OR Spectrum (2011) 33:843–883
DOI 10.1007/s00291-009-0189-0

REVIEW PAPER

Railway track allocation: models and methods

Richard M. Lusby · Jesper Larsen · Matthias Ehrgott · David Ryan

Published online: 27 December 2009

Abstract Efficiently coordinating the movement of trains on a railway network is a central part of the planning process for a railway company. This paper reviews models and methods that have been proposed in the literature to assist planners in finding train routes. Since the problem of routing trains on a railway network entails allocating the track capacity of the network (or part thereof) over time in a conflict-free manner, all studies that model railway track allocation in some capacity are considered relevant. We hence survey work on the train timetabling, train dispatching, train platforming, and train routing problems, group them by railway network type, and discuss track allocation from a strategic, tactical, and operational level.

Keywords Railway optimization · Train routing · Train timetabling

1 Introduction

The railway industry possesses a variety of rolling stock routing problems that can be modelled and solved using Operations Research techniques. Such problems exist in several forms, depend on the underlying railway network, and arise at different levels in the planning process for a railway company. Allocating the track capacity of a railway network over time in a conflict-free manner can be considered, roughly speaking, the fundamental objective of any rolling stock routing/scheduling problem. The aim of this paper is to provide a broad overview of the models and methods proposed in the literature to solve several variants of rolling stock routing problems. Special

R. M. Lusby (✉) · J. Larsen
Department of Management Engineering, Technical University of Denmark,
2800 Kongens Lyngby, Denmark
e-mail: rmlu@man.dtu.dk

M. Ehrgott · D. Ryan
Department of Engineering Science, The University of Auckland, Auckland, New Zealand

emphasis is given to the more recent work on the difficult problem of routing trains through complex junctions/stations. Within this field of research several approaches exist; however, a survey that collectively summarizes all contributions is noticeably absent in the literature. While work on single track networks is included, this is by no means exhaustive. We review several important studies in this area to not only highlight the variety of solution approaches that exist, but also to show how models have developed over time to cater for more complex railway networks.

To set the context we begin by providing an overview of the planning process commonly adopted by most railway companies. Railway companies typically adopt a hierarchical decision-making approach when planning their operations (see Bussieck et al. 1997). This decomposition results in tractable problems at each level and circumvents the need to deal directly with the entire, highly complicated planning process. Figure 1 indicates the main subtasks involved as well as the order in which they typically occur. In general, the optimal solution at a particular level is required as input to the subsequent level. The ordering of the subtasks is also consistent with the classification of the problems as they appear at each of the strategic, tactical, and operational levels commonly found in large industries (see Assad 1980).

Problems at the strategic level are characterized by lengthy time horizons and typically involve resource acquisition. Central to the network planning phase of this planning level are problems which concern the construction, and/or modification of existing infrastructure. In planning for the future, railway companies must evaluate possible changes to the network and assess their impact, particularly from a capacity perspective. A railway network in its simplest form can be viewed as a graph of nodes and edges. Each node is associated with a point of the railway network where significant train interaction occurs and usually represents a station or a junction. The major difference between stations and junctions is that the former contain so called *platforms* where trains can stop to allow passengers to board and alight. An edge between any two nodes indicates the section of track interlinking them. This typically consists of parallel stretches of one-way track designated to rail traffic in each direction; however,

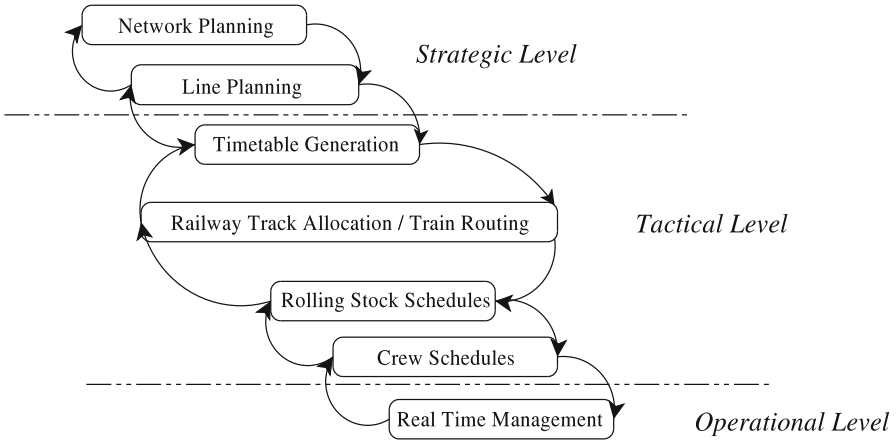

Fig. 1 The railway planning process

it is not uncommon for multiple edges (or even just a single edge) to exist between a pair of nodes. A *line* in a railway network refers to a route that starts and ends at a terminal station, connects several intermediate stations, and is operated by a particular train type. The frequency of a given line is the number of trains serving it over a certain time period. The *line planning problem* for a railway company entails selecting a set of lines (together with their frequencies). The selection of train lines is driven by forecast passenger demand between pairs of origin and destination stations, and typical objectives for this problem include minimizing operational cost or maximizing passenger satisfaction.

Tactical level problems focus more on allocating resources over an infrastructure that is assumed to be fixed. The first of four such tasks identified in Fig. 1 is timetable generation. Given the proposed line plan together with the desired frequencies, the *train timetabling problem* entails determining, for each train line, the arrival and departure time at each of the stations the line visits. The chosen times must satisfy both track capacity constraints (i.e. two trains cannot be on the same piece of track at the same time) as well as several railway company specific operational requirements. It is somewhat customary in the passenger railway industry for railway companies to operate a cyclic (or periodic) timetable. That is, a timetable that repeats itself at regular time intervals (e.g. every 30 min or every 60 min). To prevent trains from occupying the same piece of track at the same time one usually specifies a so called *headway*. The headway refers to the shortest time interval permitted between consecutive train movements on any given piece of track. This is understandably dependent on the speed capabilities of the trains under consideration. The operational requirements vary depending on the type of timetable under construction. If one is determining a timetable for passenger trains, then it is highly likely several requirements from a service perspective must be considered. For instance, one may wish to enforce certain train connections. In most, if not all, railway networks it is unlikely that direct connections between every pair of stations exist, and quite typically passengers are asked to transfer between lines at particular stations. To provide effective transfer possibilities, ideally the two train lines providing the connection should arrive at and depart from the same station during an overlapping time interval. Dwell time constraints are also likely to be enforced, i.e. how long trains may spend in the station. The time required by passengers to board and alight dictates the minimum amount of dwell time required. Placing a maximum tolerance on dwell time is usually an attempt to increase the utilization of the station's capacity by restricting the amount of time trains may remain in the station. For two trains that have a considerable portion of their respective lines in common, one may also wish to include the requirement that the two trains be coupled into one train at a given station (or similarly decoupled). Freight train timetables, on the other hand, typically have many fewer restrictions. The requirement that all train movements with a section of track in common satisfy a minimum specified headway is, however, shared. Quite typically the sequencing of trains (i.e. the order of the trains on the track) can only be altered at stations or junctions. This is definitely true if the track connecting such nodes of the network consists of two stretches of parallel track, each of which is dedicated to rail traffic in a particular direction.

There are several objectives one can consider when constructing a timetable. For timetables in passenger railways, common objectives include minimizing the

operational cost of the timetable or maximizing passenger satisfaction by providing many direct connections between stations. For the latter one could, for instance, minimize passenger waiting times (see, e.g. Wong et al. 2008). Other possible objectives include maximizing timetable *robustness*. Maximizing robustness entails finding a timetable in which delays to trains are less likely to propagate and impact the scheduled times of other trains (see, e.g. Fischetti et al. 2009; Liebchen and Stiller 2009). In some cases, however, simply finding a feasible timetable might suffice. Note that the first two objectives mentioned are conflicting in that the timetable with greatest passenger satisfaction usually incurs the highest operational cost. Other objectives involve minimizing the deviation from some ideal (i.e. most preferred) timetable, or minimizing the total time it takes the trains to complete their respective itineraries (known as the *transit time*). The latter objective typically defines the objective for the real-time version of the train timetabling problem in which the arrival and departure times of trains are adjusted so that the total accumulated delay is minimized.

Depending on the nature of the underlying network, the train timetabling problem does present itself in several different forms. In countries with extensive railway networks, such as those commonly found in Europe, nodes are highly interconnected networks of track where multiple railway lines meet, intersect and split. In the majority of cases, it is computationally prohibitive to consider the often large number of possible routings within a node when assigning arrival and departure times to each of the trains. In such situations one usually considers an aggregated rail topology in which the detailed layout of track within the node is ignored. A feasible timetable is then constructed with respect to this network. In a second step, one considers the detailed layout of the node and checks if the times found for the aggregated network allow for a feasible routing within the node to be obtained. This is what we term the *train routing problem*. If the node is a station, this second step involves the detailed allocation of trains to platforms. The related, so called *train platforming problem* is a special case of the train routing problem in which the selection of the route within the node completely dictates what platform will be used. An iterative procedure may be required if the timetable is infeasible with respect to the node. Iterative timetabling and routing procedures are described in Zwaneveld et al. (1996, 2001) and Burkolter (2005).

If, on the contrary, the network of track being considered is predominantly a long single stretch of track connecting multiple stations, where each of the stations has one (possibly two) *sidings* to facilitate train overtaking and stopping, the problem of train routing equates to finding a feasible *meet/pass* plan for the trains. The term siding refers to a section of track parallel to the main track and a meet/pass plan identifies at which siding, and at what time, two trains that require the same section of track will meet and which of the two will pass the other one. In the literature, this problem is referred to as the *train pathing problem*, or the *train scheduling problem*, and not without confusion, the *train dispatching problem*. This problem typically arises in the freight rail industry. In what follows, we refer to this problem as the train scheduling problem.

Other problems at the tactical level include the construction of rolling stock and crew schedules. Each itinerary in the chosen timetable must receive a train of a certain length as well as the necessary crew to operate it. One such problem entails the assignment of

a locomotive type to each train (see, e.g. Vaidyanathan et al. 2008). Perhaps the most important problem that arises when planning the rolling stock schedules, and which also necessitates railway track allocation, is the related, so called *shunting* problem. Shunting refers to the movement of train units to a shunting/parking yard when they are not needed for operation in the timetable. Such movements must be non-conflicting and separated by a minimum headway. The position of the train units in the shunting yard is critical since one does not want to place a train unit in such a position that it blocks the arrival or departure of another train unit. It is within the shunting yards where one must plan the composition of trains. When considering the shunting problem one might also wish to include the following aspects (among others): route selection to the shunting yard, train unit preferences for certain shunting yard tracks, and maintenance requirements. This problem is an entire research area in itself and will be omitted from this survey. We refer the interested reader to Freling et al. (2005) and Kroon et al. (2008).

Operational problems are defined to be those that occur on a day-to-day basis when operating policies determined at the tactical level need to be adjusted due to unforeseen disturbances. Efficient disruption management tools are essential to the success of any operation and have been employed in a wide range of industries. In the railway industry, late train arrival, track maintenance, or even accidents are likely to propagate through a timetable with varying degrees of severity, and quite possibly result in the planned schedules becoming infeasible. The dynamic environment in which these problems occur necessitates almost instantaneous resolution. One must quickly and efficiently adjust the planned schedules in such a way so as to return to a feasible schedule with minimal total disruption. Hence, the role of the real-time management level in Fig. 1 is to monitor the planned schedules and correct for any disruptions that occur on a daily basis. Typical operational problems hence include the rescheduling of trains (see, e.g Törnquist 2005) and crew (see, e.g. Walker et al. 2005; Rezanova and Ryan 2009) as well as the rerouting of rolling stock (see, e.g D'Ariano et al. 2008; Schöbel 2009) in managing the impact of delays.

Although Fig. 1 indicates that the problems are solved in a top-down manner, this may not necessarily be true. Bussieck et al. (1997) and Lindner (2000) point out that it may be necessary to cycle back and alter former decisions, or even consider subsequent levels when at a particular planning level in the process. For instance, it is unlikely that one would consider modifications to railway infrastructure at the network planning level without at least taking into consideration the level of rail traffic the proposed infrastructure would cater for, i.e. the line planning problem and the train timetabling problem. It may even be possible to consider two problems simultaneously. For example, Albrecht (2009) develops a two level approach that considers the line planning problem and the timetabling problem for suburban railways.

From the discussion above, it is evident that railway track allocation is an integral component in the tactical level problems of timetable generation and train routing. However, one can observe variants of these problems that arise at each of the strategic and operational planning levels. For example, in determining the capacity of some component of the railway network at the strategic level, one must allocate the associated railway track over a certain time horizon in such a way that the maximum number of trains can use it. For a junction this is often the maximum number of trains that can

be routed on it. Furthermore, on an operational planning level, when a disruption that makes the planned timetable infeasible occurs, one must reallocate the railway track in such a way that the negative impact of the disruption is minimized.

For an overview of all the main optimization problems arising at all planning levels in the passenger railway industry we refer the reader to Huisman et al. (2005) and Caprara et al. (2007b), while a previous survey in Cordeau et al. (1998) mainly considers the routing of freight trains on single track networks and reviews problems concerning railcar fleet management and locomotive assignment. In what follows we consider the train timetabling and train routing problems in more detail. Both problems share the fundamental objective of finding a set of mutually disjoint train movements that satisfy a variety of operational constraints. In particular, a minimum required headway must separate any pair of trains. We begin the review with a discussion on single track networks in Sect. 2 and move to more general networks in Sect. 3. Although Sect. 3 reviews work on more general railway networks, this is restricted to those studies that ignore the routing of trains through stations and junctions. This is the topic of Sect. 4 where we compare and contrast the work that has been completed in this area. We conclude the paper with some final remarks in Sect. 5.

2 Single track railway networks

The purpose of this section is to review the models and methods that have been employed to allocate track capacity in determining an operational timetable for single track railways. Hence, this work mainly pertains to the train scheduling problem. However, as will become apparent, some of the techniques are naturally applicable to more complicated routing situations.

