EasyChair Preprint

№ 10898

Network-Wide Mixed-Rail Traffic Scheduler: Challenges and Implementation Aspects

Sidhartha I Kumar, Karim Shahbaz, Mitul Tyagi, Samay P. Singh, Satwik V. Ramisetty, Sudarshan Pulapadi, Madhu N. Belur, Narayan Rangaraj, Merajus Salekin and Raja Gopalakrishnan

September 13, 2023

World Conference on Transport Research - WCTR 2023 Montreal 17-21 July 2023

Network-wide mixed-rail traffic scheduler: challenges and implementation aspects

Sidhartha I Kumar[a], Karim Shahbaz[a,], Mitul Tyagi[a], Samay P. Singh[a], Satwik V. Ramisetty[a], Sudarshan Pulapadi[a], Madhu N. Belur[a], Narayan Rangaraj[b], Merajus Salekin[c], Raja Gopalakrishnan[c]

[a]*Department of Electrical Engineering, Indian Institute of Technology Bombay, India*

[b]*Department of Industrial Engineering and Operation Research, Indian Institute of Technology Bombay, India*

[c]*Centre for Railway Information Systems, New Delhi, India*

Abstract

In 2020, Indian Railways took up a de novo timetabling exercise for the Golden Quadrilateral and Diagonals (GQD) connecting Mumbai-Delhi-Kolkata-Chennai, involving about 1700 trains on a weekly basis over a network of 9,099 km. Based on significant participation in this activity, this paper elaborates on the challenges faced in a network-wide simulation and scheduling of trains across a large network. The methods to address these challenges and the software implementation aspects are also described. A network-wide mixed-rail traffic simulator was used successfully to construct a daily timetable for the GQD. The two primary goals of the Zero Based Timetabling (ZBTT) project were to generate train schedules that were: (a) feasible (in terms of kinematic constraints on both trains running and block-section, running lines) and (b) conflict-free (due to multiple trains having to share resources and the need to ensure time-durations of resource usage are different, subject to safety constraints), for a proposed set of departure timings and halt patterns of all the services that were planned on the GQD.

Keywords: Mixed rail traffic; scheduler; simulator.

1. Introduction

Indian Railways (IR) operates over 13,100 passenger and 8,400 freight trains daily in a network spanning 67,956 km (6). For administrative reasons, the entire network is governed by segregating it into 17 zones, which are further divided into 70 divisions.

E-mail address: karimshahee@ee.iitb.ac.in

A passenger train consists of a set of coaches, called a rake. Each rake is maintained at a home depot in one of the zonal railways. The rake-owning zonal railway proposes a tentative departure time at the originating station and the route for the train for which a timetable is to be drawn. Passenger timetables on different sections of the network are prepared sequentially by the zonal railways governing the section. The return service is planned by the originating zone at the other end. This process is iterative as the halt pattern and speed characteristics of the train would require possession of the network resources at certain times. However, these resources may not be available because of existing train services, maintenance or freight corridors planned on those sections. The train time-table has arrived through multiple iterations between the zonal railways and concluded at a centralized coordination conference with all the zonal railways.

This method has been fairly effective over the years, with one caveat - allowances. Allowances are additional traversal times given to trains in a zonal region to account for stochastic delays. They are employed near interchange points between divisions/zones. Allowances provided for most new trains were more than the suggested norms (except for a few high-profile trains) and this did lead to a slack utilization of resources, especially near major junctions on the network. Every once in a while (at least once in a decade if not more frequently), there is a specific need to work out timings *de novo*, as there are significant changes in technology - both in rolling stock and fixed infrastructure - and also priorities and requirements of the timetable. Such an exercise cannot be done effectively by an incremental and sequential approach and needs a system-wide algorithmic approach. The Zero Based Timetabling (ZBTT) activity is one such event, where a fresh set of timings were generated on a significant part of the IR passenger traffic network: Golden Quadrilateral and Diagonals (GQD) connecting Delhi, Howrah (Kolkata), Chennai, and Mumbai. It is widely accepted that focusing on a speed-up on this sub-network using a systematic tool would allow for eventual speed-up, albeit manually, of the rest of the network too. To summarize, a *de novo* timetabling process is akin to starting from a 'clean slate', and hence also called 'Zero Base TimeTabling' (ZBTT): since this is not incremental, but starting from the zero-base. See more in (1) for benefits achieved using this tool: this paper focusses on the software changes to achieve this.

