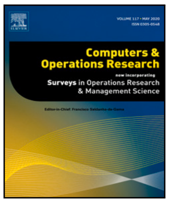
Computers & Operations Research 142 (2022) 105719

Contents lists available at ScienceDirect

Computers and Operations Research

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



A data-driven, variable-speed model for the train timetable rescheduling problem

Edwin Reynolds[a], Stephen J. Maher[b,*]

[a] STOR-i Centre for Doctoral Training, Lancaster University, Lancaster LA1 4YX, UK
[b] Department of Mathematics, University of Exeter, Exeter EX4 4QF, UK

ARTICLE INFO

Keywords:
Railway optimisation
Timetable rescheduling
Speed profile
Variable-speed

ABSTRACT

Train timetable rescheduling — the practice of changing the routes and timings of trains in real-time to respond to delays — can help to reduce the impact of reactionary delay. There are a number of existing optimisation models that can be used to determine the best way to reschedule the timetable in any given traffic scenario. However, many of these models do not adequately account for the acceleration and deceleration required for trains to achieve the rescheduled timetable. The few models that do account for this are overly complex and cannot be solved to optimality in sufficiently short times. In this study, we propose a new model for train timetable rescheduling that uses statistical methods and historical data to parsimoniously take train speed into account. The model is tested using a new set of instances based on real data from Derby station in the UK. We show that the improved accuracy of the proposed model comes with little to no trade-off in terms of run time compared to fixed-speed timetable rescheduling models.

1. Introduction

Reactionary delays are a significant problem in railway systems. These are delays that are caused by the knock-on effect of prior delays. In 2019, reactionary delays were responsible for 64.35% of total train delay minutes in Great Britain (statistic provided by Network Rail). The problem is particularly acute in large, busy station areas, where the limited capacity of the station creates a bottleneck.

Reactionary delays can be reduced by performing *timetable rescheduling*. Timetable rescheduling involves changing the planned schedules of trains in real-time to respond to unexpected delays. The aim of solving the Train Timetable Rescheduling Problem (TTRP) (Cacchiani et al., 2014) is to find a way to reschedule the timetable that is achievable in practice and optimises some objective. TTRP models must take into account the *speed profile* of each train, which describes how the velocity of the train changes over time. If the speed profiles of trains are not modelled with sufficient accuracy, then TTRP solutions may perform worse than expected in practice. This has been demonstrated by Hosteins et al. (2019) using a detailed railway simulator. Lack of attention to speed profile modelling is therefore a significant risk to the validity of TTRP models.

The focus of this paper is the development of techniques for improving the modelling of speed profiles in TTRP models. In particular, the model proposed by Reynolds et al. (2020) is extended to include approximate train speed trajectories. Using a new set of realistic instances from Derby station in the UK, we show that this extended model can be solved to optimality in times comparable to the original model of Reynolds et al. (2020).

1.1. Problem description

The TTRP is solved following a disturbance to the timetable to calculate an optimal rescheduled timetable. This disturbance could consist of any set of delays that results in two trains requiring the same infrastructure at the same time (a *conflict*), making the current timetable infeasible. The rescheduled timetable — the solution to the TTRP — consists of a new route and set of timings (i.e. a new schedule) for each controlled train. Controlled trains are those that are forecast to be inside a defined area of track during a time horizon. New routes can involve stopping at different platforms from those originally planned, taking different approaches to planned platform stops, or cancelling stops altogether. The rescheduled timetable must contain no conflicts and therefore be capable of being carried out in practice, respecting the constraints of the signalling system. A solution is considered optimal if it maximises a utility function representing the preferences of Network Rail, the infrastructure manager in Great Britain. This is modelled as the total weighted utility over all train stops which are carried out,

* Corresponding author.
E-mail address: s.j.maher@exeter.ac.uk (S.J. Maher).

https://doi.org/10.1016/j.cor.2022.105719
Received 5 February 2021; Received in revised form 16 December 2021; Accepted 11 January 2022
Available online 4 February 2022

where at each stop the utility disfavours lateness, platform change and cancellation.

1.2. Structure of the paper

Key concepts and relevant literature are reviewed in Section 2. Section 3 provides an overview of our proposed model and approach to modelling speed profiles. Methods for estimating traversal times are developed in Section 4. The model is then described in full in Section 5. Section 6 presents a computational study. Section 7 contains our conclusions and suggestions for future research.

2. Literature review

Many different variants of the TTRP have been described in the literature. As a result, many different models have been proposed. Much of this research is detailed in the surveys of Cacchiani et al. (2014), Fang et al. (2015) and Corman and Meng (2015). The variants of the TTRP discussed in this paper involve modelling the speed that trains traverse sections of track using speed profiles. A speed profile is commonly used to describe the evolution of speed either over time or space. In this paper, speed profiles are defined over time. Formally, the *speed profile* of a train k is a continuous, non-linear function $v^k : [0,T] \rightarrow \mathbb{R}^+$ mapping each time t in the time horizon to the velocity $v^k(t)$ of the train at that instant. However, it is computationally impractical to optimise such a function for each train within a TTRP model. Our literature review will focus specifically on the ways in which train speed profiles have been approximated in TTRP models.

2.1. Fixed-speed and variable-speed models

Timetable rescheduling models can be classified as either *fixed-speed* or *variable-speed* models, depending on how speed profiles are modelled. An early reference to this terminology appears in Cordeau et al. (1998, p. 393–396). In fixed-speed models, speed profiles are implicitly modelled via the specification of a fixed minimum time that is required for each train to traverse each segment of railway track. These are called *minimum traversal times* because whilst trains are permitted to stop in any segment and hence spend longer than this minimum time, they cannot traverse the segment in a shorter amount of time. In fixed-speed models, minimum traversal times are pre-computed and apply regardless of the rescheduling actions that are proposed by the model or the speed profiles required in practice to achieve them. The fixed minimum traversal time of a segment may be the same for every train that traverses it. Alternatively, minimum traversal times may be calculated based on assumptions about the likely speed profile of a particular train carrying out its originally planned schedule. There are many examples of fixed-speed TTRP models, such as those presented by Corman et al. (2010), Meng and Zhou (2014), Pellegrini et al. (2014), Lamorgese et al. (2016) and Reynolds et al. (2020).

Fixed-speed models are an approximation of railway operations and, as such, produce solutions that may be infeasible in practice. This situation is illustrated in Fig. 1. Consider a train traversing three segments of track r_0, r_1 and r_2 in sequence. If the train maintains a constant velocity, then the traversal times of the segments are t_{r_0}, t_{r_1} and t_{r_2}, respectively. Now suppose that the train comes to an unplanned stop in r_2 as a result of a rescheduling decision. A fixed-speed model will use the same fixed traversal times for each segment, with an additional waiting time of D in r_2. However, in reality the train's speed profile must change so that it decelerates in order to stop, and accelerates afterwards. This changes the real traversal times to $t^{var}_{r_0} > t_{r_0}$, $t^{var}_{r_1} > t_{r_1}$ and $t^{var}_{r_2} > t_{r_2}$. As a result, the disparity between actual traversal times and the traversal times in a fixed-speed model can be large. The negative effects of these disparities have been observed by Hosteins et al. (2019). They find that high-quality solutions produced by the fixed-speed optimisation model of Pellegrini et al. (2014) do not always perform well when tested using a microscopic railway simulation. They find that the extent of the issue is dependent on the granularity of the model and the objective function used.

Variable-speed models attempt to overcome this problem by making traversal times dependent on rescheduling actions. One way of achieving this is by iterating between a fixed-speed rescheduling model and a speed profile optimisation model. Speed profile optimisation models are detailed kinetic models that optimise the speed profile of a single train carrying out a fixed schedule to minimise journey times or energy consumption. The relevant literature is summarised by Yang et al. (2016), Scheepmaker et al. (2017) and Yin et al. (2017). The traversal times in the fixed-speed model are updated in each iteration according to the speed profiles required to achieve the rescheduling solution that it produced in the previous iteration. This iterative approach has been explored by both D'Ariano et al. (2007b) and Mazzarello and Ottaviani (2007). Whilst iterative approaches laudably avoid the need for fundamental changes to rescheduling models, they do not guarantee convergence to a solution that is feasible for both models. Moreover, it can be time consuming to solve a fixed-speed model multiple times, which makes the iterative approach especially unsuitable for the real-time environment in which rescheduling takes place.

To overcome these problems with the iterative approach, speed profiles can be modelled directly within the rescheduling model. Several authors have suggested simple ways to account for the disparity that arises between fixed traversal times and actual traversal times following unplanned stops (as described in the example above). Hosteins et al. (2019) suggest adding a fixed additional time to the traversal time of any segment in which a train comes to a stop. Rodriguez (2007) suggests adding an additional time to the traversal time of the subsequent segment that depends linearly on the amount of time the train stops for. Whilst these methods have the merit of being simple and leading to linear constraints, they are clearly very approximate. In reality, the additional acceleration time is not a fixed constant, and it is not linear in the amount of stopping time. Lusby et al. (2013) show that speed profiles can be modelled within the subproblem of a model solved using column generation. This allows speed profiles to be modelled in a more sophisticated way using kinematic formulae. A similar column generation approach to that of Lusby et al. (2013) is used in this paper. However, speed profiles are modelled differently to overcome some specific problems that are discussed in Section 4.

A different approach is to select the speed profile of each train in advance and then disallow rescheduling actions that would compromise their validity. This approach, taken by both Corman et al. (2009) and Caimi et al. (2012), is suitable if a *green wave* policy is in operation. A green wave policy dictates that trains must only come to a stop within stations, thereby reducing unnecessary braking and acceleration and saving energy. However, it does not solve the problem of modelling speed profiles within the general TTRP, where trains may come to a stop anywhere on the track network.

More recently, different ways of integrating the TTRP with speed profile optimisation have been proposed by Xu et al. (2017), Zhou et al. (2017) and Luan et al. (2018a,b). These models seek to determine the actual speed profile to be used by each train simultaneously with carrying out timetable rescheduling. The principal benefit of this integration is that by solving the two problems simultaneously, solutions with better overall quality can be achieved. However, to integrate the problems it is necessary to assume that precise real-time information about both train speed profiles and signalling states are centrally available, and that both can be centrally and automatically controlled. In other words, these models are only useful for railways in which the functions of *traffic control* and *train operation* are integrated (see Yin et al., 2017, p. 568 for a discussion of this integration). They are typically only integrated on new and expensive high-speed lines. As a result, the integration of the TTRP with speed profile optimisation is inappropriate for practical use on the majority of railways.

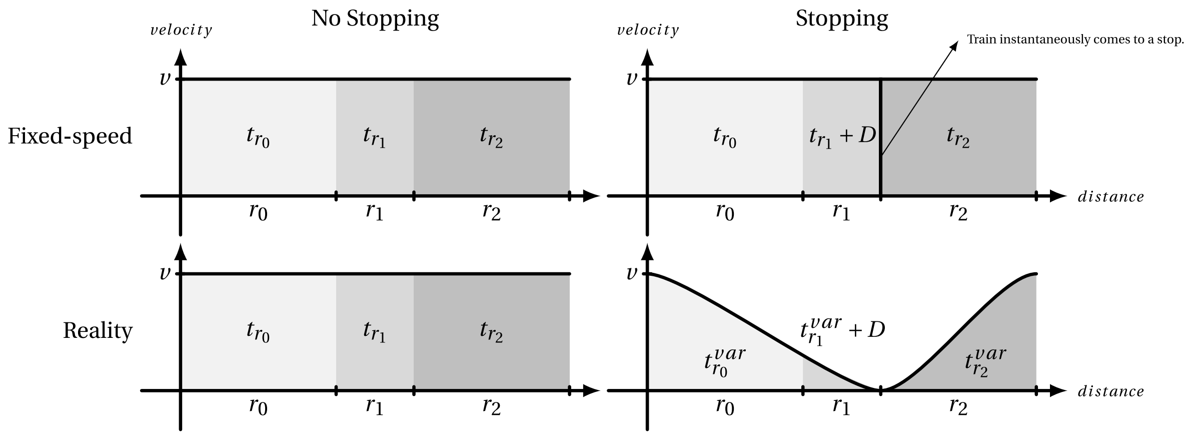


Fig. 1. An example of how the speed profile assumptions made in fixed-speed models compare to actual speed profiles. The plots depict a scenario in which a train traverses r_0, r_1 and r_2 in sequence. On the left, the train does not stop; on the right, the train stops in r_2.

