller number is set for the CDRSBK algorithm because it often finds a feasible solution very fast and its solution is updated multiple times in one iteration. In the case study, we consider the weight $\zeta = 0.55$ for the ILP problem proposed in Appendix B for the GEO decomposition, which is appropriate for getting a result with an acceptable difference of the size of regions.

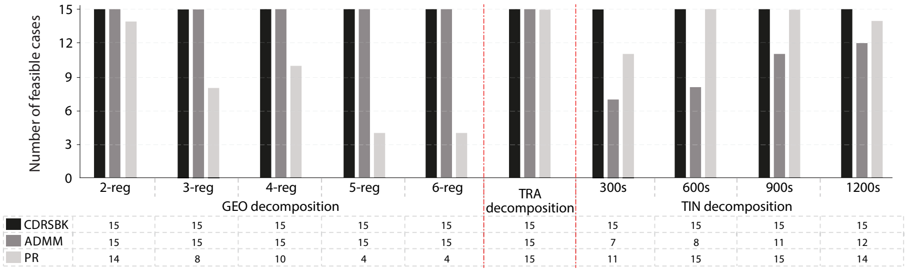
We adopt the CPLEX solver version 12.6.3 implemented in the MATLAB (R2018a) TOMLAB toolbox to solve the MILP problems. The experiments are performed on a computer with an Intel® Core$^{\text{TM}}$ i7 @ 2.00 GHz processor and 16GB RAM.

5.2 Experimental Results

This section shows the (average) results of 15 delay cases from the viewpoints of feasibility, estimated optimality gap, solution quality, and computational efficiency.

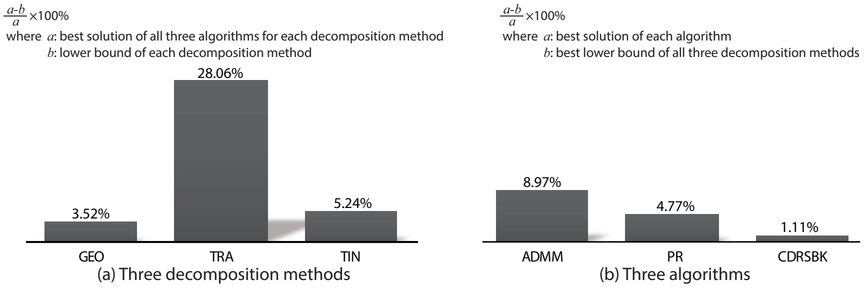
Figure 4 presents the number of cases that we can find feasible solutions within the maximum number of iterations. We can conclude that, for achieving feasibility, the TRA decomposition performs best among the three decomposition methods, and the CDRSBK algorithm is the best among the three algorithms. Considering a larger number of regions for the GEO decomposition or considering a smaller time interval for the TIN decomposition can make feasibility difficult to achieve, as they lead to a larger number of couplings among subproblems.

In Figure 5, an estimated optimality gap for each decomposition method and each algorithm is given. As shown, the estimated optimality gap of the GEO decomposition is 3.52%, the lowest among the three decomposition methods, and the CDRSBK algorithm has the smallest estimated optimality gap (only 1.11%) among the three algorithms. A large estimated optimality gap does not reflect a bad solution quality; it may be caused by a loose lower bound, as in the case of the TRA decomposition.



	2-reg	3-reg	4-reg	5-reg	6-reg	TRA decomposition	300s	600s	900s	1200s
■ CDRSBK	15	15	15	15	15	15	15	15	15	15
▨ ADMM	15	15	15	15	15	15	7	8	11	12
□ PR	14	8	10	4	4	15	11	15	15	14

Figure 4: Feasibility of the three decomposition methods and three algorithms



(a) Three decomposition methods

(b) Three algorithms

Figure 5: Estimated optimality gap of the three decomposition methods and three algorithms

Figure 6 shows the cumulative computation time (on the X-axis) and the objective value (on the Y-axis). The cumulative computation time is the CPU time consumed for finding the best feasible solution. Dashed circles around symbols indicate that feasible solution(s) can be found for all 15 delay cases by using the corresponding decomposition method and algorithm. When focusing on the three decomposition methods (represented by colors), the GEO decomposition (in pink) leads to a large range in computation time and a small range in objective value. This implies that the GEO decomposition results in small differences in the solution quality, but the computational efficiency is quite different for different algorithms. For the TRA decomposition (in blue) and the TIN decomposition (in green), ranges still exist in the two dimensions, and their results show a general trade-off between solution quality and computational efficiency. Let us now focus on the three algorithms (indicated by symbols). The CDRSBK algorithm (indicated by diamonds) overall yields the best solution quality, and the computation efficiency becomes much better when the TRA decomposition is applied. The performance of the ADMM and PR algorithms is highly variable. For the ADMM algorithm (indicated by circles), the best solution quality is achieved when using the GEO decomposition, and the best computation efficiency is achieved when the TRA decomposition is adopted. The PR algorithm (indicated by triangles) has the best performance on solution quality when the GEO decomposition is used and on computational efficiency when the TIN decomposition is applied. A black dashed circle around a symbol indicates that feasible solution(s) can be found for all 15 delay cases by using the corresponding decomposition method and algorithm. Moreover, the lower bound of the TRA decomposition (indicated by a blue cross symbol) is the loosest, which leads to its large estimated