Apart from providing the shortest effective connection between the four major metros, the GQD forms the spine of high-speed services between many pairs of cities in India. Our simulation is confined to trains that use one or more resources from the GQD network. This was acceptable because the GQD is generally acknowledged to be a bottleneck in the IR network. However, a few trains in the simulation also traverse the non-GQD portions of the network. Such trains are handled in one of the two alternative ways - for high-priority trains by fixing their path on the GQD timetable, and for others by simulation along with the other trains using the GQD network.

This paper deals with the challenges faced in such a large-scale ZBTT, with a focus on the network-wide aspect of the scheduling. The challenges are summarized below and revisited later in Section 4. This paper dwells on the software implementation aspects of how these challenges were overcome in achieving the ZBTT exercise.

1.1. Scheduler vs simulator

In this paper, we use scheduler and simulator interchangeably, and we elaborate on the reasons why the software that this paper focuses on has important features of both: scheduling and simulation. The output of the software is a timetable in which different trains avail and share infrastructural resources at different times and this involves systematic priority-based scheduling. The priority aspects in the sequencing and the concerns about the deadlock in a scheduling procedure are elaborated below in Section 2.2. The scheduling of resources in a station and within a block section between two stations involves a meticulous calculation of time durations to arrive based on multi-aspect signalling (green, yellow, double-yellow and red), and this involves a significant amount of back-calculation of when (and where) the deceleration ought to begin. These aspects of the software are typical of a simulation of a train-running on a block section. Further, the back-calculation of time duration (and distance location) also depends on which running line in the station is used for passing through or stopping; this crucially depends on the maximum entry speeds of the running lines: typically different for the 'main-line' and a 'loop-line'. This aspect of the software also makes the 'simulator' an apt description. This is elaborated below in Section 2. These aspects, together with others that we do not elaborate on here, are reasons why do we use scheduler and simulator interchangeably in this paper?

1.2. Challenges of a network simulator: a summary

In this section, we summarize the key challenges that arise when scheduling on a network in comparison to a linear rail section simulation. The IR network deals with mixed-rail traffic wherein passenger, and freight trains have differing priorities. Further, there are multiple categories of passenger trains with their own set of priorities. Therefore, overtaking of a train by another is inevitable. This paper focuses on the passenger train timetable and excludes the scheduling of freight trains. However, as an offshoot, a benefit of the so-called 'compaction' achieved due to the ZBTT exercise are longer and better-quality freight time-windows on the GQD network.

We summarize some challenges arising from scheduling a large number of trains in a network. The simulator must have the capability to handle both a linear and/or a network with the same consistent definitions of inputs and outputs. The concept of the 'next' station is central to our simulation to compute arrival times. However, at junction stations, the search space to determine the arrival time is significantly larger at the 'next' station in a network. In contrast to simulation in a linear section, the simulator must be able to resolve cycles, handle route reversals, and also decide, in real-time, from the potentially multiple routes to a station. A 'route-resolver' is necessary to address the ambiguities that may arise during a network-wide simulation. Further, while simulating multiple routes on the GQD, the overlapping segments must be handled simultaneously, an aspect that is well-studied in route relay locking challenges (7; 10). The tools of speed-distance graphs that are effective in visualizing simulation outputs - velocity profile, overtakes, or allowances - do not support the requirements for a network.

When the number of trains in a simulation increases, so does the possibility of 'deadlock', a problem well-studied in literature. A policy of sequencing the multiple trains is essential to avoid deadlocks. Further, train journeys in a large network tend to be long, with a few trains taking over 48 hours to reach their destination. In such situations, the number of trains that must be handled simultaneously increases 5-fold, thereby increasing the computational burden.

Further, issues of data quality and data integration surface while simulating operations on a network. The multiple zero mile-posts, from where distances to a station are computed, need to be reconciled when two or more routes merge. In addition, many network resources (lines in a block section or loop lines at a station) have availability only for a specific direction(s) of traffic. Data quality in the accuracy of the infrastructure information becomes critical in resolving potential deadlocks. Some of these challenges are revisited in Section 4.

1.3. Separation of simulator software and case-specific data

This paper also deals with the software implementation of the solutions we propose for addressing the above challenges. One of the important features that increases the usability of software is a certain separation between the tool itself and the input/output data that the tool processes. This is elaborated further below. Our tool takes note of this feature and has this important separation.

