# A review of principles and methods to decompose large-scale railway scheduling problems

EURO Journal on Transportation and Logistics 12 (2023) 100107

Contents lists available at ScienceDirect

EURO Journal on Transportation and Logistics

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

# A review of principles and methods to decompose large-scale railway scheduling problems

Florin Leutwiler, Francesco Corman *

*Institute for Transport Planning and Systems, ETH Zürich, 8092 Zürich, Switzerland*

ARTICLE INFO

*Keywords:*
Scheduling
Timetabling
Benders
Dantzig–Wolfe
Multi-agent
Hierarchical methods

ABSTRACT

Providing punctual, reliable and performant services to customers is one main goal of railway network operators. The railway scheduling problem is to determine, ahead of time (timetabling), a plan describing the timing of the operations in a railway network, or updating such plan during operations (rescheduling). By optimization and automation, it is possible to operate more trains on the network, closer to the infrastructure capacity. Especially when the scale and complexity of the scheduling problem is increasing, for large-scale networks and multiple interconnected problems, this is of great value for network operators. When planning or adjusting railway operations becomes increasingly complex, modern scheduling algorithms can bring significant performance and economic benefits. In this survey we review approaches in the state of the art for the problems of railway scheduling. We show how the many different approaches of decomposition proposed in the literature of railway scheduling can be categorized into two general principles. We study different solution methods and identify a list of open topics for dealing with large-scale problems for future research.

## 1. Introduction

Railway operations are normally following a predetermined plan (timetable), which specifies times for arrival, departure and passing at stations. Such plan can be designed well ahead of time (timetabling) or adjusted during operations, when delays occur (rescheduling). When following the timetable, trains should not encounter congestion or result in unsafe situations with regards to infrastructure or other traffic on the network. The timetable is also the key to an efficient usage of railway resources and infrastructure. Careful planning of resource usage, especially for large-scale railway networks and dense traffic, enables a railway system to achieve high performance (Kroon et al., 2009).

The timetable can be designed (offline) a few days up to months before actual operation, normally based on a given planned demand for railway services. The objectives are often to maximize the amount of services running, to fulfill the demand, to ensure a limited deviation from an ideal offer which could be politically determined. During operations, the timetable is adjusted, when delays occur, and real-time deviations need to be taken into account. The adjustment to unforeseen delays and disturbances is often targeting the minimization of deviations between the offline timetable and what is actually produced; or a quick recovery of the offline plan. This problem is known as railway rescheduling. Both problems deal with the time allocation of railway services, considering scarce resources such as infrastructure capacity or vehicles. They identify choices in timing of services; their explicit relative order or sequence; and the route that trains follow in the network or at stations. The present paper considers both problems, under the unifying term of railway scheduling.

In railway scheduling, operations of trains are generally described by means of events. One event can represent the start or end of an operation. Typical operations are traversals of trains over the railway network, or dwell processes at stations. The problem can be considered at different levels of detail, resulting in two typical models, macroscopic and microscopic. In macroscopic railway scheduling, railway stations are abstracted into nodes, such that the network is represented in a set of nodes and lines. A single operation of a train in macroscopic modeling is the traversal of a train from one station to another. In microscopic railway scheduling, the railway network is considered in segments of few hundred meters of railway track, i.e., block sections, which provide enough precision in control, also for safety systems. A single operation of a train in microscopic modeling is the traversal of a single block section. The time of an event in railway scheduling is generally represented either in continuous or time-indexed form. In case of indexed time, often a time–space graph is used which connects points, i.e., events, in the indexed space according to operations. Points outside of the graph are neglected for scheduling. Extensive surveys on models of railway scheduling can be found, e.g., in Fang et al. (2015) and Cacchiani and Toth (2012).

* Corresponding author.
*E-mail address:* francesco.corman@ivt.baug.ethz.ch (F. Corman).

https://doi.org/10.1016/j.ejtl.2023.100107
Received 14 December 2021; Received in revised form 4 February 2023; Accepted 21 March 2023
Available online 25 March 2023

From a computational point of view, railway scheduling has been proven NP-Hard in many different versions of the problem (e.g., Odijk, 1996; Mascis and Pacciarelli, 2002; Caprara et al., 2002). Scalability issues for large-scale problems are thus unavoidable and it is crucial to keep these issues under control through the design of novel methods. Current practice in the industry is to split up (often geographically) a large-scale scheduling problem and schedule separated parts individually. Manually or computer-aided, the local timetables are computed and afterwards merged together to a consistent (i.e., globally feasible) network-wide timetable. The merging process is in general suboptimal. Often such process is a kind of heuristic, rule-based process (e.g., high-speed trains have priority over suburban trains, first come first serve strategies etc.), where resource utilization potential is lost. Due to the heuristic merging, it is unavoidable that not all possible scheduling solutions are considered and evaluated.

Complementary to those suboptimal processes stands a continuously increasing demand in public transport and freight transport on railway systems, which requires an increasingly higher level of performance from railways. While the acquisition of new network resources, e.g., infrastructure or vehicles, is a possible way to increase performance, it is in general very expensive, slow and with little flexibility. Many railway companies, e.g., the Swiss Federal Railways, instead focus on improving the utilization of existing resources. Railway operators show an increasing interest in novel methods, which solve the railway scheduling problem at an optimal level (Borndörfer et al., 2010a). If large-scale practical instances could be solved to optimality, the existing resources could be used more efficiently, going beyond the shortcomings of today's practices and matching the potential already identified in academia, e.g. in Lamorgese et al. (2016).

A promising direction towards handling scalability issues of large-scale railway scheduling problems is the idea of decomposition. Similar to current practices, the problem is separated into smaller subproblems. In contrast to current practice, appropriate mathematical methods can reliably deliver good solutions, or even guarantee to find an optimal one. Decomposition has already been proven promising in numerous planning problems for railways (see e.g., Borndörfer et al., 2017; Lamorgese et al., 2016). Still a gap between academia and industry exists. The approaches of academia have been evaluated on scenarios of increasing complexity up to 80 trains over 600 block sections (Corman et al., 2010), 150 train over 968 block sections (Luan et al., 2020) or 130 trains over 53 block sections (Lamorgese et al., 2016). Real life large-scale networks, like the one of Switzerland, operate more than 10,000 trains per day over more than 5000 block sections. The need for novel methodologies to further scale up the complexity in automated railway scheduling becomes evident, and motivates our review on principles and solution methods of decomposition for railway scheduling.

In this review, we analyze the literature of decomposed railway scheduling and provide insights on the individual decomposition principles and solution methods, highlighting similarities and differences between different existing approaches in the literature. This review is thought to be interesting and useful for both researchers and practitioners. For researchers, we comprehensively analyze the railway scheduling problems in the literature, identifying which specific aspects and mathematical structures have been shown to be particularly suitable for decomposition. We also identify and motivate alternative mathematical descriptions of the problem. Moreover, we can identify combinations of mathematical models, decomposed according to some principles, and solution methods. This identifies some combinations which seem promising, even though have not been studied yet. For practitioners, we discuss how decompositions based on physical characteristics of the problem have been more popular, possibly because they can be more directly understood. Nevertheless, there are other approaches possible, which can be identified. We believe our review of solution methods enables the discovery of new practical applications as we review advantages of different methods.

We believe, with a coherent discussion of mathematical models, decomposition principles, and solution methods, we can inspire further development from an academic point of view and motivate possible industrial application of those concepts. This review focuses on the problems of railway timetabling and rescheduling, i.e., respectively the design and adjustment of a timetable for the operation of a railway system. Many other problems are also relevant, in a supply of railway services: determining which infrastructure is needed, how to structure lines of the required services, which vehicles should be available, and for which services they should be used, how much personnel is required and on which services it should be driving. In the same spirit of the decomposition that this paper discusses, those problems are most often treated independently, due to different time scales, and different stakeholders and objectives. In this review, we assume that the input to a scheduling problem is given, from a solution to some problems upstream, and we can assume that the scheduling problem will influence other problems downstream. We will neglect upstream and downstream problems (such as line planning, timetable rolling stock and crew scheduling), and focus only on timetabling and rescheduling.

The remainder of this review is structured as follows. In Section 2 we review different domains of decomposition in the literature and analyze decompositions in the literature from a railway point of view. In Section 3 we discuss principles of decomposition to understand the decompositions of Section 2 from a mathematical point of view. In Section 4 we review solution methods of mathematical optimization from the literature to understand how a decomposed problem is addressed to determine a globally feasible (and optimal) solution. A discussion on strengths and weaknesses of different decompositions, but also on future research needs, is given in the final Section 5.

## 2. Decomposition in railway scheduling

Decomposition is a technique to solve mathematical problems, especially beneficial when dealing with large-scale instances, which generally show issues of scalability. A decomposition will operate on an original large-scale problem, reformulating it into a set of different, smaller and possibly multiple problems. We call *decomposition principle* the way an original problem is reformulated into multiple problems. Any of those problems can be solved with specific *solution methods*, which need to solve the problems themselves, but also harmonize their solution into something which satisfies the constraints of the original problem. The decomposition is the combination of a decomposition principle and a solution method.

We label a decomposition as *hierarchical*, if one problem has responsibility of the coordination, and results in one-to-many (vertical) communication to many other problems. This identifies a *master* problem with the responsibility of coordination (hierarchically higher) and (possibly) many *subproblems* (hierarchically lower). The subproblems do not need to be of the same mathematical type and structure, and most often, they are not. Moreover, the subproblems communicating with a single master can also have same, or different, mathematical type and structure. With a small abuse of notation, we might call subproblems the entire set of problems in a decomposition, or just those problems, that, compared to a master problem, are at the same lower level.

We label a decomposition as *decentralized*, if it results in many-to-many (horizontal) communication between problems, that are of the same mathematical type and structure. We call those problems to be at the same level.

The two types of decompositions are not exclusive and can appear both in some complex approaches. Moreover, boundary cases exists which can be hard to classify. Depending on specific characteristics, we can call the multiple problems of one level as subproblems (which highlights that they are constrained by some other problem), and some other as masters (which highlights that the problem is coordinating some other problems). If there are multiple levels, it can be that

**Table 1**
Decompositions in railway scheduling.

| Domain | Aspect | Publications |
|---|---|---|
| Geographic (GEO) | Geographic Areas | Corman et al. (2010), Corman et al. (2014), Corman et al. (2012), Kersbergen et al. (2016), Parkes and Ungar (2001), Sinha et al. (2016), Luan et al. (2020), Luan et al. (2018), Toletti et al. (2020) Leutwiler and Corman (2022) |
| | Lines and Stations | Lamorgese and Mannino (2015), Lamorgese et al. (2016), Lamorgese et al. (2017) |
| | Junctions | Liu et al. (2019) |
| Temporal (TMP) | Overlapping Periods | Zhan et al. (2016) |
| | Disjoint Periods | Luan et al. (2020) |
| Entity (ENT) | Single Train | Shang et al. (2018), Bretas et al. (2021), Proença and Oliveira (2004), Bretas et al. (2019), Luan et al. (2017), Cacchiani et al. (2012), Cacchiani et al. (2008), Luan et al. (2020), Fay (2000) Brännlund et al. (1998), Caimi et al. (2009), Caimi et al. (2012), Borndörfer et al. (2010b), Perrachon et al. (2020), Borndörfer and Schlechte (2007), D'Ariano and Hemelrijk (2006), Parkes and Ungar (2001), Narayanaswami and Rangaraj (2015), Meng and Zhou (2014) Khadilkar (2019), Caprara et al. (2002), Caprara et al. (2006) |
| | Group of Trains | Herrigel et al. (2013), Liu and Dessouky (2017) |
| | Infrastructure | Borndörfer and Schlechte (2007), Borndörfer et al. (2010b) |
| Generic (GEN) | Ordering Binaries | Lamorgese and Mannino (2019), Keita et al. (2020), Liu and Dessouky (2017), Kersbergen et al. (2016) |
| | Routing Binaries | Keita et al. (2020), Liu and Dessouky (2017), Zhang et al. (2019), D'Ariano et al. (2008) |
| | Macroscopic Scheduling | Bešinović et al. (2016) |
| | Duration of Operations | Matos et al. (2021) |

the subproblem for one master is actually itself a master for another subproblem. Finally, we call *centralized* the original, non-decomposed problem.