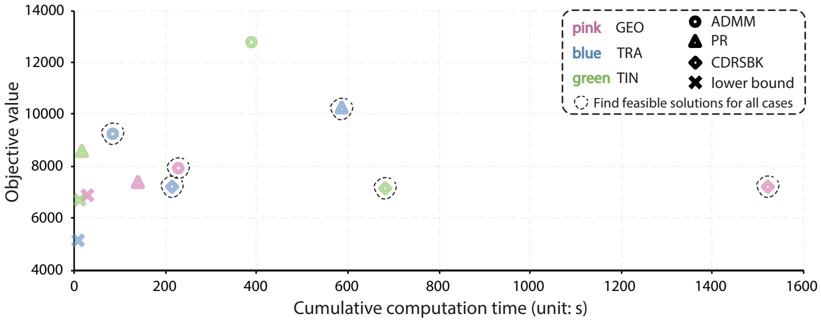


Figure 6: Solution quality and computational efficiency

optimality gap in Figure 5.

Overall, the CDRSBK algorithm with the TRA decomposition, the ADMM algorithm with the GEO decomposition, and the ADMM algorithm with the TRA decomposition have good overall performance. All these three combinations can find feasible solutions for all delay cases. In comparison, the first two combinations have the best performance on solution quality and a satisfactory performance on computational efficiency. The last combination shows the best computational efficiency (roughly half-shorter computation time than the first two combinations) but at the cost of relatively bad solution quality.

Moreover, when using the CDRSBK algorithm together with the TRA decomposition, Opt_3 described in Section 4.4 yields the best performance on both solution quality and computational efficiency. For Opt_1, Opt_2, and Opt_3, the average objective value for the 15 delay cases is 7934.43, 7334.86, and 7217.08 respectively, and the average cumulative computation time is 255.19 seconds, 224.64 seconds, and 104.75 seconds.

6 Conclusions

We have introduced distributed optimization approaches, aiming at improving the computational efficiency of the integrated optimization problem for large-scale railway networks. Three decomposition methods have been presented to split the whole optimization problem into several subproblems, and three distributed optimization approaches have been proposed for dealing with the couplings among subproblems.

The performance of the proposed approaches has been examined in terms of feasibility, estimated optimality gap, solution quality, and computational efficiency. The TRA decomposition and the CDRSBK algorithm have the best performance from the perspective of feasibility. The GEO decomposition and the CDRSBK algorithm yield the smallest estimated optimality gap. The CDRSBK algorithm with the TRA decomposition and the ADMM algorithm with the GEO decomposition achieve the best performance on solution quality and satisfactory performance on computational efficiency. The ADMM algorithm with the TRA decomposition shows the best computational efficiency but gives a relatively bad solution.

For practical applications, a promising two-step procedure can be used: first generate a feasible solution in short time (e.g., by applying the ADMM algorithm) and then improve the solution quality (by using the CDRSBK algorithm) based on that feasible solution if time permits. This leads to one direction of the future research on exploring the interactions of algorithms and decompositions so that we can play with their advantages, in order to further achieve best overall solution. Moreover, we are going to test the performance of the proposed approaches on larger-scale railway instances.

Acknowledgments

This work is supported by China Scholarship Council under Grant 201507090058.

Appendix A The complicating constraints in the MILP problem (1)

As explained in Section 3, there are some complicating constraints in the MILP optimization problem (1), causing the couplings among subproblems and making a non-separable structure of the whole problem.

When applying the GEO decomposition, the complicating constraints are the time and speed transition constraints, which can be written as follows:

$$d_{f,i,j} = a_{f,j,k}, \forall f \in F, (i,j) \in E_f, (j,k) \in E_f \tag{15a}$$

$$v^{\text{out}}_{f,i,j} = v^{\text{in}}_{f,j,k}, \forall f \in F, (i,j) \in E_f, (j,k) \in E_f \tag{15b}$$

Constraint (15a) enforces the transition time between two adjacent block sections, i.e., the departure time of train f on the preceding block section (i,j) equals the arrival time of train f on the successive block section (j,k), if two adjacent block sections (i,j) and (j,k) are used consecutively by train f. Constraint (15b) ensures the consistency of the train speed between two adjacent block sections, i.e., the incoming speed of train f on block section (j,k) equals to its outgoing speed on the preceding block section (i,j).