It is essential that software takes in data (both infrastructural and train kinematic parameters) in a format such that a user need not understand the internal workings of the software; in fact, a user need not know the programming language of the software (in our case: Java), but the user only ought to be able to edit/modify/add the infrastructural and train-parameters in a format that is accessible through a spreadsheet tool, like LibreOffice-Calc, or MS-Excel, or Google-Sheets. This is achieved in our implementation in which the inputs and outputs of the scheduler (infrastructural/train-running parameters, start timings) are spreadsheet editable/readable, and so is the generated timetable.

Another important benefit of this separation is that the input/output data almost always would need significant reformatting for the purpose of integration with a railway-centric-database. The need to integrate with such a database is inevitable given that the tool's success is directly linked with the overall ability to work on real input data fetched from a railway-centric database and the ability to later push the tool's output back into this database. Separation of the tool from the input/output data is central to this end-user database-specific reformatting of the tool's input/output.

1.4. Organization of this paper and summary of contribution

The rest of this paper is organized as follows. The next section elaborates on the typical difficulties that are faced in a *single* long section and focus on the kinematic aspects that arise in a scheduling process. Section 3 describes the so-called Golden

Quadrilateral and its Diagonals (GQD) sub-network of the IR Network and its key features. These features were addressed by the simulator, and hence we take this example as a case study to convey the key challenges/contributions described in this paper. Section 4 is where the challenges in a typical network scheduler (summarized above) are addressed through various solutions. This section also contains the software implementation of the proposed solution. In Section 5, we explain the need to simulate 'linked' trains; these are trains where one train's simulation affects the start time of the next train for a variety of reasons, for example, reversal, or route-change, or leaving the GQD sub-network and rejoining elsewhere in the GQD sub-network. A few concluding remarks and items of future work are summarized in Section 6. The rest of this section dwells on the contribution in this paper.

In this group, we had previously developed a single-section scheduler (as described in the following section): however, significant changes had to be made in the scheduler to incorporate the network aspect in the ZBTT requirements. This paper focusses on the software challenges and solutions implemented to overcome the challenges. In particular, the challenges summarized in the last two sections (i.e. Sections 1.2 and 1.3) are addressed and this is elaborated below.

2. Single section scheduler: mixed-rail traffic

As mentioned above in Section 1.1, the generation of a timetable for even a single linear section involves two important aspects: scheduling of resources across the multiple trains that share/avail these resources, albeit at different time instants. Unlike typical suburban/metro trains, long-distance trains on a section are not a homogeneous set of vehicles but have different kinematic capabilities (speed, acceleration, deceleration) and, in fact, different priorities also. Thus shorter-distance so-called passenger trains are scheduled such that the long-distance 'express' trains' schedules are not affected due to planned overtakes of the lower-priority trains by higher-priority trains. This overtake is scheduled to happen at a station where there are running lines (usually with lower max-speed capabilities) where the lower priority train is scheduled to arrive early for getting overtaken by the higher priority train. These aspects are typical of any mixed-rail traffic simulation; freight trains (often without a schedule) are usually the lowest on the priority list.

It is very useful to classify resources into up/down/common for the benefit of railway personnel to give inputs/use the output, understand and modify the inputs appropriately. In fact, up/down directions is a fairly universal 'language' across the world railway communities, though it is also acknowledged that a pure scheduling procedure does not require the up/down language.

2.1. Kinematic constraints

In this section, we describe the various kinematic constraints in the scheduling of trains. The timetable is a compilation of the outcome of both

1. simulation of a single train, and
2. scheduling of a group of trains with network resource constraints

The type of coaches and rake composition determines train characteristics such as maximum speed, deceleration, acceleration, and length. We assume a constant value for acceleration and deceleration. The kinematics of infrastructure elements are more complex. It consists of physical limits of maximum permissible speeds for different block sections and the state of the section itself, determined largely by the signalling system. Multi-aspect signalling system dynamically indicates the number of segments of the track ahead that is free for possession by the train that is being simulated. The aspect of the signal determines the precise location and time when a train must start braking. Thus, the train schedule in a block-section is influenced by limits imposed by the rolling stock, track, and signalling.

