Routing Trains Through Railway Junctions: A New Set Packing Approach

R. Lusby
Department of Engineering Science
University of Auckland
r.lusby@auckland.ac.nz

Abstract

Arguably the most important decisions facing railway companies today concern the effective utilization of available resources. One such problem, the motivation for this paper, entails allocating the track capacity of a junction to a timetabled set of trains so as to ensure quality routings are obtained. Large junctions are highly interconnected networks of track where multiple railway lines meet, intersect and split. The huge number of routings possible makes this a very complicated problem. We show how this problem can be logically formulated as a set packing model and solved efficiently with a pricing routine in which the columns of the constraint matrix are represented using tree structures. Such a structure is effective as dual variable values can be accumulated at nodes in the tree, and hence the individual pricing of variables can be avoided. A discussion of the variable generation phase (train paths), which takes into account the dynamics of the trains, is also included. The decision support system currently being developed will enable planners to solve strategic, tactical, and operational variants of the problem. An example junction is used to illustrate the proposed methodology.

1 Introduction

The railway industry is rich in problems that can be modelled and solved using Operations Research techniques. In this day and age, arguably the most important of these are the ones that concern the effective allocation and utilization of available resources. One such problem, the focus of this paper, entails allocating the track capacity of a junction to a timetabled set of trains to ensure quality routings are obtained whilst adhering to a variety of operational constraints.

The diverse range of problems facing railway companies are typically categorized according to the required planning horizon. This leads to the customary three stage approach with problems being defined as strategic, tactical, or operational in nature. The problem of routing trains through railway junctions arises at each of the levels. A short description of each variant is given next.

Problems at the strategic level are characterized by lengthy time horizons and typically involve resource acquisition. Viewed in this context the problem considered
here appears in the form of capacity assessment. Railway management often face
the task of deciding between a number of possible investment alternatives concerning
proposed infrastructure modifications to junctions. Perhaps the most influential
factor in making the final decision is capacity. Railway management are very inter-

tested in knowing, with precision, what level of rail traffic the modified infrastructure
would cater for. This involves determining the maximum number of trains that could
be routed through the junction within a given time horizon. The construction or
modification of infrastructure involves high capital investment and has long lasting
ramifications. It is essential that one has proper tools to assist in this process.

Tactical level problems focus more on allocating resources on an infrastructure
which is assumed to be fixed. Such problems normally have a mid-term planning
horizon. On this level the problem considered here answers the question of timetable
feasibility. In most countries it is not uncommon for the railway system to be divided
into two main areas; those responsible for the infrastructure, and those responsible
for the rolling stock (train operating companies). The train operating companies
each submit a preferred timetable, and the infrastructure managers determine if
there exists a conflict free routing for all the trains in the amalgamated timetable.
There is no clear objective at this stage, however, one typically aims to schedule the
maximum number of trains while considering the route preferences of the trains.

Operational problems are defined to be those that occur on a day-to-day op-
erational basis when pre-determined operating policies need to be adjusted due to
unforeseen disturbances. The dynamic environment in which these problems occur
necessitates real time resolution. The impact of late train arrival, track mainte-
nance, or even accidents will propagate through the timetable with varying degrees
of severity and quite possibly result in the pre-determined operating policy becom-
ing infeasible. The variant of the problem occurring here entails reassigning train
routes so as to return to the original schedule with minimal required disruption.

Despite its apparent potential benefits, the application of Operations Research
techniques to the problem of routing trains through railway junctions has been sur-
prisingly limited with manual approaches still widely employed. The purpose of this
paper is to present a set packing model applicable to all instances of the problem out-
lined above. We demonstrate how the train routing problem can be logically formu-
lated as a set packing model, and discuss its flexibility from a modelling perspective.
While being perhaps the most logical formulation of the problem, its dimensions do
impose some restrictions from a computational point of view. However, we show
that the dual of this formulation does possess a number of nice properties that one
can exploit in solving the problem, and present a solution procedure which entails
solving the dual through the dynamic addition of violated cuts (primal variables).
The primal variables are train paths and are constructed using a generator which
takes into account the dynamics of the trains. An efficient pricing routine in which
the primal variables are represented by several tree structures is also described. We
illustrate the proposed methodology with the aid of an example junction.