One of the earliest published works on the application of optimization in the field of train scheduling is Szpigel (1973). The motivation for the work is a long single track railroad in eastern Brazil catering for trains which are used to transport iron ore in both directions. The track is divided into a number of track sections, with each track section linking two stations. Additional track is assumed to be available at the stations for trains to stop or overtake one another. The author is perhaps the first to identify strong similarities between the train scheduling problem and the well known job shop scheduling problem. With job shop scheduling problems one is given a set of jobs and a set of machines. Associated with each job is a chain of operations, where each operation requires a given amount of uninterrupted processing time on one of the machines. The operations for a particular job must be performed in sequence, and no machine can host more than one job at a time. The aim is to schedule jobs in such a way that the completion time (referred to as the *makespan*) for all jobs is minimized.

In the train scheduling context we have the following. Trains (jobs) require the use of several track sections (machines) to complete their designated itineraries. An operation is associated with the traversal of a track section, with the processing time of a track section being the minimum amount of time the train requires to traverse the track section. Operations are constrained from both a temporal and a spatial perspective. Temporal constraints are required to ensure that a train cannot traverse any given track section without having traversed the track section immediately preceding

it, whereas spatial constraints stipulate that no track section can host more than one train at any given time. Szpigel (1973) thus defines, for each train, decision variables pertaining to the entrance time to each track section of its route. The constraints of the model enforce the requirement that the time difference between the entrance times of consecutive track sections is at least the time required to traverse the first track section of the two. The departure time of a train at its station of origin is assumed to be known.

To prevent track sections from hosting more than one operation at any given time, Szpigel (1973) identifies so called *ordering* constraints. An ordering constraint ensures that, for trains sharing a common track section, the entrance time of each train to the track section is separated by the minimum specified headway. The model is solved using a branch-and-bound algorithm. The initial linear programming (LP) problem excludes any ordering constraints. Branching is required if the solution contains two trains in conflict; in other words, two trains are on the same track section at the same time. Two branches are enforced by imposing each of the respective ordering constraints. Imposition of an ordering constraint forces one of the trains to wait until there is sufficient headway between the trains, and thus incur a delay. The resulting linear programs are solved until a feasible meet/pass plan has been constructed with the objective being to minimize a weighted sum of train transit times. With the limited computing power available at the time, Szpigel (1973) considered instances involving five track sections and ten trains.

Constructing feasible meet/pass plans is also the topic of Sauder and Westerman (1983) and Jovanović and Harker (1991). Both approaches consider a similar network to that of Szpigel (1973) and present enumeration based methods. Given a proposed timetable for the trains, the method by Sauder and Westerman (1983) first identifies all potential conflicts as well as all stations (referred to as sidings in the paper) where such conflicts can be resolved. A complete enumeration of feasible meet/pass plans is constructed in the form of a tree structure. Each level of the tree represents the resolution of one conflict, whereas the nodes on any particular level identify the train that must wait and the siding where the conflict is to be resolved. Trains incur a delay cost in waiting at sidings, and hence the cost on any node defines the accumulated cost of the partial resolution strategy. Leaf nodes define feasible meet/pass plans. The objective minimizes the total delay weighted by train priority.

Jovanović and Harker (1991) propose a similar enumeration strategy to that of Sauder and Westerman (1983). Unlike Sauder and Westerman (1983), however, the model has no explicit objective function as it is primarily concerned with showing feasibility of a proposed schedule and providing the user with a number of feasible meet/pass plans. A mixed integer programming formulation (MIP), which the authors state is similar in structure to that of a flow shop scheduling problem commonly found in manufacturing, is presented. Binary variables dictate the ordering of pairs of trains on track sections, whereas continuous variables govern the selection of arrival and departure times for the trains. The constraints of the model enforce a number of temporal restrictions on the trains, restrict the number of trains that can be at any siding simultaneously to one, and also ensure that a siding can only be used to resolve a conflict if it is at least as long as the train that must use it. The SCAN I system the authors developed can consider single or double track sections between sidings and includes a simulation module to model train movements. Computational experiments

reported in the paper deal with instances involving 24 train lines and schedules for 100 freight and passenger trains.

Kraay et al. (1991) consider a slightly different version of the problem, which they term the *train pacing problem*. The authors argue that in determining a feasible meet/pass plan the speed of the train should be determined endogenously. That is, one should determine a speed profile for each train (i.e. pace the train) that minimizes some criterion (fuel consumption, delay, etc.) when determining a meet/pass plan. For this reason, the traversal time of a track section is considered variable, and the objective aims to minimize the fuel costs, which are obviously dependent on the speed of the train. The authors present a mixed integer nonlinear program for the problem. A branch-and-bound procedure combined with cutting planes is proposed to construct a feasible meet/pass plan. The model can, however, take as input any feasible meet/pass plan. Given a feasible meet/pass plan the objective is to optimize the train velocities. Hence the problem is typically solved for a set of feasible meet/pass plans. The paper discusses results for a 102 mile stretch of track interlinking 13 sidings. The corridor is traversed by 22 trains that have potentially 34 meet/pass conflicts.

A completely different solution approach is presented by Carey and Lockwood (1995). Carey and Lockwood (1995) consider an almost identical network to that of Szpigel (1973). However, the track section between any two stations (referred to as a link in the paper) is designated to rail traffic in the same direction. The authors present are large 0–1 MIP formulation similar to that of Jovanović and Harker (1991). Each binary variable governs the order of a pair of trains on a given track section. Temporal constraints identical to those of Szpigel's model preserve the time–space continuum of train motion. Unlike Szpigel (1973), however, the model includes all ordering constraints. The inclusion of the binary variables as well as a large positive number in such constraints governs which ones are activated and deactivated. Other constraints enforce the consistency of the binary variables. The objective minimizes the deviation from some ideal arrival and departure times for each of the trains. Given the size of the formulation, the authors adopt a heuristic approach which sequentially schedules trains. The order in which the trains are considered typically follows the chronological order of departure times or train priority. To schedule trains sequentially the authors suggest constructing a subproblem from the initial formulation by holding fixed the sequence of already scheduled trains, but still permitting arrival and departure times to vary. Each subproblem is then solved using a branch-and-bound technique. The approach also allows for the possibility of rescheduling trains once a feasible solution has been obtained in an attempt to improve the solution. Subsequent work by Carey (1994a,b) shows that this approach not only extends to the case where there are multiple lines between stations that have multiple platforms, but also the case in which the single stretch of track is bidirectional.

Other heuristic approaches for the train scheduling problem have also been proposed by Cai and Goh (1994), Higgins et al. (1996, 1997), and Cai et al. (1998). The focus is slightly different in that the authors consider developing tools for real-time decision support of the problem. That is, allocating track capacity in a real-time environment possibly as a disruption recovery measure. However, as Higgins et al. (1996) point out, the methods are equally applicable in an evaluative capacity when considering potential timetables for the trains.

Higgins et al. (1996) consider a single stretch of bidirectional track that is commonly found in the freight rail industry in Australia. To construct a feasible meet/pass plan, the authors present a 0–1 MIP formulation that is very similar in structure to that of Carey and Lockwood (1995). Binary variables define the order of two trains on a track section, whereas continuous variables model the arrival and departure times at the sidings on the track. Higgins et al. (1996) include extra flexibility in terms of train dynamics in that each train has a minimum as well as a maximum achievable velocity on any given track section. Hence, the model also includes some variability in the amount of time it takes a train to traverse any track section. The constraints of the model, as is customary with this problem, stipulate the temporal restrictions trains must respect. A large proportion of the constraint set consists of the train ordering constraints (identical to those of the model proposed by Carey and Lockwood 1995) which prevent train conflicts. The objective is to minimize the total train delay weighted by train priority.

Higgins et al. (1996) present a branch-and-bound framework that is quite similar to that of Szpigel (1973). The initial problem considered is a relaxed version of the problem in which none of the constraints on train interaction are included. Hence only a subset of constraints is included in the original problem. Conflicts are identified and resolved by creating two branch-and-bound nodes with the respective train interaction constraints enforced. At each node one calculates not only the immediate cost incurred as a result of delaying the train that prompted the branch, but also a lower bound on the expected total delay remaining due to the unresolved conflicts. This sum defines the cost of a node. The algorithm iteratively proceeds in this manner, adopting a depth first search of the tree by picking the node of the last resolved conflict with lower total cost. Instances involving up to 30 trains and 12 sidings are reported in the paper. In a subsequent paper, Higgins et al. (1997) develop and compare several heuristic techniques. In particular, a local search heuristic applied to the first solution found in the branch-and-bound tree method of Higgins et al. (1996) is compared with a genetic algorithm approach, a Tabu search heuristic, and two hybrid approaches. Instances of up to 50 trains with between 103 and 113 conflicts are considered in the paper.

Cai and Goh (1994) propose a simple greedy heuristic for the same problem. The authors make the assumption that all trains travelling in the same direction have not only the same speed, but also the same terminating siding. The heuristic considers trains in chronological order and assumes that the start time and initial siding location of each train are known. Given an incumbent train, an iteration first determines two sets of conflicting trains and then updates the current time and siding location of each set of trains. To determine the two sets of conflicting trains, the algorithm does not only consider all conflicts the incumbent train will encounter on travelling to the next siding, but also any conflicts between opposing trains travelling on the same track section as the incumbent train within a certain time interval. The algorithm determines the cost associated with delaying each group of trains (or each of the two trains if the only conflict is between the incumbent train and one other train) at the respective sidings and selects the allocation that results in lowest delay cost. Hence, it iteratively proceeds and resolves conflicts at the local level only and attempts to minimize total delay. A discussion on how to extend the model to incorporate train

passing is also provided. The paper presents two examples. Both of which had 12 sidings and considered 12 and 20 trains, respectively.

Cai et al. (1998) extend the work of Cai and Goh (1994). In particular, the authors relax the requirement that the initial location for a train be at a siding. In a real-time environment trains can be positioned anywhere on the network. Cai et al. (1998) present a two-phase approach with the first phase updating the current time and position (which the authors refer to as *position time pair*) so that all trains are positioned at a siding. The second phase then implements a refined version of the greedy heuristic in Cai and Goh (1994). A successful implementation of the algorithm is reported for an Asian railway company, where up to 400 trains run per day with as many as 60 trains in the system at any given time.

Brännlund et al. (1998) introduce the notion of packing constraints to restrict the number of trains using any track section (which is termed *block section* in the paper) at any given time to at most one. A bidirectional single stretch of track connecting 17 stations in Sweden provides the motivation for the work. The authors propose a large set packing integer programming formulation. The model discretizes the scheduling horizon into 1-min intervals, and a constraint is then identified for every track section in every minute. A unique acyclic time–space network consisting of six different arc types is used to model each train's movement, with paths in the time–space network reflecting different strategies for the associated train to complete its itinerary. The variables of the model are hence associated with possible train paths, with each variable contributing to the constraints that reflect its movement on the rail network. The objective attempts to minimize unnecessary waiting along the track. The authors suggest solving the model with Lagrangian relaxation techniques. Brännlund et al. (1998) relax all constraints of the model, and hence separate the problem into n independent subproblems, where n is the number of trains. The authors test and compare several dual variable update schemes and combine this with a train priority based heuristic to solve the problem. Computational results reported indicate that instances involving up to 30 trains (both freight and passenger) could be quickly solved within a few percent of optimality.

Şahin (1999) considers the real-time scheduling of trains on a single track railway. The author presents a heuristic which considers conflicts in the order that they appear and sequentially resolves them. For both of the trains involved in a conflict, the algorithm considers the delay necessary in order to resolve the conflict. A look ahead feature, which determines the expected arrival time of all other trains at their respective destinations as a consequence of delaying one of the trains, is also included. The train resulting in least expected delay to the rest of the trains is delayed. Computational results for between 6 and 20 trains on a 163 km stretch of track in Turkey are given.

Adenso-Diaź et al. (1999) propose a MIP model for the problem of online optimization of train rescheduling in a regional railway network. Although no explicit reference is given to the nature of the network, it would appear that the authors consider a single track railway. The MIP model presented is considered to be too complicated to solve in real-time, and the authors adopt a more practical heuristic approach. The disruption which produces the conflict is assumed to affect *services* over a certain time horizon. A train service corresponds to the movement of a train between two stations. Given the number of train services affected, the aim of the heuristic is to reassign trains to

services in such a way that total delay is minimized. This is somewhat different to the other approaches in that the authors reallocate the trains to the itineraries to minimize the delay. The heuristic method is built on a tree enumeration approach. Each level of the tree represents a service with the nodes on any level corresponding to the trains that can operate the service. Based on its priority, number of passengers, and delay, each node is assigned a cumulative score to reflect its attractiveness. The authors describe a pruning strategy to prevent complete enumeration of such a tree.

Oliveira and Smith (2000) have proposed a job shop scheduling formulation for the problem of scheduling trains along a single track railway in a disruption recovery environment. The formulation is fundamentally the same as that of Szpigel (1973) excluding any ordering constraints. It is solved using constraint programming techniques. Conflicts are resolved in chronological order, and the objective minimizes the total delay. The model also incorporates several extra constraints representing practical restrictions. In particular, it possesses the capability to force two trains to reside at the same station for a certain amount of dwell time and also allows the same train to perform multiple itineraries.

A greedy travel advance strategy based on a discrete event model is described in Dorfman and Medanic (2004) for the scheduling of trains on a single line. The proposed algorithm includes a nonlocal capacity check to avoid deadlocks and can efficiently handle time perturbations to the schedule. The authors show that the approach easily extends to networks with double track sections, can handle heterogeneous rail traffic, and performs well on three time-performance criteria. A fictitious network containing seven nodes and 36 trains over a 12-h period is considered.

Törnquist and Persson (2007) consider the rescheduling of trains under disturbances on so called N-tracked networks. This extends the single track network by allowing certain sections of track to consist of multiple parallel sections, each of which can accommodate one train. The authors present a MIP formulation of the problem and describe four different solution approaches. A real-life application from the Swedish rail network is used to test the proposed methodology. The network tested contained 169 interlinked stations while the daily timetable considered had 92 freight and 466 passenger trains. Computational experiments assume a single delayed train. Note that the work reviewed on the operational problem of train rescheduling is not exhaustive, it has been included to illustrate what can be done and a more extensive survey on this topic can be found in Törnquist (2005).