In addition, the schedule of a train in a station depends on the type of lines that are utilized. A train passing through a mainline does not encounter any curves and, therefore, has higher speed limits in comparison to when utilizing a loop line. The allocation of the type of station resource (mainline or loop line) determines the signal aspect and also the train speed at the time of entry into it. This requirement is catered to by the simulator through a rule-based allocation among the available resources at the station. Further, a back-calculation may be required to compute the location and time when the train must start decelerating. This back-calculation is also relevant for two additional infrastructure characteristics:

1. Gradients
2. Permanent speed restrictions.

2.1.1. Gradients

While negotiating a rising gradient, the train weight and the hauling capacity of the locomotive restrict the acceleration that is feasible. The location and length of the gradient and also the train length influence the traversal time.

2.1.2. Permanent Speed Restriction (PSR)

In IR terminology, a PSR is a length of track with a lower speed limit when compared with the maximum speed for the section. The PSR could be the result of restrictions posed by either the track structure or geometry. The speed limit is applicable when the entire or a part of the train is traversing the PSR. Based on the length of the PSR and train length, a back-calculation is needed to determine the location and time when the deceleration must begin, thereby increasing the traversal time. The effect of the PSR further depends on the proximity to locations where speeds are restricted - such as halts, track segments such as other PSRs, or even the state of the section ahead. A blocked section ahead restricts the train speed, and correspondingly, the impact of the PSR on the increase in traversal time would be lower when compared to the situation when the section ahead is free. These aspects of the kinematic properties of the trains and infrastructural resources are relevant for both a single-section and a network-wide simulation.

2.2. Simulation sequence

In any large-scale scheduling and resource allocation amongst multiple entities (here, trains), a concern of deadlock needs to be addressed. Deadlock avoidance is a standard problem studied in many areas of computer-based resource allocation and scheduling; see (10), (11), (12). In this section, we elaborate on the simulation logic: the same logic is also used for the network-scheduler.

The simulator is currently a serial one, both for a linear section and for a network; trains are ordered as one list of trains-to-be-simulated, and paths are found and allocated one by one. Below is the sorting scheme:

1. First trains are sorted priority-wise: higher-priority trains get allocated resources (i.e. get scheduled, get timetabled) first.
2. Then the next-priority trains are scheduled, and so on.
3. Freight trains are usually scheduled last in the gaps made available between coaching trains.
4. Between trains of the *same* priority, trains starting *earlier* are scheduled first.
5. Ties (if both priority and start-time are the same) are resolved by the mere sequence of occurrence in the list of trains to be scheduled. If the proposed start time is the same for two same-priority trains proposed to start at different stations, then the sequence of scheduling them is quite unlikely to affect the two trains' schedules.

2.3. Object-oriented implementation of blocks/running-lines, trains

The simulation of features described above in a single-section simulation needs a meticulous record of the infrastructural resources and of the trains and their characteristics. This is implemented in the simulator using an object-oriented programming method in which infrastructural resources (blocks, loops) are of one class, in which the different objects are linked to one another at both station end-points into multiple running lines using the up/down/common classification of these infrastructural resources. The linking of the resources uses the individual objects' lengths and the multi-aspect signalling (and the train lengths) in a way that the zero-milestone, albeit common in a single-section simulation, does not play a central role. This non-centrality of the zero-milestone in the linking and simulation is crucial when extending the single-section

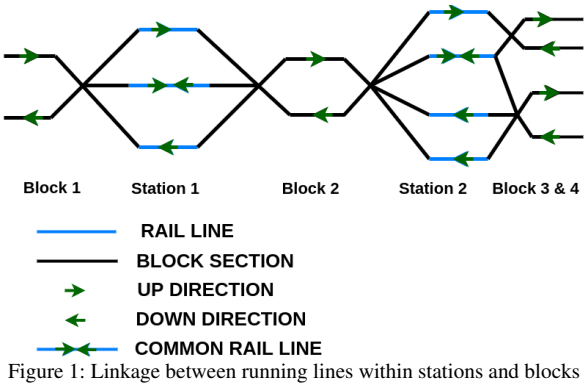

Figure 1: Linkage between running lines within stations and blocks

simulation features to a network-wide simulator. The network-wide aspect is pursued further in Section 4. In this section, we describe further the object-oriented implementation aspects.

A train's route is made up of block sections (often multiple blocks between a pair of stations) and loop lines (holding/stopping lines with a railway station). By definition, trains are not supposed to stop in blocks, and they may or may not halt in loop-lines.

Blocks and loops are objects with attributes of start-milepost, end-milepost and length (calculated from the difference). Both: blocks and loops have directions assigned: up/down/both, and each of these has a head and a tail: the blocks are linked using links, and each link has a head and a tail. Thus blocks are linked using links, and the directionality plays a role in the linking to form a route.