This paper is organized as follows. Section two gives a detailed description of
the problem, introduces some definitions and previous work. Section three describes
the set packing model and its dual as well as our path generator, while section
four details the solution approach and also the primal variable tree structures. We
conclude with an example in section five, and conclusions are given in section six.


2 Problem Definition

A junction is a highly interconnected network of track where multiple railway lines meet, intersect, and split. Quite typically it includes a station, although this is not assumed in the definition. The network of track comprising the junction can be divided into a number of track sections. These are essentially segments of track on which for safety reasons there can be at most one train at any given time. Some examples include switches, crossings and platform track sections. For completeness, and for the purpose of remaining consistent with the previous literature, we define the perimeter of the junction to consist of a number of entering points and leaving points. These are rather self-explanatory and just indicate the points at which it is possible for trains to enter and leave the junction. Furthermore, they also identify the scope of the infrastructure concerned. A route through the junction is thus a sequence of track sections connecting an entering point to a leaving point. The route may or may not involve stopping at an available platform. Depending on the number of such points as well as the number of switches in the junction, there may be a significant number of routes possible. We distinguish this from a train path which refers to a possible traversal of a given route in time. Figure 1 below illustrates the typical characteristics of a junction as well as a possible route from entering point A to leaving point E.

![Figure 1: A possible junction showing a possible route](image)

A formal definition of the problem we are addressing is as follows. Given the detailed track layout of the junction (defined by the entering and leaving points) as well as a proposed timetable, i.e. the respective arrival and departure times for a set of trains, what is the maximum number of trains that can be assigned a route through the junction? This is what is referred to as the feasibility problem. Other objectives are of course possible, however this one has been chosen for expository purposes. This simple objective function does illustrate the subtle difference between the strategic and tactical level variants of the problem. If the solution to this problem is less than the cardinality of the train set, the proposed timetable is infeasible, and a saturated solution (for this timetable only) is given by the solution to the optimization problem. On the other hand, if the solution to this problem is equal to the cardinality of the train set (i.e. a conflict free routing has been found for all trains), the proposed timetable may or may not be saturated. One could look at introducing saturating trains in such a situation. Irrespective of which problem variant we are concerned
with, the chosen routes must satisfy a number of constraints.

The most important requirement for the selected routes is that no two trains share any part of the junction infrastructure at the same time. Routes that do not satisfy this criterion are said to be in conflict, and obviously cannot be assigned simultaneously. To prevent trains from getting too close to one another within a junction railway companies tend to implement one of two (or possibly both) systems. The route locking and sectional release system enforced by the Dutch (among others) stipulates that trains must lock a sequence of track sections prior to using them. For instance, when a train arrives at an entering point to the junction it must lock all the track sections it is going to use in reaching its designated platform, and then prior to departure, lock all the track sections it is going to use in leaving the junction. Trains successively release each of the locked track sections after traversing them. Such a system allows trains to proceed without interruption within a junction as no track section can be simultaneously locked by two or more trains. Some buffer time is usually incorporated into the release time of the individual track sections to build some robustness into the route. The block signalling system implemented by the French and German railway companies is similar. Essentially the railway network is divided into a number of blocks delimited by signals, where each block may contain one or more track sections. On entry to a block, all track sections are simultaneously locked. These are released when the tail of the train has exited the block and some additional clearing time has elapsed. Any realistic model must accurately incorporate these features.

Zwaneveld et al. (1996) detail a number of other constraints pertaining to customer considerations as well as train connections. For instance, it is often beneficial to have all trains travelling in a similar direction leave from platforms that are close to each other in proximity. It is also necessary to account for trains that must be coupled and decoupled. There may of course be other constraints not listed here that refer to particular operating policies implemented by individual railway companies.