3 General railway networks

So far we have considered models and solution techniques for allocating the capacity of long single track railways where the aim is to construct a feasible meet/pass plan that typically minimizes the deviation from some ideal schedule. We now turn our attention to more complicated railway networks, particularly those in the passenger railway industry.

Caprara et al. (2002) consider the problem of timetabling trains along main *corridors* within the Italian railway network. A *corridor* typically connects two main stations, interlinks several intermediate stations, and consists of parallel stretches of

one way track in either direction. Although a corridor is, by nature, a single track network, it has been included in this section for two main reasons. Firstly, the corridor timetabling studies reviewed in this section are components of much bigger railway networks and hence connect larger more complicated stations. As a result, one must typically include additional constraints on station capacities and perform the subsequent step of train routing in developing an operational timetable. Secondly, and more importantly, from a modelling perspective the methodology for this problem is noticeably similar to the other studies reviewed in this section.

By reduction from the maximum independent set problem, Caprara et al. (2002) prove the problem they consider is *NP*-hard. The authors model the problem using an acyclic directed multigraph $G = (V, A)$, with the node set V consisting of *arrival* and *departure* nodes as well as artificial source and sink nodes. Each arrival and departure node corresponds to a possible station as well as a time instant at which the respective action can occur. There may be many nodes representing the same station depending on the number of possible time instants during which a train may arrive at (or depart from) the particular station. Each arc $a \in A$ is associated with the duration between two events, i.e. the travel time between stations or the dwell time at stations. Given both the origin station and terminal station for each of the trains, it is trivial to see that any path connecting the two defines a feasible schedule for the respective train.

Based on this network, Caprara et al. (2002) present an arc based multicommodity flow formulation with additional packing constraints. Binary decision variables govern the inclusion of any arc a in the solution. Flow constraints are required to ensure that the selected arcs for a train define one of its feasible paths. Two additional sets of binary variables are used to define constraints that prevent the simultaneous selection of conflicting arcs. The first set y_i (for all $i \in V$) indicates whether some train path visits node i, whereas the second set z_{ji} (for all $j \in T, i \in V$) indicates whether or not train j visits node i. Two arcs are said to be in conflict if they represent the same physical piece of track, and the headway time between the arrival (or departure) of the two train movements is not satisfied. For example, two arcs that define the same physical section of track and indicate that one of the two trains overtakes the other are in conflict. The objective attempts to minimize the difference from each train's ideal timetable.

One could omit the last two sets of binary variables and simply have packing constraints directly preventing the selection of incompatible arcs, i.e. each packing constraint is of the form

$$\sum_{a \in C} x_a \leq 1,$$

where C is a maximal subset of pairwise incompatible arcs. If $\mathscr{C}$ denotes the set of all such subsets, this is equivalent to saying each $C \in \mathscr{C}$ corresponds to a *maximal clique* in the undirected *conflict graph* $G_C = (A, A_C)$, where $A_C = \{\{a, b\} \in A^2 : a \neq b \text{ and in conflict}\}$.

The authors propose a Lagrangian relaxation solution approach and state that relaxing the large number of constraints in $\mathscr{C}$ would be impractical due to the overhead associated with keeping track of all Lagrangian multipliers. Caprara et al. (2002) thus

adopt the alternative approach of relaxing all constraints (defined using the additional binary variables) which prevent the simultaneous selection of conflicting arcs. The approach is tested on a number of real life instances arising in Italy. The problems considered have as many as 73 stations and 500 trains over 24 h divided into 1-min intervals. In the majority of cases all problems could be solved to within 2% of optimality. However, on other instances, particularly highly congested cases characterized by many trains, the optimality gap was shown to be as high as 20%. In a follow-up paper, Caprara et al. (2006) extend the formulation to include additional constraints representing practical restrictions. In particular, station capacities, maintenance operations, and fixed timetables for certain trains are included. Problem instances concerning up to 49 stations and 221 trains are considered.

In more recent work, Cacchiani et al. (2008) describe a column generation approach to train timetabling on a corridor. The underlying integer linear program formulation is a natural variant of those proposed by Caprara et al. (2002, 2006). An identical time–space network is used to model the problem. Cacchiani et al. (2008), however, present a path based formulation as opposed to the arc based multicommodity flow formulations described in both Caprara et al. (2002, 2006). Binary decision variables are associated with the inclusion of a path (a feasible sequence of arcs connecting the source node with the sink node). This is in comparison to the arc based multicommodity flow formulations that have binary decision variables governing the inclusion of a particular arc. The constraints of the model simply prevent the selection of conflicting paths. To solve the resulting set packing model the authors describe an exact branch-and-cut-and-price algorithm as well as two heuristic approaches. Both heuristics initially iteratively construct feasible solutions by fixing paths at the solution to the LP relaxation. Several local search strategies then improve this solution. The local search strategies attempt to improve the solution by finding the optimal path for a train given a fixed set of paths for the other trains. The authors show that the column generation subproblem entails computing an optimal path in an acyclic graph and that the addition of violated constraints does not destroy the subproblem's structure. Computational experiments considered 11 real life instances arising at the Italian railway company and contained up to 102 stations and 221 trains. The three methods are compared with the Lagrangian method of Caprara et al. (2006). Results show that the LP relaxation of the path based formulation yields a better upper bound than that of the Lagrangian approach and, more importantly, in a much shorter time. The solutions obtained using the first heuristic (which implements an additional local search strategy as well as a different path fixing routine in constructing an initial solution to that of the second) almost always obtains a better solution than that of the Lagrangian approach. The solution time is, however, significantly longer. The second heuristic obtains solutions of similar quality to that of the Lagrangian approach in a similar time frame. However, one observes more variation in the solution times. The exact approach was able to solve 3 of the 11 instances to proven optimality (for the first time) in less than 100,000 s.

Borndörfer et al. (2005) present a formulation that is very similar to that of Caprara et al. (2002). Unlike Caprara et al. (2002), however, the authors do not just consider corridors linking main stations. Borndörfer et al. (2005) propose an auction based approach to optimally allocate the track capacity of a railway network. The

authors present a framework which allows train operating companies (TOCs) to *bid* for so called *slots*. Roughly speaking, a slot refers to a sequence of track over time. In requesting a slot, a TOC must specify the monetary amount they are prepared to pay for the slot, the type of train they wish to operate, the sequence of stations the train will visit, and time-value specifications. Time-value specifications express, in monetary terms, the preference for particular arrival and departure times. As Borndörfer et al. (2005) explain, an auction approach allows one to consider multiple interdependent slots simultaneously without having to resort to the somewhat customary approach of allocating slots based on train priority.

In modelling a railway network Borndörfer et al. (2005) adopt a macroscopic view. The rail network is hence modelled as a digraph. The nodes of this digraph correspond to junctions, whereas the arcs represent the sequence of track interlinking the junctions. Each node has an assigned capacity, reflecting the number of trains that can pass and/or stop at the node at any given time. To model the temporal nature of train movements, the authors construct a time expanded model by making multiple copies of the node set. The time horizon is discretized into 1-min intervals, and a copy of the node set is constructed for every minute. An arc in this time expanded network connects two nodes if the time duration of the arc is consistent with train driving times. A slot can be identified as a directed path in the time expanded network. Depending on its kinematic capabilities, a train is able to use a subset of the arcs in the network.

With this underlying time–space network, the optimal track allocation problem (referred to as OPTRA in the paper) is formulated as an arc-based multicommodity flow problem with additional packing constraints. The formulation is equivalent to that of Caprara et al. (2002) with binary variables dictating the allocation of arcs to bids. The flow constraints enforce the requirement that the arcs selected for a particular bid define a slot in the time–space network connecting the departure and arrival station associated with the bid. The additional packing constraints restrict the selection of arcs that do not satisfy headway requirements for trains to be less than one and also enforce node capacities by restricting the number of trains arriving at any node in a particular time interval to be less than the respective capacity of the node. Such constraints are of the form as that given in Eq. (3). The objective maximizes the total monetary value of the bids.

Borndörfer et al. (2005) also present an extension of this initial formulation which includes several constraints representing practical restrictions. In particular, time windows as well as penalties on deviations from desired arrival and departure times are included. The model is also extended to include the capability of accepting combinatorial *AND* bids and *XOR* bids. An AND bid states that either a subset of individual bids must be accepted if one bid from the subset is chosen or none of them. An XOR bid, on the other hand, states that at most one bid from a subset of bids can be selected.

The authors test the extended model on a subnetwork of the long distance rail network in Germany, encompassing Hannover, Kassel and Fulda. The base case considers 946 train requests with all trains having a known origin station and known terminal station. Extra test cases are generated by allowing flexibility (in the form of time windows) on train departure times. Five extra test cases are constructed by allowing longer time windows (increments of 1 min up to a maximum of five) on all departure times for trains. Computation times reported in the paper increased from

6 s without time windows to 3 days when a 5 min time window was permitted. The second test considers an iterative combinatorial auction procedure in which a bid generator attempts to anticipate the bids of future players in the railway auction. Each round of the auction involves solving an OPTRA problem. At the end of each auction, train operating companies have the opportunity to increase any non-assigned bids, with the auction continuing until no bids change. The impact of synchronized bids through AND bids is also studied. The authors conclude that competition is much more pertinent for individual bids, with synchronized bids remaining quite stable.

In subsequent work, Borndörfer and Schlechte (2007b) view the optimal track allocation problem from a slightly different perspective. The authors recognize that models which attempt to rule out train conflicts through an enormous number of packing constraints, such as the multicommodity flow formulation proposed by Borndörfer et al. (2005), typically have weak LP relaxations, and thus make the resulting integer programs extremely difficult to solve. The model proposed by Borndörfer and Schlechte (2007b) hence attempts to rule out conflicts through the addition of extra variables rather than constraints. The formulation is shown to have a constant number of rows and be amenable to column generation techniques.

In a similar vein to Borndörfer et al. (2005), Borndörfer and Schlechte (2007b) formulate the optimal track allocation problem in terms of a digraph $G = (V, A)$. Every node $v \in V$ represents the arrival (or the departure) of a train at (from) one of the stations at a certain time. Once again the arc set A defines the track interlinking stations. Given the temporal nature of the network, several arcs may correspond to the same physical section of track; however, the time interval for which it is said to be *blocked* can be different. Paths through the digraph correspond to train routes on the railway network (again only at the macroscopic level), and two routes are said to be in conflict if there is an arc conflict, i.e. the respective blocking times for the arcs overlap, and the arcs represent the same physical piece of track. Furthermore, Borndörfer and Schlechte (2007b) introduce the concept of an arc *configuration*. An arc configuration refers to a subset of arcs representing the same physical piece of track that is conflict free. Given a set of train requests, one attempts to allocate the track capacity of the network such that each train receives at most one non-conflicting path.

Borndörfer and Schlechte (2007b) define binary variables to govern the selection of train paths as well as binary variables that dictate which arc configuration each segment of track is assigned. Packing constraints enforce the requirement that at most one path can be chosen for any train as well as the requirement that at most one arc configuration can be selected for any segment of track. Linking constraints then relate train paths with each track configuration to ensure a conflict free solution. The authors present a column generation framework in which the pricing problems for both train paths and configuration variables entails computing longest paths in appropriate acyclic networks. Computational results for three problems containing 146, 250, and 570 trains on a subnetwork of the German long distance rail network are provided. Empirical evidence suggests these problems can be solved close to optimality with this approach.

In a comparative study, Borndörfer and Schlechte (2007a) consider the track configuration based formulation of Borndörfer and Schlechte (2007b) and packing based formulations in more detail. Packing based formulations are those that do not include

arc configuration variables and typically prevent train conflicts through the inclusion of packing constraints.

Given the same digraph model $G = (V, A)$ as that proposed in Borndörfer and Schlechte (2007b) the authors describe two possible packing formulations. The first one is an arc based multicommodity flow formulation (APP), similar to those proposed by Borndörfer et al. (2005) and Caprara et al. (2002), whereas the second is a path based set packing formulation (PPP) similar to that of Brännlund et al. (1998). The majority of the packing constraints for each formulation rule out conflicting movements on railway infrastructure. As we have established earlier, such constraints are clique inequalities and define the set of all maximal cliques in an undirected conflict graph $G_C = (A, A_C)$. These two formulations are compared with an arc based track configuration formulation (ACP) and a path based track configuration formulation (PCP). The authors provide a proof showing that the LP relaxation of each of the formulations yields the same objective value.

Computational experiments reported in the paper focus on a portion of the German railway network consisting of 37 stations connected via 120 sections of track and compare the performance of the APP, ACP and PCP methods. The APP method used in the comparison is a somewhat weakened version (which the authors refer to as APP′) as the packing constraints only prevent arc conflicts between pairs of arcs as opposed to the much stronger aforementioned clique inequalities. The authors elect not to test the PPP method as they claim the majority of work in this field is based on the APP approach. Three test scenarios containing 146, 285, and 570 trains consisting of 6 different train types form the basis of 48 test instances. The additional test instances are constructed by incorporating an increasing amount of flexibility in the arrival and departure times for the trains. The primary objective of the problems was to schedule as many of the trains as possible. The authors conclude that the APP′ approach produced noticeably weaker LP bounds than both the ACP and PCP methods. The PCP method is considered the method with the most potential as it is the only method that is able to solve some of the larger problem instances.

3.1 The periodic event scheduling problem

All of the work reviewed so far considers allocating the track capacity of a railway network over time when a proposed timetable is provided. Furthermore, the preceding discussion has largely ignored the notion of periodicity of timetables. All of the reviewed work assumes the timetable repeats on a daily basis. In what follows we discuss techniques that have been proposed to construct timetables with much shorter periods (i.e. 30 or 60 min). Where relevant we discuss how these methods ensure train movements within the network are conflict free.