Decomposition is often motivated by special structural properties of the original problem, which lead the subproblem(s) to be a problem significantly easier to be solved than the original problem. For instance, it may generate, from one large problem, numerous smaller subproblems that are computationally much easier to solve (e.g., due to a superlinear relation between instance size and complexity). Decomposition is advantageous in case the computational benefits from solving the decomposed problems prevails over the additional efforts of coordinating them. We here identify as typical the structure of a single master and multiple subproblems, but some approaches might be considered having a single subproblem; and/or no master problem.

A *coordination scheme* and a corresponding *solution method* is the way to solve the resulting problems of a decomposed reformulation. A coordination scheme is an unavoidable necessity in decomposition to achieve globally feasible (and optimal) solutions. Most solution methods are iterative schemes, where alternately master and subproblems are solved. Through the coordination scheme, solutions of the subproblems have an influence on the master problem to find feasible and optimal solutions. The combinations of principles to decompose and solution methods to coordinate result in the variety of decompositions we can find in the literature.

In this section we review decomposition approaches, looking at *domains* and aspects by which academic approaches have decomposed the problem of railway scheduling. Domains classify structures in the mathematical model of the scheduling problem, which are exposed for a decomposed reformulation of the problem. We use the concept of domains, to link structures of the mathematical model to physical or mathematical characteristics of the scheduling problem and identify possible advantages and disadvantages for different domains. In our review, we identified four domains of decomposition in the literature: the geographic, temporal, entity and generic domain. Within each domain, a few aspects can be identified, based on which an original problem is separated into master problem and subproblems. While the geographic, temporal and entity domain have strong ties to the physical application of railways, we categorize a decomposition as generic if the related structure exploited has ties to general mathematical optimization. Still for most of the decompositions in the generic domain, a railway related interpretation can be made. In general, the geographic, temporal and entity domain can be seen as special cases of the generic domain. Table 1 summarizes the domains and aspects in the decompositions of the literature. We will show later in Section 3 how particular decompositions are achieved at a mathematical level.

### 2.1. Geographic domain

In decomposition approaches of the geographic domain (GEO), the original (centralized) scheduling problem is separated such that different subproblems correspond to different geographic areas, based on the underlying structure of the railway network. Events in the same subproblem correspond to operations which take place in the same area. The majority of geographic decompositions in the literature propose areas containing several stations, which generally happen to be of uniform size. The master problem in such decompositions considers the traffic in-between areas. A smaller number of geographic decompositions propose a decomposition by lines and stations. In this case, each individual station is defined as an individual area, corresponding to an individual subproblem; one large area contains what might remain of the entire railway network, defining the master problem.

### 2.2. Temporal domain

In decomposition approaches of the temporal domain (TMP), the original scheduling problem is separated into subproblems, each of

which covers a reduced time period, such that all together cover the temporal span of the original problem. The master problem in such decompositions considers the railway traffic at temporal boundaries. For a temporal decomposition it is necessary to know, for each event, a time window when it can take place, such that events can be assigned to a particular time period (i.e., a subproblem). In rescheduling, such assignment can be done using the original timetable. Reduced time periods may be overlapping in time, such as Zhan et al. (2016) or be disjoint such as Luan et al. (2020). The consequences of such choice will be analyzed in Section 3. Decompositions in the temporal domain are structurally very similar to decompositions in the geographic domain (see also Section 3). We can consider the concept of neighboring subproblems as those sharing some events or having events that are somehow constrained to events of each other. In this case, geographic decompositions have a variable number of neighbors, depending on the topology. Temporal decomposition instead have at most two neighboring subproblems (one earlier, one later).

### 2.3. Entity domain

In decomposition approaches of the entity domain (ENT), the original scheduling problem is separated into subproblems, each of which considers the scheduling a single entity. Entities can be any arbitrary set of railway resources (e.g., a train, or an infrastructure element). The master problem in such decomposition contains the interactions amongst entities. Possible entities identified in the literature are individual trains as individual entities or a group of trains as a single entity; there exists also the case of a decomposition where one entity is the set of all trains and a second entity is the set of all block sections in the network.

For the majority of decompositions in the domain of entities, individual trains are considered as individual entities. In this case, each train is scheduled in a single subproblem. Other decompositions in the domain of entities define a group of trains as a single entity. In this case, a single subproblem concerns the scheduling of all trains in the respective group. Grouping of trains can be conducted based on, e.g., priority (Herrigel et al., 2013) or train-type (Liu and Dessouky, 2017). A special case in the domain of entities are Borndörfer and Schlechte (2007), Borndörfer et al. (2010b) where exactly two entities are defined. One entity represents all trains of the scheduling problem and a second entity represents all block sections of the scheduling problem.

In this decomposition, trains and infrastructure usage are optimized independently and the master problem is to assure the match between them. In the subproblem that is related to the entity of trains, all trains are scheduled without considering their interaction on the infrastructure. Therefore, each train can be scheduled independent of the other trains in an individual, smaller subproblem. In the subproblem related to the entity of infrastructure (block sections) the interactions of trains are modeled to assure a conflict free schedule.

### 2.4. Generic domain

In decomposition approaches of the generic domain (GEN), the original scheduling problem is separated into subproblems, based on the structure of the underlying mathematical optimization problem, not directly motivated by the physical railway system. In the generic domain, the literature shows exclusively decompositions, which exploit structures related to particular types or classes of variables. Generic decompositions exploiting structures in the constraints of a scheduling problem could not be found. In case of structures related to variables, subproblems are defined by means of variables to be optimized in the subproblem. Variables not in the subproblems are optimized in the master problem. Differently from the previous domains, where subproblems are scheduling problems and often numerous, in the generic domain it is possible that subproblems are not scheduling problems and only a single subproblem is considered.

As an interesting fact, all decompositions in the generic domain reviewed for this work, with the exception of Matos et al. (2021), are based on time-continuous formulations. Furthermore, compared to other domains the generic domain is the most diverse domain, where subproblems related to different aspects substantially differ in problem-type and complexity. We discuss this aspect in more detail in Section 3.1.

## 3. Principles of decomposition

From a perspective of mathematical optimization, a problem suitable for decomposition should have a special structure, which causes the decomposed problems to be significantly simpler than the original problem. If such structure is absent, a decomposition is usually not beneficial with respect to computational speed.

Assuming a problem can be written as a (linear) mathematical optimization problem in standard form, i.e., $min\ c^T x \quad s.t. \quad Ax \geq b$, the structure of the constraint matrix $A$ may highlight a possible decomposition. If such matrix has a block-diagonal structure, the individual blocks can be solved independently, leading naturally to a decomposition into multiple problems of smaller size. Often, problems that are suitable for a decomposition show an almost block-diagonal structure. That is, apart from a few entries in the constraint matrix, all others are within a block-diagonal structure. In this case, the problem can be reformulated, considering entries outside of the block-diagonal structure in the master problem and each block as an individual subproblem. The resulting subproblems, which are smaller in size can effectively be parallelized and are often much easier to solve (e.g., due to a superlinear relation between instance size and complexity).

Even if the problem is not itself transformable into a block-diagonal structure of constraints, decomposition can be interesting. In fact, a problem is suitable for decomposition if it can be separated into master and subproblem, such that the subproblem (even without a block-diagonal structure) becomes of a different problem type, that by itself is significantly simpler to solve (e.g., the subproblem belongs to *P* instead of *NP* as the original problem).

In both cases, a coordination scheme (used by a solution method) makes sure the individual solution(s) of master and subproblem(s) are modified accordingly, to result in a solution for the original (centralized) problem.

In the following, we review how existing approaches exploit the two discussed properties (natural decomposition into smaller problems; easier class of problems). Two big classes of approaches exist: the principle of decomposing by *complicating variables* and the principle of decomposing by *complicating constraints* (Conejo et al., 2005). We review both in the subsections below with respect to the literature.

### 3.1. Complicating variables

In the principle of complicating variables, the term “complicating variables” refers to a subset of variables in an optimization problem, which is known to cause a significant part of its computational complexity. If the complicating variables are set to constant value, the residual optimization problem over the remaining variables is computationally significantly simpler, either due to a natural decomposition or an easier problem class, e.g., *P* instead of *NP*.

In a decomposition of complicating variables the optimization over the complicating variables is separated from the optimization of the remaining, i.e., non-complicating, variables. The non-complicating variables are optimized in a subproblem, while considering the complicating variables as fixed. The optimization over the complicating variables is done in the master problem. We summarize different identifications of complicating variables from the literature (together with a railway

**Table 2**
Complicating variables in decomposed railway scheduling.

| Publication | Domain | Description | Interpretation (Master/Subproblem) |
|---|---|---|---|
| Lamorgese and Mannino (2015), Lamorgese et al. (2016), Lamorgese et al. (2017) Leutwiler and Corman (2022) | GEO | Variables of traffic between stations. | Scheduling between stations/ Scheduling a station. |
| Corman et al. (2010), Corman et al. (2012), Corman et al. (2014) Sinha et al. (2016), Proença and Oliveira (2004), Kersbergen et al. (2016) | GEO | Variables of traffic between areas. | Scheduling between areas/ Scheduling an area. |
| Parkes and Ungar (2001) | GEO | Variables of traffic between areas only of a single train. | Scheduling a single train/ Scheduling an area. |
| Liu et al. (2019) | GEO | Variables of traffic between areas of different junctions. | -/Scheduling a junction area. |
| Zhan et al. (2016) | TMP | Variables of traffic between time periods. | -/Scheduling a time period. |
| Herrigel et al. (2013), Liu and Dessouky (2017) | ENT | Variables related to groups of trains. | -/Scheduling a group of trains. |
| Shang et al. (2018), Bretas et al. (2019), Bretas et al. (2021) Perrachon et al. (2020) | ENT | Variables related to individual trains. | -/ Scheduling a single Train. |
| Bešinović et al. (2016) | GEN | Variables related to macroscopic scheduling model. | Macroscopic scheduling/ Microscopic scheduling. |
| Lamorgese and Mannino (2019), Liu and Dessouky (2017), Keita et al. (2020) Kersbergen et al. (2016) | GEN | Variables related to precedence of trains. | Ordering of trains/ Scheduling of trains. |
| Keita et al. (2020), Liu and Dessouky (2017), Zhang et al. (2019) D'Ariano et al. (2008) | GEN | Variables related to routing decisions of trains. | Routing of trains/ Scheduling of trains. |
| Matos et al. (2021) | GEN | Variables related to durations of operations. | Setting durations/ Scheduling with given durations. |

specific interpretation of master and subproblem) in Table 2 and report on the resulting decomposition in Table 3. In most publications, complicating variables have not been identified explicitly as such by the authors themselves. We instead systematically apply this concept of complicating variables, as we believe that it helps in bringing different decompositions to a common denominator. In Table 3, we report on the type of master and subproblem, where the type "Scheduling" refers to a scheduling problem (explicitly including ordering decisions), as such a mixed-integer linear problem. In case ordering is excluded, we denote it by "(LP)" to indicate the non-integrality of such problem. The types "Ordering" and "Routing" refer to integer linear problems; integrality is required to describe the ordering and routing decisions of railway scheduling respectively. Other types used in Table 3 are self-explanatory. The last column reports respectively the type of master and subproblem.