2.2. Physics-based and data-driven models

Each of the models proposed by Lusby et al. (2013), Xu et al. (2017), Zhou et al. (2017) and Luan et al. (2018a,b) are *physics-based*. This means that traversal times of track segments are derived from kinetic speed profile modelling. Track segment distances and train speed capabilities are combined with assumptions about the tractive force applied by the driver to calculate traversal times. This kind of kinetic modelling has traditionally been used to estimate running times for timetable construction and evaluation — an introduction to this topic is provided by Brünger and Dahlhaus (2014). Luan et al. (2018a,b) incorporate much of this kinetic modelling into a mixed-integer non-linear programming formulation, and propose two heuristics for solving a linearised version of the formulation. Heuristics are used because realistic instances cannot be solved to optimality in suitable computation times. Both Xu et al. (2017) and Zhou et al. (2017) discretise velocity, thus avoiding direct representation of the non-linear kinetics of train motion. Xu et al. (2017) extend the Alternative Graph model (see D'Ariano et al. (2007a)) to incorporate speed-dependent traversal times. Zhou et al. (2017) propose a time–space–speed network model for the offline train timetabling problem on a high-speed line with power supply constraints.

The physics-based approach to traversal time modelling has practical disadvantages. Detailed information about the physical properties of the track and trains is required, despite the fact that it can be hard to obtain. For example, Luan et al. (2018b) use physical parameters for the resistance between train and track that are individualised for each train and block section. Finding this information is not simply a data collection exercise: many parameters need to first be estimated and then calibrated within the overall model. Although sophisticated methods such as those of Bešinović et al. (2013) have been developed to estimate the parameters used in physical speed profile calculations, each estimated parameter is still subject to uncertainty. The effect of this uncertainty on rescheduling models is not well understood. Our model bypasses the need for these physical parameters, and is therefore significantly less onerous to test and deploy. It is also easier to adapt to changes in infrastructure than physics-based models, since any changes will be reflected in the data and can be used to update traversal times. These are significant advantages given that practical implementations of the TTRP are still rare (see Lamorgese et al. (2018) for a description of the state of implementation).

There are also important modelling disadvantages to using the physics-based approach. In a physics-based model, the modeller must make assumptions about which speed profiles should be feasible. For example, Lusby et al. (2013) and Zhou et al. (2017) assume constant rates of acceleration over each section of track and each time interval, respectively. Lusby et al. (2013) allow trains to travel at any real-valued speed below the speed limit, whilst Zhou et al. (2017) and Xu et al. (2017) allow trains to travel only at one of a few pre-defined speeds in each block section. Zhou et al. (2017) allow trains to transition between any two speed levels provided limits on the acceleration and deceleration capabilities of the train are respected, whilst Xu et al. (2017) allow trains to transition only between adjacent speed levels. All of these approaches to constraining speed profiles run the simultaneous risks of both eliminating perfectly reasonable speed profiles from the feasible space, and including many speed profiles that are very unlikely to arise in practice.

In this paper, we avoid these problems by taking a *data-driven* approach. This terminology refers to the fact that we infer traversal times from historical observations, leading to an innovative synthesis of statistical techniques with optimisation. Our data-driven approach provides a natural way to model traversal times that reflects speed profiles that have actually arisen in the past on the parts of track that are modelled. Rather than making assumptions about how to constrain speed profiles, our approach allows the data to speak for itself. Results for our application show that the use of more than one traversal time is justified by the data on only a subset of the routes. This highlights an additional advantage of our data-driven approach. It is able to target increased model complexity (a higher number of possible traversal times) at parts of the track where it can be best justified by the data. The result is a significantly more parsimonious model than any of the physics-based models that have been mentioned.

2.3. Contributions

The contributions made by this paper can be summarised as follows:

1. We propose a new variable-speed model for the TTRP in complex station areas. The model utilises a time–space–type graph to approximate train speed profiles.
2. We show how the application of statistical methods to historical data can be used to model traversal times in a variable-speed model. This data-driven approach results in a parsimonious model that is less onerous to test and deploy than existing physics-based models. Moreover, it avoids the need to make restrictive assumptions about speed profiles.
3. We present a new set of instances based on real data from Derby station in the UK.
4. We show that these instances can be solved to optimality or provably near to optimality in times suitably short for real-time operations. In particular, the solving times are comparable with the fixed-speed model proposed by Reynolds et al. (2020).

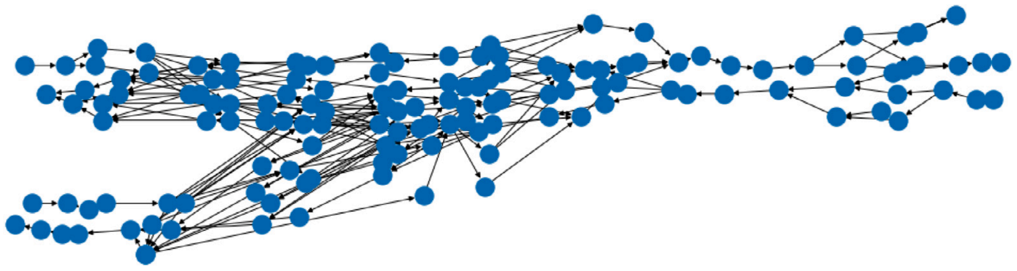


Fig. 2. The route graph G_r for the area centred on Derby station that is used in our computational experiments.

3. Model overview

This paper presents a model for the TTRP that is based on a time–space–type (TST) graph. This is a directed graph $G = (N_0 \cup \{source, sink\}, A)$ that models the state of each train over the time horizon using a path from the *source* node to the *sink* node. Each node in N_0 represents a state in which a train could be, whilst each arc represents a possible transition between these states.

Each node in N_0 corresponds to a combination (r, t, v) of a route r, a time interval t and a speed profile type v. These are defined as follows:

- A *route* is a short length of track that runs from one railway signal to another. The controlled area of track is formed of a set of routes N_r, that can be modelled as a route graph $G_r = (N_r, A_r)$, where the arcs in A_r represent feasible route transitions. Fig. 2 shows the route graph for an area centred on Derby station.
- *Time intervals* are periods of time, each of length 10 s, that together form a partition $\mathcal{T}$ of the time horizon. The time horizon begins at the time the model is solved and lasts for one hour.
- Each traversal of a route $r \in N_r$ by a train is modelled as having a *speed profile type*. The minimum number of time intervals L_r^v required to traverse r is dependent on the speed profile type v that is used. In general, the set of speed profile types is unrestricted, i.e. $N_v = \{v_0, v_1, \dots, v_n\}$. Whilst there is a general set of all types N_v, the possible speed profile types for any given route r are $N_r^v \subseteq N_v$. The definition of these speed profile types and the calculation of the traversal times is the subject of Section 4.

The transition between speed profile types is modelled using a speed profile type graph $G_v = (N_v, A_v)$, where $A_v = \{(v_i, v_i), (v_i, v_{i+1}), (v_{i+1}, v_i) | i \in 0, 1, 2, \dots, n-1\}$. The speed profile type used by a train on a given route is constrained to be adjacent in the type graph G_v to the type used on the preceding route. This requirement is enforced by the arc set A_v. While the transition through consecutive speed profile types can be prohibitive for large values of n, i.e. transitioning from v_n to v_0 requires the use of n routes, we will show in Section 4 that for small n this restriction is supported by railway operational data. The rationale for only allowing consecutive speed profile type transitions is that fast speeds and stopping must be separated by periods of acceleration or deceleration that involve slower speeds. Note that this assumption is similar to the constraint of Xu et al. (2017) stating that the speed levels on consecutive block sections must be at most one level apart. However, we use speed profile types instead of physically defined speed levels (e.g. 250–300 km/h).

3.1. Example TST graph

A small explanatory example of a TST graph is depicted in Fig. 3. In this example, the route graph is given by

$$N_r = \{AB, BC, CD, DE\} \text{ and } A_r = \{(AB, BC), (BC, CD), (CD, DE)\}$$

and the time horizon $\mathcal{T} = \{0, \dots, 7\}$ consists of eight time intervals. There are three speed profile types, v_0, v_1 and v_2, although type v_2 is not possible in routes CD or DE ($N_v^{CD} = \{v_0, v_1\} = N_v^{DE}$). Type v_0 corresponds to stopping, since the arcs join nodes corresponding to the same route at consecutive time intervals. The traversal times of type v_1 are $L_{AB}^1 = 3$, $L_{BC}^1 = 2$ and $L_{CD}^1 = 2$. The traversal times of type v_2, $L_{AB}^2 = 1$ and $L_{BC}^2 = 1$, are shorter. The graph G contains arcs allowing movement between speed profile types, but movement between types v_0 and v_2 is not possible. The thick blue line is an example $source - sink$ train path in G. It corresponds to a sequential traversal of all routes, with AB and CD traversed with speed profile type v_1, BC with type 2, and the train coming to a stop in DE.

A solution to the problem (i.e. the new schedule) comprises one $source - sink$ path in G for each controlled train $k \in \mathcal{K}$. This path completely describes the sequence of routes to be traversed by k, the time intervals in which each route traversal begins and the speed profile type of each traversal.

3.2. Speed profile types

In fixed-speed models, the traversal time of a route $r \in N_r$ is represented by a single value L_r. Trains are usually permitted to come to a stop on any route. As a result, L_r is merely the minimum traversal time because a train may spend longer than L_r in route r by stopping in it. Whilst L_r may depend on the train, route or pre-planned speed profile, it does not depend on the speed profile required to achieve the rescheduled solution. We refer to stopping (denoted by v_0) and ordinary traversal (denoted by v) as *speed profile types*. In this case, the speed profile type graph G_v contains only two nodes, with the arcs permitting the transition between these two speed profile types. Fig. 4 presents G_v for fixed-speed models. This figure visualises the fact that both stopping and ordinary traversal are possible on any route, regardless of which type occurred on the previous route.

In this paper, we propose a natural extension to this idea by defining one or more traversal times for each route. As such, the set of possible times for a train to traverse route r is denoted by $\mathcal{L}_r = \{L_r^1, L_r^2, \dots, L_r^n\}$, where $L_r^1 > L_r^2 > \cdots > L_r^n$. The traversal time that is selected from $\mathcal{L}_r$ depends on the speed profile required to perform the rescheduled solution. Replacing the single traversal time of fixed-speed models with a set of traversal times leads to a variable-speed model for the TTRP.

The speed profile type graph for variable-speed models is presented in Fig. 5. This figure shows that it is only possible for a train to transition between adjacent speed profile types for consecutive routes. Such speed profile type graphs approximate the actual speed profile types that are used by trains in railway operations. The discretisation of the speed profile types given by G_v provides flexibility in the modelling of the TTRP where different sets of speed profile types are used for each route r. The estimation of traversal times and the methods to construct the most appropriate G_v for each route r are discussed in Section 4.