When applying the TRA decomposition, the couplings result from the competitive use of infrastructure by trains, i.e., the capacity constraint is the complicating constraint, formulated as follows:

$$\begin{aligned} a_{f',i,j} - \tau^{\text{approach}}_{f',i,j} - \tau^{\text{sig\_set}} + (1-\theta_{f,f',i,j}) \cdot M \geq d_{f,i,j} + \tau^{\text{clear}}_{f,i,j} + \tau^{\text{rel}}, \\ \forall f \in F, f' \in F, f \neq f', \rho_f = \rho_{f'}, (i,j) \in E_f, (i,j) \in E_{f'}, \end{aligned} \tag{15c}$$

$$\begin{aligned} a_{f',j,i} - \tau^{\text{approach}}_{f',j,i} - \tau^{\text{sig\_set}} + (1-\theta_{f,f',i,j}) \cdot M \geq d_{f,i,j} + \tau^{\text{clear}}_{f,i,j} + \tau^{\text{rel}}, \\ \forall f \in F, f' \in F, f \neq f', \rho_f \neq \rho_{f'}, (i,j) \in E_f, (j,i) \in E_{f'}. \end{aligned} \tag{15d}$$

where M is a sufficiently large positive number, $\tau^{\text{sig\_set}}$ is the setup, sight, and reaction time to lock a block section before the arrival of a train, and τ^{rel} is the release time to unlock a block section after the departure time of a train. Constraints (15c) and (15d) ensure that any pair of trains using one block section in the same or different direction respectively are conflict-free, by avoiding the overlap between the block section release time for a preceding train and the block section occupancy time for a successive train.

For the the TIN decomposition, all constraints in (15) can be complicating constraints.

Appendix B An integer linear programming approach for the geography-based decomposition

The 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 set E_f

belong to different regions, otherwise, $(\beta_f)_j = 0$. The binary vector α_r indicates the assignment of all block sections for region r, e.g., if $(\alpha_r)_i = 1$, then the i^{th} block section in 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 set E, then $B_f = \begin{bmatrix} 1 & 0 & -1 & 0 & \dots \end{bmatrix}$. The integer vector $\mu \in (\mathbb{Z}^+)^{|E|}$ indicates the index of regions that each block section $e \in E$ belongs to. We use $\|\cdot\|_1$ to denote the 1-norm. The objective function is formulated as follows:

$$\min_{\alpha,\beta} \left[\zeta \cdot \left(\sum\nolimits_{f\in F} \|\beta_f\|_1 \right) + (1-\zeta) \cdot \left(\sum\nolimits_{r=1}^{|R|} \left| \|\alpha_r\|_1 - \frac{|E|}{|R|} \right| \right) \right], \tag{16}$$

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

We consider four constraints, presented as follows:

$$\frac{\left| (B_f \cdot \mu)_j \right|}{|R|-1} \leq (\beta_f)_j, \quad \forall f \in F, j \in \{1, \dots, |E_f|-1\}, \tag{17}$$

guarantees that $(\beta_f)_j > 0$ if the two consecutive block sections along the route of train f belong to different regions, i.e., $\left| (B_f \cdot \mu)_j \right| > 0$.

$$\mu_i \in \{1, \dots, |R|\}, \quad \forall i \in \{1, \dots, |E|\}, \tag{18}$$

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

$$(\alpha_r)_i \leq 1 - \frac{|\mu_i - r|}{|R|-1}, \quad \forall r \in \{1, \dots, |R|\}, i \in \{1, \dots, |E|\}, \tag{19}$$

and

$$\|\alpha_r\|_1 \geq 1, \quad \forall r \in \{1, \dots, |R|\}, \tag{20}$$

are used to avoid solution in which no block section is assigned to some region(s). Specifically, in (19), 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 (20), we ensure that at least one block section is assigned to each region. As a result, (19) and (20) imply that the number of the resulting regions must equal the given number $|R|$. An illustrative example is provided in Appendix C to explain the above formulations.