In the geographic domain, we see by Table 2 that such decompositions are achieved by fixing (i.e., determining as complicating) variables related to traffic passing the borders between different areas, stations or junctions. Complicating variables describe the sequence and timing of traffic over borders, therefore the constraint matrix shows a block-diagonal structure for non-complicating variables and multiple subproblems can be identified. We see such phenomena in Table 3 as all decompositions in the geographic domain show as many subproblems as areas, stations or junctions, which are in general independent. We further see in Table 3 multiple approaches with no master problem. In such cases complicating variables are assigned to subproblems; and subproblems then are dependent on each other. In Section 4, we will see that particular solution methods can be used, which result in independent solution processes for the subproblems, such that they can be solved in parallel. A special case is the approach of Proença and Oliveira (2004), where complicating variables are further divided into smaller groups, in particular one group for each train. The result are multiple master problems, as many as trains. Those problems are independent, because interactions among trains are handled in the subproblems, which consider the traffic inside different areas.

**Table 3**
Decompositions of complicating variables in railway scheduling.

| Publication | Domain | Master problem | | | Subproblem | | |
|---|---|---|---|---|---|---|---|
| | | *N* | Indp. | Type | *N* | Indp. | Type |
| Lamorgese and Mannino (2015), Lamorgese et al. (2016), Lamorgese et al. (2017) | GEO | 1 | – | Scheduling | # Stations | Yes | List Coloring |
| Leutwiler and Corman (2022) | GEO | 1 | – | Scheduling | # Stations | Yes | Scheduling |
| Proença and Oliveira (2004), Kersbergen et al. (2016) | GEO | – | – | – | # Areas | No | Scheduling |
| Corman et al. (2010), Corman et al. (2012), Corman et al. (2014) Sinha et al. (2016) | GEO | 1 | – | Scheduling | # Areas | Yes | Scheduling |
| Parkes and Ungar (2001) | GEO | # Trains | Yes | Shortest Path | # Areas | Yes | Scheduling |
| Liu et al. (2019) | GEO | – | – | – | # Junctions | No | Scheduling |
| Zhan et al. (2016) | TMP | – | – | – | # Time Periods | No | Scheduling |
| Herrigel et al. (2013), Liu and Dessouky (2017) | ENT | – | – | – | # Train Groups | No | Scheduling |
| Shang et al. (2018), Perrachon et al. (2020) | ENT | – | – | – | # Trains | No | Scheduling |
| Bretas et al. (2019), Bretas et al. (2021) | ENT | – | – | – | # Trains + # Stations | No | Scheduling |
| Bešinović et al. (2016) | GEN | 1 | – | Scheduling | 1 | – | Scheduling (LP) |
| Lamorgese and Mannino (2019) | GEN | 1 | – | Ordering | 1 | – | Scheduling (LP) |
| Keita et al. (2020) | GEN | 2 | No | Routing Ordering | 1 | – | Scheduling (LP) |
| Kersbergen et al. (2016) | GEN | – | – | – | # Train Groups | No | Scheduling |
| Liu and Dessouky (2017), | GEN | 2 | No | Ordering Routing | 1 | – | Scheduling |
| Zhang et al. (2019) | GEN | 1 | – | Scheduling with Routing | 1 | – | Scheduling with Maintenance |
| D'Ariano et al. (2008) | GEN | 1 | – | Routing | 1 | – | Scheduling |
| Matos et al. (2021) | GEN | 1 | – | Scheduling Durations (LP) | 1 | – | Scheduling with fixed Durations |

Indp.: Independent Problems.

In the temporal domain, we see similar considerations as in the geographical domain. Instead of geographic areas, time periods are used to identify complicating variables (see Table 2); complicating variables are events between subsequent time periods. In our review of the literature we discovered only one related publication, where complicating variables describe traffic operating beyond the temporal boundary of two subsequent time periods. The single decomposition found keeps the complicating variables in the subproblem, such that subproblems remain dependent (see Table 3).

In the entity domain, we see decompositions where complicating variables are a subset of variables related to an entity. In particular, complicating variables are those variables related to an entity, which appear in a constraint together with variables of other entities. Entities used for the identification of complicating variables are single trains or groups of trains (see Table 2). Depending on the entities used, different decompositions are reported in Table 3. In the literature, no decomposition on complicating variables in the entity domain has been found, where a master problem is used and subproblems are independent due to an entity related block-diagonal structure in the subproblem. In the literature, complicating variables for the entity domain are in general kept inside the subproblems, such that these remain dependent. We will see that few publications apply solution methods, which are able to treat such subproblems independently by appropriate heuristics (see Section 4). A possible reason for no exploitation of block-diagonality may be that complicating variables in the entity domain are in general numerous, leading to a large master problem and are usually hard to identify.

In the generic domain, we find four classes of complicating variables in the literature (see Table 2). In the literature, complicating variables are identified either related to macroscopic scheduling, ordering or routing of trains, or the duration of operations (see Table 2). Variables for the order of trains exclusively appear in continuous-time models, such that any decomposition exploiting such variables can only be applied to a continuous-time model of railway scheduling. Furthermore, depending on the particular railway scheduling problem addressed, e.g., including of routing, different decompositions are reported in Table 3. In case of complicating variables related to ordering and routing, the master problem is a pure integer linear program, to optimize only over the binary variables related to ordering or routing decision. In case of Bešinović et al. (2016) the master is a macroscopic scheduling problem and in case of Matos et al. (2021) the master is a scheduling problem with no binary decisions, i.e., a linear program (LP). We can see in Table 3 that all proposed decompositions in the generic domain exploit not a block-diagonal structure in the subproblem, but a simplification of the problem class, such that there is always a single subproblem. Exceptions are Matos et al. (2021), where the subproblem cannot be simplified and Kersbergen et al. (2016) where complicating variables are kept in the subproblem and optimized in different groups (see Section 4). In two publications (i.e., Keita et al., 2020; Liu and Dessouky, 2017), complicating variables have further been grouped into different sets. In both cases binary variables related to routing decisions and ordering decisions are addressed separately.

### 3.2. Complicating constraints

In the principle of complicating constraints the term "complicating constraints" refers to subset of constraints in an optimization problem, which are the main cause for its computational complexity. If the complicating constraints are removed from the problem, the optimization over the remaining, non-complicating constraints is computationally, significantly simpler, either due to a block-diagonal structure in the

**Table 4**
Complicating constraints in decomposed railway scheduling.

| Publication | Domain | Description | Interpretation (Master/Subproblem) |
|---|---|---|---|
| Toletti et al. (2020), Luan et al. (2020), Luan et al. (2018) | GEO | Constraints in-between different areas. | Coordination of Areas/ Scheduling of an Area |
| Luan et al. (2020) | TMP | Constraints in-between different time periods. | Coordination of Time Periods/ Scheduling of a Time Period |
| Brännlund et al. (1998), Cacchiani et al. (2012), Caprara et al. (2002) Caprara et al. (2006) | ENT | Constraints of capacity between individual trains (AB-TSG). | Coordination of Trains/ Scheduling a single Train |
| Caimi et al. (2009), Caimi et al. (2012), Cacchiani et al. (2008) | ENT | Constraints of capacity between individual trains (PB-TSG). | Coordination of Trains/ Scheduling a single Train |
| Meng and Zhou (2014), Luan et al. (2017) | ENT | Constraints of capacity between individual trains (TI). | Coordination of Trains/ Scheduling a single Train |
| D'Ariano and Hemelrijk (2006), Luan et al. (2020) | ENT | Constraints of capacity between individual trains (TC). | Coordination of Trains/ Scheduling a single Train |
| Khadilkar (2019), Narayanaswami and Rangaraj (2015), Proença and Oliveira (2004) | ENT | Constraints of capacity between individual trains. | Coordination of Trains/ Scheduling a single Train |
| Borndörfer and Schlechte (2007), Borndörfer et al. (2010b) | ENT | Constraints of capacity between trains and block sections (PB-TSG). | Coordination of Blocks and Train/ Scheduling of Blocks and Trains |

(AB/PB)-TSG: Arc/Path-Based Time–Space Graph Model.
T(I/C): Time-Indexed/Continuous Model.

remaining constraints and/or as the problem belongs to a different complexity class of problems, e.g., *P* instead of *NP*.

In the decomposition of complicating constraints, the optimization over the complicating constraints (in the master) is separated from the optimization over the non-complicating constraints (in the subproblem). We summarize complicating constraints identified in decompositions approaches of the literature in Table 4 and report on the corresponding decompositions in Table 5. Similar to the case of complicating variables, in most publications, complicating constraints have not been identified explicitly as such by the authors themselves. We instead systematically apply this concept of complicating constraints, as we believe that it helps in bringing different decompositions to a common denominator. In Table 5, we report on the type of master and subproblems in the decomposition, where we refer by "Penalty Update" to a linear optimization problem over penalty parameters used to coordinate subproblems. We refer by "Set Packing" to an integer program with the basic structure of a set packing problem and possibly additional constraints, and by "Scheduling" to a mixed-integer problem, likewise as for Table 3. "Conflict Detection" and "Utility Evaluation" refer to an evaluation of solutions rather than to an optimization problem; the remaining types of problems in Table 5 are self-explanatory.

In the geographic domain, we can identify complicating constraints as the constraints between the different geographic areas (see Table 4). Areas might be defined differently in different publications, but all with the same goal of exploiting a block-diagonal structure leading to multiple subproblems. We can see the result of the block-diagonal structure in Table 5 as all subproblems in such decompositions are independent.

In the temporal domain, decompositions are very similar to decompositions of the geographic domain. The only difference to the geographic domain is that complicating constraints are not identified between different areas, but different time periods (see Table 4). As in the geographic domain, a block-diagonal structure is exposed, leading to independent subproblems (see Table 5).

In the entity domain, for all publications in the literature, we can exclusively identify constraints of network capacity as the complicating constraints. In this case, a train related block-diagonal structure is exposed; this leads to multiple smaller, independent subproblems, one for each individual train. In Table 4, we identify different complicating constraints related to network capacity, in relation to different underlying models of railway scheduling. In arc- and path-based models using a time–space graph, and also general time-indexed models, network capacity constraints result in clique constraints. Each clique is the set of arcs, paths, or time-indices, which would lead to a conflict on a single particular block section if used simultaneously in the timetable. In time-continuous models, network capacity constraints are often modeled as disjunctive precedence constraints between pairs of trains on each resource. The number of possible pairs for $n$ trains is at most $n^2$, such that the amount of network capacity constraints in time-continuous models is in the worst case $n^2 * m$, where $m$ is the number of resources. For time-indexed models, capacity constraints usually grow linearly with the number of resources, i.e., a single capacity constraint for each resource, in general resulting in smaller amount of capacity constraints. Special cases are the decompositions of Khadilkar (2019), Narayanaswami and Rangaraj (2015), Proença and Oliveira (2004). In these approaches, network capacity constraints are not explicitly considered, but network capacity is validated after the scheduling process of individual trains. We argue this is the same rationale as complicating constraints.