The Periodic Event Scheduling Problem, or PESP as it is commonly referred to, has been the model of choice for the construction of periodic timetables in the railway industry. First introduced by Serafini and Ukovich (1989), the PESP entails scheduling a set of periodic events E given a set of periodic time window constraints C. By definition, a periodic event is one that repeats itself at recurring intervals of time λ. For instance, if an event $i \in E$ is scheduled to occur at time $\pi_i \in [0, \lambda)$, then this

event will also occur at all times $\pi_i + k\lambda$, $k \in \mathbb{Z}$. All constraints of the PESP model relate pairs of events and restrict the duration between the occurrence of each event to be within some specified time interval. Hence, any constraint $c \in C$ is of the form

$$L_c \leq \pi_j - \pi_i + z_c\lambda \leq U_c, \tag{1}$$

where L_c and U_c define the lower and upper limits on the time interval permitted between the occurrence of events $i, j \in E$, and z_c is an integer decision variable needed to model the periodic nature of the problem. The basic version of PESP is purely a feasibility problem. That is, either one finds vectors $\boldsymbol{\pi} \in [0, T)^{|E|}$ and $\boldsymbol{z} \in \mathbb{Z}^{|C|}$ such that all periodic constraints are satisfied or shows that the problem is infeasible. Any PESP instance may be viewed as a graph theoretic problem. A so called *constraint graph*, also known as an event-activity network, is modelled as a digraph $D = (E, C, L, U)$ in which there is a node $i \in E$ for each event and an arc $(i, j) \in C$ for every constraint of the form (1).

In the context of train timetabling each event is associated with the arrival or departure of a train at a particular station, with the constraints of the model suitably enforcing restrictions on the time intervals between (arrival, arrival), (arrival, departure), (departure, arrival), and (departure, departure) event pairs. We refer the reader to Peeters 2003, Chapter 3 for a detailed discussion of typical constraints appearing in PESP formulations. In particular, examples of constraints enforcing certain train connections, trip times between stations, and those that enforce a required headway are provided. Liebchen and Möhring (2004) describe how the constraints of the PESP formulation can be used to model more sophisticated features of railway timetabling such as train line bundling.

PESP models adopt a macroscopic view of the railway network. Furthermore, they assume that the section of track a train will use in travelling between stations has been determined a priori; they do not possess the capability to consider alternative routes for trains. A PESP model ensures a conflict free timetable between nodes of the network through headway constraints on pairs of events. The ability to enforce the required headway is dependent on knowledge of the respective trip times for each train type between stations. However, a variable trip time model based on PESP has been proposed by Kroon and Peeters (2003).

A variety of solution techniques for the PESP model have been proposed in the literature. Serafini and Ukovich (1989), in their pioneering work, present a branch-and-bound based approach which essentially constructs a timetable by sequentially satisfying the constraints. Odijk (1996) uses a constraint generation approach to construct a set of feasible timetables in an attempt to evaluate the infrastructural capacity of a station and its surrounding area. Cutting planes for the PESP model are also presented in Nachtigall (1996), while Nachtigall and Voget (1996) describe a genetic algorithm based methodology. In addition to this, a constraint programming approach that also incorporates a local search heuristic to improve the quality of a timetable has been proposed by Schrijver and Steenbeek (1994). This particular algorithm, known has CADANS, is one of the core modules of the decision support system DONS (Designer of Network Schedules) currently used by the Dutch railway company to develop operational timetables (see Hooghiemstra et al. 1999).

An alternative formulation of the PESP model, known as the *Cycle Periodicity Formulation* (CPF), has also been widely studied (see, e.g. Nachtigall 1998; Lindner 2000; Liebchen and Peeters 2002; Liebchen 2003, 2006; Peeters 2003). Unlike the classical PESP formulation, which has a decision variable for each event time, the CPF defines *periodic tension* decision variables x_c for all $c \in C$. The periodic tension x_c can be interpreted as the time difference between events i and j of constraint c. That is,

$$x_c = \pi_j - \pi_i + z_c\lambda.$$

To ensure that the chosen tension times do produce periodic event times, the CPF formulation requires that for any cycle in D the sum of the tension values along the arcs is an integer multiple of λ. This alternative formulation yields a better LP relaxation than the classical PESP approach and appears to be easier to solve. Liebchen et al. (2008) provides a detailed empirical study of five different solution approaches for the PESP. The performance of constraint programming, genetic algorithms, and simulated annealing each applied to the classical PESP formulation is compared with the performance of Cplex 8.1 on two MIP models. Furthermore, a different approach for constructing periodic timetables is presented in Vansteenwegen and Van Oudheusden (2006). The proposed methodology combines an LP model with simulation. Note also that, although we have included a discussion on the PESP model in Sect. 3 due to its prevalence in constructing timetables in the passenger railway industry, it is equally applicable when timetabling trains on single track networks.

One can see from the preceding discussion that the routing of trains plays an indispensable role in determining an operational timetable for a railway company. Although the methods described in Sect. 3 ensure mutually disjoint train movements between nodes of a railway network, the aggregated topology considered means that the detailed assignment of trains to particular routes through the disaggregated infrastructure is still undetermined. Indeed, in the case of the PESP approach, the macroscopic solution may be infeasible with respect to particular junctions. The work of Borndörfer et al. (2005), Borndörfer and Schlechte (2007a,b), and Caprara et al. (2006) ensures that the provisional macroscopic timetable is feasible with respect to junctions through the inclusion of capacity constraints. However, the subsequent step of route assignment (including platform allocation within stations) must still be performed.

4 Junction train routing

We begin this section by introducing the problem of routing trains through railway junctions in more detail. In particular, we introduce some essential terminology and discuss variants of the problem. This description is from a German railway perspective; however, parallels can be drawn with most European railway companies.

Figure 2 illustrates the main components of a railway network. In Germany, track is classified as either *tracks of the open line* or *station track*. The former define the arcs of the aforementioned aggregated topology, while the latter refer to track within the station boundary and includes platforms. This boundary is indicated by *home signals*.

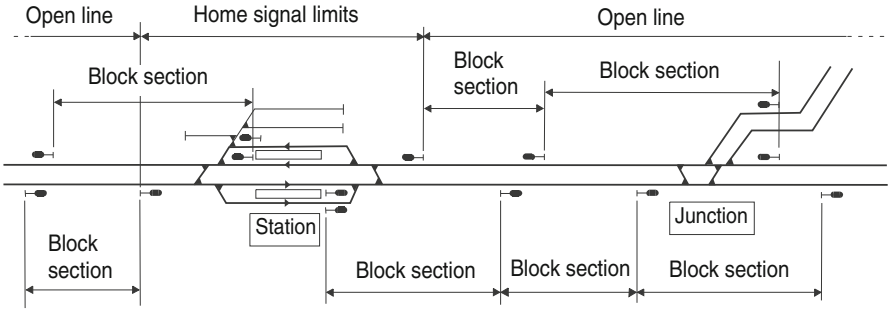

Fig. 2 Home signal limits and block sections (from Pachl 2004)

In a similar way, the entry to non-station junctions is controlled by *block sections*. Depending on their complexity, a junction may contain just station tracks, just tracks of the open line, or both. For instance, although the network in Fig. 2 contains both a station and a junction on the open line, one could refer to this entire network as a junction.

Train movements are controlled by a signalling system. This ensures, among other things, that trains do not get too close to one another by enforcing the minimum required headway between trains. To implement a signalling system, the tracks of the open line are divided into several block sections. Such blocks are delimited by block signals and ensure that there is at most one train on any block at any given time. Note that the track leading from a home signal to a platform is not referred to as a block section. However, trains use it in much the same way. We will refer to this as an interlocking section. Both block and interlocking sections may contain a number of smaller track sections. Interlocking (block) sections may have track in common with other interlocking (block) sections. For instance, they may share a switch or an intersection. In general, a switch is a track device that allows a train to change railway lines, whereas an intersection indicates that one set of railway lines crosses another set. In some cases, however, an intersection can also be used to change railway lines. Track sections denote the boundary of the infrastructure that is common to multiple blocks.

On entering a block section, the respective entrance times of all track sections within the block are synchronized with that of the first section of the block. For this reason trains are said to claim track sections. In other words, trains will request the use of several track sections before actually occupying them. On traversing a sequence of claimed track sections, trains will successively release each one once the tail of the train has exited, and a small amount of buffer time has elapsed. Released track sections may then be claimed by other trains. Conflicts occur if trains simultaneously attempt to claim the same track sections. For a more detailed explanation of the operational processes within the German rail network we refer the reader to Pachl (2004).

The perimeter of a junction consists of a number of *entering* and *leaving points*. These indicate the points at which a train may enter and/or leave the junction. They also identify the scope of infrastructure concerned. A *route* defines a sequence of track sections connecting a possible entering point with a possible leaving point. If the junction

coincides with a station, this may involve the train stopping at an available platform. Depending on the number of switches within a junction, there may be many possible routes between pairs of entering and leaving points. A *path*, on the other hand, refers to the traversal of a given route over time. For any given route, there may be a number of alternative paths representing different acceleration and deceleration strategies for the traversal of the underlying route. The problem of routing trains through railway junctions (henceforth referred to as the train routing problem) entails assigning each of the trains in a proposed timetable a conflict-free path through the junction while adhering to several operational constraints.

The importance of this problem to railway planning is emphasized by the fact that one can identify a variant of the problem at each of the strategic, tactical, and operational planning levels. On a strategic level it answers questions concerning the capacity of a junction, while on a tactical level it is required to validate the feasibility of a proposed timetable. It also appears at the operational level where one is forced to reroute trains when a disruption has made the planned routings infeasible. In what follows, we group contributions in this field by the model proposed and where relevant discuss what variant of the problem is being dealt with.

4.1 Conflict graph approaches

The most common approach when modelling the train routing problem is to use conflict graph methodology. Since this forms the basis of the majority of work in this field we begin our review here. Within this methodology, two quite different approaches have been used to model variants of the train routing problem. These are known as the node packing problem and the graph colouring problem, respectively. The former is typically used when routing trains through railway junctions, while the latter is used to model the train platforming problem. We now discuss each in turn.

The structure of the node packing problem means it is perhaps the most intuitive formulation for the train routing problem. The aim is to find a conflict free set of paths for a set of trains and by representing each train path as a node and connecting any two nodes for which the corresponding train paths are in conflict by an edge, the problem can be explicitly modelled as a node packing problem. Figure 3 illustrates an example conflict graph with ten train paths. In this example, the edge between v_1 and v_8 indicates that the corresponding train paths cannot be assigned simultaneously.

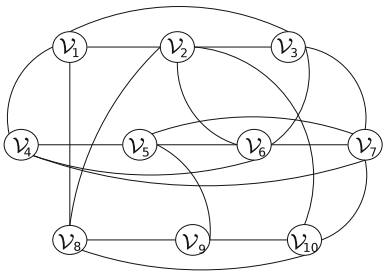

Fig. 3 An example conflict graph

4.1.1 Node packing formulation

A generic integer programming formulation of the node packing problem in the context of the train routing problem is given next. Assume that there are n variables in total, that $\boldsymbol{x}$ denotes the vector of binary decision variables governing the selection of the particular train paths, and that $\boldsymbol{\rho}$ denotes a vector of weights. Each element of $\boldsymbol{\rho}$ reflects the benefit received in assigning the corresponding train path. The objective of the model (given by (2) below) is to find a node packing of maximum weight. For example, in Fig. 3 the set $\{v_2, v_4, v_9\}$ gives a feasible solution to the routing problem.

$$\text{Maximize:} \quad \boldsymbol{\rho}^T \boldsymbol{x} \tag{2}$$

Subject to:

$$x_i + x_j \leq 1 \qquad \text{for conflicting train paths } i \text{ and } j, \tag{3}$$

$$\boldsymbol{x} \in \{0, 1\}^n. \tag{4}$$

Constraint set (3) enforces the requirement that one can pick at most one of any two train paths in conflict, while constraints (4) state the binary restriction on the decision variables. In what is perhaps the first attempt to model and solve the train routing problem via Operations Research techniques, Zwaneveld et al. (1996), in the development of the decision support system DONS for the Dutch railway company NS, propose a node packing formulation. DONS was developed to assist strategic planners evaluate infrastructural capacity requirements given different possible scenarios of expected future demand for rail transportation. It consists of the two separate modules CADANS and STATIONS, the first of which was introduced in Sect. 3. Zwaneveld et al. (1996) develop the STATIONS module. This receives as input a provisional cyclic timetable from CADANS (at the macroscopic level) and then determines whether or not this is feasible with respect to each of the stations in the network. A node packing formulation is the underlying model of STATIONS. Note that previous literature terms the traversal of a sequence of track a "route" where we use "path". In order to keep descriptions consistent, we will use our terminology when describing previous work.

Zwaneveld et al. (1996) argue that in order to handle the operational requirements imposed by NS, one should adopt a modelling approach in which the paths are dealt with explicitly as opposed to dealing directly with the track sections of a station. This particular line of reasoning naturally gives rise to a node packing formulation. In order to determine whether any two paths are in conflict, one must calculate the exact claim and release times of the track sections comprising the two paths. Although, as the authors explain, the node packing formulation can be used to incorporate more general pairwise incompatibilities between paths. Indeed, without any fundamental change to the formulation, one may prevent the simultaneous selection of any two paths, regardless of the reason, simply through the inclusion of additional edges in the conflict graph. This flexibility is particularly beneficial if one wishes to enforce train connections at adjacent platforms or the coupling/decoupling of two trains.

The particular node packing formulation Zwaneveld et al. (1996) consider is a unicost instance. In other words, all train paths are uniformly assigned a benefit of one in the objective function. The authors do permit time deviations on the provisional

arrival and departure times for the trains. For instance, deviations of 1 min on either side of a train's provisional arrival and departure time are allowed. Such deviations are considered in the form of additional variables and are merely duplicates of their undeviated counterparts shifted forward (or back) in time by 1 min. The inclusion of these extra variables is an attempt to increase the likelihood of finding a feasible solution. The authors describe two types of conflict. Firstly, paths for the same train are conflicting as at most one path for any train is required in a solution, and, secondly, any pair of paths for two different trains that are pairwise incompatible are conflicting. The latter form of conflict is most likely to arise from the two train paths simultaneously claiming some part of the station infrastructure.