Blocks have different attributes: start (from a specific zero-milepost), block-end-km (from the same specific zero-milepost). And up/down attributes. The next few block linkages are important for calculating the velocity profile: each block length is all that is used for the velocity profile (of new few blocks), and the linking of these blocks/loops do not need the exact distance from the zero-milepost. This is pursued further when we describe the implementation of the network-aspect of the scheduler. Figure 1 shows linkages between block-sections (between stations) and running-lines (within stations) for a single-section.

3. Case study: the Golden Quadrilateral and diagonals: Indian Railways

In this section, we describe a concrete example: the case of the Golden-Quadrilateral and its diagonals (GQD); this is an important sub-network of the IR, and using this as a case study helps us elaborate on the challenges faced in a typical network-scheduler. The tool and methodology are not GQD specific but are valid for any general railway network. Successful implementation of the tool helped in getting a *de novo* timetable for this sub-network based on only the proposed start-timings of various trains.

A schematic of the GQD railway sub-network is shown[1] in Figure 2. Some of the key features of the GQD are listed below.

1. At each of the four nodes and at the diagonal intersection, instead of merely *intersecting*, there are about 50 km stretches of resource *overlap*. In other words, trains going on different routes need to be scheduled together and thus calling for a combined network-wide simulation.
2. Majority of the trains to be scheduled remain on just one route, thus often needing a single-route simulation also. Ideally, the same tool should be able to handle both.

[1] Though an up/down the classification of resources might be inessential from a scheduling viewpoint, and classification might also be potentially infeasible in a consistent way, we report the up/down direction that was used in the simulation, and this does not necessarily coincide with the IR's directional nomenclature.

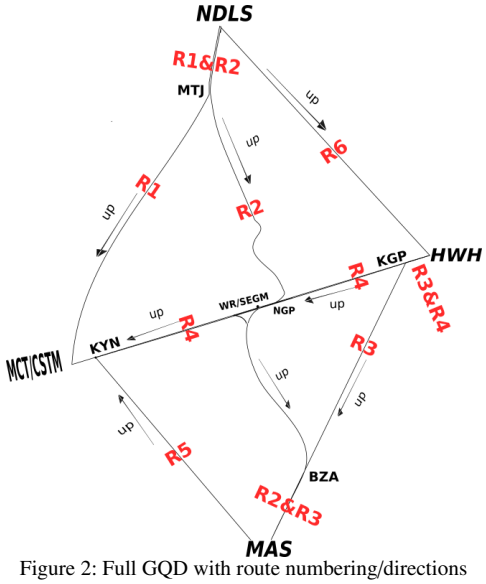

Figure 2: Full GQD with route numbering/directions

3. In a large network, it is unreasonable to have to model exhaustively, but preferable to model only the congested or high-volume sub-network. In Figure 2, it was decided to *not* model many sections that connect points on this network, though there are trains that use a significant part of the GQD and these non-modelled routes. The scheduler then needs to 'link' the simulation of train-journey portions; this is elaborated in Section 5.1.

4. Another typical feature is that some trains often *reverse* at a junction and continue along a different route. Here too, different sub-routes need to be simulated in a linked fashion, and further, keeping note of the up/down reversal of the trains. This is elaborated in Section 5.2

5. For a single-route simulation, certain bottle-neck stations often have a 'bypass': see Figure 4, where a majority of the trains go via the important VSKP station, and a small number of 'superfast' trains do not go within this bottleneck station. To avoid complex up/down renaming of the overlapping/reversing portions, the common stations were renamed for the scheduling purpose and re-modified for the central railway database integration. This Railway database integration centrally has been elaborated in Section 1.3.

4. Network-wide scheduler

In this section, we pursue the main contribution of this paper. We dwell on the implementation aspects that arise in resolving the challenges faced in the development of a network-wide simulator. We first touch upon an important analogy in the area of transportation networks in which items originate from a source and reach a destination, and items are usually sent as fast as possible. There are many other graph-network applications where analysis of different routes from a source to a destination is relevant: see (8), for example.

4.1. Internet traffic router: an analogy

The multi-route railway infrastructure network can be compared with a computer network (for example, the Internet or an Intranet), where a packet represents a train with an analogous source and destination. In order to route packets to the correct destination in a computer network, the packets are transferred on a hop-to-hop basis, and each node is responsible for making "local" decisions for each packet, depending upon the destination. This local decision is stored as a per-node look-up table and appropriate meta-data from the packet is used to query this table.