Appendix C An illustrative example

In this appendix, we use a small instance to explain the proposed decomposition methods and algorithms. As illustrated in Figure 7, the instance includes 4 trains following the pre-

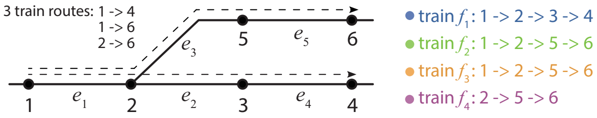
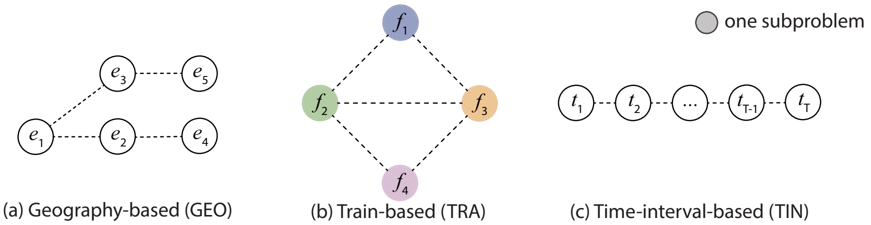

3 train routes: 1 -> 4
1 -> 6
2 -> 6
e_3
5
e_5
6
1
e_1
2
e_2
3
e_4
4
● train f_1: 1 -> 2 -> 3 -> 4
● train f_2: 1 -> 2 -> 5 -> 6
● train f_3: 1 -> 2 -> 5 -> 6
● train f_4: 2 -> 5 -> 6

Figure 7: A small instance



(a) Geography-based (GEO) (b) Train-based (TRA) (c) Time-interval-based (TIN)

Figure 8: Subproblems and couplings

defined routes, i.e., train $f_1 : 1 \rightarrow 2 \rightarrow 3 \rightarrow 4$, train f_2 and $f_3 : 1 \rightarrow 2 \rightarrow 5 \rightarrow 6$, and train $f_4 : 2 \rightarrow 3 \rightarrow 4$.

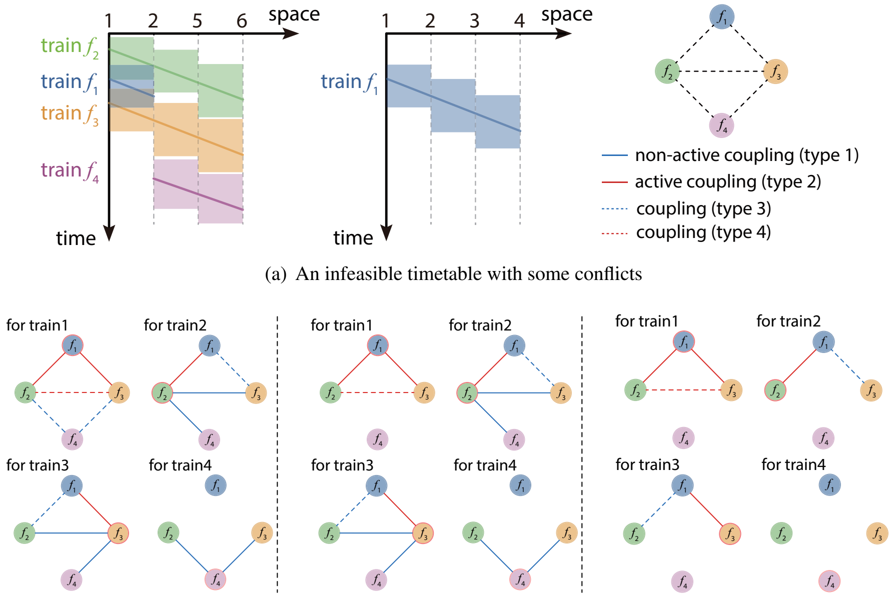
We now illustratively explain the formulation of the ILP problem proposed in Appendix B. We can write the set of block sections as $E = \{e_1, e_2, e_3, e_4, e_5\}$. The route matrix B_{f_1} and the variable vector β_{f_1} for train f_1 and the variable vector μ for block sections can be expressed as

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

Consider the consecutive block sections e_1 and e_2 in the route of train f_1; the (17) results in the inequality $\frac{|\mu_1 - \mu_2|}{|R|-1} \leq (\beta_{f_1})_1$. If the two block sections belong 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 strictly in range $[0, 1)$ and B_{f_1} is an integer matrix. Constraints (19)-(20) 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., 5 regions and each region contains only one block section, and denote T as the number of subproblems for the TIN decomposition. By applying the three propose decomposition methods, the resulting subproblems and (primary) couplings can be shown in Figure 8. As illustrated, the GEO decomposition results in 5 subproblems, corresponding to 5 block sections respectively; the TRA decomposition leads to 4 subproblems, corresponding to 4 trains respectively; and the TIN decomposition gives T subproblems connected in an order of time horizon.