The authors do acknowledge that there are several limitations in formulating the train routing problem this way. From a computational point of view, the resulting formulation is typically very large. Computational experiments reported in Zwaneveld et al. (1996) suggest that instances involving up to 27 trains can have in excess of 3,000 nodes and as many as 1.5 million packing constraints. Furthermore, node packing formulations have a notoriously weak LP relaxation. It is trivial to see that the authors' formulation will always fractionally achieve an objective of routing all trains even if it is not possible to route all trains. Each constraint (3) only restricts pairs of paths for different trains. Hence, if each train has at least two paths, selecting any two at value $\frac{1}{2}$ will always be feasible.

To overcome both the formulation's intractable nature as well as its weak LP relaxation Zwaneveld et al. (1996) show that the model can be strengthened through the addition of valid inequalities and reduced in size through the removal of dominated train paths. A path for a particular train is considered dominated if the train has an alternative path that leaves at least the same routing options open for all other trains. Valid inequalities for the node packing problem are the well known clique inequalities. The search for cliques in a conflict graph is exponential in the problem size, and the authors suggest adding only a subset of the clique inequalities prior to solving the problem. For example, one may add the following constraint for any train $i \in T$:

$$\sum_{j \in P_i} x_j \leq 1, \tag{5}$$

where $P_i \subseteq V$, and each node $j \in P_i$ defines a possible path for train i. All nodes corresponding to paths for the same train define a complete subgraph in G, and hence only one can be selected. In addition to these constraints, the authors also suggest looking for cliques in the subgraph of the conflict graph defined by the nodes corresponding to two trains. The addition of the clique inequalities results in a reduction in problem size as such constraints naturally dominate a number of the much weaker pairwise constraints. Computational experiments conducted suggest that these reduction techniques had a dramatic effect on problem size.

Zwaneveld et al. (1996) present a branch-and-cut approach to solve the reduced node packing formulation. The cut part of the procedure attempts to identify violated clique inequalities at each node of the branch-and-bound tree. Two heuristics for the train routing problem are also described. The first randomly selects a train and assigns it a path, where the path selected attempts to maximize the routing possibilities left for

the remaining trains. The second also sequentially assigns paths to trains. However, trains with the least number of routing possibilities are given priority. As the solutions obtained using these heuristics always coincided with the truncated value of the objective function, the branch-and-cut routine was not needed. It was concluded that the most time consuming part of the solution approach was in the generation of the sets of compatible train paths for each pair of trains (required for the conflict graph generation). Zwaneveld et al. (2001), in a follow-up paper, however, stated that this approach was unable to solve the train routing problem for two of the larger problems in The Netherlands.

Zwaneveld et al. (2001) can be considered an extension of Zwaneveld et al. (1996). The authors adopt the same modelling approach; however, they attempt to improve the performance of the solution method. By making the reasonable assumption that trains will prefer certain paths to others, the problem is formulated as a weighted node packing problem and is hence more aligned at the tactical level than Zwaneveld et al. (1996). The authors also elect to partition a train path into three components: namely, *inbound path*, *platform path*, and *outbound path*. An inbound path defines a sequence of track sections from an entering point to a platform, whereas an outbound path defines a sequence of track sections connecting a platform with a leaving point. The platform path refers to the track sections corresponding to a platform. As Zwaneveld et al. (1996) explain, such a partitioning allows the shunting of train units to be considered.

Unlike Zwaneveld et al. (1996), the decision variables of the problem do not pertain to train paths, but rather the partitions of the paths. To be assigned a path through the junction a train can receive at most one inbound path, one outbound path, and one platform path. Constraints of the form (3) can still enforce the possible connections between path partitions. Zwaneveld et al. (1997) show that such a partitioning can lead to a reduction in the number of variables.

The calculation of the preference coefficient for a particular inbound/outbound or platform path is discussed in detail in Zwaneveld (1997). Essentially it consists of three elements. The first is the train's preference for the inbound/outbound or platform path. This is typically a reflection of the number of switches in the preferred position as well as the speed limits on the track sections. The second reflects the preference for a time deviation from the train's provisional arrival and departure times, whereas the third reflects the importance of the inbound/outbound or platform path. Zwaneveld (1997) shows how by selecting suitable integer values for each element, a hierarchical list of objectives can be considered. The first aims to route the maximum number of trains, whereas the second looks at maximizing the preferences associated with the assigned paths. The third objective seeks to minimize the total number of shunting movements.

Zwaneveld et al. (2001) include more sophisticated preprocessing and reduction techniques in the branch-and-cut solution approach. In particular, the techniques of node-dominance, set-dominance and iterating set-dominance are developed for the weighted node packing problem. All approaches attempt to show that a certain node can be replaced by another node in all feasible solutions without a reduction in the objective function value and hence reduce the size of the problem instance. We refer the reader to Zwaneveld (1997) for a more detailed discussion of each preprocessing technique.

The authors conclude that the refined solution approach is sufficient for solving the train routing problem arising at any Dutch railway station. The implementation of the improved preprocessing techniques were very successful in reducing problem size. The largest instance considered had 79 trains. The initial 9,328 variables were reduced to just 706, whereas the a priori addition of clique inequalities reduced the number of constraints from 11,629 to 1,231. The authors also showed that in determining whether two train paths were in conflict one only needed to consider a subset of track sections that they termed "relevant track sections". By considering only relevant track sections the set of compatible train paths for each pair of trains could be generated faster.

A node packing formulation is also the basis of the model in Delorme (2003). The author's PhD thesis considers the problem of evaluating the capacity of railway junctions. Delorme (2003) presents a formulation that is essentially equivalent to that of Zwaneveld et al. (2001). An entire train path through a junction is assumed to consist of a number of smaller subpaths through so called *partitions* of the junction, where each partition consists of a number of *zones* (synonymous with track section). Constraints of the form (3) are used to enforce a range of incompatibilities between pairs of subpaths. The binary variables of the model govern the inclusion of particular subpaths in the solution. All train paths are predetermined, and trains have a set of possible entering times. Hence, associated with each binary variable is a time stamp reflecting the time the train enters the respective partition.

In contrast to Zwaneveld et al. (1996), the author defines multiple objective functions, which are considered lexicographically in the following order. The problem is primarily concerned with capacity assessment, and hence determining the maximum number of trains that can be routed through the junction is given the highest priority. The author also suggests that one should consider introducing additional so called *saturating* trains as well as maximizing the preferences associated with the assigned train paths. The last two objectives are considered of equal importance. In addition to this, Delorme (2003) introduces the notion of stability for a train routing. The stability measure is calculated as the maximum propagated delay resulting from some initial delay. This is solved via several shortest path problems, where the network considered reflects the direct or indirect impact of a late train on all other trains. The nodes of the network correspond to the trains, whereas the edges represent the immediate delay one train causes another. For example, if the late arrival of train t causes a delay of 45 s to train t', then there is an edge between the respective nodes with weight 45.

Delorme (2003) develops two solution methods for the proposed node packing formulation. The first is noticeably similar to that of Zwaneveld et al. (2001) and involves extensive preprocessing through variable dominance as well as the a priori addition of several classes of clique inequalities. The reduced formulation is then solved using Cplex 8.0. The author tests this approach on both railway specific instances as well as several randomly generated data sets. The random instances had up to 2,000 variables and 10,000 constraints. The railway instances, on the other hand, had up to 3,720 variables (from 200 trains) and 482,887 constraints. The application of the preprocessing techniques appears to be very successful in reducing problem size, particularly for the railway specific instances for which it is reported that on average 16% of the variables and 90% of the constraints are removed. However, this exact solution approach appears quite inefficient. The gap between the best known integer solution and the LP bound

can be as bad as 400% for the random instances and up to 25% for railway instances after 50,000 s of computation time.

A second, more promising, approach using the GRASP (Greedy Randomized Adaptive Search Procedure) metaheuristic is presented. GRASP methods typically consist of two phases. The first is the implementation of a greedy heuristic to yield an initial feasible solution. The second phase then attempts to improve the solution through a local search heuristic. The neighbourhood often considered for the local search heuristic is based on the *k-p* exchange. The *k-p* exchange neighbourhood for a solution $\boldsymbol{x}$ is the set of solutions obtained from $\boldsymbol{x}$ by changing the value of k of the variables from 1 to 0, and changing p variables form 0 to 1. One usually considers the 0–1, 1–1, 1–2, or 2–1 exchange due to the combinatorial nature of the exchange possibilities. Empirical evidence shows that the GRASP metaheuristic is able to solve the problems close to optimality in a much more reasonable time frame. Results reported show that the GRASP metaheuristic finds solutions within 3% of optimality and requires, on average, less than 800 s of computing time. The particular GRASP method implemented by the author includes both reactive GRASP and path relinking, among other features. We omit an explanation of the details here and refer the interested reader to Delorme et al. (2004). This paper not only provides an overview of GRASP, but also compares various types of GRASP on a number of routing problems arising at the Pierrefitte-Gonesse junction north of Paris. The algorithmic work of Delorme (2003) forms the basis of the RECIFE decision support system used by the French national railway company SNCF to evaluate railway infrastructure capacity.

Gandibleux et al. (2004, 2005) propose Ant Colony Optimization (ACO) metaheuristics as an alternative to GRASP. One major difference between ACO and GRASP is that at each iteration ACO will produce many feasible solutions, whereas GRASP will produce just one. With the ACO approaches, initially one defines a pheromone matrix ϕ with ϕ_i reflecting the probability that variable x_i is in a good solution. At each iteration, starting from the trivial solution $x_i = 0$ for all variables, each ant constructs a feasible solution by sequentially adding variables. The selection of which variable to add is usually random or determined by the highest pheromone value. Typically a local search heuristic is applied to each feasible solution at each iteration, and the pheromone matrix is updated so that ants *remember* what constitutes a good solution.

The algorithm described in Gandibleux et al. (2005) is an evolution of that described in Gandibleux et al. (2004). In particular, new data structures, a new stopping criterion, and an improved local search heuristic are implemented. The ACO algorithms were tested on a number of railway instances arising at the Pierrefitte-Gonesse railway junction. Results in Gandibleux et al. (2004) suggest that ACO is competitive with GRASP procedures despite the fact that it is a much simpler algorithm. The authors identified the numerical instability as a possible drawback of the method, since ACO could return solutions with a wide range of quality for multiple runs on the same test case. Results reported in Gandibleux et al. (2005) suggest that the improved ACO algorithm can deal with larger problem instances and find better quality solutions quicker than the earlier version.

The problem of finding a robust routing for the train routing problem is the topic of Caimi et al. (2005). The node packing formulation used is identical to that of Zwaneveld et al. (1996). The authors state, however, that the station of Bern in Switzerland, which

formed the test case of the study, was too large to allow for the successful application of the branch-and-cut algorithm and propose a fixed-point iteration heuristic to yield an initial feasible solution. The authors adopted the fixed-point iteration heuristic for its ability to exploit the clique structure of the underlying conflict graph.

The fixed-point iteration heuristic involves first assigning each train path a probability, where the probability reflects the likelihood of finding a feasible routing if the train path is selected. In a second step, the probabilities are iteratively updated until they have converged (i.e. successive updates yield the same probabilities and are termed *fixed*). Updated probabilities are obtained by first calculating the probability of selecting no conflicting train path for another train, and then adjusting this by the probability of not picking an alternative path for the same train. Attractive fixed points are those that correspond to a conflict free routing. It is, however, also possible for the algorithm to terminate at a fixed point that contains conflicts.

If a conflict free solution is obtained using the fixed point heuristic, the authors attempt to improve its robustness through the implementation of a local search heuristic. The improvement heuristic attempts to maximize the total weighted sum of *time slots*. A time slot is defined to be the time interval during which a train may arrive and find its designated path open. Time slots are weighted to reflect the importance of increasing the length of shorter time slots. The local search heuristic looks to reroute one or two trains in an attempt to improve the objective value.

By using random initial probabilities, this fixed-point heuristic allows a variety of solutions to be found. Results reported suggest that this approach yields feasible solutions to the train routing problem quickly. However, since unattractive fixed points (i.e. the heuristic terminates with an infeasible solution) exist, there were occasions when the heuristic needed to be restarted (as many as 50 times in some cases). Increasing the length of the time slots proved more time consuming; however, evidence suggests one can improve the length of the time slots dramatically in some cases. Computational tests involved between 11 and 19 trains with up to 6,800 paths.

Herrman (2006) uses the same fixed point heuristic to obtain an initial feasible solution and considers both deterministic as well as probabilistic robustness measures for train timetables. The time slot concept can be considered a deterministic robustness measure as it is independent of any probability distribution. Herrman (2006) discusses probability distributions of delays for train timetables and introduces four different robustness measures. The author's work forms a component of a larger model developed by Burkolter (2005) to generate dense timetables within station areas. Burkolter (2005) considers an aggregated topology of the station infrastructure and models train movement with Petri Nets. Simulated annealing is used to determine a timetable with minimum cycle time. This timetable is then tested for feasibility with the fixed point heuristic of Herrman (2006) by considering the detailed infrastructure of the station.

Caprara et al. (2007a) study a version of the problem which the authors term *The Train Platforming Problem*. Recall that the train platforming problem entails assigning the arriving trains at a station to an available platform. Note that this problem is implicitly dealt with in the studies of Zwaneveld et al. (1996, 2001) and Caimi et al. (2005) since with these methods one assigns trains to a complete path through a station, the platform allocated being one component of this. The train platforming problem has been studied by De Luca Cardillo and Mione (1998), Billionnet (2003),

Carey and Carville (2003), and Cornelsen and Di Stefano (2007), and each will be discussed in detail later. As has been mentioned in Sect. 1, the difference in terminology arises from the characteristics of the junctions that have been considered in the respective studies. If the nature of the station is such that each platform can be reached via a number of alternative paths, the routing of the trains will play a more influential role. However, in the studies of De Luca Cardillo and Mione (1998), Billionnet (2003), Carey and Carville (2003), and Cornelsen and Di Stefano (2007), the allocation of the platform uniquely determine a train's path into and out of the station, meaning the emphasis was less on the routing and more on the platform assignment.