4. Traversal time estimation

It is important to address the question of how to calculate the variable traversal times for each $L \in \mathcal{L}_r$. This is because the extent to which these values are representative of the actual traversal times of trains with different speed profiles is a major determinant of whether

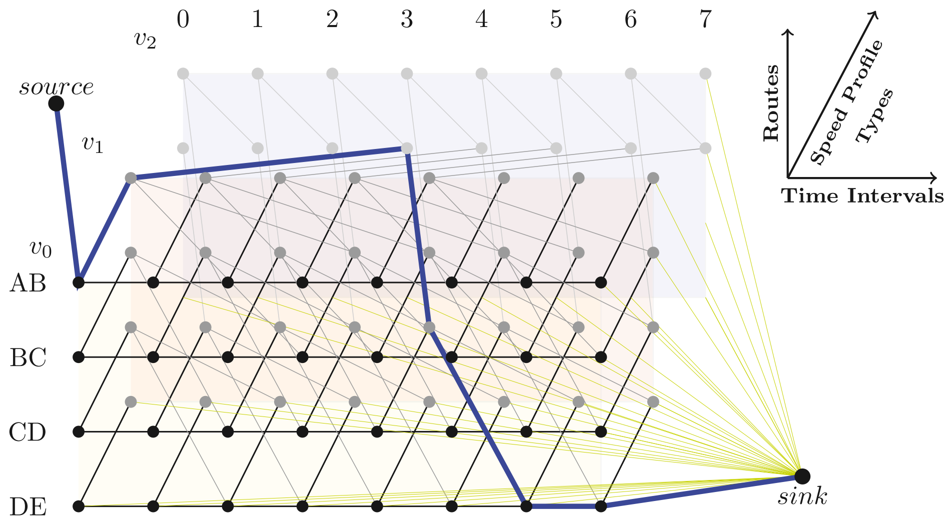


Fig. 3. An example of a time–space–type graph.

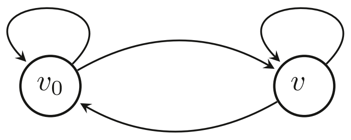


Fig. 4. Speed profile type graph for a fixed-speed model.

the solutions produced by the model are achievable in practice. The calculation of traversal times is therefore an inextricable part of the modelling methodology. In the practical application of the TTRP considered in this paper, the railway operational data indicates that at most two traversal times are required for each route. Thus, the traversal times L_r^1 and L_r^2 must be calculated for each route r.

In fixed-speed models, traversal times can be estimated using classical running time estimation techniques — see (Brünger and Dahlhaus, 2014) for a review. Stochastic running time estimation methods have also been proposed for use in timetable simulation (Yuan and Medeossi, 2014). However, the question of how to calculate traversal times for variable-speed TTRP models has not been adequately addressed. This task poses several unique challenges that straddle the topics of running time estimation and timetable rescheduling. In our case, two times L_r^1 and L_r^2 that represent trains with different speed profiles are required. These must be meaningful in the sense that it should be rare or impossible for a train to use types v_0 and v_2 in consecutive routes, but possible for any other combination to occur. An additional challenge is posed by the discrete nature of time in the model, meaning that times must correspond to a whole number of time intervals. Finally, routes in station areas are often much shorter than the distances over which running times are typically calculated.

Our approach involves estimating traversal times based on historical data. The data that is used was collected by a *Train Describer* over a seven month period. A Train Describer is a real-time information system that records the sequence of routes traversed by each train and the number of seconds spent in each one.

4.1. Method 1: Estimating a single time (fixed-speed)

First we summarise the method employed by Reynolds et al. (2020) for estimating a single traversal time L_r for a given route $r \in N_r$ in a fixed-speed model. The data used is $\mathbf{y}^r = (y_1^r, \dots y_{n_r}^r)$, a vector containing the number of seconds spent in r by n_r different trains. Unimodal statistical distributions are generally not appropriate for modelling $\mathbf{y}^r$. This is because the times arise from different processes according to the speed profile used by a train in r. By modelling each speed profile type as a different process, we can model $\mathbf{y}^r$ as a mixture of unimodal distributions. The speed profile type of each observation is unobserved, but this can be estimated by fitting a mixture model. Specifically, we fit a Gaussian Mixture Model (GMM) to each $\mathbf{y}^r$ using the Expectation-Maximisation (EM) algorithm. This is a model-based clustering technique that identifies groups of historical times (called *clusters*) that approximately follow a Gaussian distribution. An introduction to GMMs and the EM-algorithm is provided by Bouveyron et al. (2019).

To fit a GMM to $\mathbf{y}^r$, the number of clusters (each corresponding to a speed profile type) must be specified in advance. The appropriate number is different for each route. For example, on the open line where trains rarely travel below the line speed there is often only one process occurring and therefore a single cluster is appropriate. Conversely, up to three different speed profile types are discernible on many routes within stations. To decide the number of clusters, we fit three models with 1, 2 and 3 clusters respectively, and choose the model that optimises the Bayesian Information Criterion (Bouveyron et al., 2019, p. 51).

In the fixed-speed model, a single time L_r must be chosen for each route from the fitted mixture distribution. The cluster with the smallest mean is selected, since this reflects the times of trains travelling close to the speed limit. From this component, the mean is rounded up to the nearest number of whole time intervals to produce L_r. Whilst rounding introduces approximation error, rounding up ensures that the feasibility of a solution to the model is not compromised. An example of the result of this clustering process is shown in Fig. 6.

4.2. Method 2: Extension to multiple times (variable-speed)

The method presented in Section 4.2 can be extended to produce two traversal times, L_r^1 and L_r^2, for routes with three clusters. The traversal times L_r^1 and L_r^2 are calculated by taking the smallest and second smallest means of the clusters, respectively, and rounding up to the nearest whole number of time intervals. This allows more speed profile types to be represented in the model. Trains travelling close to the speed limit have a traversal time of L_r^1, whilst trains that accelerate/decelerate or coast below the speed limit have a traversal time of L_r^2. Observations in the third cluster are discarded, since these correspond to stopping, and that is not modelled using a traversal time.

Given the assumptions made about how clusters correspond to speed profile types, we expect to observe that when trains traverse two routes consecutively, the respective traversals are rarely of types

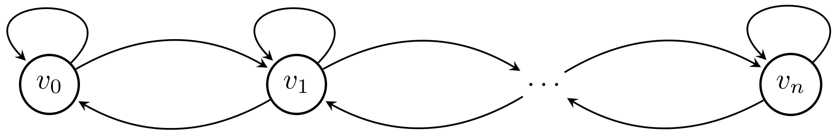


Fig. 5. A general speed profile type graph $G_v = (N_v, A_v)$ for variable-speed models.

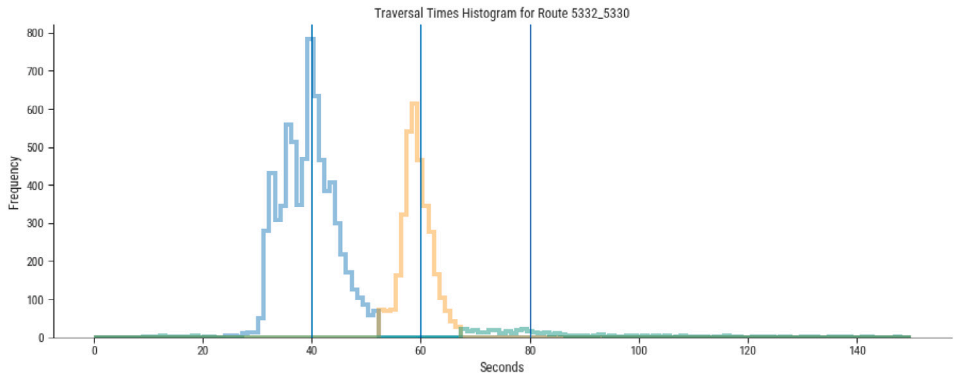


Fig. 6. The histogram of historical traversal times for route 5332_5330. The three clusters identified are shown in different colours. The rounded mean of each cluster is shown as a vertical line. The left-most vertical line (40 s, or L_r=4) is used as the final estimate of the route traversal time.

$v_0 \to v_2$ or $v_2 \to v_0$. This is because these transitions represent an abrupt change between stopping and a fast speed within the space of a single route. To test whether this is the case, a model of transitions between different clusters is created using historical data from the Train Describer. For each historical train journey j, this transition data consists of both the sequence $(r_1^j, \dots, r_{n_j}^j) \in (N_r)^{n_j}$ of routes traversed, and the corresponding sequence $(t_1^j, \dots, t_{n_j}^j) \in \mathbb{R}_+^{n_j}$ of traversal times for each route in seconds. Since the clustering process classifies each traversal time t_i^j as belonging to a particular speed profile type, the journey can also be represented as a sequence $(i_1^j, \dots, i_{n_j}^j) \in \{v_0, v_1, v_2\}^{n_j}$ of speed profile types.

In order to test the suitability of the clustering method, we model the speed profile type sequences probabilistically using a discrete Markov chain $(X_t)_{t\in\mathbb{N}}$ with state space $N_v = \{v_0, v_1, v_2\}$. This means that for each time step $t \in \mathbb{N} = \{0, 1, 2, \dots\}$, X_t is a discrete random variable taking values in N_v, such that

- **Markov Property**
  $\mathbb{P}\left(X_{t+1} = i_{t+1} | X_t = i_t, \dots, X_0 = i_0\right) = \mathbb{P}\left(X_{t+1} = i_{t+1} | X_t = i_t\right)$ for any $t \geq 1$ and $i_t \in N_v$.
  In our application, the Markov property means that the conditional probability of a train using a speed profile type, given the types used on all previous route traversals, only depends on the type used on the most recently traversed route.
- **Time homogeneous**
  $\mathbb{P}\left(X_{t+1} = j | X_t = i\right) = \mathbb{P}\left(X_t = j | X_{t-1} = i\right)$ for any $t \geq 1$ and $i, j \in N_v$.
  This means that the probability of transitioning from state i to j is independent of t, and we denote this transition probability by p_{ij}.

The transition probabilities of this Markov chain can be inferred from the data. If n_{ij} is the number of times j immediately follows i in a speed profile type sequence for a train journey in the data, then p_{ij} is estimated by

$$p_{ij} = \frac{n_{ij}}{\sum_{m\in N_v} n_{im}}. \tag{1}$$

The clustering and transition probabilities were calculated using data from Derby station, and the estimated transition probabilities are shown in Fig. 7. Whilst $p_{v_0v_2} = 0.01$ is small, which is expected, $p_{v_2v_0} = 0.21$ is higher than expected. It would be problematic to disallow

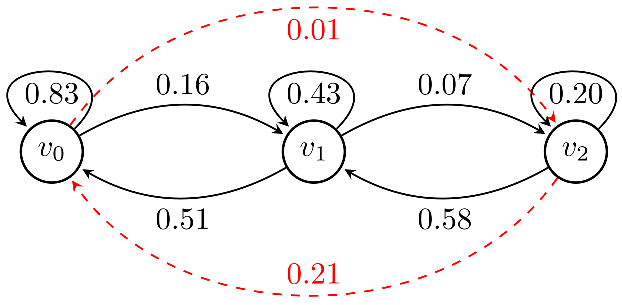


Fig. 7. The Markov chain $(X_t)_{t\in\mathbb{N}}$, with estimated transition probabilities shown. Probabilities are rounded to two decimal places.

transitions from v_2 to v_0 in our optimisation model when the historical data shows that this occurs after as many as 21% of type v_2 traversals.

The higher than expected value of $p_{v_2v_0}$ could result from inadequacies in the clustering methodology that lead to misclassification of traversal times. One particular concern is that the Gaussian distribution may not be appropriate to model clusters. Another possibility is that distinct and meaningful clusters do not exist for some routes, or do not strongly correspond to speed profile types. A third possibility is that $p_{v_2v_0}$ is a poor estimate of the true transition probability. This could have arisen because for some routes, the cluster representing type v_2 has only a small number of data points. Finally, it may be inappropriate to assume that the Markov property holds. For example, the distribution of X_t given the values at each previous time step might be dependent on the value of X_{t-2} in a way that our probabilistic model has failed to reflect.