In the train-simulation case, the current node id (i.e. the block/loop number) is used as the meta-data. An entry (a, b) in a train-specific lookup table is interpreted as follows: if the train has reached node a in the simulation, then the next allowable

track segment is b. That is, only links connecting to b are considered for further simulation. It is worth mentioning that even though a is used as the query key for the lookup table, multiple entries with a as the key are allowed. This just means that if the train is currently in node a, then the next allowable track segment can be one of $b[i]$'s where $(a, b[1])$, $(a, b[2])$, and so on, are the entries in the lookup table with a as their key. Note that slight differences in these linkings are merely different lines/platforms at intermediate stations/blocks and not a different routes.

Unlike the Internet network, our simulator needs to unambiguously determine the route for each train, and hence it is up to the simulator's user to specify the look-up table data, which is detailed enough to resolve all the decision points, for example, a 'fork' in the network, on the path of the train. However, the simulator can work with ambiguous route-resolvers as well, meaning that if there are multiple possible paths even after considering the route-resolver entries (provided these paths lead to the specified destination of the train), the simulator selects a path depending upon the priority of the links at the diverging node. This inherent choice in the path can allow the simulator to avoid congestion and use other options possible due to this ambiguity.

4.2. Route-resolver: multiple routes for OD pair

In this section, we elaborate on how it is essential to include a 'route-resolver' in the train's input data. Consider Figure 3, where a sub-network of the GQD is shown. Assume we are simulating a train from HWH to MAS. Clearly, there are two routes: one via KGP-NGP-SVGM-BZA, and another via VSKP-BZA. While for the internet, only the origin and destination are relevant, for a train, it is essential that the simulator does not suggest the next destination through a different route. Further, the search for the next route is also significantly minimized if a route-resolver is encoded into the train's input data. Here, the input data of a train would typically contain the list of stations and the halt durations. The simulator's search space also decreases drastically due to the inclusion of the route-resolver.

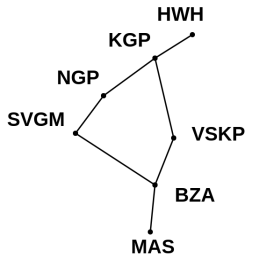

Figure 3: Multiple paths between an OD pair: route-resolver

4.3. Implementation in the simulator

The simulator was designed for linear infrastructure as per the brief description of the logic (specific to traversal over the track segments) in Section 2.2. From the logic, it is evident that modelling a non-linear network is possible with this version of the simulator as it relies on the linking described in the input files, and the linking can be specified such that the resources form a network as opposed to a section. However, this does not guarantee that a train will run on the desired path, as there may be points on the path of the train where there are multiple choices for the next track segment. In such a case, the old simulator will just loop over the available links and pick the first free link. To make the train path deterministic and exactly follow the desired path, the next track segment to be traversed at these decision points needed to be specified to the simulator.

To reduce search space, whenever a route diverges into multiple possibilities (at junctions, due to network aspect), then the next link is also explicitly specified. This speeds up the simulation and ensures wrong route to the final destination is not taken.

Train path router implementation - For each train entry in the input file, a field was added that contains the route map - a mapping between track elements which can be described as follows. If the route map contains a key-value pair B1:B2, and the train reaches B1, then the train is allowed to go over B2 only. In other words, given that the train is currently on B1, then the train can only go to B2 even though there are other routes in the network that are connected beyond from B1. Inside the BlockScheduler.traverseBlock() method, the simulator, without the route-resolver feature, would have explored all links

available for a particular block/loop. The route-resolver feature in the simulator uses the train route map data to selectively explore the links: if the current block is present as a key in the route map, then consider the corresponding value link.

While the proposed mechanism of specifying the path router demands significant analysis from the user's side, it has many advantages. Since the mechanism allows the user to direct a train through a possibly complicated nonlinear infrastructure at a very low level of abstraction, complex network structures can be implemented. The GQD network (see Figure 2) is an example.

5. Connected trains

In this section, we elaborate on another key feature of the network scheduler; the linking of trains. Often a train's long journey is comprised of multiple routes and also maybe contain a reversal at a junction. While in practice, the train continues to be called up or down; it is too complicated and cumbersome to model this entire journey as one simulation. In such a situation, we break the long journey as sub-journeys and hence different simulations which are 'linked': these are linked in the sense that the second subjourney start time depends on the first journey's end-time by a duration that is specified for that train explicitly. This is elaborated further using concrete examples.