The entity domain is the most commonly used domain for decompositions based on complicating constraints. A possible reason to this might be the fact that in decompositions based on train entities on time-indexed formulations, subproblems are not only independent, but also reduce to shortest, least-cost or similar path-based problem (see Table 5), which in general can be solved extremely efficient. In absence of a time–space graph, i.e., in general time-indexed models, a time-dependent shortest path problem results. Hybrid situations occur when the time–space graph is updated iteratively according to possible train trajectories computed by a train dynamics model (Caimi et al., 2009, 2012). A special case are the decompositions in Borndörfer and Schlechte (2007), Borndörfer et al. (2010b), which are based on a reformulation of a time–space graph model. Additional variables are introduced to model the occupation of a block section by train, such that complicating constraints can be identified as those constraints between the variables of scheduling (events) and variables of occupation.

**Table 5**
Decompositions of complicating constraints in railway scheduling.

| Publication | Domain | Master problem | | | Subproblem | | |
|---|---|---|---|---|---|---|---|
| | | *N* | Indp. | Type | *N* | Indp. | Type |
| Toletti et al. (2020), Luan et al. (2020), Luan et al. (2018) | GEO | 1 | – | Penalty Update | # Areas | Yes | Scheduling |
| Luan et al. (2020) | TMP | 1 | – | Penalty Update | # Time Periods | Yes | Scheduling |
| Brännlund et al. (1998), Cacchiani et al. (2012), Caprara et al. (2002) Caprara et al. (2006) | ENT | 1 | – | Penalty Update | # Trains | Yes | Shortest Path |
| Caimi et al. (2009), Caimi et al. (2012) | ENT | 1 | – | Set Packing | # Trains | Yes | Trajectory Planning |
| Cacchiani et al. (2008) | ENT | 1 | – | Set Packing | # Trains | Yes | Shortest Path |
| Meng and Zhou (2014) | ENT | 1 | – | Penalty Update | # Trains | Yes | Shortest Path[a] |
| Luan et al. (2017) | ENT | 1 | – | Penalty Update | # Trains | Yes | Least-Cost Path[a] |
| Luan et al. (2020) | ENT | 1 | – | Penalty Update | # Trains | Yes | Scheduling |
| D'Ariano and Hemelrijk (2006) | ENT | 1 | – | Scheduling | # Trains | Yes | Utility Evaluation |
| Khadilkar (2019), Narayanaswami and Rangaraj (2015) | ENT | 1 | – | Conflict Detection | # Trains | Yes | Scheduling |
| Proença and Oliveira (2004) | ENT | #Areas | Yes | Conflict Detection | # Trains | Yes | Scheduling |
| Borndörfer and Schlechte (2007), Borndörfer et al. (2010b) | ENT | 1 | – | Penalty Update | #Trains + #Block Sections | Yes | Shortest Path |

Indp.: Independent Problems.
[a]: Time dependent.

The result are two subproblems, where one is to schedule all trains and another to schedule all occupations of block sections. In both these subproblems a block-diagonal structure is exposed and both problems naturally decompose into a shortest path problem for scheduling each train and each occupation of a block section individually.

The literature in decomposed railway scheduling does not show any decomposition based on complicating constraints in the generic domain.

## 4. Solution methods in decomposed railway scheduling

Complementary to the principles of decomposition, different solution methods are used to solve and coordinate master and subproblems back to a solution, valid for the original (centralized) problem. In the following, we review solution methods considered in the literature of decomposed railway scheduling. In general we will differentiate between exact methods and heuristic methods.

In the literature, we can identify purely hierarchical solution methods; purely decentralized solution methods; and a small group which has both characteristics at once.

For decompositions showing a hierarchical aspect, problems of the decomposition (master and subproblems) are arranged in hierarchical levels. In general problems of the same type are (considered to be) on the same level. In hierarchical solution methods, problems on the same level are not coordinated directly with each other. A (master) problem on a hierarchically higher level can be identified, responsible for coordination (i.e., one-to-many (vertical) communication). For decompositions showing a decentralized aspect, in general the problems (often of the same type) are on a single level and are coordinated directly with each other (i.e., many-to-many (horizontal) communication). No specific problem within the decentralized structure can be identified as responsible for coordination of others.

The way of coordination directly determines which problems can be solved in a specific order, or can be solved at the same time, i.e., parallelized. In decentralized solution methods, the problems can be solved in arbitrary order. Instead, in hierarchical solution methods problems of the same level can be solved independently (and parallelized), but different levels must be addressed in some specific (partial) order, e.g., a master problem has to be solved before/after all subproblems. In this sense, hierarchical solution methods are partially parallelizable, as the master must be solved only after all subproblems are solved, which means the total execution time depends on the computation time of both the master, and the slowest subproblem. Decentralized solution methods can achieve full parallelization.

We review purely hierarchical solution methods in Section 4.1, purely decentralized solution methods in Section 4.2, and we discuss in Section 4.3 solution methods which have a combination of both aspects. Furthermore, we review in Section 4.4 a small group of methods that are non-iterative and apply only unidirectional coordination over the problems of a decomposition.

### 4.1. Purely hierarchical solution methods

In purely hierarchical solution methods, problems of a decomposition are solved in a hierarchy over multiple levels, where in general the master problem(s) of a decomposition are considered hierarchically higher than the subproblem(s).

In hierarchical solution methods, coordination is achieved exclusively via different levels of hierarchy to establish consistency over all problems of a decomposition. The coordination between hierarchical levels (vertical communication) is in general bidirectional; from higher to lower hierarchies and vice versa.

In hierarchical decomposition, the coordination from higher to lower levels imposes some aspect of the solutions of hierarchically higher problems onto problems of lower level. We find two different schemes for this in the literature, either enforcing by constraints, or guiding through adjusted penalties. For simplicity we refer respectively to those concepts as *strict imposition* or *soft imposition*. With the exception of D'Ariano and Hemelrijk (2006), all publications considered for

this review show a direct relation between the kind of impositions used, and the principle of decomposition applied. Strict impositions are mainly the result of using the principle of complicating variables. Soft impositions are the result of using the principle of complicating constraints. In solution methods applying strict impositions, solutions of higher levels affect directly the solution space of problems of lower levels. In other terms, solutions of higher hierarchy result in determining some values for variables, to be imposed: complicating variables are fixed in the subproblems. In solution methods applying a soft imposition, solutions of higher levels are considered as parameters of a penalty term in the objective of problems of lower levels, guiding in this way their solutions process. Soft impositions are used in decompositions of complicating constraints to substitute complicating constraints through coordinating penalties.

Strict imposition has the advantage that solutions of the master problem and subproblem(s) are always consistent, building together a feasible solution for the original (centralized) problem. Therefore, once solutions for master and subproblems have been found, a solution for the original problem is given. As a drawback, strict imposition must consider the possibility that the constraints imposed on the subproblems by the master make them infeasible. Therefore, the solution process needs to handle infeasibility of subproblems (e.g., Corman et al., 2012; Lamorgese et al., 2016; Leutwiler and Corman, 2022). Moreover, it is often the case that no intermediate solution is available until a master solution has been found, for which also a feasible solution to all subproblems exists.

Soft imposition does not constrain subproblems. Therefore, subproblems are in general always feasible but not necessarily consistent with each other and thus not feasible with regard to the original (centralized) problem. As a consequence of soft impositions, especially for problems with integer variables (e.g., railway scheduling), it is often difficult to achieve consistency between subproblem solutions. In many publications, heuristics are used to recover a feasible solution for the original problem, from the inconsistent solutions of subproblems (e.g., Brännlund et al., 1998; Caprara et al., 2002; Luan et al., 2018). As an advantage, these heuristics can also be used to determine a solution at any intermediate point of the solution process.

The coordination from lower levels to higher levels in hierarchical solution methods provides feedback from subproblem(s) to the master problem(s). For such coordination we can identify in the literature five types of coordination schemes: *constraint generation*, *column generation*, *penalty functions*, *solution passing* and *negotiation*.

In Table 6 we summarize hierarchical solution methods of the literature and report the type of imposition, coordination scheme, particular solution method and possible optimality of the method. We discuss the coordination schemes of the literature in more detail below.

#### 4.1.1. Coordination by additional constraints: Constraint generation

In the coordination scheme of constraint generation, subproblems report feedback to the master problem in form of additional constraints, which are not part of the original problem. In general, these constraints are incrementally generated and considered throughout iterations of sequentially solving master and subproblems. The generated constraints are meant to represent the feasible space and optimal solutions of subproblems, using only variables of the master problem. Constraint generation is exclusively used for decompositions based on complicating variables and thus all related methods use strict impositions.

Logic Benders (Hooker and Ottosson, 2003) and classical Benders decomposition (Geoffrion, 1972) are well-known schemes from the field of mathematical optimization. Benders decomposition is a constraint generation scheme with proven optimality. Generated constraints, i.e., Benders cuts, are based on proofs of infeasibility and suboptimality of subproblems, which can be used inside the master problem. While in classical Benders decomposition a clear procedure is given to generate new constraints, in logic Benders decomposition the actual procedure for constraint generation has to be designed individually for each specific application. The authors, which proposed such logic Benders decomposition in Hooker and Ottosson (2003) simply propose guidelines for the constraint generation, rather than an exact procedure. For the logic Benders decomposition in Lamorgese and Mannino (2015), Lamorgese et al. (2016, 2017) the authors propose a novel constraint for their particular subproblem, i.e., scheduling traffic at a station. In Leutwiler and Corman (2022), a logic Benders decomposition is used, where subproblems share the same mathematical structure with the master and the centralized problem. For the Benders decomposition, a novel type of constraint is introduced, based on the infeasibility of a general railway scheduling problem. Instead, (Lamorgese and Mannino, 2019) applies classical Benders decomposition to railway scheduling. In Keita et al. (2020) a three-layer Benders decomposition has been introduced, where the master problem of a classical Benders decomposition is split into two layers. One master problem determines optimal routing in railway scheduling, and a subordinate master problem is to determine optimal precedences in railway scheduling. In Corman et al. (2012) the authors designed constraints for a subproblem that is the scheduling of a geographic area, which has the same spirit of Benders decomposition as they address infeasibility and suboptimality in subproblems similarly. In Bešinović et al. (2016) the master problem is a macroscopic scheduling problem, which includes macroscopic precedence constraints. These constraints are iteratively adapted based on the analysis of the subproblem, that is a microscopic scheduling problem.

Benders decomposition is particularly suitable for a decomposition, where the subproblems have large solutions spaces, that are rather independent from the solution space of the master problem. In this case only few additional constraints are necessary to represent the solution space of the subproblem, in the master; quick convergence can be expected. A railway related example is the decomposition of Lamorgese et al. (2016) where a subproblem relates to a station with many alternative routes for passing trains, all taking the same amount of time to pass. In such case, scheduling the trains outside the stations (which would be the master problem) is basically independent of routes used by trains in the station. A single constraint for the master problem would be sufficient to represent travel times of all routes in the station to the master.