The potential problems that have been identified may affect different route transitions to different extents. It is possible to estimate transition probabilities $p_{ij}^{r,r'}$ for each route transition $(r, r') \in N_r$ separately. The probabilities $p_{ij}^{r,r'}$ are calculated in the same way as p_{ij} using formula (1) with one difference. The difference is that n_{ij} is replaced by n_{ij}^r, the number of observations of $(r, t) \to (r', t')$ in the transition data such that t is classified as type i and t' is classified as type j. The distributions of $p_{v_0v_2}^{r,r'}$ and $p_{v_2v_0}^{r,r'}$ over all route transitions $(r, r') \in N_r$ are shown in Fig. 8. There is considerable variation across different route transitions, with $p_{v_2v_0}$ being acceptably low for some, and unacceptably high for others.

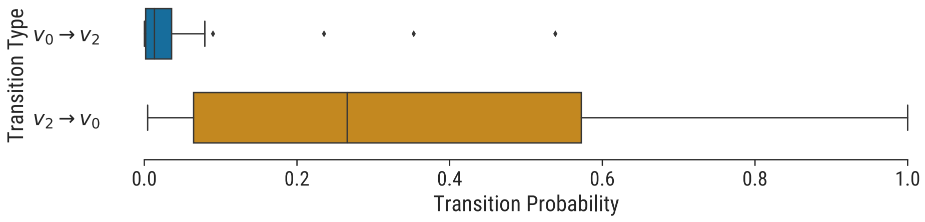


Fig. 8. Boxplots showing the distribution of transition probabilities for types $v_0 \rightarrow v_2$ and $v_2 \rightarrow v_0$.

A severe limitation of this approach is that the transition data is used only for validation and not for the clustering, which is performed separately for each route. The transition data potentially contains useful information that could be used when performing the clustering. For example, a transition from a cluster of type v_2 to a cluster of type v_0 might indicate that the second observation is misclassified and should be in cluster v_1. Transition patterns also contain useful information for selecting the number of clusters. This limitation is addressed in the next section, where the full transition data is utilised.

4.3. *Method 3: Transition-based multiple traversal time estimation*

This method for estimating differentiated traversal times is designed to address the problems identified with Method 2. It uses the full transition data to estimate the traversal times, and it does not rely on the Markov property or assume that the data follows a Gaussian distribution.

There are four main steps involved in Method 3. The first step is to identify, for each route $r \in N_r$, which observations from $\mathbf{y}^r$ arise from a train coming to a complete stop in r (type v_0). This is performed using traversal time data alone, without transition data. The distributions of times recorded for each route are very heterogeneous, which makes using standard parametric distribution fitting challenging. We therefore take an *ad hoc* approach. Any observation that is longer than 120 s, or in the top 10% of observations is assumed to arise from a train coming to a stop. These values (120 s and 10%) are selected using our familiarity with the specific area being modelled.

The second step is to classify the remainder of the traversals as either type v_1 or type v_2. This is performed using the transition data alongside our classification of type v_0 transitions from the first step. Specifically, a route traversal is classified as type v_1 if it occurs immediately before or after another route traversal that has been classified as type v_0. Conversely, route traversals that are not adjacent to traversals of type v_0 are classified as type v_2. This completes the classification of each route traversal into a speed profile type.

The third step is to calculate the traversal times L_r^1 and L_r^2 using the classification from step two. For $i = 1, 2$, the median l_r^i (in seconds) of type v_i observations is divided by 10 (the length of a time interval) and rounded up to the nearest whole number to obtain L_r^i. Medians are used because they are not unduly influenced by more extreme observations in each group. The rounding process is necessary as a result of the discretisation of time in the TST graph.

The fourth and final step is to decide, for each route, whether the data supports using two different traversal times or whether a single traversal time is more appropriate. Due to rounding, routes r for which $l_r^1 - l_r^2 < 10$ have $L_r^1 = L_r^2$, so these should have only one traversal time i.e. $N_r^v = \{v_0, v_1\}$. For other routes, we check for statistical evidence that the true values $\check{l}_r^1$ and $\check{l}_r^2$ of the medians of groups v_1 and v_2 are significantly different (l_r^1 and l_r^2 are estimates of $\check{l}_r^1$ and $\check{l}_r^2$, respectively). Mood's test for a difference in medians (Mood, 1950) is used to assess this. A one-tailed test is performed at significance level 95% to test the null hypothesis that $\check{l}_r^2 = \check{l}_r^1$ against the alternative hypothesis that $\check{l}_r^2 > \check{l}_r^1$. When the null hypothesis is rejected, both of the traversal times, L_r^1 and L_r^2, are used. For the remaining routes, only L_r^1 is used because there is a lack of statistical evidence that using a second traversal time is justified. An illustration of the method for a particular route that is given two traversal times is shown in Fig. 9. The results of carrying out this method on all routes for Derby station are described and discussed in Section 6.2.

5. Model description and solution method

The results obtained from traversal time estimation are used to determine both the TST graph, and sets of graph arcs that are used to model track capacity constraints. These objects, in turn, are used to define a Mixed Integer Programming model for the TTRP that is solved using a branch-and-price algorithm.

5.1. *TST graph*

Recall from Section 3 that $G_r = (N_r, A_r)$ is the route graph and that $G_v = (N_v, A_v)$ is the speed profile type graph. The set of discrete time intervals is denoted by $\mathcal{T}$, whilst $\mathcal{K}$ is the set of trains. For each route r, $N_r^v \subseteq N_v$ is the set of possible speed profile types for route r.

The TST graph is given by

$$G = (N_0 \cup \{source, sink\}, A),$$

where, in addition to the artificial source and sink, the nodes of G are given by

$$N_0 = \left\{(r, t, v) \in N_r \times \mathcal{T} \times N_v \, : \, v \in N_r^v\right\}.$$

The directed arc set $A = \bigcup_{i=1}^{6} A_i$ of G consists of six different arc types. These six types and their interpretations are given below. Fig. 10 shows an example arc of each different type in a small artificial example of G.

- $A_1 = \{(source, (r_0^k, a_0^k, v_0^k)) \, : \, k \in \mathcal{K}\}$
  Entering from the source node to the first known position r_0^k of train k within the time horizon and modelled area, at time interval a_0^k, and speed profile type v_0^k.
- $A_2 = \{((r, t, v_0), (r, t+1, v_0)) \, : \, (r, t, v_0) \in N_0 \text{ and } (r, t+1, v_0) \in N_0\}$
  Waiting in route r for one time interval when speed profile type is 0 (i.e. train is stopped).
- $A_3 = \{((r, t, v_0), (r, t, v_1)) \, : \, (r, t, v_0) \in N_0 \text{ and } (r, t, v_1) \in N_0\}$
  Transitioning from speed profile type 0 (stopped) to type 1 so that the train can begin traversing r again.
- $A_4 = \{((r, t, v), (r', t + L_r^v, v')) \, : \, (r, r') \in A_r, (v, v') \in A_v,$
  $(r, t, v), (r', t + L_r^v, v') \in N_0, v \neq v_0\}$
  Traversing r with speed profile type v, and arriving in a successive route r' with speed profile type v' after a traversal time of L_r^v.
- $A_5 = \{((r, T, v), sink) \, : \, r \in N_r, v \in N_v\}$
  Exiting to the sink node at the end of the time horizon.
- $A_6 = \{((r, t, v), sink) \, : \, \sigma^+(r) = \emptyset \text{ and } (r, t, v) \in N, v \neq v_0\}$
  Exiting to the sink node from a node at the boundary of the area of track modelled. The train cannot exit whilst stationary.

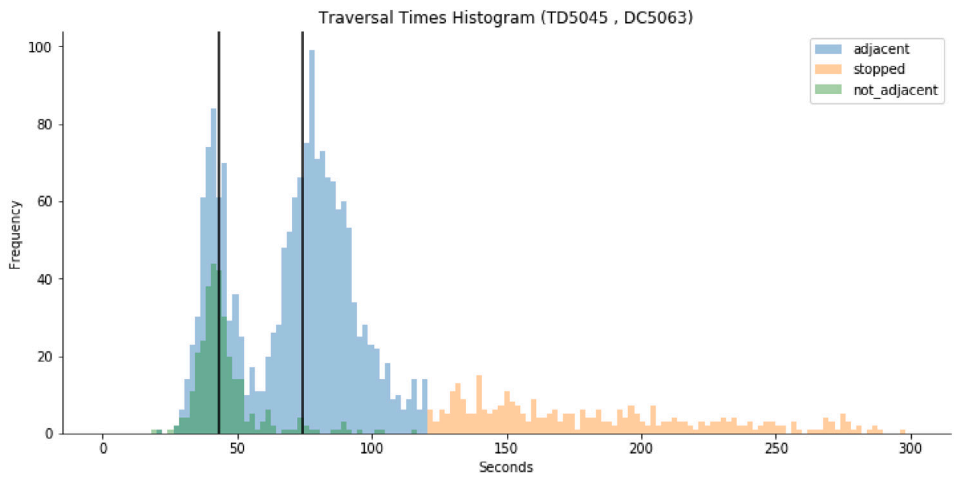


Fig. 9. A histogram of historical traversal times for a particular route. The three estimated groups are shown in different colours, and estimates for L_r^1 and L_r^2 are marked vertical lines.

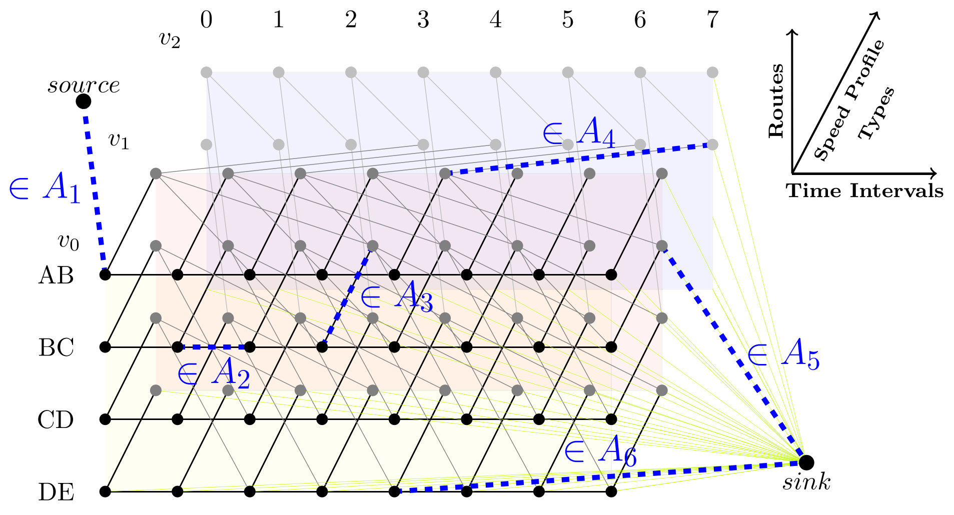


Fig. 10. An example TST graph, with examples of arcs from different sets A_1–A_6 labelled.

5.2. Track capacity constraints

A feasible solution to the TTRP is a set of *source* − *sink* paths in G, one for each train, that collectively satisfy the constraints of the signalling system. The signalling system is a safety system that limits track capacity and regulates train movements to avoid physical collisions. The type of signalling system modelled is a *sectional-release interlocking* system. This takes into account track subdivisions smaller than routes, called *track circuits*. All of the track circuits in a route are *locked* before a train may enter the route, and then *released* individually once the train has vacated each track circuit. Between locking and release, a track circuit cannot be locked by any other train. Because track circuits can form part of more than one route, the capacities of routes are not always independent.

The constraints induced by the signalling system can be modelled as capacities in the time–space–speed graph. Each pair $(r,t) \in N_r \times \mathcal{T}$ of one route and one time interval is regarded as a *time-space resource*. The capacity of these resources can be consumed by trains in two different ways: *occupying* and *banning*. Resource (r,t) is *occupied* by a train if and only if it has traversed r and at least one of the track circuits in r is still locked during time interval t. By contrast, a train *bans* a time-space resource (r',t) if it makes route r' unavailable during time interval t as a result of occupying (r,t), where r is a distinct route with track circuits in common with r'. These terms are illustrated in more detail by Reynolds et al. (2020).

