Automatic dispatching of train operations using a hybrid optimisation method

Stefan Wegele, Roman Slovák, Eckehard Schnieder
Institute for traffic safety and automation engineering
Technical University of Braunschweig
Langer Kamp 8, Braunschweig, Germany

Abstract
The dispatching of a large area in a railway net is a very challenging task because of the many constraints that have to be taken into account during the decision making process. Nowadays, dispatching is mainly taken care of by human operators. None of the algorithmic approaches attempted in the last three decades could solve this task. Their computational complexity or simulation accuracy was insufficient for practical application. The dispatching method presented in this work brings a new algorithmic approach with practical application. Based on the optimisation method specially developed for this purpose, optimal dispatching decisions are calculated. The resulting conflict-free timetable can be directly implemented in the railway operation. Additionally, the resulting timetable represents an input for the energy saving train speed control.

Introduction
Rail transport differs from road transport in that vehicles move on a very restricted topology, resulting in strong intravehicular interaction. Deviations from the timetable can thus have implications for space and time allocation. It is the aim of dispatching to minimise the impact of such deviations from the timetable, and it can thus be seen as predictive control (cf. Fig. 1 and Fig. 2). Based on the current operating state, a forecast for future system behaviour is given, and on this basis, the control variable (dispatch measures) is optimised.

Fig. 1: Integration of dispatching in the operations control system
Fig. 2: Dispatching following the principle of predictive control: Optimisation problem at point in time $t_0$ with system state $x_0$.

To determine potentially successful dispatch measures, the quality of a newly computed timetable has to be assessed. One obvious rating system involves awarding penalty points to disruptions that are noticed by customers, i.e. delays, missed connections, platform changes. It is the aim of dispatching to achieve maximum customer satisfaction, i.e. to minimise penalties.

$$f = \sum_{i \in \text{Trains}} \left( \sum_{j \in \text{Stations}} k_{i,j} \Delta t_{i,j} + \sum_{j \in \text{Stations}} g_{i,j} \Delta G_{i,j} + \sum_{l \in \text{Trains}} s_{i,j} s_{i,l} \right) \Omega \rightarrow \min$$

- $k_{i,j} \Delta t_{i,j}$: Weighted penalty for the delay $\Delta t_{i,j}$ of a train $i$, at station $j$ [euros]
- $g_{i,j} \Delta G_{i,j}$: Weighted penalty for a platform change for a train $i$, at station $j$ [euros]
- $s_{i,j} s_{i,l}$: Weighted penalty for a missed connection between trains $i$ and $l$ [euros]

Since all rail operators can define their own penalties, multiple factors can be taken into account, such as the number of passengers affected, waiting time for the next train, type of service, ticket price, etc.

Research on optimising conflict resolution in rail transport has been carried out for over thirty years (cf. Table 1). Given the great complexity of this task, and previously inadequate computer technology, initial models of operations were simple, using constant trip times. As computer technology improved, it became possible to look at more complex models and handle more difficult problems. But calculations were still very time-consuming, so that these methods were normally only used for train path planning.
This optimisation approach combines evolutionary algorithms with the branch-and-bound method. One by one, the individual train paths are recomputed using branch-and-bound, and the constraints of evolutionary algorithms are varied (cf. Fig. 2). During train pathing trip times are determined in a microscopic simulation. It has been shown that greater optimisation can be achieved by calculating trip times very accurately in simulations, with a relative deviation of less than $10^{-7}$ (8 milliseconds deviation per day), rather than using tables with pre-calculated trip times, since the scattering of trip times is eliminated by the simulation variance in repeated iterations. For this purpose, we have developed a simulation module that can calculate 48,000 train kilometres in only 80 ms.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Model</th>
<th>Coding</th>
<th>Optimisation method</th>
<th>Problem size</th>
</tr>
</thead>
<tbody>
<tr>
<td>[2], 1981</td>
<td>macroscopic</td>
<td>conflicts</td>
<td>Branch-and-Bound</td>
<td>single line, 12 trains</td>
</tr>
<tr>
<td>[3], 1991</td>
<td>macroscopic</td>
<td>time based</td>
<td>Branch-and-Bound</td>
<td>net, 100 trains, 3000 km</td>
</tr>
<tr>
<td>[4], 1995</td>
<td>macroscopic</td>
<td>train chains</td>
<td>Branch-and-Bound</td>
<td>net, -</td>
</tr>
<tr>
<td>[5], 1998</td>
<td>macroscopic</td>
<td>time based, Lagrange-Relaxation</td>
<td>Branch-and-Bound, conjugate gradients</td>
<td>single line, 26 trains, 17 stations</td>
</tr>
<tr>
<td>[6], 1999</td>
<td>Fuzzy Petri nets</td>
<td>conflicts</td>
<td>Rule-propagation (coordinate search)</td>
<td>-</td>
</tr>
<tr>
<td>[7], 2002</td>
<td>microscopic</td>
<td>time based</td>
<td>time based conflict solution</td>
<td>net, -</td>
</tr>
<tr>
<td>[8], 2002</td>
<td>-</td>
<td>train priorities</td>
<td>genetic algorithms</td>
<td>single line, 64 trains, 250 km</td>
</tr>
<tr>
<td>[9], 2002</td>
<td>-</td>
<td>conflicts</td>
<td>genetic algorithms</td>
<td>single line, 20 trains</td>
</tr>
<tr>
<td>[10], 2002</td>
<td>macroscopic</td>
<td>time based</td>
<td>genetic algorithms</td>
<td>net, 126 trains, 31 stations</td>
</tr>
<tr>
<td>[11], 2005</td>
<td>microscopic</td>
<td>train chains, conflicts</td>
<td>Branch-and-Bound, Tabu-Search, Evolutionary Algorithms</td>
<td>net, 1024 trains, 500 km, 104 stations</td>
</tr>
</tbody>
</table>