#### 4.1.2. Coordination by additional solutions: Column generation

In the coordination scheme of column generation, subproblems report solutions to the master problem, which result in additional variables, i.e., columns in the constraint matrix of the master problem. Column generation can only be applied to decompositions in complicating constraints; and all related methods use soft impositions only. Dual values from a solution of the master problem are used for penalties in the subproblems (soft imposition) to generate appropriate new subproblem solutions, i.e., columns in the master. Approaches of column generation typically work iteratively adding columns to the master problems based on the incumbent solutions of the subproblems. Those latter are guided through iterative updates of the penalty in their objective, based on the incumbent master solution. The only solution method used for column generation in all reviewed works is the Dantzig–Wolfe reformulation.

Dantzig–Wolfe reformulation (Vanderbeck and Savelsbergh, 2006) is an approach to reformulate the solution space of a mathematical optimization problem through its vertices, i.e., solutions at the boundary of the solution space. The optimization problem after a Dantzig–Wolfe reformulation is to find an optimal convex combination of vertices. The vertices are the columns of such optimization problem. The reformulation is in general addressed by a column generation, where new vertices are iteratively generated. In case of decomposition, the Dantzig–Wolfe reformulation is used to reformulate only the non-complicating constraints. In this case, generating new vertices (i.e., subproblem solutions) is then, per definition of complicating constraints, a significantly simpler problem. Further, if the non-complicating constraints inherit

**Table 6**
Hierarchical solution methods in decomposed railway scheduling.

| Publications | Imposition | Coordination scheme | Solution method | Optimal |
|---|---|---|---|---|
| Lamorgese and Mannino (2015), Lamorgese et al. (2016), Lamorgese et al. (2017) Leutwiler and Corman (2022) | Strict | Constraint Generation | Logic Benders | Yes |
| Lamorgese and Mannino (2019) | Strict | Constraint Generation | Classical Benders | Yes |
| Keita et al. (2020) | Strict | Constraint Generation | Three-Layer Benders | Yes |
| Corman et al. (2012) | Strict | Constraint Generation | Benders-Like | Yes |
| Bešinović et al. (2016) | Strict | Constraint Generation | Constraint Adaptation | No |
| Borndörfer et al. (2010b), Borndörfer and Schlechte (2007), Cacchiani et al. (2008) | Soft | Column Generation | Dantzig–Wolfe | Yes |
| Caimi et al. (2009), Caimi et al. (2012) | Soft | Column Generation | Dantzig–Wolfe | No |
| Brännlund et al. (1998), Luan et al. (2017), Meng and Zhou (2014) Toletti et al. (2020), Cacchiani et al. (2012) | Soft | Penalty Function | Lagrangian Relaxation | No |
| Caprara et al. (2002), Caprara et al. (2006) | Soft | Penalty Function | Relax-and-Cut | No |
| Luan et al. (2018), Luan et al. (2020) | Soft | Penalty Function | ADMM | No |
| Corman et al. (2010), Corman et al. (2014), Sinha et al. (2016) Zhang et al. (2019), D'Ariano et al. (2008), Kersbergen et al. (2016) Matos et al. (2021) | Strict | Solution Passing | Multi-Shot | No |
| D'Ariano and Hemelrijk (2006), Narayanaswami and Rangaraj (2015) | Strict | Negotiation | Auction | No |

a block-diagonal structure, a single new vertex can be computed in parallel, over multiple independent subproblems.

In Borndörfer and Schlechte (2007), Borndörfer et al. (2010b), Cacchiani et al. (2008) a Dantzig–Wolfe reformulation is applied, where the column generation is carried out within a branch-and-bound scheme; these schemes are known as branch-and-price schemes, where pricing refers to the generation of new columns by appropriate penalties. Complicating constraints are capacity constraints, such that each column generated, that is a partial vertex and a solution of a subproblem, is the schedule of an individual train. Only with a branch-and-price scheme, the Dantzig–Wolfe reformulation and column generation is able to propose optimal solutions in case of mixed-integer programming, e.g., the railway scheduling problem.

In Caimi et al. (2009, 2012), Dantzig–Wolfe reformulation is applied similarly, but column generation is performed in a heuristic manner. Therefore, it is possible that the columns, which are part of the optimal solution, are never generated; if this is the case, the optimal solution cannot be found when solving the master problem. The approaches in Caimi et al. (2009, 2012) distinguish themselves from the literature, as they include train dynamics. Each column generated from a subproblem in such decomposition corresponds to a dynamically feasible train trajectory. Caimi et al. (2012) considers in comparison to Caimi et al. (2009) strengthened capacity constraints.

Column generation is effective if new columns, i.e., subproblem solutions, can be computed efficiently. In general, many different solutions from each subproblem (corresponding to columns in the master) are necessary, to make sure the master can find a combination of them that is an optimal solution to the original (centralized) problem. Therefore, it is crucial in column generation to compute subproblem solutions quickly. The most effective examples in railway scheduling are decompositions in the entity domain, where subproblem solutions are schedules of single trains and reduce to problems of shortest-path, which can be solved extremely efficient.

#### *4.1.3. Coordination by objective function: Penalty functions*

In the coordination scheme of penalty functions, solutions of subproblems are considered as parameters of a penalty function in the objective of the master problem. In particular the inconsistency between solutions of subproblems (a mismatch in solutions of the subproblems, determining a global infeasibility) manifests as a penalty in the master problem. The master problem on the other hand is an optimization problem over variables used to parameterize penalty functions in the subproblems. This scheme is known as Lagrangian relaxation (Fisher, 1985). The penalty function in subproblems, parameterized by the master, directs the optimization of subproblems inside the respective feasible area, away from solutions potentially inconsistent with

other subproblem solutions. Coordination by penalty functions can only be applied to decompositions in complicating constraints and is exclusively found in the literature, in the form of Lagrangian relaxation (Fisher, 1985); all methods of the coordination scheme of penalty functions use soft impositions only. Three different types are given: those applying the general scheme of Lagrangian relaxation, those using a relax-and-cut scheme, and those using ADMM (Alternating Direction Method of Multipliers).

In Brännlund et al. (1998), Cacchiani et al. (2012), Meng and Zhou (2014), Luan et al. (2017) a classical Lagrangian relaxation is applied to the constraints of network capacity. For (Brännlund et al., 1998; Cacchiani et al., 2012; Meng and Zhou, 2014) the subproblem reduces to a (time-dependent) shortest path problem. In Luan et al. (2017) maintenance constraints lead to a least-cost path problem. In all schemes above, the master problem computes a subgradient step for an update on variables used to parameterize penalty functions inside subproblems, based on the latest subproblem solutions. Often, e.g., in Brännlund et al. (1998), subproblem solutions are first adapted to be consistent with each other (i.e., build a centralized solution), and then used for the subgradient step. These two steps of first ensuring feasibility, and then updating the penalty, increases the convergence amongst subproblem solutions. Still, the scheme of Lagrangian relaxation is known to converge slowly; in fact, all related publications use some heuristic to compute a feasible upper bound solution at any intermediate point of the scheme. Most common is a heuristic where the penalty of a solution of the master problem is used to determine priorities to trains (subproblems); those are then, train by train, sequentially scheduled based on their priority. In Toletti et al. (2020) a Lagrangian relaxation has been proposed for a geographic decomposition. Complicating constraints at the border of geographic regions are relaxed. The solution scheme follows the standard Lagrangian relaxation scheme with a subgradient step to update penalty parameters (i.e., master variables). In case convergence cannot be achieved after a given number of iterations, a heuristic solution scheme is applied to coordinate the transitions of trains between different geographic areas.

In Caprara et al. (2002, 2006), Lagrangian relaxation has been paired with a lazy constraints scheme into a relax-and-cut scheme. In this case, during the iterative adaptation of the penalty parameters (i.e. master variables), only complicating constraints that are violated by the latest subproblem solutions are considered. This allows to reduce the computation of the subgradient step in the master problem. Also here, heuristics are proposed to compute intermediate feasible solutions.

In Luan et al. (2018, 2020) the authors used the model of their previous publication, i.e., Luan et al. (2017), to propose a decomposition in the geographic domain. To solve the decomposed problem, an alternating direction method of multipliers (ADMM) is applied. ADMM uses a Lagrangian relaxation of the complicating constraint to generate a penalty function for subproblems, but applies a different scheme to update the penalty parameters in the master. In particular, subproblems are solved sequentially; before the solution of each subproblem, the penalty parameters in the master are adapted based on the latest subproblem solutions. Like standard Lagrangian relaxation, ADMM has no guarantee to converge for problems of railway scheduling, and in fact a similar priority scheduling heuristics as in Caprara et al. (2002) has been used to compute intermediate feasible solutions. Luan et al. (2020) extends the decomposition principle of Luan et al. (2018) to different decompositions, and compares it to other heuristic coordination methods.

A coordination by penalty functions (Lagrangian relaxation) is most effective if subproblems can be solved efficiently. Often many updates of penalty parameters in the master problem are necessary to find consistent solutions for all subproblems and subproblems must be solved many times. The most effective examples in railway scheduling are, as for column generation, decompositions in the entity domain, where subproblem solutions are schedules of single trains, extremely efficient to compute, by solving, e.g., a shortest-path problem.

#### 4.1.4. Coordination by sharing solutions: Solution passing

We denote as solution passing a coordination scheme in which subproblems report feedback by communicating (passing) their entire solutions to the master, where in the master, depending on the actual implementation of solution passing, some part of the reported solution(s) is fixed in its own optimization to coordinate with subproblems. Solution passing is exclusively applied to decompositions based on complicating variables; and all methods of solution passing use strict impositions. According to our interpretation, a single solution method has been identified in the literature. We name such method as multi-shot; multiple tries (shots) are undertaken to find a global solution. Each try is the sequence of first solving the master problem and subsequently all subproblems. The important part is, that each try is carried out with different initial conditions, i.e., a slightly different master problem. The initial conditions to be included in the master are the results of solutions of subproblems passed to the master problem from the previous try (shot), hence multi-shot in solution passing. Most often, one master problem and several subproblems are considered (Corman et al., 2010, 2014; Sinha et al., 2016; Zhang et al., 2019; D'Ariano et al., 2008).

A special case on solution passing and multi-shot is the approach of Kersbergen et al. (2016). Compared to the approaches above, the decomposition of Kersbergen et al. (2016) has no clear master problem but only multiple dependent subproblems; subproblems are sequentially solved. The multi-shot characteristic in such approach is brought in, as after solving all subproblems in sequence, the sequence is re-solved again from the beginning, but this time considering the solutions of the previous iteration.

A different special case is the approach of Matos et al. (2021), where the decision at the master level is to determine the duration of all operations in the timetable; and a single subproblem evaluates if a feasible timetable exists for the durations determined by the master. At the master level, a multi-agent approach is used: each agent defines the duration of a single operation. We classify the scheme as solution passing, as agents adapt their solutions based on solutions of the subproblem.

All approaches of solution passing and multi-shot heuristically optimize different subsets of variables from the original scheduling problem in different problems of the decomposition. In Corman et al. (2010, 2014), Sinha et al. (2016) furthermore a block-diagonal structure is exploited. In general we understand the heuristics of this section to work best in cases where subproblems have large solution spaces with little ties to (i.e., few variables fixed by) the master problem (or other subproblems). A large solution space increases the likelihood for a feasible solution in the subproblem while few variables fixed by the master solution decrease potential coordination effort.

#### 4.1.5. Coordination by sharing intentions: Negotiation

In coordination schemes of negotiation, subproblems report feedback to the master problem in form of an intention. The intention represent the desire of a subproblem to take a certain solution, i.e., schedule. In an iterative process, the master problem allows or denies specific solutions based on previously shared intentions from subproblems. Based on the master's decision, subproblem update their intentions. In the literature we identified a single solution method in negotiation with purely hierarchical structure: auctions.