The capacity of each time-space resource (r,t) is subject to two constraints:

1. (r,t) cannot be occupied more than once; and
2. (r,t) cannot be both banned and occupied.

To enforce these constraints using the capacities of arcs in G, two sets of arcs, $A_{r,t}$ and $\bar{A}_{r,t}$, are defined for each time-space resource (r,t). These are defined such that a train path contains an arc in $A_{r,t}$ if and only if it occupies (r,t), whilst a train path contains an arc in $\bar{A}_{r,t}$ if and only if it bans (r,t). Reynolds et al. (2020) show how these sets can be constructed on a time-space graph without speed profile types. We extend this to show how they can be constructed on the TST graph G. Fig. 11 visualises an example of a set $A_{r,t}$.

A train occupies (r,t) if and only if the *source* − *sink* path in G assigned to that train contains a node from the set

$$W_{r,t} = \left\{(r,t',v') : v' \in N_v^r, t' \in \{t-(L_r^{v'}+h_r)+1,\dots,t\}\right\},$$

where h_r is headway time left for the release of route r to occur. To see this, suppose that a train path contains such a node (r,t',v'). Then the track circuits of r cannot all be released until the train has traversed r, and the headway time h_r has elapsed. This cannot occur before time interval $t'+L_r^{v'}+h_r \geq t-(L_r^{v'}+h_r)+1+(L_r^{v'}+h_r) = t+1$, meaning that at least some track circuits are still locked during time interval t.

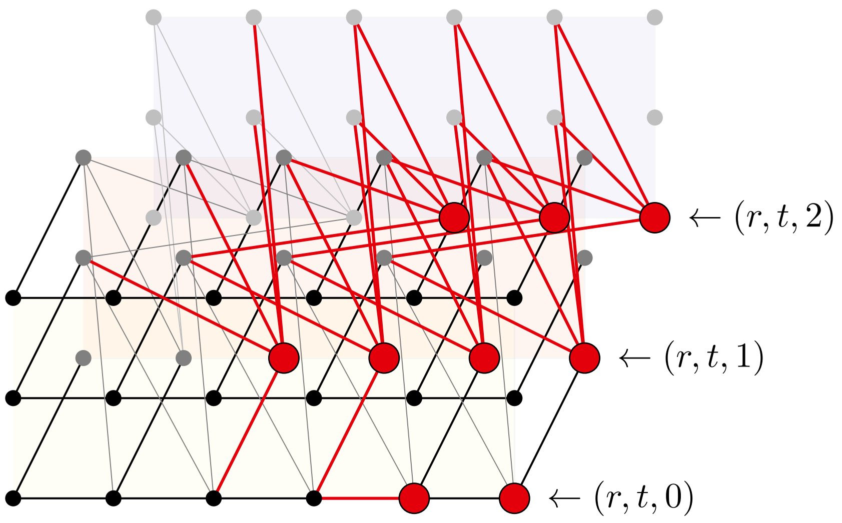


Fig. 11. An illustration of $W_{r,t}$ and $A_{r,t}$ for a specific time-space resource $(r,t) \in N_r \times \mathcal{T}$ in an example graph G. The nodes in $W_{r,t}$ and arcs in $A_{r,t}$ are highlighted in red. The relevant times are $h_r = 1$, $L_r^0 = 0$, $L_r^1 = 2$ and $L_r^2 = 1$.

A train bans (r,t) if and only if the $source-sink$ path in G assigned to that train contains a node from the set

$$\bar{W}_{r,t} = \left\{(r',t',v') : r' \in S_r, v' \in N_{v}^{r}, t' \in \{t - t(r',v',r) + 1, \dots, t\}\right\},$$

where S_r is the set of routes distinct from r that share at least one track circuit with r. The quantity $t(r',v',r)$ is the minimum number of time intervals between the track circuits of route r' being locked, and all of the track circuits common to r and r' being released, when r' is traversed using speed profile type v'. This is given by

$$t(r',v',r) = \lceil \theta(r',r) L_{r'}^{v'} + h_{r'} \rceil,$$

where $\theta(r',r)$ is the proportion of the traversal of r' after which r is released. That quantity is calculated from the track circuit data as described by Reynolds et al. (2020).

Using the definitions of $W_{r,t}$ and $\bar{W}_{r,t}$, $A_{r,t}$ and $\bar{A}_{r,t}$ can be defined as:

$$A_{r,t} = \bigcup_{n \in W_{r,t}} \sigma^-(n) \setminus \bigcup_{n \in W_{r,t}} \sigma^+(n)$$
$$\bar{A}_{r,t} = \bigcup_{n \in \bar{W}_{r,t}} \sigma^-(n) \setminus \bigcup_{n \in \bar{W}_{r,t}} \sigma^+(n),$$

where $\sigma^-(n)$ and $\sigma^+(n)$ are used to denote the set of directed arcs of G entering a node n and leaving a node n, respectively. A path in G contains an arc in $A_{r,t}$ (respectively $\bar{A}_{r,t}$) if and only if it contains a node in $W_{r,t}$ (respectively $\bar{W}_{r,t}$). Therefore a path in G contains an arc in $A_{r,t}$ (respectively $\bar{A}_{r,t}$) if and only if it occupies (respectively bans) time-space resource (r,t).

The track capacity constraints described above can be formulated in the following way. A set of $source-sink$ paths in G is a feasible solution to the problem if and only if among all of the paths:

1. At most one arc in $A_{r,t}$ is used; and
2. If an arc in $A_{r,t}$ is used then no arcs in $\bar{A}_{r,t}$ are used.

5.3. Antichain condition

Analogously to the model presented by Reynolds et al. (2020), our formulation of the track capacity constraints is correct if and only if it is not possible for a train to violate the operational constraints single-handedly. We give conditions under which this is true below.

Definition 1. An antichain in a directed graph $G = (N, A)$ is a set $Z \subseteq A$ of arcs such that for all (i,j) pairs, where $i,j \in N$, each path between i and j contains at most one arc in Z.

Definition 2. For two routes $r_1, r_2 \in N_r$, let

$$L(r_1,r_2) = \min\left\{\sum_{r' \in p \setminus r_2} \min_{v=1,2} L_{r'}^{v} : p \text{ is an } r_1\text{-}r_2 \text{ path in } G_r\right\}$$

be a lower bound on the minimum total traversal time from r_1 up to but not including r_2. For example, if $(r_1,r_2) \in A_r$ then $L(r_1,r_2) = \min\{L_{r_1}^1, L_{r_1}^2\}$.

Assumption 1. For each $(r,t) \in N_0$, $A_{r,t} \cap \bar{A}_{r,t} = \emptyset$ and $A_{r,t} \cup \bar{A}_{r,t}$ is an antichain in G.

Proposition 1. *Assumption 1 is true if for each $r \in N_r$, either*

(i) $S_r \cup \{r\}$ is an antichain in G_r, or
(ii) $L(r_1,r_2) \geq \max_v L_{r_1}^v + h_{r_1}$ for each pair $r_1, r_2 \in S_r \cup \{r\}$.

Proof. Let $(r,t) \in N_0$. Since $r \notin S_r$ by definition, $W_{r,t} \cap \bar{W}_{r,t} = \emptyset$ and hence $A_{r,t} \cap \bar{A}_{r,t} = \emptyset$.

Suppose for contradiction that at least one of (i) and (ii) holds, and that there is a path p in G that contains two distinct arcs $((r_0,t_0,w_0), (r_1,t_1,w_1))$ and $((r_2,t_2,w_2),(r_3,t_3,w_3))$ in $A_{r,t} \cup \bar{A}_{r,t}$, where without loss of generality $t_1 \leq t_3$. By the definitions of $A_{r,t}$ and $\bar{A}_{r,t}$, the routes r_1 and r_3 are in $S_r \cup \{r\}$. However, since $t_1 \leq t_3$, there must be a subpath q of p from (r_1,t_1,w_1) to (r_3,t_3,w_3), and hence a path from r_1 to r_3 in G_r, contradicting (i). Furthermore,

$$\begin{aligned} t - (L_{r_1}^{w_1} + h_{r_1}) + 1 + L(r_1,r_3) &\leq t_1 + L(r_1,r_3) && (r_1,t_1,w_1) \in W_{r,t} \cup \bar{W}_{r,t} \\ &\leq t_3 && q \text{ must span at least } L(r_1,r_3) \text{ time intervals} \\ &\leq t. && (r_3,t_3,w_3) \in W_{r,t} \cup \bar{W}_{r,t} \end{aligned}$$

This rearranges to $L(r_1,r_3) \leq L_{r_1}^{w_1} + h_{r_1} - 1$, which contradicts (ii). Therefore neither (i) nor (ii) holds, and the result is proved by contradiction. □

Note that the conditions of Proposition 1 can be checked much more easily than checking Assumption 1 directly. Condition (i) is satisfied for all routes in our instances for Derby station.

5.4. Objective function and arc weights

The quality of a given solution to the TTRP is measured as the sum of the weights of the paths in the solution. A smaller sum corresponds to a better solution, so the problem is a minimisation problem. The weight of a given path for train k is the sum of the weights c_a^k of the arcs a that make up the path. These weights depend on both the arc and the particular train, so each arc a in G has a set of weights $\{c_a^k : k \in \mathcal{K}\}$.

The weights are selected such that the negative of the sum of the weights of the paths in a solution corresponds to the utility associated with the solution by Network Rail. This utility is modelled as the total weighted utility over all trains k and scheduled events j, given by

$$U = \sum_{k\in\mathcal{K}} \alpha_k \sum_{j=1}^{J^k} \beta_k^j U_k^j.$$

Train priorities and event priorities are controlled by parameters α_k and β_k^j, respectively. The set of events J_k for each train k includes all of its station stops and its exit from the modelled area. Each event j specifies both a route r_j^k and a time interval a_j^k in which the train should arrive into the route. The quantity U_k^j is the utility accrued by train k at stop j. It is equal to zero if train k does not visit r_j at all, and $\gamma(t-a_j^k) = \phi^{-\omega|t-a_j^k|}$ if train k enters r_j in time interval t (and hence $t-a_j^k$ time intervals late).

To facilitate the representation of this objective function, the weights c_a^k take the following values:

- $c_a^k = 0$ if $a \in A_2, A_3, A_5, A_6$ for all $k \in \mathcal{K}$.
- $c_a^k = \infty\mathbb{1}_{\{j\neq k\}}$ for all $k \in \mathcal{K}$ and $a = (source, (r_0^j, a_0^j, v_0^j)) \in A_1$. This ensures that trains begin in their initial position, rather than the initial position of a different train.
- Let $j \in J^k$ be a scheduled event for train $k \in \mathcal{K}$ that requires train k to arrive into route r_j^k at time interval a_j^k. Let $t \in \mathcal{T}$ be a time interval. Then:
  - If the event requires the train to stop at a platform, then all arcs $a \in A_4$ that enter node (r_j^k, t, v_0) have weight $c_a^k = -\alpha_k\beta_k^j\gamma(t-a_j^k)$.
  - If the event doesn't require the train to stop (e.g. passing a junction), then the weight $c_a^k = -\alpha_k\beta_k^j\gamma(t-a_j^k)$ applies to arcs in $a \in A_4$ that enter any of the nodes (r_j^k, t, v) where $v \in N_v^{r_j^k}$.
  - If the event has a departure time that is later than t, then all arcs $a \in A_4$ starting at any of the nodes (r_j^k, t, v), where $v \in N_v^{r_j^k}$, have $c_a^k = \infty$. This prevents train k from leaving r_j^k for another route before the scheduled departure time.

  All other arcs $a \in A_4$ have weight $c_a^k = 0$ for each k.

Note that weights c_a^k with value ∞ ensure that an optimal solution will never contain a path for train k that contains arc a. In computations, a sufficiently large floating point value is used to represent ∞.