3 The Set Packing Model and its Dual

The problem of routing trains through railway junctions is a natural application of the set packing model. Recall that to provide a conflict free routing through the junction for a set of trains, one must ensure that at most one train is locking any track section at any given time. To capture both the spatial and temporal components inherent in such a restriction, one needs to identify a constraint for each track section in a sequence of uniform time intervals. This is analogous to taking a snapshot of the junction in each time interval. The time interval used is obtained through a discretization of the timetable period. This is consistent with what is done in planning in practice although the duration of the time interval is railway
company dependent. A typical value might be about 15 seconds. This modelling approach allows one to represent a path through the junction for a particular train as a column of zeros and ones. A one in a particular row indicates that the train is claiming that particular track section at that particular time, while a zero indicates otherwise. As was mentioned earlier, trains may have a number of possible routes (each with many paths) through the junction, including of course the null route which pertains to the train not being routed at all. We therefore, in addition to the set packing constraints discussed above, generalise the model through the inclusion of a generalized upper bound (GUB) constraint for each train, thus ensuring we pick one and only one of the possible paths for each train. Our complete model is given below. We will refer to this model as our primal problem.

Maximize \( \sum_{i=1}^{t} \sum_{j=1}^{n_i} \rho_{ij} x_{ij} \)

subject to

\[
\begin{bmatrix}
T_1 & T_2 & \ldots & T_t \\
R_1 & R_2 & \ldots & R_t
\end{bmatrix}
\begin{bmatrix}
x_{ij}
\end{bmatrix}
\leq
\begin{bmatrix}
b_1 \\
b_2
\end{bmatrix}
\]

where \( t \) is the number of trains, \( b_1 \) and \( b_2 \) are vectors of ones, \( P_i \) is set of paths for train \( i \) (indexed 1 to \( n_i \)), \( \rho_{ij} \) is the benefit received in assigning train \( i \) path \( j \in P_i \), \( \pi_1 \) and \( \pi_2 \) are the dual vectors corresponding to the GUB constraints, and the time period track section constraints respectively. The constraint matrix consists of submatrices \( T_i \) and \( R_i \) (\( i = 1, 2, \ldots, t \)). These are defined as follows.

\( T_i = e_i e_i^T \) with \( e_i \) the \( i \)th unit vector and \( e^T = \begin{bmatrix} 1 & 1 & \ldots & 1 \end{bmatrix} \)

\( R_i = (r_{l,k}) = \begin{cases} 1 & \text{if the } k \text{th path for train } i \text{ uses time interval track section pair } l \\ 0 & \text{otherwise} \end{cases} \)

Our decision variables are given by the binary variables

\[
\begin{cases}
1 & \text{if train } i \text{ is assigned path } j \in P_i \\
0 & \text{otherwise}
\end{cases}
\]

Unlike the node packing models proposed by Zwaneveld et al. (1996) and Zwaneveld, Kroon, and van Hoesel (2001), this particular approach implicitly deals with the train routes as it attempts to identify and resolve conflicts between trains during the solution procedure rather than having to generate them all a priori and explicitly represent the infeasibilities. This being the case, we can easily consider additional trains, additional routes, and delayed trains. Identifying and resolving conflicts is embedded in the optimization and there would be a limited amount of additional effort required to include such variables. Providing that they represent feasible paths through the junction, they are nothing more than extra columns in the model.

The above model is very similar in structure to a well known optimization model known as the crew rostering model developed by Ryan (1992). Indeed, one could regard a train as being a crew member and the sequence of time period track section pairs it claims as its sequence of duties. Thus the columns defining routes for trains
are analogous to the columns defining lines of work for crew members. From a modelling perspective the only noticeable difference between the two models appears in the right hand side. In the crew rostering model this is not restricted to being of unit value, nor packing for that matter. In terms of the dimensions of the two models there is a major difference. The crew rostering model is characterized by a small row dimension with many variables. It is not uncommon for crew members to have hundreds of thousands of possible lines of work, however there are only relatively few constraints corresponding to the duties that must be allocated. The set packing model above, on the other hand, is characterized by a constraint matrix with many constraints and relatively few variables. Any realistic problem will have thousands of time period track section constraints. In contrast, trains only have a limited number of possible routes through the junction.

