

DONS: computer aided design of regular service timetables

J.H.A. van den Berg^a & M.A. Odijk^b

^a*Railned Innovation, Utrecht, The Netherlands*

^b*Delft University of Technology, Delft, The Netherlands*

Abstract

In this paper the construction of periodic timetables as the dominant procedure in long term planning of railroad networks at Dutch Railways is briefly described. An efficient algorithm is outlined that generates periodic timetables for a given set of constraints. Also given is a short description of the way this algorithm has been embedded in the decision support system DONS and the way this system is being used to speed up the construction of periodic timetables and increase the flexibility of network designs.

1 Introduction

Dutch Railways (NS) offer railway passengers a regular service (periodic) timetable, repeating a basic schedule each hour. At rush hours extra trains are added to satisfy peak traffic demands. At the moment timetable construction is the dominant method in long term railroad planning.

The process of planning infrastructure for railroad operations aims at determining flexible extensions and adjustments of the railroad network. Its starting point are the forecasted traffic flows for the period 2010- 2015, derived from a macro-economic transport model. Lines are determined in an iterative procedure using an algorithm that allocates passengers to proposed lines, using some assumptions about the service level. In this way a line system is determined that minimizes the total number of transfers.

Subsequently, weights for correspondences between trains at stations are determined. The resulting set of lines, frequencies, correspondences and weights are the input for the timetable construction process.

Based upon experience and expert judgement, planners have selected several variants for the intended layout of the railroad network. These are used in the process of timetable construction and subsequently evaluated using performance measures like costs, revenues, service level and robustness. Because of the extremely long time frame of 10 to 15 years, the construction of a few single schedules does not result in a robust design of the future railroad network. However, constructing a timetable is extremely time consuming. At the moment it takes 3 to 6 months to construct one single timetable, so it is hardly possible to assess the flexibility of a network design.

Because NS aim at doubling its volume of traffic and presented Rail 21 as its vision on railroad transportation in the next century, the current railroad network will undergo considerable changes during the next 15 years. Because future traffic flows are uncertain and decisions have to be taken now, there is a need for methods that support determining flexible adjustments and extensions of the infrastructure. Therefore a project called DONS started at the end of 1992 aiming at the development of a decision support system to speed up the timetable construction process considerably.

The DONS project is highly innovative. To our knowledge, no comparable system has been developed before, to our view because of the computational complexity of constructing periodic timetables. The next two sections provide a brief description of the model and the algorithms that underlie the DONS system. Subsequently, we describe how this system supports the design of flexible railroad networks.

2 Designing a timetable structure

Constructing a timetable amounts in fact to solving a system of constraints that define the structure of the aspired timetable. This system of constraints is hereafter referred to as a timetable structure and in this section we will explain what it looks like. Then, in the next section, we will describe several algorithms to solve a timetable structure, thus constructing a timetable.

In a timetable structure all constraints are of the form

$$\tau_i \in \{(\tau_j + x) \bmod P \mid x \in [a_{ij}, b_{ij}]\} , \quad (1)$$

where τ_i and τ_j are the unknown event times of event i and j respectively. For instance, event i can be the departure (arrival) of a certain train from (at) a certain station. Furthermore, a_{ij} and b_{ij} are specified numbers such that $0 < b_{ij} - a_{ij} < P$ and P expresses the periodicity. In the Dutch case

$P = 60$ minutes. This expression must be understood as ‘event i takes place between a_{ij} and b_{ij} minutes (modulo 60) after event j took place’. We assume that all events occur at integral points in time, though this is theoretically not a severe restriction. Also, with constraints like in (1) more complicated constraints can be modeled, see Odijk [2].

The constraints in a timetable structures may stem from various practical requirements a timetable is supposed to meet. For one thing, a constraint might express that two trains must correspond at a certain station. Think, for instance, of passengers transferring from one train to another or two trains being physically coupled. Constraints like these are specified by the Marketing Department within NS.

Other constraints arise because the limited infrastructure does not allow for some simultaneous train movements. For example, if train i can cross a track section no sooner than 2 minutes after train j crossed it (and vice versa), a constraint like

$$\tau_i \in \{(\tau_j + x) \bmod P \mid x \in [2, 58]\} \quad (2)$$

is part of the timetable structure. To generate all such infrastructural constraints an ExSpect model has been developed at Eindhoven University of Technology (Eindhoven, The Netherlands). ExSpect is a family name denoting a model concept based on High-level Petri nets, an executable specification language and a software tool to support the execution of specifications. The infrastructure is modeled with it and by executing the specification all mutual exclusions of trains on a certain track can be enumerated.