We believe the study by Caprara et al. (2007a) is more closely related to the train routing problem as the selection of a platform may not uniquely determine a train's path into and out of the station. Each train to be assigned a platform is assumed to have a number of possible *patterns*, where each pattern consists of an inbound path, an outbound path, a platform, and both an arrival and departure time at the platform. In a similar way to Zwaneveld et al. (2001), deviations from the arrival and departure times are considered in the form of alternative patterns. The authors make the rather restrictive assumption that a train takes a constant amount of time to traverse any inbound path, regardless of the platform it visits.

Caprara et al. (2007a) present an integer programming formulation based on a *pattern incompatibility graph*, the nodes of which correspond to patterns, whereas the edges connect any two patterns that are pairwise incompatible (i.e. they occupy the same platform during an overlapping time interval). The constraints of the model enforce the requirement that one pattern must be selected for each train, and that conflicting patterns cannot be chosen. To achieve the latter, two sets of constraints are identified. The first set defines cliques in the pattern incompatibility graph associated with sets of patterns that use the same platform, whereas the second set defines all the cliques in the pattern incompatibility graph.

Given the exponential nature of the defined constraint system, the authors present alternative versions of the clique inequalities. Each clique inequality relating to platform occupation corresponds to a set of time intervals, any pair of which is overlapping. The authors hence elect to replace the clique inequalities with an enumerated set of time interval constraints for each platform. Caprara et al. (2007a) also suggest considering only those clique inequalities defining incompatible patterns for pairs of trains, rather than determining all cliques in the pattern incompatibility graph.

The objective of the model is to minimize the sum of costs associated with platform use, pattern allocation, and a quadratic function indicating the costs incurred from the pairwise assignment of patterns to any two trains. Through the introduction of additional variables and linear constraints, the authors show how the quadratic term in the objective function can be linearized. Due to the large number of constraints and variables of the model, the authors propose a branch-and-cut-and-price solution approach. In particular, the initial LP contains only a subset of the pattern variables. Separation routines are defined for the clique inequalities (corresponding to incompatible paths for pairs of trains) as well as the additional constraints that are required to linearize the objective. The authors consider several case studies from the Italian railway involving up to 237 trains over a 24-h period. Although some instances show a significant gap

between the best known integer solution and the LP relaxation, the approach yielded significantly better results than the heuristic currently in use at the railway company.

4.1.2 Graph colouring approaches

De Luca Cardillo and Mione (1998), Billionnet (2003), and Cornelsen and Di Stefano (2007) propose graph colouring approaches for the train platforming problem. The respective studies consider stations that have limited routing options to and from platforms. They are, however, particularly relevant as one must still ensure that the allocation of platforms to trains is conflict free. A conflict graph also forms the basis of graph colouring approaches; however, unlike the node packing formulations discussed in Sect. 4.1, the aim is not to select nodes of the conflict graph, but rather to assign each a colour such that no two adjacent nodes have the same colour. For this reason, graph colouring methods are not suited to complex junctions where trains have a number of alternative paths to and from platforms. For instance, there is no possibility of including path preference for a train if it has more than one path through a junction that uses the same platform. A formal definition of the graph colouring problem is given below.

Definition 1 (*Graph Colouring Problem*) Let $G = (V, E)$ be an undirected conflict graph, and let $C = \{1, 2, \ldots, k\}$ be a set of colours. A colouring of G with at most k colours is a function $f : V \rightarrow C$, where $f(v_1) \neq f(v_2)$ for all edges $e = \{v_1, v_2\} \in E$.

Several extensions of the basic definition above also exist. In particular, a k L-list colouring of G is a colouring γ of G with k colours such that $\gamma(v) \in L(v)$ for all $v \in V$, where $L(v) \subseteq C$. In other words, each node $v \in G$ has a list of colours $L(v)$ that it may receive. De Luca Cardillo and Mione (1998) subsequently introduced the k L-list τ colouring problem. A k L-list τ colouring is a k L-list colouring with a set of extra restrictions τ. Each element of τ specifies an incompatible assignment of two colours to adjacent vertices. That is, each element of τ is of the form $\{v_1, v_2, i, j\}$, where $\{v_1, v_2\} \in E$, $i \in L(v_1)$, and $j \in L(v_2)$. It specifies that one cannot assign $L(v_1) = i$ and $L(v_2) = j$ simultaneously.

As De Luca Cardillo and Mione (1998) explain, the train platforming problem is a natural application of a k L-list τ graph colouring problem. One can represent every train in the timetable as a node in the conflict graph, and then connect any two nodes for which the respective trains cannot be assigned the same platform. Each platform represents a possible colour. The set τ is used to identify the incompatibilities between different platforms for two trains due to conflicting paths for the trains to each of the platforms. De Luca Cardillo and Mione (1998) propose a heuristic algorithm with backtracking to colour the resulting conflict graph.Computational experiments for relatively simplified problems involving up to 242 trains and 16 platforms over a 24-h period are reported with the authors acknowledging that as the problem size grew their algorithm was unable to find solutions in reasonable time. Problem reduction techniques were required to decrease the computation time.

The work of Billionnet (2003) builds on that of De Luca Cardillo and Mione (1998) by showing that the k L-list τ colouring problem has two integer programming formulations. Both formulations are based on a node packing formulation. For instance, one

can define the binary variables x_{ir}, equal to 1 if vertex v_r $(r = 1, \ldots, n)$ is assigned colour $i \in L(V_r)$, and formulate the k L-list τ graph colouring problem as follows:

$$\text{Find:} \quad x_{ir} \tag{6}$$

Subject to:

$$\sum_{i \in L(v_r)} x_{ir} = 1, \qquad r = 1, \ldots, n, \tag{7}$$

$$x_{ir} + x_{is} \leq 1, \qquad r < s,\ \{v_r, v_s\} \in E,\ i \in L(v_r) \cap L(v_s), \tag{8}$$

$$x_{ir} + x_{js} \leq 1, \qquad \{v_r, v_s, i, j\} \in \tau, \tag{9}$$

$$x_{ir} \in \{0, 1\}, \qquad r = 1, \ldots, n,\ i \in L(v_r). \tag{10}$$

The model has no explicit objective function since it is purely a feasibility problem. Constraint (7) ensures each node receives one of its possible colours, while constraint set (8) enforces the requirement that adjacent vertices must receive different colours. Constraints (9) prevents the incompatible assignments of different colours to adjacent vertices, as specified by the τ set. The final set of constraints enforce the binary restrictions on the decision variables.

Viewed in a train platforming context, the binary variables govern the selection of a particular platform for a given train. With such decision variables, any objective function would obviously seek to maximize (minimize) the use of a certain platform within the station. The constraints of the model are also easy to interpret: each train must be assigned one of its available platforms (7), no two trains can be assigned the same platform at the same time (8), and two trains cannot be assigned different platforms if there respective paths to the platforms conflict (9).

Billionnet (2003) compares model (6)–(10) with a strengthened version. The strengthened formulation is obtained by observing that if vertex v_r receives platform i, then an adjacent vertex v_s cannot receive colour i, or colour j if $\{v_r, v_s, i, s\} \in \tau$. Thus, the weaker pairwise packing constraints are replaced by the following set of constraints:

$$x_{ir} + x_{is} + \sum_{j:\{v_r, v_s, i, j\} \in \tau} x_{js} \leq 1, \quad i = 1, \ldots k,\ i \in L(v_r),\ r < s,\ \{v_r, v_s\} \in E.$$

A discussion on how the two models can be further strengthened through the addition of clique inequalities is also provided. The models are tested on a number of randomly generated timetables using a custom integer programming solver and consider daily instances involving 200 trains and 14 platforms. The work of both De Luca Cardillo and Mione (1998) and Billionnet (2003) can be considered fairly simplified as no deviations to train arrival or departure times are permitted.

A graph colouring approach is also the basis of the study by Cornelsen and Di Stefano (2007). The authors similarly assume trains have fixed arrival and departure times. However, the network considered is perhaps even more simplified; the station has only one inbound path leading to a number of parallel platforms and one outbound path (although these are both bi-directional). The aim is to assign trains to platforms in such a way that they can arrive and depart on time without being blocked

by other trains. Cornelsen and Di Stefano (2007) propose an identical conflict graph formulation. The authors look at various timetables (linear and cyclic) as well as consider minor variations of the problem and report on the complexity of solving the resulting graph colouring problem.

4.2 Constraint programming approach

Constraint programming has also been used to model the train routing problem. The models presented in Delorme et al. (2001) and Rodriguez et al. (2002) are directed towards the strategic level problem of capacity assessment, whereas those presented in Rodriguez and Kermad (1998) and Rodriguez (2007) are an attempt to model the operational variant of the problem. There is, however, only a subtle difference between the two respective constraint programming formulations.

Constraint programming approaches view the movement of a train through the junction as a job. The similarities between job shop scheduling and train scheduling have been discussed in Sect. 2. Similarly in the junction setting, each track section is considered a machine, and an operation is associated with the traversal of a track section by a train. A sequence of operations thus results in an entire train path through a junction.

In contrast to node packing formulations, allocating a train path entails determining start times for a set of operations. Precedence constraints are required to ensure that the scheduled times define a physically possible train path, whereas disjunctive constraints are required to prevent two conflicting operations being scheduled during an overlapping time interval. If two operations are scheduled at the same time, they must use different track sections. In determining entrance times to track sections, one must calculate the exact claim and release times of the track section. In this regard, constraint programming methods must explicitly take into account the role of the signalling system. Typically simulation is used to determine the amount of time a given train requires to traverse a given track section (this also includes some buffer time).

The operational level variant of the problem permits trains to wait on track sections if the subsequent track section is unavailable. This is precisely the difference between the constraint programming formulations of Delorme et al. (2001) and Rodriguez et al. (2002) and those of Rodriguez and Kermad (1998) and Rodriguez (2007). The latter two papers incorporate a possible waiting time (or delay) on the traversal time of each track section. The release time of a track section is adjusted to be the train's earliest possible exit time plus the accumulated delays to that point. No such waiting is permitted in the capacity assessment related work of Delorme et al. (2001) and Rodriguez et al. (2002).

Rodriguez (2007) defines the waiting time on a track section to consist of two types of delay: the time spent decelerating and the time spent waiting. A third possible delay, the wasted time during acceleration (since the train must start from a stationary position on entering the next track section), is associated with the delay of the subsequent section. The objective of the constraint programming formulation is to minimize the sum of delays.

The author tests and compares two variants of the constraint programming formulation. The first considers all three possible causes of delay, whereas the second ignores the time wasted in the acceleration phase (i.e. it assumes a train can reach maximum speed instantaneously if starting from rest). Both formulations are solved using a labelling and consistency procedure for constraint programs. Not surprisingly the method that ignores the time spent accelerating underestimates the actual total delay. However, the authors show that this approach required much less computing time and still produced good solutions. Test instances had between 6 and 24 trains and were based on the Pierrefitte-Gonesse railway junction in Paris.

Delorme et al. (2001) compares the constraint programming approach for capacity assessment with the GRASP procedure by applying each to a number of railway instances at the Pierrefitte-Gonesse junction. The variant of GRASP used was applied to a formulation not dissimilar to that of Delorme (2003), whereas the constraint programming formulation was similar to that of Rodriguez et al. (2002) and solved using constraint propagation. The authors cannot conclusively say one approach is superior to the other as both have positive aspects. The constraint programming formulation has limitations in the number of paths it can consider due the huge number of variables and constraints of the model. Despite this, it does appear to perform better on the instances where choosing a suitable start time was more influential than path selection. Contrastingly, the node packing formulation with the GRASP solution approach was able to consider alternative train paths effectively. It did, however, struggle on the instances where the selection of suitable start times was required in achieving a good routing. The authors suggest that future research should be directed toward a hybrid model.

4.3 Heuristic approaches

Carey and Carville (2003) consider the problem of platforming trains at busy stations. In contrast to De Luca Cardillo and Mione (1998), Billionnet (2003), and Cornelsen and Di Stefano (2007), the authors permit deviations on the arrival and departure times of trains. The considered station, however, is such that the selection of a platform still uniquely determines a train's inbound and outbound path. The authors propose a greedy heuristic that aims to emulate what is done in practice by experienced planners for platforming a set of trains.

The algorithm is noticeably similar to that of Carey and Lockwood (1995), in that trains are considered sequentially. The order is usually determined by train priority. For each train the cost of assigning it to each of its possible platforms is calculated. The cost incurred in assigning a train to a platform is a function of the time adjustments that need to be made to obtain a conflict free slot at the platform for the train. Obtaining a conflict free slot at a platform involves ensuring the respective headways are satisfied. The time adjustments pertain to the incumbent train and measure the deviation from the train's desired arrival, dwell, and departure times. Each train is assigned the platform that yields the lowest cost. Computational experiments reported in the paper focus on the Leeds railway station in England and consider a daily timetable with 491 trains.

More recently, this work has been extended by Carey and Crawford (2007) to look at scheduling trains on a network of stations connected by multiple one-way lines in each direction. In particular, the headway for trains departing from the same station onto the same line is modified to ensure that if there is disparity in the speeds of the trains, then a faster train will not catch up to and be delayed behind a slower one. Essentially this amounts to including extra buffer time in setting the departure time of a train in such situations. The aim of Carey and Carville (2003) was simply to resolve conflicts within a particular station regardless of the ramifications the sequencing of the trains would have on the lines between stations.

The underlying heuristic of Carey and Crawford (2007) is identical to that described in Carey and Carville (2003), apart from the inclusion of the modified headway restriction described above. Trains are still assigned one after the other and receive a platform that results in the lowest cost. However, Carey and Crawford (2007) do modify the algorithm of Carey and Carville (2003) slightly to include some extra flexibility when resolving conflicts. When determining the cost of a particular platform for a train, if either departure headway or platform occupation conflicts exist (i.e. the dwell time at the platform of the train overlaps with an already scheduled train), time adjustments for both trains are considered depending on which results in the lower, immediate cost. Carey and Carville (2003) assess the impact of each modification by running and comparing three versions of the heuristic on a synthetic example which contains 25 linked stations.