To determine the best solution approach to adopt, one should consider what properties an optimal solution to the primal problem (LP) would have. We would expect there to be, comparatively speaking, few variables. We also believe that there would be a significant number of inactive constraints. It seems highly unlikely that all track sections will be locked in all time intervals. The dimensions of the primal problem naturally suggest one should consider the dual formulation. In the dual setting at optimality these observations would be mirrored in the form of very few active constraints, and most of the dual variables being non-basic at value zero. The dual formulation has a similar dimension to the crew rostering model. Experience tells us that such models can be solved efficiently. The solution approach that we have adopted, which is explained in detail in Section 4, hence focuses on the dual and attempts to exploit its small row dimension as much as possible. For completeness, the dual formulation of (1) is given below (all notation is as previously defined).

\[
\begin{align*}
\text{Minimize} & \quad b_1^T \pi_1 + b_2^T \pi_2 \\
\text{subject to} & \quad A^T \begin{pmatrix} \pi_1 \\ \pi_2 \end{pmatrix} \geq \rho \\
& \quad \pi_2 \geq 0
\end{align*}
\]

(2)

4 Solution Approach

The solution approach we propose consists of two main steps. The first step involves finding all possible train paths (variable generation), while the second entails solving the problem. A detailed description of each is given next.

4.1 Variable Generation

In our modelling we assume that all train routes are generated a priori, but not train paths. Modelling train paths essentially involves determining the distribution of zeros and ones in a particular column. In other words, indicating when and at what time a train will lock particular track sections during its passage through the junction. This depends on two key factors. The safety system enforced by the rail company stipulates how many additional track sections will need to be locked, while the dynamics of the trains (the acceleration and deceleration capabilities) will determine the exact running time of a train on a given track section.
In modelling train dynamics we make several assumptions. Whenever a train enters a new track section on its route, it is limited to doing the following: (1) Proceeding with constant velocity, (2) Accelerating at a constant rate, and (3) Decelerating at a constant rate. This approach of constant acceleration and deceleration rates is consistent with previous work on modelling train dynamics; Zwaneveld et al. (1996) and Lu, Dessouky, and Leachman (2004) both adopt a similar approach. We believe it to be a realistic approach in that it generates only practically feasible paths. Extra restrictions are also included to ensure a train will not accelerate immediately after decelerating and vice versa. This ensures only smooth train paths are considered.

We assume that for each of a train’s possible paths, the entering track section, entering time, and entering speed of the train is known. We also make the assumption that a train will not be accelerating/decelerating on entry to a junction. This defines the initial point of the train path, and is represented as a time space node which stores this information. Similarly, a time window in which the train may depart the junction, as well as the leaving track section are also known. In addition to this each train is assigned a maximum permitted amount of time it can be in the junction. The first time space node is then extended based on the dynamics of the train given the possibilities outlined above. A new time space node will be generated each time a train enters a new track section on its route. This process continues until all possible train paths have been enumerated. Train paths will terminate as feasible if they leave the junction in the designated time window. However, they will terminate as infeasible if this is not the case, or if the duration of the train run through the station exceeds the maximal tolerance. A diagram illustrating this process is given below.

![Diagram of modelling train paths](image)

**Figure 2: Modelling Train Paths.**

The figure above represents the possible ways a train may traverse the first three track sections of its route. The number of the track section it is entering is given on the node. The successor nodes are generated based on the dynamics of the train. The arrows emanating from nodes correspond to the different possible kinematic options available. Each track section is assumed to have both a minimum and maximum permitted velocity. This could be an infrastructure restriction (i.e. speed limit), or the maximum permitted speed of the train. As a result of this, one needs to be careful when extending nodes. To ensure feasible extensions are guaranteed a backwards preprocessing step is used to adjust the maximum and minimum velocities.
on a given track section. This prevents a train from reaching a situation in which it will be unable to comply with the speed requirements of the next track section.