In auctions, subproblems report their intentions in form of bids. A bid represents the worth of a particular subproblem solution in perspective of the subproblem objective. The master problem acts as an auctioneer. The master receives all bids and then decides which subproblem wins the auction, and consequently the solution. Auctions are carried out multiple times, till a solution for each subproblem has been found. Auctions may also consider only parts of the solution of a subproblem, e.g., particular departure or arrival times of trains. In general, all auction-based decomposition methods we discovered in the literature of railway scheduling are also agent-based. That is, each

**Table 7**
Decentralized solution methods in decomposed railway scheduling.

| Publications | Imposition | Coordination scheme | Solution method | Optimal |
|---|---|---|---|---|
| Perrachon et al. (2020), Liu et al. (2019), Bretas et al. (2019) Bretas et al. (2021) | Strict | Negotiation | Multi-Agent | No |

subproblem is represented by an agent which computes the bids, as well as the auctioneer is represented by an additional agent. Auction based methods belong to methods using strict impositions; the result of an auction is strictly enforced to the subproblems, such that only the winning subproblem is allowed for a particular schedule.

In D'Ariano and Hemelrijk (2006) agents represent subproblems, each modeling a train; they bid on possible schedules provided by the master problem, i.e., the auctioneer. In Narayanaswami and Rangaraj (2015) agents are trains bidding for particular departure or arrival times at stations.

Auction-based methods are a category of multi-agent methods where convergence to a feasible solution is guaranteed, through the structured interaction of auctioneers and bidding agents. In general, there is no guarantee that other methods based on multi-agents will converge, i.e., agree on a common consistent solution.

### 4.2. Purely decentralized solution methods

In purely decentralized solution methods, the master and/or subproblems can be solved in an arbitrary order. All problems which are at the same level in the decomposition can be coordinated in a direct exchange of information with each other (vertical communication). A recent, more extensive review on the topic of decentralized railway scheduling can be found in Marcelli and Pellegrini (2021). Such review includes also applications beyond railways, where decentralized solution methods have shown potential.

The key difference in decentralized solution methods is that only methods using strict impositions are possible, where impositions are made on a single level of hierarchy. If otherwise soft impositions would be used, feasibility of a solution could not be guaranteed and centralized coordination would be necessary in some form, to recover a feasible solution for the original (centralized) problem (e.g., see Khadilkar, 2019). As strict impositions are required, decentralized methods of the literature are exclusively applied to decompositions based on complicating variables. Decentralized solution methods are applied in particular to those decomposition in Table 3 where no master problem or multiple master problems are considered; hence vertical communication is required. In these cases, the solution methods enforce a certain degree of parallelism among the various subproblems.

Regarding the coordination schemes in decentralized decomposition, the literature shows only a single scheme, i.e., the scheme of negotiation, where in particular agents negotiate between each other. We summarize all decentralized approaches of the literature in Table 7.

#### 4.2.1. Coordination by sharing intentions: Negotiation

The coordination scheme of negotiation is a coordination scheme that appears in both, hierarchical and decentralized decompositions. In hierarchical negotiation a master problem may only guide, reject, or restrict subproblem solutions in the process of negotiation. Instead, approaches in decentralized negotiation have no master problem and subproblems negotiate directly, mutually restricting each other's solution by strict impositions. In the literature reviewed, all solution methods using negotiation on a decentralized decomposition are agent-based approaches. Agents negotiate by exchanging (details of) their (current best) solution.

In Perrachon et al. (2020), Liu et al. (2019), Bretas et al. (2019, 2021) decentralized agent-based approaches are proposed. A full decentralization is possible due to strict impositions, i.e., in these approaches agents are not allowed to propose solutions with would conflict with solutions they received from other agents. In Perrachon et al. (2020), Bretas et al. (2019, 2021) agents represent trains. Each agent respects the solutions it has received from other agents, such that in case of a possible conflict, it determines for itself a different conflict-free solution accordingly. In Liu et al. (2019) agents represent different geographic areas. Agents exchange entry and exit times of trains; each agent adapts its schedule according to exit and entry times of trains it receives from other agents. In general, the decision processes of agents above are designed to converge to a consensus in an iterative exchange and adaptation of solutions.

Decentralized agent-based approaches can be parallelized to a high extent, as each agent is almost independent from anything else in its execution. The parallelism can be implemented locally on a single machine or spread over multiple physical machines.

### 4.3. Hierarchically decentralized solution methods

Hierarchically decentralized solutions methods have at the same time both aspect of hierarchical and decentralized decomposition. In these approaches, the problems of the decomposition show a hierarchical structure; within the hierarchical structure, some problems on the same hierarchical level are coordinated directly, in a decentralized manner.

In the literature, hierarchically decentralized solution methods have been found with both strict and soft impositions. For solution methods using strict impositions it is not necessary to have a single problem at the hierarchical top for coordination. Multiple problems can exist on the highest hierarchical level; these problems can be addressed in parallel. For solution methods using soft impositions it is necessary to have a single problem at the hierarchical top for coordination. We summarize hierarchically decentralized solution methods in Table 8.

#### 4.3.1. Coordination by sharing intentions: Negotiation

In all the literature, hierarchically decentralized solution methods are either auction or multi-agent methods where agents negotiate towards a consistent solution.

While auction-based methods of Section 4.1.5 have a single auctioneer, auction-based solution methods of this section have multiple auctioneers. In Parkes and Ungar (2001) agents represent trains, bidding at different auctions for entry and exit times into different areas of the network. Auctions are held for different geographic areas of the railway network. We consider the decomposition of Parkes and Ungar (2001) as a special case where neither auction agents, nor train agents communicate among agents of the same type. Rather global coordination is achieved as train agents participate at multiple auctions, i.e., the train agent assures consistency of a single train ride, and multiple train agents participate at a single auction, i.e., the auction guarantees no conflicts inside an area. In Table 3 we classified train scheduling as the master problems, and auctions as subproblems. In reality, there is no clear superiority of one the two of types of problems over the other, as train agents coordinate the result of different auctions but also auctions impose restrictions on train agents.

**Table 8**
Hierarchically decentralized solution methods in decomposed railway scheduling.

| Publications | Impositions | Coordination scheme | Solution method | Optimal |
|---|---|---|---|---|
| Parkes and Ungar (2001) | Strict | Negotiation | Auction | No |
| Proença and Oliveira (2004) | Strict | Negotiation | Multi-Agent | No |
| Khadilkar (2019) | Soft | Negotiation | Multi-Agent | No |

Compared to the multi-agent methods of Section 4.2.1, in the multi-agent methods discussed in this section, agents are partially arranged in a hierarchy. In such setting, there are both hierarchical coordination of problems; and decentralization. This latter is present when, for some levels, problems are coordinated directly instead of hierarchically. In other terms, there might be multiple problems (or agents), which are coordinated directly, on the highest hierarchical level. The other possibility is that on a lower hierarchical level, subproblems (or agents) are coordinated directly with each other, partially without the input from hierarchically higher levels.

In Proença and Oliveira (2004) a multi-agent approach is proposed where agents represent geographic areas on a hierarchically higher level, and train agents on a hierarchically subordinate level. Train agents exclusively communicate with agents of areas, about available block sections for their operations. Agents of areas coordinate with each other to assure consistency for inter-area traffic.

In Khadilkar (2019) a multi-agent approach is proposed where agents only represent trains plus a single coordination entity. Agents communicate directly to each other exchanging the incumbent best solutions, i.e., timetables. Train agents do not necessarily respect solutions received from other agents, but rather the received solutions are input to a penalty function, which is considered in the decision process of the receiving agent (soft impositions). Therefore, if the incentive for the agent is big enough, the agent may decide for a solution, whose execution would lead to a conflict with another agent. As a consequence, a final centralized recovery step is required, to adjust solutions of train agents, i.e., subproblems, for feasibility. The recovery step is a hierarchical element in the otherwise decentralized solution method. Building a schedule by a centralized (recovery) agent assures that no train agent may plan operations, which would lead to conflicts or deadlocks in the system.

Hierarchically decentralized solution methods combine the benefits of both hierarchical and decentralized methods. Ideally, such solution methods result in multiple problems equal in type (suitable for decentralization), where each subproblem is rather difficult to solve. Then, it makes sense to solve these individual problems by a second decomposition, i.e., a using a hierarchical solution method.

### 4.4. No coordination

In our review of the literature, we also identify several decompositions with solution methods where no explicit coordination scheme is applied. We classify as single-shot approaches with no coordination. That is, after a sequence of all problems of a decomposition is solved, either a feasible solution is found, or it is required that the fundamentals of the scheduling problem are changed, either rule-based or by human interaction. No retry is undertaken using different initial conditions and we may consider these solution methods as non-iterative.

In the literature all solution methods of no coordination apply strict impositions, such that if a solution is found after a single shot, such solution is guaranteed to be feasible regarding the original (centralized) problem. As such all these methods are decompositions of complicating variables (see Table 3). Furthermore no solution method in this class proposes an explicit master problem to exploit block-diagonality. Rather all subproblems are dependent on each other and solved sequentially.

In approaches of Shang et al. (2018), Herrigel et al. (2013), Liu and Dessouky (2017) different subproblems describe different trains or groups of trains. In Zhan et al. (2016) different subproblems relate to different time periods. An overview is given in Table 9.

Solution methods with no coordination exploit exclusively the super-linear decrease of complexity for decreasing problem size. Because each subproblem is significantly smaller than the original problem, the sequence of subproblems is in general solved faster than the original problem. Despite the computational speedup, methods of no coordination are not guaranteed to provide any solution.

## 5. Discussion

In this paper, we reviewed the academic literature discussing decomposed railway scheduling. We included early approaches, following mostly methods of mathematical optimization, but also more novel approaches based on data-driven solutions or agents.

We identified mathematical principles for decomposition, which depend on the structure of the mathematical models. A complementary analysis focused on solution methods; two properties have been identified, the presence of a hierarchy in the solution process, and the presence of decentralized communication within subproblems at the same level. For each type of structure, we identified different ways by which the decomposed problems are coordinated towards globally feasible and (possibly) optimal solutions. Here, we take a step back and reflect on the strength and potential for specific approaches.

### 5.1. Decomposition in different domains

Different decomposition domains often result in different sizes of master and subproblem, different complexities of these problems and also determine whether the subproblem(s) shows a block-diagonal structure. In the ideal decomposition, the complexity of the master and subproblem(s) is as low as possible. For the master problem, the complexity often directly relates to the amount of complicating variables or complicating constraints in a decomposition, i.e., the amount of variables or constraints in the master. For subproblem complexity, there is no similar easy pattern. We discussed ways of decomposition by which we may expose a block-diagonal structure in the subproblem or fundamentally change the class of complexity of the subproblem. The size alone does not determine the complexity of a subproblem, but rather the complexity strongly depends on the domain and principle of decomposition. It is therefore important to choose the right domain and principle for a specific problem at hand, to result in a performant decomposition approach. The ideal decomposition should have a small amount of complicating elements (variables or constraints), and the subproblems should display a block-diagonal structure, where blocks propose an optimization problem of an easier class than the original problem. We report in Table 10 on the characteristics of decompositions in different domains, as seen in the literature. We report for domain and principle an order of magnitude of the computational complexity as depending on complicating elements and for related subproblems; and how complexity is reduced, e.g., through a block-diagonal structure or a computational complexity problem class (e.g., $P$ instead of $NP$).