4.4 Multicommodity flow formulation

One noticeable limitation with the node packing formulation of the train routing problem is that, for conflicts arising from conflicting movements within the junction for different trains, there is no indication in the constraint set where (in both space and time) the conflicts occur. To achieve this, each train path must by unpacked into a sequence of topology points. Fuchsberger (2007) introduces the notion of a resource tree conflict graph for the train routing problem. This approach attempts to find a conflict free routing for a set of trains by tying the path conflicts to particular track sections within the junction.

A resource tree graph is a tree structure that is defined for each train and contains an enumerated list of all the routes for the train between its entering point at the junction and its leaving point. Each node of the tree structure is associated with a topological element of the junction and contains information concerning the time at which the train reaches the corresponding topological element as well as the train's speed. For example, the root node of the tree structure contains information on the train's entering track section, entering time and entering speed, while the leaf nodes each correspond to the train leaving the junction at the train's designated leaving track section via one of the train's possible routes. A branch in the tree structure is defined if the topology point represented by the node provides the train with two alternatives (i.e. it indicates a switch or an intersection). An edge between two nodes indicates the section of track connecting the two topological elements and the times on the nodes can be used to calculate the train's traversal time.

By adding, for each train, a dummy source node that precedes the root node and a dummy sink node that succeeds all leaf nodes, the train routing problem can be converted into a multicommodity network flow problem. To state this formulation more formally, we introduce the following notation. We assume that each of the t trains in the problem has a set O_k^+ $(k = 1, \ldots, t)$ of nodes connected to the source node of train k and a set of nodes D_k^- connected to the sink node of train k. Furthermore, any node j is assumed to have a set of successor nodes Δ_j^+. The parameter c_{ij} reflects the travel time associated with nodes i and j of the network, the binary decision variables x_{ij} govern the inclusion of a particular edge (delimited by nodes i and j) in the solution. The basic multicommodity flow model can hence be stated as follows:

$$\text{Minimize:} \quad \sum_i \sum_j c_{ij} x_{ij} \tag{11}$$

Subject to:

$$\sum_{j \in O_k^+} x_{kj} = 1, \quad k = 1, \ldots, t, \tag{12}$$

$$\sum_{i \in D_k^-} x_{ik} = 1, \quad k = 1, \ldots, t, \tag{13}$$

$$x_{ij} = \sum_{m \in \Delta_j^+} x_{jm} \quad \text{for all } i, j : j \notin \bigcup_{k=1}^{t} D_k^-, \tag{14}$$

$$x_{ij} \in \{0, 1\}. \tag{15}$$

The objective function (11) minimizes the total travel time of the trains. Constraints (12)–(14) are the flow constraints and ensure that each train receives a feasible path through the junction. Finally, constraints (15) impose the integrality requirements on the flow variables.

Additional constraints are, however, required to ensure that the train paths selected are conflict free. In determining resources (track sections) where a conflict occurs, one must calculate the time each resource is claimed. Overlapping time intervals for train paths on a particular track section indicate that the train paths are in conflict. Fuchsberger (2007) defines conflict cliques pertaining to each resource. Multiple cliques may exist for the same track section if there is more than one time instant in which a conflict occurs. The resource tree conflict graph has a node for each resource and includes all resource tree graphs (i.e. the resource tree graph for each train). Conflict cliques are identified by edges between the resource nodes and nodes of the resource tree graphs. The multicommodity flow formulation is modified to include the following constraints:

$$\sum_{(i,j) \in C} x_{ij} \leq 1 \quad \text{for all } C \in \mathscr{C}_s, \quad \text{for all track sections } s, \tag{16}$$

where $\mathscr{C}_s$, the set of all conflict cliques arising on track section s. The full multicommodity formulation is hence given by (11)–(16).

Fuchsberger (2007) also introduces the concept of *pulsed train times*. This extends the resource tree graph structure described above to include variation in the train's arrival time at the junction. The author suggests duplicating the resource tree graphs for the trains. Each resource tree graph for a particular train uniquely corresponds to a possible entering time at the junction.

Computational experiments reported in the author's thesis focus on the Bern railway station in Switzerland and show that the tree conflict graph requires significantly fewer constraints than an equivalent node packing formulation. By binding the conflicts to track sections one obtains fewer yet stronger constraints. Computational experiments are also performed for the pulsed train case. Although the solution times are still acceptable, the author acknowledges that the approach of duplicating trees to model arrival time variation requires significant amounts of computer memory. All tests are solved using Cplex 9.1.

4.5 Alternative graph formulation

D'Ariano et al. (2007) propose an alternative graph formulation for the operational variant of the train routing problem. The concept of an alternative graph was first introduced by Mascis and Pacciarelli (2000), and has been further studied by Mascis and Pacciarelli (2002). An alternative graph is a generalization of the disjunctive graph introduced by Roy and Sussman (1964) and is used to model job shop scheduling problems.

An alternative graph G_A is characterized by the triple (V, F, A). The node set V corresponds to the set of operations and includes two dummy nodes, 0 and n. The source node 0 is equivalent to a dummy operation that precedes the first operation of every job, whereas node n is a dummy task that succeeds the last operation of every job. The set F corresponds to a set of conjunctive arcs. A conjunctive arc links a pair of consecutive operations for the same job and defines a precedence relation. The weight of such an arc reflects the processing time of the operation it emanates from. Set A defines a set of so called *alternative* arcs. Alternative arcs are defined in pairs and are used to model the situation in which there is a potential conflict between pairs of operations (typically because they require the use of the same machine simultaneously). Such arcs are equivalent to the ordering constraints defined in the model by Szpigel (1973). By selecting either one of the two arcs, a processing order is specified for the two conflicting operations.

Figure 4 highlights the main characteristics of an alternative graph. Let i and k be two conflicting operations. Conjunctive arcs link these operations with their respective successor operations j and l, whereas dashed arcs indicate the pair of alternative arcs required to enforce a processing order. The weight of each alternative arc can be interpreted as the *setup* time. This is the time that must elapse before the subsequent job can be performed on the same machine.

D'Ariano et al. (2007) equate the traversal of a block section with an operation. Hence a chain of nodes connecting node 0 with node n via conjunctive arcs denotes a train path. The setup time in this situation reflects the necessary headway that must exist between two trains that require the use of the same block section. One should

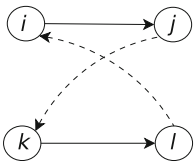

Fig. 4 The characteristics of an alternative graph

note that an alternative graph explicitly models the priority decisions associated with sequencing trains that share block sections. As the work is aimed at the operational level, in an attempt to deviate as little as possible from the planned schedule, all trains are assumed to remain on their allocated paths. The alternative graph formulation tries to determine the best sequencing order of trains at block sections where potential conflicts occur.

D'Ariano et al. (2007) define a *complete selection* S to be the set of arcs obtained by fixing exactly one arc from each pair of alternative arcs. The selection is said to be *consistent* if the graph $(V, F \cup S)$ has no positive length cycles. The authors show that the length of the longest path in a complete consistent selection (i.e. the path from the dummy start node to the end node) represents the maximum propagated delay of the corresponding train sequencing. The authors present a truncated branch-and-bound algorithm to minimize the length of the longest path. Given an initial selection (which may be empty), the procedure chooses an unselected alternative arc pair at each step and creates two extensions of the selection by considering the two sequencing possibilities. To keep the branch-and-bound tree within a tractable size, both static and dynamic implication rules are implemented. Essentially these just reduce the number of unselected alternative arc pairs by automatically setting an arc based on implications from previously set arcs. For each partial selection a lower bound on the extension of the selection is obtained via the *Jackson preemptive schedule*, and if this is greater than the best current upper bound on the maximum propagated delay, the node is fathomed.

The model is tested on a section of the Dutch railway and compared with three heuristic methods. The test instances had a time horizon of 1 or 2 h and had 54 or 108 trains. Based on 300 test instances the branch-and-bound algorithm proved to be far superior in that it could produce optimal or near optimal solutions quickly.

4.6 Set packing problem

First proposed in Velásquez et al. (2005) and further studied by Lusby (2008) is a set packing inspired model for the train routing problem. This model is very similar in definition to that which is proposed in Brännlund et al. (1998), and like Fuchsberger (2007) ties conflicts to particular resources (in both space and time) within the junction. While the node packing formulations discussed in Sect. 4.1.1 are, by definition, set packing problems, there are fundamental differences from a modelling perspective between such models and the methodology discussed in this section. For this reason they have been classified differently.

Unlike conflict graph approaches, the approach presented in Velásquez et al. (2005) and Lusby (2008) does not require the a priori generation of all conflicting train paths.

The models have a resource based constraint system identical to that of Brännlund et al. (1998); however, a much shorter time interval is observed. Each constraint corresponds to the use of a particular track section during a given 15-s time interval. Such constraints are termed *time interval track section* (*tints*) *constraints* and allow one to explicitly represent the movement of trains on the junction. Each column of the model corresponds to a feasible train path, and the non-zero elements of the column indicate which track sections the train is claiming and during what time intervals. One can observe that the binary decision variables for this model are identical to that of the node packing formulation. The paths are, however, unpacked into topological points of the junction over time and constraints are enforced on the track sections of the routes.

One can formally state the model as follows. Let us assume we have a set of trains $N(|N| = t)$, a set of tints constraints S, and a set of paths $\Omega(|\Omega| = n)$. We define the two matrices T and R, each of which consists of n columns. Matrix $T = (T_{i\omega})$ contains a row for each train, and $T_{i\omega} = 1$ if column $\omega \in \Omega$ is a path for train $i \in N$. Each column of T contains just one non-zero element. Matrix $R = (R_{s\omega})$ contains a row for each tints constraint $s \in S$. Each element $R_{s\omega}$ is usually either zero or one, indicating whether or not path $\omega \in \Omega$ claims tints resource $s \in S$; however, Lusby (2008) also considers $\frac{1}{2}$ elements. The cost of path $\omega \in \Omega$ is denoted as c_ω. Finally, the binary decision variable x_ω is equal to one if path $\omega \in \Omega$ is used in the solution and is zero otherwise. The full formulation is given below.

$$\text{Minimize:} \quad \boldsymbol{c}^T \boldsymbol{x} \tag{17}$$

$$\text{Subject to:}$$

$$T\boldsymbol{x} = \mathbb{1}, \tag{18}$$

$$R\boldsymbol{x} \leq \mathbb{1}, \tag{19}$$

$$\boldsymbol{x} \in \{0, 1\}^n. \tag{20}$$

The objective function (17) minimizes the total cost in routing all trains. Constraints (18) ensure that all trains receive a path, while constraints (19) enforce the restriction that at most one train can claim any track section during each time interval. Finally, constraints (20) give the binary restrictions on each of the decision variables. The nature of the constraint system is such that it inherently guarantees a conflict free routing for the trains. Furthermore, greater flexibility is provided as the structure of the model allows one to dynamically add train paths. This is primarily because the row dimension of the model is independent of the number of conflicting train movements.

From a modelling perspective one can also identify similarities with the aforementioned constraint programming approaches. In particular, in calculating the exact claim and release times (to the nearest time interval) of the track sections on any of the possible routes for a train the role of the signalling system must be explicitly modelled. Lusby (2008) models the movement of a train over a given route using an acyclic time–space network. Such a tree structure is similar to that proposed in Fuchsberger (2007); however, one noticeable difference exists—the route tree structure proposed by Fuchsberger (2007) contains all the possible routes through the junction for a train, whereas Lusby (2008) defines a tree structure for each of train's possible routes. This

Table 1 Junction routing literature review

Author	Level	Model	Solution approach
Zwaneveld et al. (1996)	S	Conflict graph	Branch-&-cut
De Luca Cardillo and Mione (1998)	T	Conflict graph	Backtracking Heuristic
Zwaneveld et al. (2001)	T	Conflict graph	Branch-&-cut
Delorme et al. (2001)	S	Constraint program	Constraint propagation
Billionnet (2003)	T	Conflict graph	Branch-&-bound
Carey and Carville (2003)	T	–	Greedy heuristic
Delorme (2003)	S	Conflict graph	GRASP metaheuristic
Caimi et al. (2005)	T	Conflict graph	Fixed point + Local search
Gandibleux et al. (2005)	S	Conflict graph	ACO metaheuristic
Velásquez et al. (2005)	T	Set packing	Branch-&-bound
Herrman (2006)	T	Conflict graph	Fixed point + Local search
Caprara et al. (2007a)	T	Conflict graph	Branch-&-price-&-cut
Carey and Crawford (2007)	T	–	Greedy heuristic
Cornelsen and Di Stefano (2007)	T	Conflict graph	Graph colouring algorithms
D'Ariano et al. (2007)	O	Alternative graph	Branch-&-bound
Fuchsberger (2007)	T	Multicommodity flow	Branch-&-bound
Rodriguez (2007)	O	Constraint program	Constraint Propagation
Lusby (2008)	S, O	Generalized set packing	Branch-&-price

difference comes from the fact that the traversal time of a track section is not assumed to be constant in Lusby (2008). The route tree structure contains all the different acceleration and deceleration strategies (i.e. paths) for the train on the particular route. Lusby (2008) shows that the track section discretization of a junction is not trivial to model; the same physical piece of track can be defined by more than one track section and for this reason one cannot use a strict set packing model for the problem.

To overcome the models large row dimension Lusby (2008) presents a branch-and-price based solution methodology that utilizes the dual representation of any basic feasible solution. The model is first applied to the strategic level variant of the train routing problem. Instances of up to 45 trains are considered and results suggest that the improvement in the bound obtained from the LP relaxation over an equivalent, unpreprocessed node packing formulation can be as high as 66.7%. The author exploits the flexibility of the model and shows that it can be used to solve the operational level variant of the train routing problem. Here the objective is to minimize the total delay incurred in recovering feasibility of a timetable. A real-life instance in Germany, and supplied by Deutsche Bahn, is the focus of the operational work. Instances of up to 66 trains are considered and results show that near optimal solutions can be found quickly with this approach.

To conclude our discussion on junction train routing, Table 1 summarizes all the reviewed contributions. In particular, the planning level the study focuses on is given (i.e. strategic, tactical, or operational), the underlying model, and the solution method are reported.