Table 1: Overview of approaches to conflict resolution in rail transport

**Optimisation method**

During train pathing trip times are determined in a microscopic simulation. It has been shown that greater optimisation can be achieved by calculating trip times very accurately in simulations, with a relative deviation of less than $10^{-7}$ (8 milliseconds deviation per day), rather than using tables with pre-calculated trip times, since the scattering of trip times is eliminated by the simulation variance in repeated iterations. For this purpose, we have developed a simulation module that can calculate 48,000 train kilometres in only 80 ms.
The branch-and-bound method is acting on an event-driven Petrinet model (cf. Fig. 4).

![Petrinet model of a station with two tracks, T1-3 are trains moving from left to right](Fig. 4)

Train sequences are assigned to places and the transitions fire in accordance to them. The branch-and-bound method constructs the decision tree, where every train is inserted into train sequences on places. A typical decision tree for insertion of a local train is shown in Fig. 3.

![Decision tree for a local train. First value - id of the Petrinet place, second - position of the train inside of its train sequence, third - value of objective function after the decision, (third) - lower bound estimation for objective function](Fig. 5)

As the branch-and-bound method inserts trains one by one the resulting timetable quality depends on the order in which the trains were inserted. The aim of the global optimization with evolutionary algorithms is to find an optimal insertion sequence. It also tries to modify the objective function to enforce globally optimal and not locally optimal decisions during insertion of single trains.

**Test cases**

To test this dispatching algorithm, we looked at a section of the network in Northern Germany, in collaboration with Deutsche Bahn AG (cf. Fig. 6). This section comprises 104 stations and 1,006 km of line, including both single- and double-track stretches. Depending on the day of the week, between 750 and 1,000 services per day run on the network.
In a first stage, we assessed the quality of the suggested solutions generated. Let us assume an ICE train (marked in black in Fig. 7) has an entry delay of 20 minutes. In a territory that is so densely utilised, this results in multiple occupational conflicts, which are resolved using signal box logic, i.e. without changing the order of the trains. This causes delays for 43 trains, resulting in 78,194 penalty points for the timetable.
Fig. 7: Situation following disruption notification - all occupational conflicts are resolved using signal box logic.

Fig. 8: Optimised timetable.
With this optimisation, which took approx. 2.2 seconds to process, the total penalty could be reduced from 78,194 to 39,633 points or 50.6%. No further manual improvement of the timetable was possible. Since disruptions spread within a network, and a linear time-path chart only shows one section, Fig. 8 can only model some of the dispatch measures taken, namely:

1. The station stop of an IR train was moved to enable overtaking on the main track.
2. At station HLEH, the disrupted train was placed on a siding to enable a meet with another ICE train.
3. At station HBS, the disrupted train’s stopping time was extended by 3 minutes, allowing an SE train and an RB train to pass. After that station, the line is single-track, and the options for letting trains pass are thus limited.
4. The meeting point of the SE and ICE trains was moved from HHEG to HWWI.
5. In HBTM, a meet with another ICE train was implemented.

Additional time-path charts could be used to show all other dispatch measures. Of the remaining 39,633 penalty points, 24,131 result from the ICE train’s primary delay; another 11,091 had accumulated earlier. The secondary penalty thus amounts to

\[39633 - 24131 - 11091 = 4411\] points.

**Other Areas of Application**

This project has had to handle the great complexity of dispatching, and a high level of software complexity. Using the dispatching procedure as a basis, other planning tasks in a similar field could be resolved, including operations simulations and train path planning:

- Operations simulation (stochastically disrupted simulation runs to analyse timetable stability) can be implemented relatively easily. Disruptions are generated by a random generator, rather than manually in the graphical user interface. They are added to the module via the CORBA interface.
- Train path planning is another area of application for the module. As already shown in Fig. 5, train path planning forms part of the dispatch module. Planning a new train path is quite similar to dispatching for a severely disrupted train path. When a train is delayed by, say, 40 minutes, it will miss all scheduled “slots” and have to be inserted again. The only difference is that the dispatch process also has to take into account and make changes to all other services, whereas in train path planning, the other services are unaffected. This behaviour can be easily modelled by giving a very low weighting to the reinserted service. Since all other services have far higher weightings, possible disruptions would receive high penalties and would thus be avoided by the optimisation procedure.

A further stage of development will involve testing this procedure on a larger section of the DB AG rail network, to provide evidence of its suitability for use in a train control centre.

**Impact on railway business**

After implementation and installation of the presented method, the railway company would have available the following tools:

- Global score for the quality of railway operation with proved non-discriminating decisions
- List of primary and secondary delays for penalty money transfers
- Automated support for optimal dispatching decisions, resulting in more predictable and punctual operation
- Possibility of coupling this algorithm to an energy saving driver assistant for saving of up to 40% energy due to avoidance of train stops at blocked tracks [14].

**References**

Transportation Science, 25(1), 1991, pp. 46-64