Geographic and temporal decompositions are promising, under the aspects of small sized master problem and block-diagonality in subproblems, but subproblems remain in the same complexity class as the original problem. The number of complicating elements relates to the amount of railway traffic between different geographic areas or time periods, and is usually much lower for those two domains,

**Table 9**
Non-coordinating solution methods in decomposed railway scheduling.

| Publications | Imposition | Coordination scheme | Solution method | Optimal |
|---|---|---|---|---|
| Shang et al. (2018), Herrigel et al. (2013), Liu and Dessouky (2017) Zhan et al. (2016) | Strict | None | Single-Shot | No |

**Table 10**
Characteristics of decompositions in different domains.

| Domain | Principle | # Complicating elements | Reduction of complexity |
|---|---|---|---|
| Geographic | CV | $\mathcal{O}$(Traffic between Areas) | Block-Diagonal |
| Geographic | CC | $\mathcal{O}$(Traffic between Areas) | Block-Diagonal |
| Temporal | CV | $\mathcal{O}$(Traffic between Time Periods) | Block-Diagonal |
| Temporal | CC | $\mathcal{O}$(Traffic between Time Periods) | Block-Diagonal |
| Entity | CV | $\mathcal{O}$(Single Entity) | Block-Diagonal |
| Entity | CC | $\mathcal{O}$(Capacity Constraints) | Block-Diagonal (& Problem Class) |
| Generic | CV | $\mathcal{O}$(Decisions) | Problem Class |

than for the entity or generic domain (e.g., see Luan et al., 2020); moreover the subproblems show in general a block-diagonal structure. For temporal decompositions, one has to be careful when separating the original time horizon into smaller periods. As Luan et al. (2020) shows, if arbitrary periods are chosen (regular in time against a very variable traffic, or irregular in time against a rather constant traffic), a temporal decomposition can result in large numbers of complicating elements, e.g., more than for a decomposition in the entity domain. A possible solution for temporal decompositions can be to adjust the length of time periods according to traffic density on the network over time. Choosing time periods for the decomposition dynamically, according to traffic density over time, we can minimize the amount of traffic operating in more than one time period and thus minimize ties (complicating elements) between different time periods. An interesting open challenge is to exploit recurrent patterns in both decompositions, for instance based on periodic timetables, to simplify the solutions. For instance, Benders cuts can be computed for a recurrent service pattern and then propagated to all relevant subproblems.

The entity domain is by far the most explored domain in decomposed railway scheduling. The decomposition with train entities shows in general very beneficial properties with many very small subproblem due to block-diagonality and moreover subproblems of a complexity class usually extremely simple (e.g., shortest path problems). Decompositions in the entity domain have also drawbacks, which include a very large number of complicating elements. In decompositions based on complicating constraints, capacity constraints are in general identified as complicating; those are especially numerous in dense traffic situations, resulting in increased coordination burden. Furthermore, the entity domain shows only few decompositions by means of complicating variables. We see here a big potential for novel approaches. If entities are trains or block sections (e.g., Borndörfer and Schlechte, 2007), a specific analysis may identify the potential conflicts amongst entities and identify those entities with many conflicts, such that variables related to those entities can be isolated as complicating variables. A possible result can be a partition of the problem in a complexity core; and a set of non-complicating entities, which have only little or almost no interference with each other, and can be therefore scheduled independently.

The generic domain results in very different approaches concerning the amount of complicating elements (variables) and the type of subproblem, in comparison to other domains. Often all or a subset of discrete decisions are identified as complicating. An example such as Lamorgese and Mannino (2019) shows, that if a decomposition is appropriately chosen, the structures in the railway scheduling problem can be exposed, which lead to impressive speed-ups. An open challenge in the generic domain is how to determine an appropriate choice of complicating variables to expose structures; several cases have already been reported, where exposed structure lead to a significantly reduced complexity of the subproblem. To this end, data driven methods based on clustering or other unsupervised learning, as well as domain transformation, can help solve the problem faster. Further, we did not come across any generic decomposition based on complicating constraints. It is important thus to analyze a railway scheduling problem for complicating constraints, whose removal can lead to a decomposition with favorable properties, i.e., few complicating constraints, block-diagonal or simple subproblems. In this sense, graph representations of the problem might be used to analyze connected components, and/or by flow or cut approaches.

In conclusion, no domain used for decomposition seems to outperform the other in terms of general potential. The specific characteristics of the railway scheduling problem to be solved can strongly suggest some or other approaches, depending on which variables and constraints it imposes at global and local level. We expect decompositions with a well balanced complexity between master and subproblems as favorable for large-scale applications.

### 5.2. Important aspects of decompositions

In the following we discuss a few important requirements and potentials for decompositions.

*Parallelization*. Computer architectures today are designed for efficient parallel computing, especially modern GPU's contain a very high number of cores, designed for highly parallelized computing. Still, whether such potential can be explored depends on the problem to be solved by the computer. With regards to parallel computing, decentralized solution methods (Section 4.2) have a clear advantage over hierarchical solution methods (Section 4.1). Hierarchical structures can also be parallelized, but the master (or single entity to which all subproblems communicate) must be run at once, and after all subproblems have been executed. Instead, every decentralized structure features many subproblems that can be run in parallel, lacking the possible bottleneck of a single master problem. An important issue in the parallelization is of course the communication, which should be fast, with high bandwidth (in case of geographically dispersed subproblems) and pertinent, i.e., exchanging what is needed to achieve consistency, but not much more. Instead, for hierarchical solution methods, we can revisit Tables 3 and 5 to identify cases of multiple independent subproblems. Those are in general very effective for parallel computing. In hierarchical solution methods the bottleneck remains the slowest subproblem in each iteration, as for each solution iteration, normally

all subproblems must be solved. This would suggest to balance the complexity of all subproblems (rather than the amount of sections, length of time interval, or trains).

*Dedicated Hardware*. In case the task for a computer is always the same in terms of operations to be done, the usage of dedicated hardware such as GPU's or FPGA's can lead to substantial speedup in computation. Revisiting Tables 3 and 5 to see which problems are faced during the different decompositions, we see two possible applications for dedicated hardware: In most publications reviewed on train-entity based decompositions, the subproblems reduce to problems of shortest path. Such problems are strongly structured and solved by efficient algorithms, which repeat only few operations many times. As such, these decompositions are very suitable for being solved via dedicated hardware. The second use case for dedicated hardware relates to agent-based approaches. Agents in such approaches are in general rule based, or taking actions based on machine learning algorithms. Both implementations require for the computation of an agents decision, several matrix multiplications, for which GPU's have proven extremely effective.

*Explainability of Solutions*. An important aspect in industrial applications is that the experts accept the computed solutions. Here, in general decomposition approaches could bring a benefit. Since solutions for the original problem are the assembly of different subproblem solutions, experts have the possibility to study also the subproblem solutions, and get more insight in the determination of a particular solution. Moreover, decomposition domains rooted in the real life problem (for instance based on geography, or time) are commonly used in real life, thus easier to understand by most operators. This could help for the acceptance of novel decomposition methods.

*Fast Availability*. An important aspect with respect to industrial applications is the fast availability of a first good solution, or even better, being extremely fast in determining the optimal solutions. There, a clear difference exists between decompositions of complicating constraints, and complicating variables. In decompositions of complicating variables, especially in the exact approaches, a subproblem determines a feasible and optimal solution given the latest master solution. It is likely that many iterations are necessary before a first feasible solution for a subproblem is encountered, making it hard to have a intermediate feasible solutions available quickly. The extreme case is that the first time a feasible solution for all subproblems has been found, is also the end of the algorithm, as it is the optimal solution (see e.g., Lamorgese et al., 2016). In decompositions of complicating constraints, subproblems often report some possible solutions. Either one can find an intermediate feasible solution for the original problem based on given subproblem solutions, or one of the many heuristics in the literature can determine intermediate feasible, yet not optimal, solutions. This could be an advantage in real-time truncated solution processes.

*Railway specific vs. Generic*. Looking at Tables 3 and 5 we see that many decompositions are inspired by traditional elements of railway, e.g., areas, junctions or trains. We see great potential in getting away from railway specific concepts and looking at the problem of railway scheduling in a more generic way. In other terms, studying and understanding the problem not by its physical aspects, but by exploiting its mathematical structure. An example of this idea is Lamorgese and Mannino (2019), where the variables in the mathematical problem are decomposed only by the type of variables, and not by domain understanding. The paper proposes to separate all integer from all non-integer variables, under a standard Benders decomposition approach, and reports a significant speedup. Generic decomposition approaches are a great opportunity to use the variety of generic tools which are increasingly made available in mathematical optimization.

*Indexing, or continuous Relaxation*. Many of the decompositions reviewed in this work rely for their decomposition on a time-indexed formulation of the railway scheduling problem. These formulations generally suffer under a more detailed time granularity, as the size of the solution space depends on the amount of time indices. In other terms, if operators desire timetables with high accuracy, i.e., a fine discretization of time, this becomes a big disadvantage for such models and decompositions. This mainly concerns decompositions where capacity constraints are identified as complicating constraints, as these decompose in time-indexed form. Continuous formulations suffer from a more detailed granularity only for numerical approximations, e.g., concerns of floating-point arithmetic, which can be needed in case of nonlinear constraints or objective functions. In contrast, continuous formulations in general include big-M constraints to model the discrete decisions of a railway scheduling problem. These constraints often show a poor linear relaxation. Instead, in general good (tight) linear relaxations are crucial for the efficiency of many algorithms that are used to solve mixed-integer problems (railway scheduling).

*Separability of objective function terms, extension to multi objective optimization*. One important factor we found in most of the decompositions we reviewed, is that the objective should be separable in terms of variables, i.e., does not contain any bi-linear or similar terms. If this is not the case, a separation into master and subproblem is often not possible, or possible only partially. In fact, it must be ensured that the variables, which appear together in a nonlinear term in the objective, have to be in the same problem. This can be either any of master or subproblem.

No decomposition approach reviewed deals with multi-objective optimization. Nevertheless, we believe that many decomposition approaches are easily extendable and some even as particularly beneficial, compared to the non-decomposed problems. In multi-objective optimization, where different objectives are optimized sequentially, e.g., when using methods of column or constraint generation, the columns/constraints generated during the optimization over one objective could partially or maybe even fully be reused in the optimization of other objectives.

### 5.3. Current research gaps, and directions for future research

The literature of decomposed railway scheduling shows many different promising decompositions. Nonetheless as shown in the introduction of this paper, a gap is still existing between the size of instances addressed in academia, and the size of instances, which the operators need to handle in real life, especially in microscopic scheduling. Most decompositions we reviewed, sacrifice to some extent optimality for the sake of speed. On the contrary, it is the main goal of railway operations to increase the capacity of their railway network through highly effective timetables. As such there needs to be a dialog between academia and industry how much optimality may be sacrificed.