5.5. MIP formulation and solution

Although the model presented in this paper differs from that presented by Reynolds et al. (2020), it can be formulated as a Mixed Integer Program analogously and therefore solved using the same branch-and-price algorithm. This is because the differences between the models are expressed in the definitions of both the graph G on which the problem is defined, and the sets $A_{r,t}$ and $\bar{A}_{r,t}$.

The formulation is a path-based flow formulation with binary variables $\lambda^{k,p}$ used to indicate which $source - sink$ path $p \in P^k$ in the TST graph is selected for each train k. Thus, the formulation of the TTRP is given by

$$\min \quad \sum_{k\in\mathcal{K}}\sum_{p\in P^k}\left(\sum_{a\in A} c_a^k x_a^{k,p}\right)\lambda^{k,p} \tag{2a}$$

$$\text{s.t.} \quad \sum_{k\in\mathcal{K}}\sum_{p\in P^k}\left(\sum_{a\in A_{r,t}} x_a^{k,p} + \delta\sum_{a\in\bar{A}_{r,t}} x_a^{k,p}\right)\lambda^{k,p} \le 1 \quad \forall (r,t)\in N_r\times\mathcal{T} \tag{2b}$$

$$\sum_{p\in P^k}\lambda^{k,p} = 1 \quad \forall k\in\mathcal{K} \tag{2c}$$

$$\lambda^{k,p}\in\{0,1\} \quad \forall k\in\mathcal{K}, p\in P^k, \tag{2d}$$

where $\delta > 0$ is a small positive constant, and each $x_a^{k,p}$ is a constant that is equal to 1 if path $p \in P^k$ for train k contains arc $a \in A$, and equal to 0 otherwise.

The objective (2a) is to minimise the total weight of all of the selected train paths. This corresponds to maximising Network Rail's utility. Constraint (2b) ensures that the track capacity constraints outlined in Section 5.2 are respected. For example, if a time-space resource (r,t) is occupied twice, then the left hand side of (2b) is 2, violating the constraint. Similarly, if (r,t) is both occupied and banned, then the left hand side of (2b) is $1+\delta$, which also violates the constraint. However, the constant δ is given a small value so that the resource can be banned multiple times without violating the constraint. In our instances, a value of $\delta = 0.05$ was sufficient to ensure that there was no practical constraint on the number of times a resource could be banned. Constraint (2c) ensures that exactly one path is selected for each train. Finally, constraint (2d) states that the variables are all binary.

This problem can be solved using the branch-and-price algorithm described by Reynolds et al. (2020). Column generation is used to solve the LP relaxation at each node of the branch-and-bound tree. Variables are generated in each column generation iteration from the solution of one subproblem for each train. These subproblems are shortest-path problems on G, which is a directed acyclic graph. Both partial pricing and reduced cost variable fixing are used as column generation acceleration strategies. A customised branching rule is used that branches on conflicts between pairs of trains over resources (r,t) in the track capacity constraints.

5.6. Relationship between fixed-speed and variable-speed models

The variable-speed model for the TTRP presented in this paper is an extension of the fixed-speed model proposed by Reynolds et al. (2020). These two models are distinguished by the method used to calculate the traversal times — discussed in Section 4. Specifically, the two models are considered in this paper are:

(FS) The fixed-speed model proposed by Reynolds et al. (2020), with traversal times calculated using Method 1 (see Section 4.1).
(VS) The variable-speed model proposed in this paper, with traversal times calculated using the transition-based Method 3 (see Section 4.3).

The relationship between (FS) and (VS) is dependent on the traversal times of the corresponding routes in each model. Since the traversal times used in (FS) are less than or equal to the smallest traversal time in (VS) for all routes, (FS) is a relaxation of (VS). This is because all solutions for (FS) can be translated into a solution for (VS) by making trains wait in berths for an amount of time equal to the difference between the (FS) and (VS) traversal times.

Consider the variable-speed example given in Section 3.1. The fixed-speed version of this example has only a single traversal time for each route, which are the minimum traversal times given by $L_{AB} = 1$, $L_{BC} = 1$, and $L_{CD} = 2$. The route depicted in Fig. 3, traversing AB, BC, CD and then stopping in DE, takes a total of 6 time steps in the variable-speed model. The equivalent route in the fixed-speed model would take 4 time steps, without intermediate stopping. However, with intermediate stopping, the variable-speed path could be achieved by the fixed-speed model by traversing AB in 1 time step, waiting for 2 time steps before traversing BC in 1 time step, traversing CD in 2 time steps, and finally stopping in DE. The resulting path has a total length of 6 time steps, equal to the path found in the variable-speed model.

Since (FS) is relaxation of (VS) the expected utility of the former is at least as good as that achievable by the latter. However, (FS) can deliver

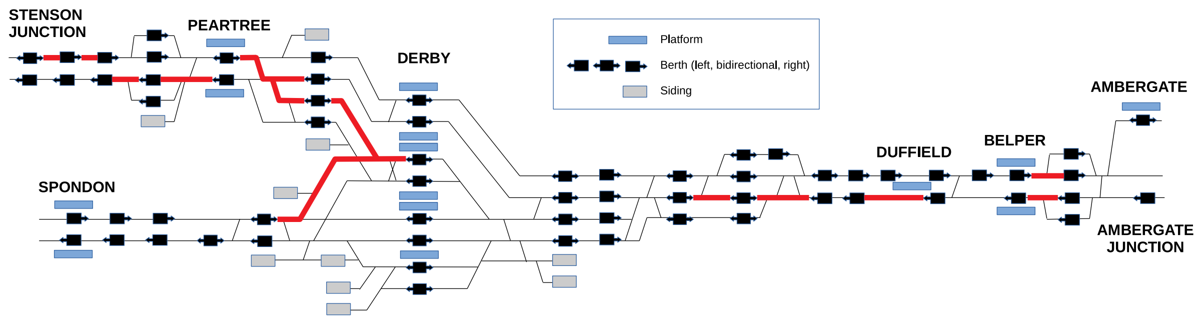


Fig. 12. A berth diagram of the modelled area (own image). Routes with multiple traversal times in (VS) are highlighted with thick red lines (see Section 6.2).

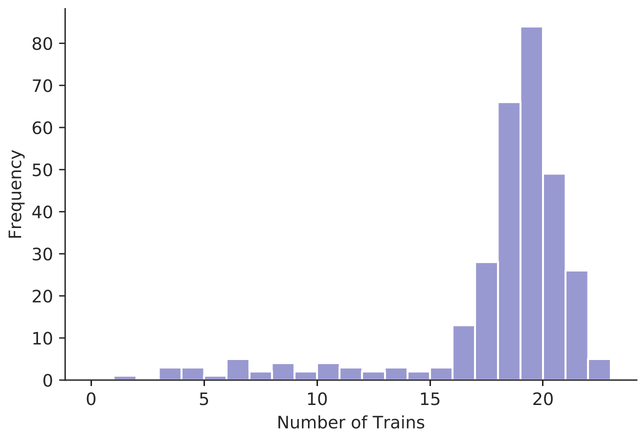


Fig. 13. A histogram showing the distribution of the number of trains in each instance.

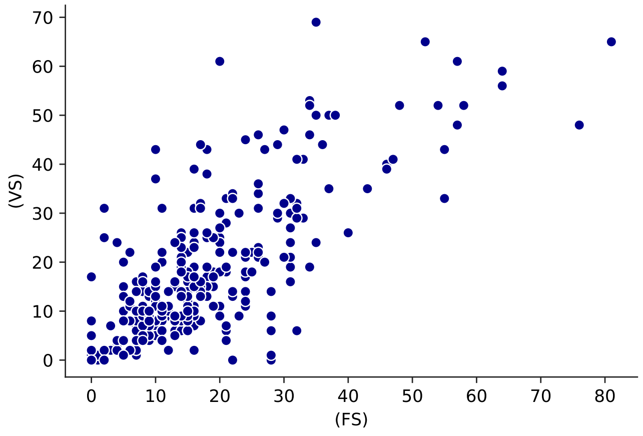


Fig. 14. A comparison between (FS) and (VS) of number of conflicts.

solutions that are infeasible in practice. This is due to sudden changes in speed that result from traversing a route and then stopping in the subsequent route without any deceleration being prescribed by the model. The increased granularity of traversal times introduced by (VS) aims to address the potential infeasibility of TTRP solutions that arises from the use of single traversal times for each route. The improved modelling of traversal times in (VS), while imposing greater restrictions on the solution of the TTRP, is expected to produce solutions that will achieve a better realised utility than solutions to (FS).

6. Computational study

A computational study has been carried out to compare (FS) and (VS). The two goals of this computational study are to (i) determine where greater accuracy in the traversal times is required and (ii) evaluate the computational cost of this increased accuracy. The first goal is a major contribution of this paper, where the traversal time estimation technique presented in Section 4.3 is used to determine locations in the railway infrastructure that require more than one traversal time. This process aims to improves the accuracy of the TTRP only for the parts of the network where it is most necessary. The second goal evaluates the impact of this increased granularity of traversal times on the computational performance of the branch-and-price algorithm when solving the TTRP. It is expected that the time required to solve (VS) will be greater than that required to solve (FS). This computational study will evaluate the extent to which the times for solving the TTRP will increase as a result of the improved modelling accuracy. Since the improvement in realised utility that can be achieved by (VS) over (FS) can only be observed in real-world railway operations or microscopic simulations, such analysis is out of scope for this paper.

Traversal time estimation methods were implemented in the Python 3.6 language, using the package Scikit-learn 0.20.3 (Pedregosa et al., 2011) for fitting Gaussian mixture models, and the package SciPy 1.2.1 (Virtanen, 2020) for performing Mood's test. The branch-and-price algorithm is implemented using SCIP 6.0.2 (Gleixner et al., 2018) as a branch-and-price framework. Custom plugins for SCIP are written in the C language, and Gurobi 9.0 (Gurobi Optimization LLC, 2020) is used as the linear programming solver. The experiments were carried out on a computing node equipped with an 18 core Intel Xeon E5-2699 v3 CPU with 2.30 GHz and 500 GB of RAM, running Ubuntu 16.04.

Section 6.1 describes the new set of instances for Derby station in the UK that are used for our computational study. Section 6.2 compares the traversal times produced by the different estimation methods. Finally, Section 6.3 evaluates the performance of the branch-and-price solution algorithm for the two models. Full tables of results are available online (see Reynolds, 2020).

6.1. Instance data

A new set of 310 instances has been created for the computational study. These instances are based on real data from an area of railway centred around Derby station in the UK. Derby station lies on both the Midland Main Line and the Cross Country Route, two heavily used, double-track, inter-city lines connecting London with Leeds, and Bristol with York via Birmingham, respectively. The station also hosts local services to Nottingham and Matlock. In 2018–2019, Derby station had 3,902,000 passenger entries and exits and 619,000 passenger interchanges. Lying at the confluence of several lines, Derby Station is regarded as a traffic bottleneck. This makes it suitable for testing our model.