5 Conclusions and future work

In this paper we have considered the important problem of routing rolling stock in the railway industry and discussed the various techniques that have been proposed to determine how the track capacity of a railway network should be allocated to trains. We have identified and reviewed several variants of the problem, and where possible tried to group the contributions according to certain characteristics. An exhaustive survey on contributions in the field of junction train routing has been included to compare and contrast the work that has been done in this area. Table 1 as well as the discussion in Sect. 4 highlight the fact that conflict graph approaches are the most widely used models for this problem. Contributions in this area suggest that the such approaches typically lead to large formulations that require significant amounts of preprocessing and are inflexible in the sense that additional train paths cannot be included easily. Hence, making them impractical in a real-time environment. More recent studies have shown that there is potential in adopting resource based constraint systems, and furthermore, that such models provide more flexibility in a dynamic environment.

For complex junctions currently the timetabling and routing phases are performed separately. Possible future work might look at integrating these two problems to see if additional benefits can be had. In addition to this, a natural extension to such routing problems would be to consider more than one junction in the routing phase. This is particularly true in a dynamic setting where delays at one junction will undoubtedly affect the routings at subsequent stations on the line.

References

Adenso-Diaź B, Olivia González M, González-Torre P (1999) On-line timetable re-scheduling in regional train services. Transp Res B 33:387–398

Albrecht T (2009) Automated timetable design for demand-oriented service on surburban railways. Public Transp 1(1):5–20

Assad AA (1980) Modelling of rail networks: toward a routing/makeup model. Transp Sci Part B 14: 101–114

Billionnet A (2003) Using integer programming to solve the train-platforming problem. Transp Sci 37(2):213–222

Borndörfer R, Schlechte T (2007a) Models for railway track allocation. Technical Report 07–02, Konrad-Zuse-Zentrum für Informationstechnik Berlin

Borndörfer R, Schlechte T (2007b) Solving railway track allocation problems. Technical Report 07–20, Konrad-Zuse-Zentrum für Informationstechnik Berlin

Borndörfer R, Grötschel M, Lukac S, Mitusch M, Schlechte T, Schultz S, Tanner A (2005) An auctioning approach to railway slot allocation. Technical Report 05–45, Konrad-Zuse-Zentrum für Informationstechnik Berlin

Brännlund U, Lindberg PO, Nõu A, Nilsson JE (1998) Railway timetabling using lagrangian relaxation. Transp Sci 32(4):358–369

Burkolter D (2005) Capacity of railways in station areas using petri nets. PhD thesis, Swiss Federal Institute of Technology Zurich

Bussieck MR, Winter T, Zimmerman UT (1997) Discrete optimization in public rail trainsport. Math Program 79:415–444

Cacchiani V, Caprara A, Toth P (2008) A column generation approach to train timetabling on a corridor. 4OR 6:125–142

Cai X, Goh CJ (1994) A fast heuristic for the train scheduling problem. Comput Oper Res 21(5):499–510

Cai X, Goh CJ, Mees AI (1998) Greedy heuristics for rapid scheduling of trains on a single track. IIE Trans 30:481–493

Caimi G, Burkolter D, Herrmann T (2005) Finding delay-tolerant train routings through stations. In: Fleuren H (ed) Operations research proceedings 2004: selected papers of the annual international conference of the German operations research society (GOR) jointly organized with the Netherlands society. Springer, Berlin, pp 136–143

Caprara A, Fischetti M, Toth P (2002) Modeling and solving the train timetabling problem. Oper Res 50(5):851–861

Caprara A, Monaci M, Toth P, Guida PL (2006) A lagrangian heuristic alogorithn for a real-world train timetabling problem. Discret Appl Math 154:738–753

Caprara A, Galli L, Toth P (2007a) Solution of the train platforming problem. In: Liebchen C, Ahuja RK, Mesa JA (eds) ATMOS 2007—7th workshop on algorithmic approaches for transportation modeling, optimization, and systems, Dagstuhl, Germany. Internationales Begegnungs und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany

Caprara A, Kroon LG, Monaci M, Peeters M, Toth P (2007b) Passenger railway optimization. In: Barnhart C, Laporte G (eds) Handbooks in operations research and management science, vol 14, chap 3, pp 129–187. Elsevier, Amsterdam

Carey M (1994a) Extending a train pathing model from one-way to two-way track. Transp Res Part B 28B(5):395–400

Carey M (1994b) A model and strategy for train pathing with choice of lines, platforms, and routes. Transp Res Part B 28B:333–353

Carey M, Carville S (2003) Scheduling and platforming trains at busy complex stations. Transp Res Part A 37:195–224

Carey M, Crawford I (2007) Scheduling trains on a network of busy complex stations. Transp Sci Part B 41:159–178

Carey M, Lockwood D (1995) A model, algorithms and strategy for train pathing. J Oper Res Soc 46(8):988–1005

Cordeau J, Toth P, Vigo D (1998) A survey of optimization models for train routing and scheduling. Transp Sci 32(4):380–400

Cornelsen S, Di Stefano G (2007) Track assignment. J Discret Algorithms 5(2):250–261

D'Ariano A, Pacciarelli D, Pranzo M (2007) A branch-and-bound algorithm for scheduling trains in a railway network. Eur J Oper Res 183:643–657

D'Ariano A, Corman F, Pacciarelli D, Pranzo M (2008) Reordering and local rerouting strategies to manage train traffic in real time. Transp Sci 42(4):405–419

De Luca Cardillo D, Mione N (1998) k L-list τ colouring of graphs. Eur J Oper Res 106:160–164

Delorme X (2003) Modélisation et résolution de problèmes liés à l'exploitation d'infrastructures ferroviaires. PhD thesis, Université de Valenciennes et du Hainaut Cambrésis

Delorme X, Rodriguez J, Gandibleux X (2001) Heuristics for railway infrastructure saturation. Electron Notes Theor Comput Sci 50(1):39–53

Delorme X, Gandibleux X, Rodriguez J (2004) GRASP for set packing problems. Eur J Oper Res 153(3): 564–580

Dorfman MJ, Medanic J (2004) Scheduling trains on a railway network using a discrete event model of railway traffic. Transp Res Part B 38(1):81–98

Fischetti M, Salvagnin D, Zanette A (2009) Fast approaches to improve the robustness of a railway timetable. Transp Sci 43:321–335

Freling R, Lentink RM, Kroon LG, Huisman D (2005) Shunting of passenger train units in a railway station. Transp Sci 39(2):261–272

Fuchsberger M (2007) Solving the train scheduling problem in a main station area via a resource constrained space–time integer multi-commodity flow. Master's thesis, Institute for Operations Research ETH Zurich

Gandibleux X, Delorme X, T'Kindt V (2004) Ant colony optimisation algorithm for the set packing problem. In: ANTS Workshop, pp 49–60

Gandibleux X, Jorge J, Angibaud S, Delorme X, Rodriguez J (2005) An ant colony optimization inspired algorithm for the set packing problem with application to railway infrastructure. In: Proceedings of the sixth metaheuristics international conference (MIC2005), pp 390–396

Herrman TM (2006) Stability of timetables and train routings through station regions. PhD thesis, Swiss Federal Insitute of Technology Zurich

Higgins A, Kozan E, Ferreira L (1996) Optimal scheduling of trains on a single line track. Transp Res Part B 30B(2):147–161
Higgins A, Kozan E, Ferreira L (1997) Heuristic techniques for single line train scheduling. J Heuristics 3:43–62
Hooghiemstra JS, Kroon LG, Odijk MA, Salomon M, Zwaneveld PJ (1999) Decision support systems support the search for win–win solutions in railway network design. Interfaces 29(2):15–32
Huisman D, Kroon LG, Lentik RM, Vromans MJCM (2005) Operations research in passenger railway transportation. Technical report, Erasmus Research Institute of Management
Jovanović D, Harker PT (1991) Tactical scheduling of rail operations: the scan *i* system. Transp Sci 25(1):46–64
Kraay D, Harker PT, Chen B (1991) Optimal pacing of trains in freight railroads: model formulation and solution. Oper Res 39(1):82–99
Kroon LG, Peeters L (2003) A variable trip time model for cyclic railway timetabling. Transp Sci 37:198–212
Kroon LG, Lentink RM, Schrijver A (2008) Shunting of passenger train units. Transp Sci 42(4):436–449
Liebchen C (2003) Finding short integral cycle bases for cyclic timetabling. Technical Report 12, Technische Universität Berlin, Institut fur Mathematik
Liebchen C (2006) Periodic timetable optimization in public transportation. PhD thesis, Technische Universität Berlin
Liebchen C, Möhring R (2004) The modelling power of the periodic event scheduling problem: railway timetables—and beyond. Technical Report 2004/20, Technische Universität Berlin, Institut fur Mathematik
Liebchen C, Peeters L (2002) On cyclic timetabling and cycles in graphs. Technical Report 761/2002, Technische Universität Berlin, Institut fur Mathematik
Liebchen C, Stiller S (2009) Delay resistant timetabling. Public Transp 1(1):55–72
Liebchen C, Proksch M, Wagner FH (2008) Performance of algorithms for periodic timetable optimization. In: Hickman M, Mirchandani P, Voss S (eds) Computed-aided systems in transport (CASPT2004). Springer, Berlin
Lindner T (2000) Train schedule optimization in public rail transportation. PhD thesis, Technical University of Braunschweig
Lusby RM (2008) Optimization methods for routing trains through railway junctions. PhD thesis, The University of Auckland
Mascis A, Pacciarelli D (2000) Machine scheduling via alternative graphs. Technical Report DIA-46-2000, Dipartimento di Informatica e Automazione, Universitá Roma Tre, Roma, Italy
Mascis A, Pacciarelli D (2002) Job shop scheduling with blocking and no-wait constraints. Eur J Oper Res 143(3):498–517
Nachtigall K (1996) Cutting planes for a polyhedron associated with a periodic network. Technical Report IB 112-96/17, Deutsche Forschungsanstalt für Luft- und Raumfahrt e.V
Nachtigall K (1998) Periodic network optimization and fixed interval timetables. Habilitation Thesis
Nachtigall K, Voget S (1996) A genetic algorithm approach to periodic railway synchronization. Comput Oper Res 23(5):453–463
Odijk MA (1996) A constraint generation algorithm for the construction of periodic timetables. Transp Res B 30(6):455–464
Oliveira O, Smith BM (2000) A job shop scheduling model for the single track-railway timetabling problem. Technical Report 2000.21, University of Leeds
Pachl J (2004) Systemtechnik des Schienenverkehrs (in German). B.G. Teubner, Stuttgart
Peeters L (2003) Cyclic railway timetable optimization. PhD thesis, Erasmus Research Institute of Management
Rezanova N, Ryan DM (2009) The train driver recover problem—a set partitioning based model and solution method. Comput Oper Res (forthcoming). doi:10.1016/j.cor.2009.03.023
Rodriguez J (2007) A constraint programming model for real-time trains scheduling at junctions. Transp Res Part B 41(2):231–245
Rodriguez J, Kermad L (1998) Constraint programming for real-time train circulation management problems in railway nodes. In: Proceedings of the international conference on computer aided design, manufacture and operation in the railway and other advanced mass transit systems, pp 597–606
Rodriguez J, Delorme X, Gandibleux X (2002) Railway infrastructure saturation using constraint programming approach. In: Computers in railways VIII. WIT Press, Lemmos, Greece, pp 807–816

Roy B, Sussman B (1964) Les problèm d'ordonnancement avec contraintes disjonctives. Note ds no. 9 bis. SEMA, Paris

Şahin İ (1999) Railway traffic control and train scheduling based on inter-train conflict management. Transp Res Part B 33:511–534

Sauder R, Westerman WM (1983) Computer aided train dispatching: decision support through optimization. Interfaces 6:24–37

Schöbel A (2009) Capacity constraints in delay management. Public Transp 1(2):135–154

Schrijver A, Steenbeek A (1994) Timetable construction for railned (in Dutch). Technical report, C.W.I. Center for Mathematics and Computer Science, Amsterdam

Serafini P, Ukovich W (1989) A mathematical model for periodic scheduling problems. Soc Ind Appl Math J Discret Math 2(4):550–581

Szpigel B (1973) Optimal train scheduling on a single line railway. Oper Res 72:344–351

Törnquist J (2005) Computer-based decision support for railway traffic scheduling and dispatching: a review of models and algorithms. In: ATMOS2005 (algorithmic methods and models for optimization of railways), Palma de Mallorca, Spain, October 2005. Dagstuhl Research Online Publication Server (DROPS)

Törnquist J, Persson JA (2007) N-tracked railway traffic rescheduling during disturbances. Transp Res Part B 41(3):342–362

Vaidyanathan B, Ahuja RK, Orlin JB (2008) The locomotive routing problem. Transp Sci 42:492–507

Vansteenwegen P, Van Oudheusden D (2006) Developing railway timetable which guarantee a better service. Eur J Oper Res 173:337–350

Velásquez R, Ehrgott M, Ryan D, Schöbel A (2005) A set packing approach to routing trains through railway stations. In: Proceedings of the 40th annual conference of the operational research society of New Zealand, pp 305–314

Walker CG, Snowden JN, Ryan DM (2005) Simultaneous disruption recovery of a train timetable and crew roster in real time. Comput Oper Res 32(8):2077–2094

Wong RCW, Yuen WY, KwokWah Fung T, Leung JMY (2008) Timetable synchronization for rail mass transit. Transp Sci 42:57–69

Zwaneveld PJ (1997) Routing of trains and allocation of passenger lines. PhD thesis, Rotterdam School of Management, TRAIL Research School

Zwaneveld PJ, Kroon LG, Romeijn HE, Salomon M, Dauzere-Peres S, van Hoesel SPM, Ambergen HW (1996) Routing trains through railway stations: model formulation and algorithms. Transp Sci 30(3):181–194

Zwaneveld PJ, Kroon LG, Romeijn HE (1997) Routing trains through railway stations: complexity issues. Eur J Oper Res 98:485–498

Zwaneveld PJ, Kroon LG, van Hoesel SPM (2001) Routing trains through a railway station based on a node packing model. Eur J Oper Res 128:14–33