Future research must address the issues of today's existing methods, which include the following: A clear issue amongst all works reviewed is the lack of studies on benchmark large-scale instances of railway scheduling. Many papers conclude stating that larger instances have been tackled by decomposition, compared to existing centralized approaches, but no exhaustive experiments have been conducted on openly available actual large-scale instances. Several papers state the issue, that in case of a non-optimal approach, it is extremely difficult to estimate the quality, or optimality gap, of an intermediate solution. It thus can hardly be estimated if it is worth to continue the computation or stop with the incumbent best solution. A number of publications, especially in column generation approaches, denote the issue of RAM as a limiting factor towards large-scale instances. RAM could be expanded, but this comes still at a high resource cost. Agent-based approaches have been mostly tested and especially been trained only on medium sized instances (i.e., 15 trains over 4 areas (Parkes and Ungar, 2001), 20 trains over 20 station (Perrachon et al., 2020), 48 trains over 2 junctions (Liu et al., 2019)). To prove a practical applicability of agent-based approaches on larger instances (e.g., 150 trains and 1000 block sections as in Luan et al. (2020)) further studies are necessary with an increased amount of agents, and associated complexity. The amount

of data (either from recorded operations, or from simulation, or both) necessary to train the agents in case of large-scale instances becomes an additional, significant factor of complexity.

Furthermore, many agent-based approaches reviewed in this work rely on more or less complex rules to guide the underlying decision process. The possibility to automatically learn such rules based on the instance and the realization is very attractive: Khadilkar (2019) and subsequently Bretas et al. (2021) use techniques from machine learning, in particular q-learning, to provide a decision process for agents. It is an interesting and yet open question whether other methods in machine learning, ranging from simple regression to sophisticated neural networks, can lead to an improved solution quality in railway scheduling. Here, the availability of benchmarks (such as the ongoing research at Swiss Federal Railways — SBB (Laurent et al., 2021)) allows collaborative investigation and benchmarks of specific approaches, which are typically only solved for specialized types of problems.

In a different direction, we see the integration of Benders decomposition very promising. First, all publications using such an approach are published within the last 6 years, which identifies it as a promising new idea. Also as we discussed earlier, especially for logic Benders decomposition, the design of appropriate constraints can be particularly elaborated to outperform what can be achieved by following a simple procedure. In this direction, the seminal paper (Hooker and Ottosson, 2003) proposes only a framework and the necessary requirements for the validity of appropriate constraints, but leaves the problem open, on how to design the constraints for different applications such as railway scheduling. We believe there is large potential for further improvement beyond the ideas of Lamorgese and Mannino (2015) and Leutwiler and Corman (2022).

Finally, we see great potential in more generic decompositions. Most of the decomposition approaches we reviewed in this work propose decompositions based on ideas backed with a physical interpretation. It is relatively easy to interpret geographic areas, time periods and the entity of a train. Instead, generic decompositions can be less easy to be interpreted, but might have computational advantages. In this case, specific subset of complicating variables or complicating constraints can be determined, that fit very simple structures, or balance the load of the subproblems. In this sense, generic decompositions might exploit approximate descriptions of complexity, and use the variables and constraints of each subproblem (i.e., its complexity) as a hyper-parameter to be determined. We believe it is possible to achieve in such way decompositions closer to the ideal of a small master problem and simple independent, numerous subproblems.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

This project was supported by the ETH Zürich Foundation, Switzerland.

## References

Bešinović, N., et al., 2016. An integrated micro-macro approach to robust railway timetabling. Transp. Res. B 87, 14–32.
Borndörfer, R., Grötschel, M., Jäger, U., 2010a. Planning problems in public transit. Prod. Factor Math. 95–121.
Borndörfer, R., Schlechte, T., 2007. Models for railway track allocation. OpenAccess Ser. Inform. 7, 62–78.
Borndörfer, R., Schlechte, T., Weider, S., 2010b. Railway track allocation by rapid branching. OpenAccess Ser. Inform. 14 (August), 13–23.
Borndörfer, R., et al., 2017. Recent success stories on integrated optimization of railway systems. Transp. Res. C 74, 196–211.
Brännlund, U., et al., 1998. Railway timetabling using Lagrangian relaxation. Transp. Sci. 32 (4), 358–369.
Bretas, A., et al., 2019. Modelling railway traffic management through multi-agent systems and reinforcement learning. In: 23rd Int. Congr. Model. Simul. - Support. Evidence-Based Decis. Mak. Role Model. Simulation, No. December. MODSIM 2019, pp. 291–297.
Bretas, A.M., et al., 2021. A decentralised multi-agent system for rail freight traffic management. Ann. Oper. Res.
Cacchiani, V., Caprara, A., Fischetti, M., 2012. A lagrangian heuristic for robustness, with an application to train timetabling. Transp. Sci. 46 (1), 124–133.
Cacchiani, V., Caprara, A., Toth, P., 2008. A column generation approach to train timetabling on a corridor. 4OR 6 (2), 125–142.
Cacchiani, V., Toth, P., 2012. Nominal and robust train timetabling problems. European J. Oper. Res. 219 (3), 727–737.
Caimi, G., et al., 2009. Conflict-free train scheduling in a compensation zone exploiting the speed profile. Int. Semin. Railw. Oper. Res. (January), 162–177.
Caimi, G., et al., 2012. A model predictive control approach for discrete-time rescheduling in complex central railway station areas. Comput. Oper. Res. 39 (11), 2578–2593.
Caprara, A., Fischetti, M., Toth, P., 2002. Modeling and solving the train timetabling problem. Oper. Res. 50 (5), 851–861.
Caprara, A., et al., 2006. A Lagrangian heuristic algorithm for a real-world train timetabling problem. Discrete Appl. Math. 154 (5), 738–753.
Conejo, A., et al., 2005. Decomposition Techniques in Mathematical Programming. Springer.
Corman, F., et al., 2010. Centralized versus distributed systems to reschedule trains in two dispatching areas. Public Transp. 2 (3), 219–247.
Corman, F., et al., 2012. Optimal inter-area coordination of train rescheduling decisions. Transp. Res. E 48 (1), 71–88.
Corman, F., et al., 2014. Dispatching and coordination in multi-area railway traffic management. Comput. Oper. Res. 44, 146–160.
D'Ariano, A., Hemelrijk, R., 2006. Designing a multi-agent system for cooperative train dispatching. IFAC Proc. Vol. 39 (3), 369–374.
D'Ariano, A., et al., 2008. Reordering and local rerouting strategies to manage train traffic in real time. Transp. Sci. 42 (4), 405–419.
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.
Fay, A., 2000. Decentralized railway control based on autonomous agents. IFAC Proc. Vol. 33 (9), 83–88.
Fisher, M.L., 1985. An applications guide to Lagrangian relaxation. Manage. Sci. 27 (1), 1–18.
Geoffrion, A., 1972. Generalized benders decomposition. J. Optim. Theory Appl. 10 (4), 237–260.
Herrigel, S., et al., 2013. Hierarchical decomposition methods for periodic railway timetabling problems. Transp. Res. Rec. 2374 (1), 73–82.
Hooker, J.N., Ottosson, G., 2003. Logic-based benders decomposition. Math. Program. A 96 (November 2000), 33–60.
Keita, K., Pellegrini, P., Rodriguez, J., 2020. A three-step Benders decomposition for the real-time Railway Traffic Management Problem. J. Rail Transp. Plan. Manag. 13 (July 2019), 100170.
Kersbergen, B., van den Boom, T., De Schutter, B., 2016. Distributed model predictive control for railway traffic management. Transp. Res. C 68, 462–489.
Khadilkar, H., 2019. A scalable reinforcement learning algorithm for scheduling railway lines. IEEE Trans. Intell. Transp. Syst. 20 (2), 727–736.
Kroon, L., et al., 2009. The new Dutch timetable: The OR revolution. Interfaces 39 (1), 6–17.
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., 2019. A non-compact formulation for job-shop scheduling problems in traffic management. Oper. Res. 67 (6), 1503–1782.
Lamorgese, L., Mannino, C., Natvig, E., 2017. An exact micro–macro approach to cyclic and non-cyclic train timetabling. Omega (United Kingdom) 72, 59–70.
Lamorgese, L., Mannino, C., Piacentini, M., 2016. Optimal train dispatching by benders'-like reformulation. Transp. Sci. 50 (3), 910–925.
Laurent, F., et al., 2021. Flatland competition 2020: MAPF and MARL for efficient train coordination on a grid world.
Leutwiler, F., Corman, F., 2022. A logic-based benders decomposition for microscopic railway timetable planning. European J. Oper. Res. 303 (2), 525–540.
Liu, L., Dessouky, M., 2017. A decomposition based hybrid heuristic algorithm for the joint passenger and freight train scheduling problem. Comput. Oper. Res. 87, 165–182.
Liu, J., et al., 2019. A multi-agent based approach for railway traffic management problems. In: 2018 Int. Conf. Intell. Rail Transp. ICIRT 2018, IEEE.
Luan, X., et al., 2017. Integrated optimization on train scheduling and preventive maintenance time slots planning. Transp. Res. C 80, 329–359.
Luan, X., et al., 2018. Distributed optimization for real-time railway traffic management. IFAC-PapersOnline 51 (9), 106–111.
Luan, X., et al., 2020. Decomposition and distributed optimization of real-time traffic management for large-scale railway networks. Transp. Res. B 141, 72–97.

Marcelli, E., Pellegrini, P., 2021. Literature review toward decentralized railway traffic management. IEEE Intell. Transp. Syst. Mag. 13 (3), 234–252.
Mascis, A., Pacciarelli, D., 2002. Job-shop scheduling with blocking and no-wait constraints. European J. Oper. Res. 143 (3), 498–517.
Matos, G.P., et al., 2021. Solving periodic timetabling problems with SAT and machine learning. Public Transp. 13 (3), 625–648.
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. B 67, 208–234.
Narayanaswami, S., Rangaraj, N., 2015. A MAS architecture for dynamic, realtime rescheduling and learning applied to railway transportation. Expert Syst. Appl. 42 (5), 2638–2656.
Odijk, M.A., 1996. A constraint generation algorithm for the construction of periodic railway timetables. Transp. Res. B 30 (6), 455–464.
Parkes, D.C., Ungar, L.H., 2001. An auction-based method for decentralized train scheduling. Proc. Int. Conf. Auton. Agents 43–50.
Perrachon, Q., Chevrier, R., Pellegrini, P., 2020. Experimental study on the viability of decentralized railway traffic management. WIT Trans. Built Environ. 199, 337–344.
Proença, H., Oliveira, E., 2004. MARCS Multi-Agent Railway Control System. In: Lect. Notes Artif. Intell. (Subseries Lect. Notes Comput. Sci.), vol. 3315, Springer Berlin Heidelberg, pp. 12–21.
Shang, F., Zhan, J., Chen, Y., 2018. Distributed model predictive control for train regulation in Urban Metro Transportation. IEEE Conf. Intell. Transp. Syst. Proc. ITSC 2018-Novem, 1592–1597.
Sinha, S.K., Salsingikar, S., SenGupta, S., 2016. An iterative bi-level hierarchical approach for train scheduling. J. Rail Transp. Plan. Manag. 6 (3), 183–199.
Toletti, A., Laumanns, M., Weidmann, U., 2020. Coordinated railway traffic rescheduling with the Resource Conflict Graph model. J. Rail Transp. Plan. Manag. 15 (December 2019), 100173.
Vanderbeck, F., Savelsbergh, M.W., 2006. A generic view of dantzig-wolfe decomposition in mixed integer programming. Oper. Res. Lett. 34 (3), 296–306.
Zhan, S., et al., 2016. A rolling horizon approach to the high speed train rescheduling problem in case of a partial segment blockage. Transp. Res. E 95, 32–61.
Zhang, Y., et al., 2019. Microscopic optimization model and algorithm for integrating train timetabling and track maintenance task scheduling. Transp. Res. B 127, 237–278.