Note that the obvious linking of a departure event at a station with the arrival at that station (so that the halt duration is as specified) is the relevant item: this obvious linking arises in a single-section simulation too. The item in this section regards reversals or route changes, where one subjourney of a train (perhaps initially in the up direction) continues into a second subjourney of the train (which perhaps is along a different route and/or in the down direction). These simulations are implemented as single-subjourney simulations in which the second subjourney's start-time is linked to the end-timing of the first subjourney's simulation, with a suitable halt duration.

The network simulation considers this aspect as two separate trains as follows. The end-timing of one train (which is an outcome of one simulation) triggers the start timing of the next train: either immediately (like just the halt duration after reversal), or after a pre-specified larger duration (e.g. due to the crossing of a non-modelled section and starting from elsewhere in the network). This latter possibility is elaborated on below.

5.1. Linked trains in a partially modelled network

When performing a *de novo* timetabling exercise on a sub-network of a larger rail network, it is essential to restrict the focus on the sub-network that is both high-volume and bottleneck. But there are some trains that could leave this sub-network and rejoin back at a different point. It might not be reasonable to model this intermediate (hitherto unmodelled) part because of the possibly low volume of rail traffic and, further, this section not being a bottleneck.

In such a case, one triggers the latter part of the journey (where the train 'rejoins') according to the end-timing of the first part of the train's journey. The difference between the end-timing of the first to the start-timing of the next is the amount of time that the train needs to traverse the unmodelled part of the railway network. This time difference is specified manually and is not obtained by the simulation exercise due to the non-modelling of this part.

We explain this using the example of Figure 4. The part of the rail network between DVD and SCM is very sparsely used because VSKP is an important passenger station, and most coaching trains go via VSKP, in spite of the significant excess time due to going into a dead-end and reversing back. However, a small number of long-distance express trains and all freight trains use the DVD-SCM link, and for this small number of trains using the DVD-SCM link, it is needless to model this part. In such a case, we model the simulation of a train via the DVD towards SCM as two trains in which the end-time at DVD triggers a start-time at SCM for the ongoing part: with an appropriate and manually determined traversal duration from DVD to SCM.

This is a simple example, and there are long stretches of unmodelled parts, though in a network simulation, it is inevitable that the modelling and simulation effort has to have a scope, and there needs to be a systematic way to address trains that traverse parts outside the simulation scope.

Figure 4 is included for another purpose: repetition of stations in a train's journey, which is typical during a reversal. The station VSKP in Figure 4 happens to be a dead-end, and the complexity arising in part near VSKP is not due to a reversal

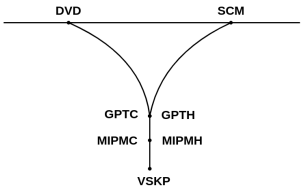

Figure 4: Connecting/Linking trains

of each train, but the fact that stations GPT and MIPM are visited twice for each train: once while approaching VSKP and once while departing VSKP. For modelling ease, and due to the up/down pre-classification of block sections and running-line resources, it was now imperative to duplicate each of stations GPT and MIPM into GPTC, GPTH and MIPMC, MIPMH, respectively (with -C and -H indicate the Chennai/DVD side and Howrah/SCM side, respectively). Thus trains from DVD reversing at VSKP will have the sequence of stations DVD-GPT-MIPM-VSKP-MIPM-GPT-SCM, which in our simulation is represented as a train following the sequence DVD-GPTC-MIPMC-VSKP followed by a train following sequence VSKP-MIPMH-GPTH-SCM with the appropriate linkage of timings. The headway constraints on the common section of the track are also modelled through suitable quantities in the simulation.

5.2. Train-reversals: up/down change

As explained earlier, to eliminate the possibility of deadlock in the scheduling procedure, it helps to have a clear up/down the classification of rail network resources: block sections, station running lines, etc. Accordingly, train simulation too is for trains that are either up or down. However, when a train reverses at a junction, that simulation would have to be two distinct simulations (one of them being up, the other being down), which are carefully 'linked' together again by the fact that the end-timing of the first simulation is used to trigger the start-timing of the second simulation.