A third type of constraints is formed by the requirement that for some lines the stations are visited every half an hour in a semi-regular fashion. For instance, two trains, say i and j , serving the same line, visit a station twice an hour in a 28/32, 29/31, 30/30 or 31/29 pattern. This, then, is captured by the constraint

$$\tau_i \in \{(\tau_j + x) \bmod P \mid x \in [28, 31]\} . \quad (3)$$

Note that if a strict half an hour pattern is specified, the timetable structure can be reduced by removing a variable (i.e. an event time) and relating all constraints specified for it to the variable it is strictly related to.

The constraints in a timetable structure can be visualized by drawing up a directed graph in which each vertex represents an event and each arc a constraint of its adjacent vertices. We call such graph a constraint graph and figure 1 shows an example. In this example there are four events and five relations between them. Note that the two relations between event 1 and 4 express that event 1 happens between 2 and 4 minutes after event

112 Railway Operations

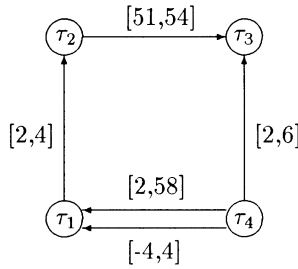


Figure 1: Example of a constraint graph.

4 or vice versa. The constraint graph will prove to play an important role when solving a timetable structure.

3 Solving a timetable structure

A timetable structure for which the underlying undirected graph of the constraint graph is a tree can be solved easily by assigning admitted values to the vertices (event times) when walking along the branches of the tree. Since a tree does not contain any circuit, no vertex is ever revisited so that there is no danger of encountering inconsistencies. Therefore, in the remainder of this paper we assume that the underlying undirected graph of the constraint graph is 2-connected, see Bondy & Murty [1].

The general problem of solving a timetable structure is NP-complete. This can be seen by reducing the Graph Coloring problem to it. This is the problem of, given a graph and an positive integer K , to determine if the vertices of the graph can be ‘colored’ such that no two adjacent vertices have the same color.

To solve a timetable structure three approaches have been developed. The first one is based on constraint propagation and backtracking and it is developed at Eindhoven University of Technology by Voorhoeve [4]. The algorithm starts by taking two constraints that share precisely one variable (event time), say

$$\tau_1 \in \{(\tau_2 + x) \bmod P \mid x \in [a_{12}, b_{12}]\} \quad (4)$$

and

$$\tau_2 \in \{(\tau_3 + x) \bmod P \mid x \in [a_{23}, b_{23}]\} . \quad (5)$$

These two constraints implicitly relate τ_1 and τ_3 by virtue of transitivity. Hence, we can deduce

$$\tau_1 \in \{(\tau_3 + x) \bmod P \mid x \in C_{13}\} \quad (6)$$

from expressions (5) and (6), where C_{13} is a segment or the union of two segments depending on the values of $a_{12} + a_{23}$ and $b_{12} + b_{23}$. Then, if there exists a constraint relating τ_1 and τ_3 , a new constraint can be constructed by intersecting C_{13} and $[a_{13}, b_{13}]$, thus creating a smaller search space when looking for feasible timetables. By continuing this process on and on until no further changes can be made a sort of 'minimal' timetable structure is created. Then the algorithm continues by choosing a value for some event time and propagating the remaining constraints again. This either ends with a further reduced timetable structure in which again some event time is fixed and propagation is continued, or it ends with the observation that one of the last choices made for an event times leads to an empty reduced timetable structure so that this choice must be reconsidered (backtracking).

The second algorithm is developed by Schrijver and Steenbeek [3] at the Centre for Mathematics and Computer Science (Amsterdam, The Netherlands). It starts with rewriting expression (1) as

$$\begin{cases} \tau_i + n_{ij}P \in \{\tau_j + x \mid x \in [a_{ij}, b_{ij}]\} \\ n_{ij} \text{ integer} \end{cases} \quad (7)$$

for each constraint of the timetable structure. An assignment to the n_{ij} 's is called an n-vector. Schrijver and Steenbeek, then, use a branch & bound technique to look for a feasible n-vector, i.e. an n-vector for which the remaining system of constraints is feasible.

Finally, the third method has been developed by Odijk [2] at Delft University of Technology (Delft, The Netherlands). It also starts with rewriting the timetable structure like in (7) but it searches for an n-vector somewhat differently. That is, it can be shown that a timetable exists if and only if for each circuit in the constraint graph the n_{ij} 's obey some linear expression called a circuit-constraint, see Odijk [2]. Then, by solving the system $\mathcal{X}$ of all circuit-constraints, either a feasible n-vector is found, or the infeasibility of the timetable structure is revealed.