The area modelled is shown in Fig. 12. Derby station has 6 bidirectional platforms and 204 track circuits. The area as a whole contains portions of three double track lines, and consists of 142 routes in total, with 228 valid berth transitions, which correspond, respectively, to the

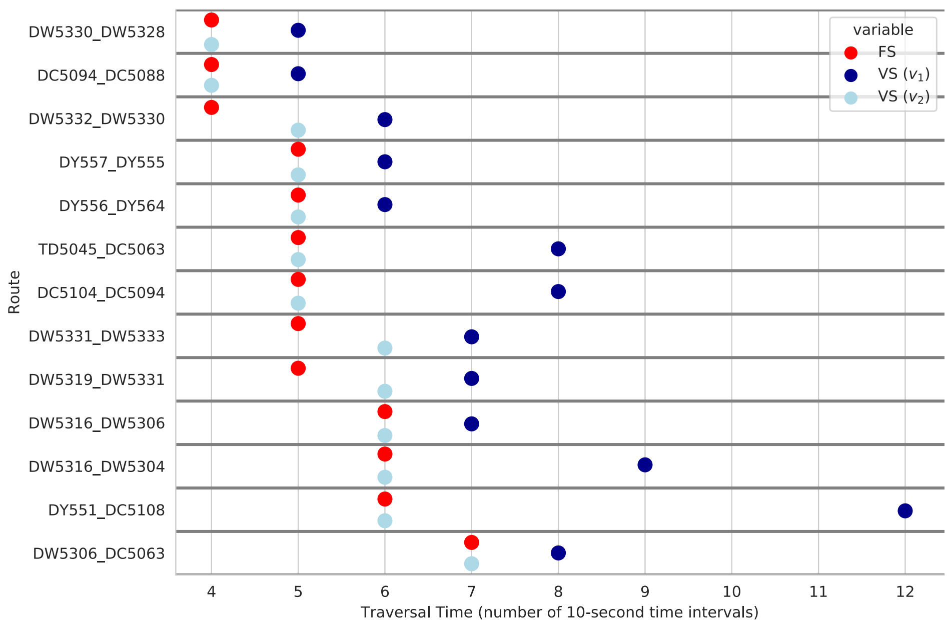


Fig. 15. A comparison between (FS) and (VS) of traversal time lengths for routes that have two traversal times in (VS).

number of nodes and edges in the route graph. The area under consideration includes five small stations in addition to Derby: Spondon, Peartree, Duffield, Belper and Ambergate.

Each one of the 310 instances covers a different hour long period. Ten instances, covering between 8 am and 6 pm, are used from each of the 31 days in January 2020. They were created using real timetables, and real data about traffic perturbations. The level of traffic perturbation varies considerably between the instances. However, they provide a representative sample of traffic conditions at Derby over January 2020.

The number of trains in each instance is shown as a histogram in Fig. 13. The most common number of trains is 19, whilst the largest number is 23. The number of *conflicts* in each instance for both (FS) and (VS) is shown in Fig. 14. This is defined as the number of track capacity constraints that are violated by the solution obtained from solving each model without any track capacity constraints. The number of conflicts in a given instance can be different for models (FS) and (VS) because the differing traversal times result in trains reaching different parts of the track at different times. Nevertheless, we see that there is a strong positive relationship because although the models and traversal times differ, the TTRP instances used are identical. Table 1 compares the number of *train pair conflicts* in the instances when using (FS) and (VS). This is the number of unique train pairs involved in at least one conflict together. Measuring train pair conflicts can be more informative than measuring conflicts. This is because each train pair can only have at most one train pair conflict, where several conflicts can occur for the same train pair in consecutive time intervals over the same route. Both the number of conflicts and the number of train pair conflicts have been shown by Reynolds et al. (2020) to be correlated with the number of branch-and-bound nodes required to solve instances to optimality. Discrepancies between (FS) and (VS) can therefore cause differences in the performance of the solution algorithm.

6.2. Traversal time estimation

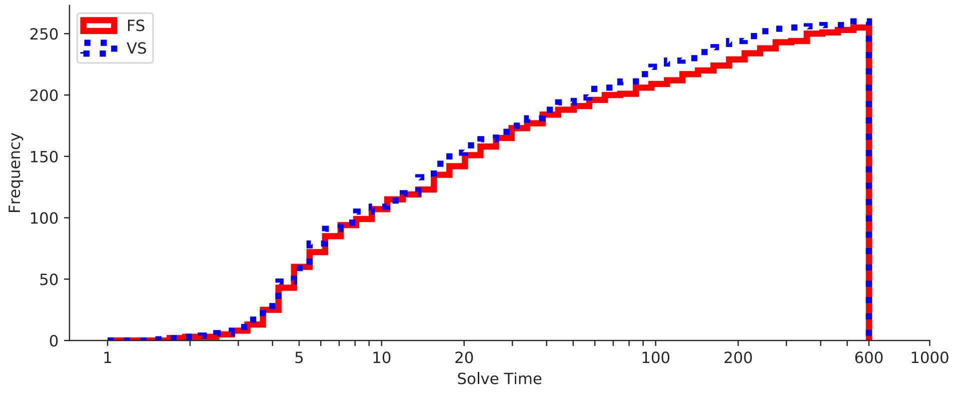
Historical Train Describer data was used to estimate traversal times for routes in Derby station using Method 3 (see Section 4.3). The application of Method 3 identified that 15 out of the 140 routes required two different traversal times for (VS). For these 15 routes, sufficient evidence was found for a difference of at least one time interval between the traversal times of speed profile types v_1 and v_2 — sufficient evidence was not found for the remainder of the routes. This shows that our statistical method requires a high standard of evidence for using two traversal times for a route. The effect of this is that nodes of speed profile type v_2 in the TST graph are included sparingly, resulting in a parsimonious optimisation model. Including two traversal times for a route introduces additional complexity and increases the size of the model, so it is important to do this only where it can be shown statistically to be most important.

Table 1
A comparison between (FS) and (VS) of train pair conflicts. Values are frequencies of instances.

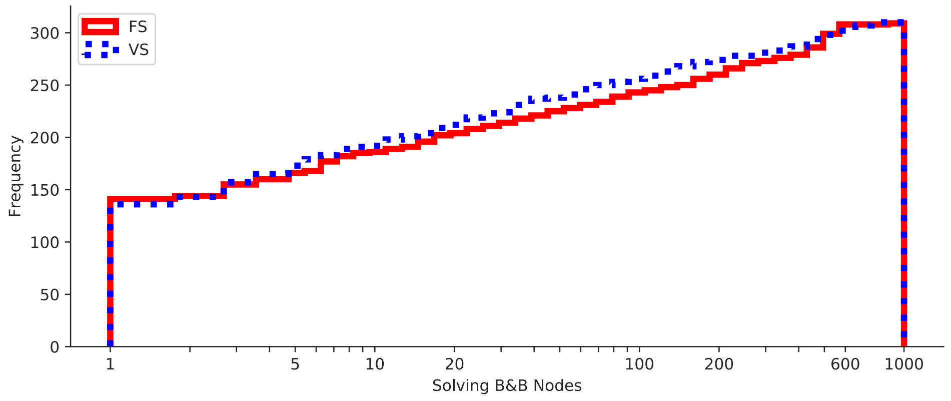
		(VS)								
		0	1	2	3	4	5	6	7	8
	0	16	6	1	0	0	0	0	0	0
	1	2	17	10	4	1	0	0	0	0
	2	1	10	40	26	6	3	0	0	0
(FS)	3	1	5	22	33	18	9	0	1	0
	4	0	0	8	10	14	9	1	1	0
	5	0	1	0	1	4	6	7	2	0
	6	0	0	0	0	2	2	4	0	1
	7	0	0	0	0	1	0	1	2	1
	8	0	0	0	0	0	0	0	0	0

The routes with two traversal times are highlighted in red in Fig. 12. Some of these routes are adjacent to routes that frequently have conflicts, such as the routes entering platform 3 (the third platform from the top in Fig. 12) at Derby station. This is likely to arise from the fact that some trains stop in preceding routes as a result of conflicts, whilst others do not — affecting the speed at which they traverse the route. Other routes with two traversal times are adjacent to stations at which only some trains stop, such as Peartree, Duffield and Belper. Trains that stop at these stations must decelerate or accelerate through adjacent routes, whilst trains that do not can continue at the line speed. These patterns conform to our expectations and therefore give us confidence that the traversal time estimation method is performing well.

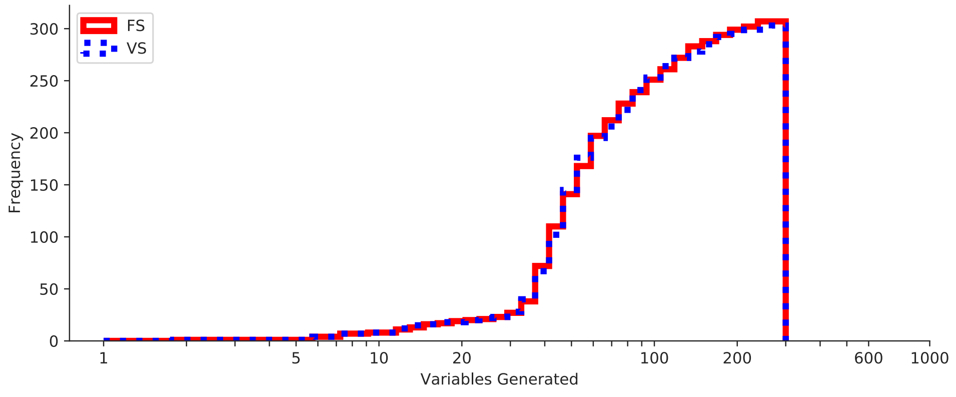
The traversal times of routes that have three speed profile types (i.e. two traversal times) in the (VS) model (using Method 3) are shown in Fig. 15 alongside the corresponding traversal times for (FS) (which uses Method 1). It shows that traversal times in (FS) are often similar to or the same as for traversal type v_2 for (VS). This is because the smallest cluster mean for each route is used as the traversal time in (FS),



(a) Solution time (seconds)



(b) Number of branch-and-bound nodes



(c) Number of variables generated

Fig. 16. Cumulative histograms comparing (FS) and (VS) over the set of instances with a time limit of 600 s. Note the x-axis is log-scaled.

and the observations in that cluster overlap strongly with observations that are categorised as type v_2 in (VS). Whilst the difference between traversal times for v_1 and v_2 in (VS) is not large for most routes, even a single time interval (10 s) difference can be enough to affect the optimal rescheduled timetable. An exception to this is the traversal times $L_r^1 = 12$ and $L_r^2 = 6$ for route $r =$ DY551_DC5108, which differ by a whole minute. This is a relatively long route on the open line in which trains stopping at Duffield station and trains not stopping there are likely to be travelling at very different speeds.

6.3. Algorithmic performance

There are many aspect to the solution algorithm that will be assessed in the computational experiments. The main goal of this section is to demonstrate that the improved accuracy of (VS) over (FS) comes at little to no computational cost.

6.3.1. Size of MIP formulations

While the (FS) and (VS) have very similar integer programming formulations, the multiple traversal times on a subset of routes introduces additional constraints to the latter problem. As a result, (FS) and (VS) comprise 139,682 and 144,362 constraints respectively. For both (FS) and (VS), the number of constraints correspond to the number of nodes in the underlying time-space and time–space–type graph, respectively. The additional 4680 constraints (nodes) arise from the 15 routes for which there is sufficient evidence to require more than one traversal time. This small increase in the model size, while gaining improved

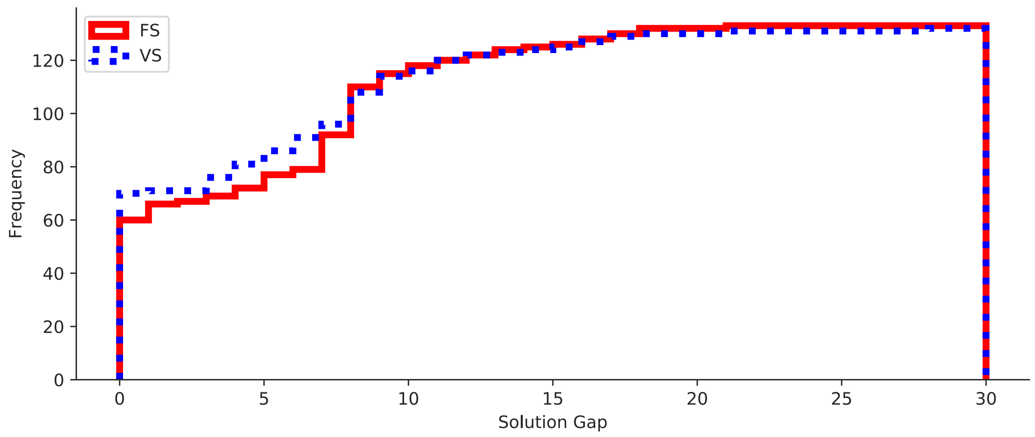


Fig. 17. Cumulative histogram of optimality gaps after 20 s for (FS) and (VS). Note the x-axis is log-scaled.

accuracy, is a significant benefit of the data-driven modelling approach proposed in this paper.