We now illustrate the three options for defining the four types of couplings in the CDRSBK algorithm with the TRA decomposition. Let us assume an infeasible timetable shown in Figure 9(a), which can be generated by independently scheduling trains one-by-one without considering their couplings. The three options are illustrated in Figure 9(b)-Figure 9(d) respectively. Let us now focus on train f_1 (i.e., subproblem f_1) to explain. In Opt_1, couplings between f_1 and f_2 is recognized as active coupling (Type_2), because train f_1 has conflict with train f_2 in the timetable shown in Figure 9(a). Both f_2 and f_3 are actively coupling subproblem of f_1; so a Type_3 coupling exists between f_2 and f_3. Train f_1 and train f_4 use completely different block sections. So subproblem f_4 only has couplings with f_2 and f_3, and their couplings are recognized as a Type 3 coupling for subproblem f_1. Train f_2 uses same block sections with all the other trains, but only has conflict with train f_1; therefore, when we focus on train f_2, 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. In Opt_2, still focusing on subproblem f_1, as the coupling between f_2 and f_4 is a non-active



(a) An infeasible timetable with some conflicts

(b) Opt_1

(c) Opt_2

(d) Opt_3

Figure 9: Three options of the CDRSBK algorithm with the TRA decomposition

coupling (Type_1, when focusing on subproblem f_2 or f_4), we consider the Type 3 coupling between f_2 and f_4 do not exist, as same as the Type_3 coupling between f_3 and f_4. In Opt_3, we consider no coupling if no conflict, which can be simply explained as removing all Type_1 couplings based on the coupling architecture of Opt_2. However, Type_3 and Type_4 couplings are generally defined, same to Opt_1 (and Opt_2).

References

Beltran Royoa, C., Heredia, F. J., 2002. “Unit commitment by augmented Lagrangian relaxation: Testing two decomposition approaches”. *Journal of Optimization Theory and Applications*, *112*, 295–314.

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., 2011. “Distributed optimization and statistical learning via the alternating direction method of multipliers”. *Foundations and Trends in Machine Learning*, *3*, 1–122.

Brännlund, U., Lindberg, P. O., Nou, A., Nilsson, J.-E., 1998. “Railway timetabling using Lagrangian relaxation”. *Transportation science*, *32*, 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”. *Transportation Research Part B: Methodological*, *63*, 15–37.

Corman, F., D'Ariano, A., Hansen, I. A., Pacciarelli, D., 2011. "Optimal multi-class rescheduling of railway traffic". *Journal of Rail Transport Planning and Management*, *1*, 14–24.

Corman, F., Meng, L., 2015. "A review of online dynamic models and algorithms for railway traffic management". *IEEE Transactions on Intelligent Transportation Systems*, *16*, 1274–1284.

D'Ariano, A., Pacciarelli, D., Pranzo, M., 2007. "A branch and bound algorithm for scheduling trains in a railway network". *European Journal of Operational Research*, *183*, 643–657.

Findler, N. V., Stapp, J., 1992. "Distributed approach to optimized control of street traffic signals". *Journal of Transportation Engineering*, *118*, 99–110.

Kersbergen, B., van den Boom, T., De Schutter, B., 2016. "Distributed model predictive control for railway traffic management". *Transportation Research Part C: Emerging Technologies*, *68*, 462–489.

Kuwata, Y., How, J. P., 2011. "Cooperative distributed robust trajectory optimization using receding horizon MILP". *IEEE Transactions on Control Systems Technology*, *19*, 423–431.

Lamorgese, L., Mannino, C., Piacentini, M., 2016. "Optimal train dispatching by benders'-like reformulation". *Transportation Science*, *50*, 910–925.

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". *Transportation Research Part B: Methodological*, *115*, 41–71.

Meinel, M., Ulbrich, M., Albrecht, S., 2014. "A class of distributed optimization methods with event-triggered communication". *Computational Optimization and Applications*, *57*, 517–553.

Meng, L., Zhou, X., 2014. "Simultaneous train rerouting and rescheduling on an N-track network: A model reformulation with network-based cumulative flow variables". *Transportation Research Part B: Methodological*, *67*, 208–234.

Nedic, A., Ozdaglar, A., 2010. "Cooperative distributed multi-agent optimization". *Convex Optimization in Signal Processing and Communications*, *340*.

Negenborn, R. R., De Schutter, B., Hellendoorn, J., 2008. "Multi-agent model predictive control for transportation networks: Serial versus parallel schemes". *Engineering Applications of Artificial Intelligence*, *21*, 353–366.

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).