As can be seen from the diagram, the first three track sections can be traversed via a number of different acceleration and deceleration patterns. The fact that we consider only constant rates of acceleration and deceleration, and that different track sections often have different lengths, means it is extremely unlikely that duplicate time space nodes will be generated via two different traversal patterns. On the rare occasion that this does occur, removing one of the labels through domination would also be unlikely. This is primarily because each time space node is defined by an entering time, entering track section, entering speed and entering acceleration. All of which would need to be equal for domination to occur. This means that the modelling described above in fact builds a tree structure. For each route for each train there will be a different tree. On traversing any tree one would obtain all possible ways to traverse the corresponding route.

4.2 Primal Variables as Tree Structures

The representation of the train routes (primal variables) as tree structures is advantageous from a computational point of view. It allows one to price the paths described by the tree very efficiently. Using Figure 2 as an example once more, the arcs indicate the length of time the train spends on the track section associated with the node it emanates from. Hence telling us which primal constraints all train paths containing this arc would cover. We can easily store an accumulated dual value at each node by summing the appropriate dual variables along the arc leading to it. Through assigning each node an accumulated dual value one has effectively priced out a component of the reduced cost for all train paths using this arc. For example, all subsequent nodes of 4* would have the accumulated dual variables to that point in common. This is much more effective than pricing individual variables as one only ever needs to back track to the first arc of difference on pricing a different train path; equivalent to a pre-order walk of the tree. If the accumulated dual of a leaf node (which represents a complete path) is favourable, the variable it represents is easily obtained by tracing back through each predecessor node.

4.3 Solution Method

To solve the set packing model above we propose an approach which attempts to exploit the small row dimension of the dual formulation as much as possible. Essentially the method involves dynamically updating the dual problem through the addition of entering variables as violated constraints.

We begin with an initial dual problem which has just one constraint for each train. We elect to assign each train its null route, although one could easily assign each train any of its possible routes. This is done because we only ever need enough constraints (primal variables) in the dual to ensure we have a primal basic feasible solution. This is analogous to having a restricted master problem in the primal setting. The solution to the dual problem gives us the dual variables for the primal set packing LP. This solution vector can easily be used to price out the tree structures for favourable train paths. A primal variable (cut) pool is also set up (initially null) which will store a subset of the paths. This pool is priced prior to the trees.

In our pricing of the tree structures we implement a form of partial pricing. We
only ever call the pricing routine for a particular train and a particular route, and thus initially focus on one tree. If a favourable train path is found in this tree it is returned immediately to the optimization in the form of a violated constraint. The reduced cost of the primal entering variable is the extent to which the corresponding dual constraint is violated. Hence, on appending the constraint to the dual problem, an artificial variable with an initial value equal to the reduced cost is included to ensure we have a starting basis for the next iteration. The dual problem is then reoptimized to obtain updated dual variables, and the process repeats itself. In effect, we equate an iteration of the primal simplex with the addition of a dual constraint. If, on the other hand, a favourable train path is not found in the tree called for, another tree (for a different train and route) is examined. This continues until either a primal entering variable has been found, or we have priced all trees. In the latter case, we would declare optimality with the optimal solution to the primal problem being the dual vector of the optimal solution to the dual problem.

In an effort to maintain a small basis throughout the solve an inactive constraint removal routine is also implemented. If at a particular solution to the dual an inactive constraint appears, it is removed. Inactive constraints in the dual indicate that the corresponding primal variable is at value zero and can be removed.