6. Concluding remarks

This paper described a method to have a network-wide scheduler/ simulation for a mixed-rail traffic scenario. We elaborated on the various challenges that are faced in such a network-wide simulation, some of the key challenges being as follows. It is essential to be able to use the same infrastructure data in a hybrid mode: both for just a single route simulation and for network-wide simulation. Having multiple zero-mile posts is inevitable in such a network, and further, typical trains that need scheduling would involve reversals, change across routes, and discontinuities in the train route (due to the inevitable non-exhaustive modelling of any network).

Using the Golden Quadrilateral and Diagonal (GQD) sub-network of the Indian Railways as a case study, we described how these challenges were successfully overcome, and we also elaborated on the software implementation aspects. The main operational consequence was that multiple iterations could be tried on the entire network in one go, and a final timetable satisfying all local constraints could be arrived at. In the whole exercise, multiple timetables were generated over a three-month period, with extensive consultation between the 12 zonal railways and the coordination of the Railway Board. Each new iteration took merely a few hours to execute: this is a drastic improvement to the time needed to reconcile across different railway divisions/zones. The few hours required are mainly due to pre-processing the inputs carefully from various sources and providing useful reports for the team. In future, as these requirements are understood, this may become faster. The actual simulation took about 30 minutes of computational time.

The benefits of the tool have been elaborated in (1) from a railways' end-user viewpoint: both qualitatively and quantitatively, since the focus on this paper is more on the software tools/changes/modifications done to achieve this.

Future work in this area would involve a systematic optimization of the timetable with respect to various performance metrics like wide freight corridors, high average speeds across all trains and other similar passenger oriented performance measures.

References

[1] K.P. Anoop, A. Madhukumar Reddy, M.S. Bhatia, A.K. Jain, R. Gopalakrishnan, Merajus Salekin, Samay Pritam Singh, R.V. Satwik, Sudarshan Pulapadi, C.S. Bobade, I.S. Kumar, S. Gopalakrishnan, Kumar Appaiah, Narayan Rangaraj, Madhu N. Belur, Rationalized Timetabling Using a simulation tool: A Paradigm Shift in Indian Railways, *INFORMS Journal on Applied Analytics*, 2023. https://doi.org/10.1287/inte.2023.1158

[2] R. Borndörfer, T. Klug, T. Schlechte, A. Fügenschuh, T. Schang and H. Schülldorf, 2016. The freight train routing problem for congested railway networks with mixed traffic, *Transportation Science*, vol. 50, num. 2, pages 408-423.

[3] S. Dutta, N. Rangaraj, M.N. Belur, S. Dangayach and K.N. Singh, 2017. Construction of periodic timetables on a suburban rail network-case study from Mumbai, *Proceedings of the International Conference on Railway Transport Technology*, Lille, France.

[4] I. Felko, 2011. Simulation-based deadlock avoidance and optimization in bidirectional AGVS, *Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques, SIMUTools 11*, pages 152-161.

[5] R. Gopalakrishnan and N. Rangaraj, 2010. Capacity Management on Long-Distance Passenger Trains of Indian Railways, *Interfaces*, vol. 40, no. 4, pages 291-302.

[6] *Indian Railways Year Book 2019-2020*, Ministry of Railways, Government of India, New Delhi.

[7] A. Kuzu, O. Songuler, A. Sonat, S. Turk, B. Birol, E.H. Dogruguven, 2011. Turkish Railways development of a method and its software implementation of automatic interlocking table generation from railway topology, *Proceedings of the IEEE International Conference on Mechatronics*, pages 64-70, Istanbul, Turkey.

[8] S. Moothedath, P. Chaporkar, M.N. Belur, 2019. Approximating constrained minimum cost input-output selection for generic arbitrary pole placement in structured systems, *Automatica*, vol. 107, pages 200-210.

[9] S. Salsingikar, N. Rangaraj and L. Sathishkumar, 2017. Analysis and planning of train movements at a railway junction, *Proceedings of the International Conference on Railway Transport Technology*, Lille, France.

[10] G. Theeg and S. Vlasenko (Eds), 2020. *Railway Signalling and Interlocking*, International Compendium, 3rd Edition, Eurail Press, PMC Media Publishing.

[11] N. Wu and M. Zhou, 2005. Modeling and deadlock avoidance of automated manufacturing systems with multiple automated guided vehicles, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 35, no. 6, pages 1193-1202.

[12] M.C. Zhou and M.P. Fanti, 2004. *Deadlock resolution in computer-integrated systems*, New York: Marcel Dekker.