Since the TTRP, for both (FS) and (VS), is solved by branch-and-price, the number of variables is dependent on the algorithmic performance. The number of variables generated while solving the TTRP will be discussed in the following section.

6.3.2. *Evaluating the branch-and-price algorithm*

To understand how our branch-and-price algorithm performs when solving both models (FS) and (VS), each of the real instances from Derby station were solved with a time limit of 600 s. Within this time limit, 282 of the 310 instances were solved to optimality with (FS), and 287 were solved to optimality with (VS). This is a good result because it means that all but the most difficult instances can be solved to optimality with the more accurate (VS). The solving times of the instances that could be solved to optimality are shown in Fig. 16(a) (note that the x-axis is logarithmic with base 2). This figure demonstrates that the distribution of solving times over the instance set are very similar for (FS) and (VS). However, the higher blue line indicates that (VS) has a better solving performance than (FS) among models that took over 100 s to solve. Specifically, the average solving times for all solved instances are 43.67 and 53.69 s for (VS) and (FS) respectively.

The explanation for the better performance of (VS) on difficult instances can be seen by studying Figs. 16(b) and 16(c). These show the number of branch-and-bound nodes explored, and the number of variables (columns) generated, respectively, during the solving process. Fig. 16(c) shows that the number of columns generated during the complete branch-and-price algorithm was almost identical. In particular, the average number of columns generated for (FS) and (VS) is 74.4 and 77.8 respectively. This indicates that convergence of the column generation algorithm for solving LP relaxations was not affected by differences between (FS) and (VS). On the other hand, Fig. 16(b) shows that there are more instances with a smaller number of branch-and-bound nodes for (VS) compared to (FS). This is particularly evident for instances requiring between 20 and 600 nodes. Further, it is observed that across the complete set of instances (FS) required an average of 92.5 branch-and-bound nodes compared to an average of 76.04 for (VS). This difference in the number of branch-and-bound nodes is the likely to be the reason for the better solution times for (VS) on difficult instances. It is conjectured that this result arises from the difference in the quantity and quality of conflicts due to using more accurate traversal times in (VS) compared to (FS).

6.3.3. *Solving with restricted time limits*

Both of the models were also evaluated using a time limit of 20 s. This reflects the short amount of time that is typically available for solving TTRP instances in practical, real-time environments. Of the 310 instances, 174 were solved to optimality by both models within this time limit. The optimality gaps for the remaining 136 instances are plotted in Fig. 17. It shows that with both (FS) and (VS), high-quality solutions that are provably within 10% of optimality were found for the vast majority of instances. For each, only two instances had remaining optimality gaps exceeding 20%. This reinforces the evidence found by Reynolds et al. (2020), using instances for a different station, that the developed algorithm is suitable for real-time operations. Fig. 17 also shows that whilst the distribution of remaining optimality gaps is similar for (FS) and (VS), (VS) has a larger concentration of instances with very small gaps. This corroborates our observation of better performance for (VS) when the time limit was 600 s.

Based on the results presented, we conclude that the benefits of variable speed modelling in (VS) compared with fixed speed modelling in (FS) come at no price for computational performance. This striking result highlights the value of applying a data-driven approach for improving the accuracy of the TTRP.

7. Conclusion

In this paper, we present a new variable-speed model for the TTRP that uses a time–space–type graph to approximate train speed profiles. To achieve this, the notion of a discrete speed profile type is introduced, and techniques for estimating traversal times based on historical Train Describer data are developed.

Our approach is tested using a new set of real instances for Derby station in the UK. These tests show that our data-driven approach results in a parsimonious model that is able to improve speed profile modelling relative to fixed-speed models in parts of the track network. In addition, these modelling enhancements come at no cost, in the sense that they do not lead to longer solving times in comparison with the fixed-speed model of Reynolds et al. (2020). This represents a major contribution to modelling approaches for the variable speed TTRP.

Further work is needed to quantify the inaccuracies that still remain in speed profile modelling. For example, a simulation study using a microscopic railway simulator could be used to evaluate solutions produced by our model and compare them to solutions produced by different variable speed models. Our data-driven modelling approach also creates exciting opportunities to improve the statistical methodology for estimating traversal times based on speed profile types. One possibility is to use a Hidden Markov Model (see Zucchini et al., 2016 for an introduction), in which speed profile types are the ‘hidden’ states, and traversal times are the observed data. This could open up the possibility of fitting mixture models for each route (as in Method 2) whilst additionally using the full transition data.

CRediT authorship contribution statement

Edwin Reynolds: Conceptualisation, Methodology, Software, Formal analysis, Writing – original draft, Visualisation. **Stephen J. Maher:** Writing – review & editing, Supervision.

Acknowledgements

SJM was supported by the Engineering and Physical Sciences Research Council (EPSRC), UK [EP/P003060/1].

References

Bešinović, N., Quaglietta, E., Goverde, R.M., 2013. A simulation-based optimization approach for the calibration of dynamic train speed profiles. J. Rail Transp. Plan. Manage. 3 (4), 126–136.

Bouveyron, C., Celeux, G., Murphy, T.B., Raftery, A.E., 2019. Model-based clustering and classification for data science: With applications in R. In: Model-Based Clustering and Classification for Data Science. Cambridge University Press, pp. 15–75.

Brünger, O., Dahlhaus, E., 2014. Running time estimation. In: Hansen, I.A., Pachl, J. (Eds.), Railway Timetabling & Operations, second ed. Eurail Press, pp. 65–90.

Cacchiani, V., Huisman, D., Kidd, M.P., Kroon, L.G., Toth, P., Veelenturf, L.P., Wagenaar, J.C., 2014. An overview of recovery models and algorithms for real-time railway rescheduling. Transp. Res. B 63, 15–37.

Caimi, G., Fuchsberger, M., Laumanns, M., Lüthi, M., 2012. A model predictive control approach for discrete-time rescheduling in complex central railway station areas. Comput. Oper. Res. 39 (11), 2578–2593.

Cordeau, J.-F., Toth, P., Vigo, D., 1998. A survey of optimization models for train routing and scheduling. Transp. Sci. 32 (4), 380–404.

Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2009. Evaluation of green wave policy in real-time railway traffic management. Transp. Res. C 17 (6), 607–616.

Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2010. Centralized versus distributed systems to reschedule trains in two dispatching areas. Public Transp. 2 (3), 219–247.

Corman, F., Meng, L., 2015. A review of online dynamic models and algorithms for railway traffic control. IEEE Trans. Intell. Transp. Syst. 16 (3), 1274–1284.

D'Ariano, A., Pacciarelli, D., Pranzo, M., 2007a. A branch and bound algorithm for scheduling trains in a railway network. European J. Oper. Res. 183 (2), 643–657.

D'Ariano, A., Pranzo, M., Hansen, I.A., 2007b. Conflict resolution and train speed coordination for solving real-time timetable perturbations. IEEE Trans. Intell. Transp. Syst. 8 (2), 208–222.

Fang, W., Yang, S., Yao, X., 2015. A survey on problem models and solution approaches to rescheduling in railway networks. IEEE Trans. Intell. Transp. Syst. 16 (6), 2997–3016.

Gleixner, A., Maher, S.J., Fischer, T., Gally, T., Gamrath, G., Gottwald, R.L., Hendel, G., Koch, T., Lübbecke, M.E., Miltenberger, M., Müller, B., Pfetsch, M.E., Puchert, C., Rehfeldt, D., Schenker, S., Schwarz, R., Serrano, F., Shinano, Y., Weninger, D., Witt, J.T., Witzig, J., 2018. The SCIP Optimization Suite 6.0. Technical Report 18-26, Zuse Institute Berlin.

Gurobi Optimization LLC, 2020. Gurobi optimizer reference manual. http://www.gurobi.com.

Hosteins, P., Pellegrini, P., Rodriguez, J., 2019. Studies on the validity of the fixed-speed approximation for the real time Railway Traffic Management Problem. In: 8th International Conference on Railway Operations Modelling and Analysis - RailNorrköping 2019. pp. 409–424.

Lamorgese, L., Mannino, C., Pacciarelli, D., Krasemann, J.T., 2018. Train dispatching. In: Borndörfer, R., Klug, T., Lamorgese, L., Mannino, C., Reuther, M., Schlechte, T. (Eds.), Handbook of Optimization in the Railway Industry. Springer, pp. 265–283.

Lamorgese, L., Mannino, C., Piacentini, M., 2016. Optimal train dispatching by Benders'-like reformulation. Transp. Sci. 50 (3), 910–925.

Luan, X., Wang, Y., De Schutter, B., Meng, L., Lodewijks, G., Corman, F., 2018a. Integration of real-time traffic management and train control for rail networks - part 1: Optimization problems and solution approaches. Transp. Res. B 115, 41–71.

Luan, X., Wang, Y., De Schutter, B., Meng, L., Lodewijks, G., Corman, F., 2018b. Integration of real-time traffic management and train control for rail networks - part 2: Extensions towards energy-efficient train operations. Transp. Res. B 115, 72–94.

Lusby, R.M., Larsen, J., Ehrgott, M., Ryan, D.M., 2013. A set packing inspired method for real-time junction train routing. Comput. Oper. Res. 40 (3), 713–724.

Mazzarello, M., Ottaviani, E., 2007. A traffic management system for real-time traffic optimisation in railways. Transp. Res. B 41 (2), 246–274.

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

Mood, A.M., 1950. Introduction to the Theory of Statistics. McGraw-Hill, pp. 394–399.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830.

Pellegrini, P., Marlière, G., Rodriguez, J., 2014. Optimal train routing and scheduling for managing traffic perturbations in complex junctions. Transp. Res. B 59, 58–80.

Reynolds, E., 2020. TTRP full results. www.github.com/edwin6/TTRP_Results.

Reynolds, E., Ehrgott, M., Maher, S.J., Patman, A., Wang, J.Y.T., 2020. A multicommodity flow model for rerouting and retiming trains in real-time to reduce reactionary delay in complex station areas. (submitted for publication).

Rodriguez, J., 2007. A constraint programming model for real-time train scheduling at junctions. Transp. Res. B 41 (2), 231–245.

Scheepmaker, G.M., Goverde, R.M., Kroon, L.G., 2017. Review of energy-efficient train control and timetabling. European J. Oper. Res. 257 (2), 355–376.

Virtanen, P., 2020. Scipy 1.0: Fundamental algorithms for scientific computing in python. Nature Methods 17, 261–272.

Xu, P., Corman, F., Peng, Q., Luan, X., 2017. A train rescheduling model integrating speed management during disruptions of high-speed traffic under a quasi-moving block system. Transp. Res. B 104, 638–666.

Yang, X., Li, X., Ning, B., Tang, T., 2016. A survey on energy-efficient train operation for urban rail transit. IEEE Trans. Intell. Transp. Syst. 17 (1), 2–13.

Yin, J., Tang, T., Yang, L., Xun, J., Huang, Y., Gao, Z., 2017. Research and development of automatic train operation for railway transportation systems: A survey. Transp. Res. C 85, 548–572.

Yuan, J., Medeossi, G., 2014. Statistical analysis of train delays and movements. In: Hansen, I., Pachl, J. (Eds.), Railway Timetabling & Operations, second ed. Eurail Press, pp. 217–236.

Zhou, L., Tong, L.C., Chen, J., Tang, J., Zhou, X., 2017. Joint optimization of high-speed train timetables and speed profiles: A unified modeling approach using space-time-speed grid networks. Transp. Res. B 97, 157–181.

Zucchini, W., MacDonald, I.L., Langrock, R., 2016. Hidden Markov Models for Time Series, second ed. CRC Press.