5 Example Junction

To demonstrate our model and solution approach we test it on the junction given in Figure 4. While this junction is fictional and purely for expository purposes, it has been created based on the Pierrefitte-Gonesse junction which is used in the test cases of Delorme, Rodriguez, and Gandibleux (2001), Delorme, Gandibleux, and Rodriguez (2004), Delorme (2003), Gandibleux et al. (2005), and Rodriguez (2002). The purpose of this was to assimilate a real life situation as much as possible. The test junction has 50 track sections. Each of which has an upper and lower speed limit and is either unidirectional (indicated by an arrow) or bidirectional. The four different directions entering/leaving the junction have been labelled A, B, C, and D.

The instance we consider consists of eight trains. All trains enter the junction within 570 seconds of each other and the timetable period is 1100 seconds. The trains are one of three types; Express, Inter City, or Freight. The composition is as follows:
3 Express trains between A and D

1 Express train and 2 Inter City Trains between A and C

2 Freight trains between B and C

The difference in train category is reflected in the acceleration and deceleration capabilities of each. The values used are consistent with those of Danish trains; again in order to achieve a real life situation. In this particular case trains have between one and three possible routes (an example of each is given in the diagram).

As was explained in Section 4, the first step in the solution process is to generate the necessary tree structures. For this problem 17 such trees are required. The largest of which contains approximately 500,000 nodes representing about 260,000 possible paths, while the smallest consists of 77 nodes representing 36 paths. In total, the 17 tree structures contain approximately 800,000 possible paths. In reality we believe it is unlikely that there would be so many possible ways to traverse the routes. However, by considering trees that are bigger than we would expect, we can gain an insight into the time required to initialise trees in general. For the problem at hand all 17 tree structures are initialized in less than two seconds.

The pricing of the trees is also very quick. To price all the paths in a particular tree involves a pre-order traversal of the tree and one would expect this to be on par with, if not faster than, the time required to generate the tree itself. Indeed this is the case. Small trees can be priced in fractions of a second, while the larger ones may take slightly longer. The largest tree in the problem considered here takes no more than half a second to price. The traversal code used to price the trees has been stress tested on a tree containing 1.7 million nodes. The 1 million variables that this tree represented took between two and three seconds to price. We believe this to be very encouraging as trees are not likely to ever contain this many possible paths.

With our approach the problem took less than five seconds to solve. This included the preprocessing time involved in generating all paths. The following statistics are interesting to note. During the solve 34 constraints (train paths) were added to the dual, 29 were removed, and there were never more than 17 constraints in a dual basis. This illustrates how we can maintain a very small basis throughout the solve.

The fractional solution obtained by the solver indicated that it is possible to achieve an objective of eight (equivalent to routing all trains) by fractionally assigning routes to trains. By recording integer solutions found by the dual during the execution of the solver a lower bound on the number of trains that can be assigned a route can be obtained. In this case, a lower bound of 7 was found. From the optimal solution we can easily identify where a conflict occurs, and which trains are involved. Although no specific branch-and-bound routine is included in the software at this stage, to ascertain whether or not an integer solution routing all trains exists, a form of primal constraint branching was implemented. Conflicts were chosen arbitrarily from the optimal solution and a decision as to which train would use the particular time period track section was made (again arbitrarily, although in reality one of the two trains may have a higher priority than the other). As a result of this certain variables were no longer considered for each of the trains. The problem was then resolved with a smaller set of variables. Two such branches were required to show there did exist an integer solution which routed all trains. This shows that this problem lends itself to the constraint branching framework. This is the topic of future research and will be implemented in later work.
If the objective value is ever less than the number of trains being considered, the proposed timetable would be infeasible and the only way of achieving feasibility would entail changing the timetable, or considering alternative routes for trains.

6 Conclusions

We have shown that the problem of routing trains through railway junctions can be formulated as a set packing problem and solved efficiently via a procedure that dynamically updates the dual problem through the addition of violated cuts. We have also demonstrated that by representing train routes as tree structures we can price a significant number of train paths very efficiently. We believe that both the modelling technique and solution approach described herein to be very promising.

References