However, there can be $\mathcal{O}(2^N)$ circuits in the constraint graph, where N is the number of vertices. So, it is virtually impossible for large and highly specified timetable structures to enumerate all circuit-constraints of $\mathcal{X}$. Fortunately, this is not necessary. By Duality Theory it can be shown that it is sufficient to consider $M - N + 1$ circuit-constraints at a time, where M is the number of constraints of the timetable structure. The algorithm, then, is an iterative procedure in which each iterate is associated with an n-vector and a set $\mathcal{C}$ of $M - N + 1$ circuit-constraints, the two of them constituting the state of the algorithm in that iterate. Given such n-vector, it is tested on feasibility and either the test is positive and a timetable can be found by solving a linear program, or the n-vector is found infeasible and a violated circuit-constraint is reported. In the latter case, the violated

114 Railway Operations

circuit-constraint is added to $\mathcal{C}$, a new n -vector is calculated and a non-binding circuit-constraint is removed from the new $\mathcal{C}$. Note that adding the violated constraint cuts away the old infeasible n -vector. Therefore, this method is a cutting-plane method.

The algorithm always terminates either with a feasible timetable or with an overspecified circuit in the constraint graph. This way the railway scheduler gets information about what he did wrong when specifying a timetable structure. He, then, can modify the timetable structure by relaxing the overspecified circuit restart the algorithm with, for instance, an n -vector that is a solution to $\mathcal{I}$.

The constraint propagation approach was found unreliable as regards computation time. It frequently happened that after more than 20 hours of computation on a single problem instance the algorithm still neither had found a timetable nor had it established its infeasibility. Therefore, this approach is rejected, although propagating constraints seems a useful way to preprocess the timetable structure when applying the other two approaches. It may, therefore, very well be that in the future a hybrid approach will be implemented as the final choice of algorithm. The branch & bound approach is very promising. Surprisingly short computation times of only a few minutes are reported using this method and, moreover, the method is flexible enough to extend it with an 'optimization mode'. Finally, the cutting-plane method is currently under study, but it is for sure the most sophisticated method meaning that it exploits the algebraic structure of a timetable structure by far the most, although it is less general and flexible than the other two approaches. This is especially true when NS also wants (and they do) to solve timetable structures with disjunctive constraints. Computational results of the cutting-plane algorithm are in order (Odijk [2]).

4 Embedding the algorithm in a DSS

The algorithm developed by Schrijver and Steenbeek [3] has been implemented in the DONS system. A modular system architecture guarantees optimal flexibility implying that other algorithms can be added to the system. The ExSpect graphical user interface is used to build railroad networks that can be saved in an Oracle database system together with the relevant parameters like maximum speed, crossing times, follow up times etc. A separate module calculates travel times for long distance trains, regional trains and local trains.

The functionality of the DONS system is being extended at the moment. Especially the feed back the system gives in case of inconsistent constraints is being elaborated in order to let the system suggest the most effective adjustments in the timetable structure. Furthermore, a new windows based graphical user interface is being developed in order to meet the high stan-

dards at NS for planning systems.

The DONS system is used to generate many timetables for a given network variant and various timetable structures. The number of qualitatively different schedules indicates the flexibility of the railroad network being considered. In this way a multitude of variants for the infrastructure can be evaluated in a few weeks, providing an enormous improvement compared to the situation in which timetables were constructed manually.

The robustness of timetables generated by the DONS system can be assessed using the network simulation model FASTA [5], developed by the Federal University of Technology of Lausanne (EPFL, Lausanne, Switzerland). Other performance measures are still calculated manually at the moment.

5 Conclusions

It turns out to be possible to use advanced Operation Research techniques to construct algorithms that generate periodic timetables, though fine heuristics remain indispensable to make the algorithm efficient. These algorithms embedded in a decision support system can be used to design flexible railroad networks. By including more details, like safety installations, signals and detailed characteristics of rolling stock, the DONS system can be made suitable for short term timetable construction.

References

1. Bondy, J.A. & Murty, U.S.R. *Graph Theory with Applications*, The MacMillan Press Ltd, London, 1976.
2. Odijk, M.A. *A Cutting Plane Algorithm for Timetable Construction*, To appear as research report, Delft University of Technology, Delft, The Netherlands, 1994.
3. Schrijver, A. & Steenbeek, A. *Dienstregelingontwikkeling voor Nederlandse Spoorwegen* (In Dutch), Unpublished report, Center for Mathematics and Computer Science, Amsterdam, The Netherlands, 1993.
4. Voorhoeve, M. *Rail scheduling with discrete sets*, Unpublished report, Eindhoven University of Technology, Eindhoven, The Netherlands, 1993.
5. *FASTA* (In French), Report 303/3, Institut des transport et de planification, Ecole Polytechnique Fédérale de Lausanne, 1993.