Adaptive, data-driven, online prediction of train event times*

Pavle Kecman¹ and Rob M.P. Goverde¹

Abstract—This paper presents a microscopic model for accurate prediction of train event times based on a timed event graph with dynamic arc weights. The process times in the model are obtained dynamically using processed historical track occupation data, thus reflecting all phenomena of railway traffic captured by the train describer systems and preprocessing tools. The graph structure of the model allows applying fast algorithms to compute prediction of event times even for large networks. Accuracy of predictions is increased by incorporating the effects of predicted route conflicts on train running times due to braking and re-acceleration. Moreover, the train runs with process times that continuously deviate from their estimates in a certain pattern are detected and downstream process times are adaptively adjusted to minimize the expected prediction error. The tool has been tested and validated in a real-time environment using train describer log files.

I. INTRODUCTION

Real-time prediction of train positions in time and space is a basic requirement for effective route setting, traffic control, rescheduling, and passenger information. However, in practice only the last measured train delays are known in the traffic control centers and dispatchers must predict the arrival times of trains using experience only, without adequate computer support. This often results in simple extrapolation of the current delays for the expected arrival delays. Some railways use a linear shift of the timetable to extrapolate the current delays to the future. This method neglects the fact that some trains may (partially) recover from a delay using running time supplements, while others may get (more) delayed due to route conflicts.

In the current literature, the problem of deriving accurate predictions of the state of railway traffic in the network has been tackled mainly by simulation tools and approaches based on graph theory. Microscopic simulation tools such as OpenTrack [1] or RailSys [2] are able to give accurate predictions of running times, and possible route conflicts and resulting delay propagation. Due to a high level of detail in modeling infrastructure and train dynamics, such models are not suitable for real-time applications in large and heavily utilized networks.

Medeossi et al. [3] used track occupation data along with train event data recorded on-board to calibrate the train motion equation parameters in the process of computing stochastic blocking times for individual trains. This work has been extended with stochastic estimates of initial delays and dwell times in order to build a simulation tool for an a priori evaluation of the impacts of timetables changes or infrastructure improvements [4].

The prediction, model introduced in this paper, is based on an acyclic timed event graph. Goverde [5] introduced a macroscopic model for train delay propagation based on timed event graphs and max-plus algebra that allows application of fast algorithms for computation of delay propagation in a short time even for large networks. Due to the fixed structure of train orders and routes in the model, it is not directly suitable for real-time predictions that need to consider current conditions on the network and changes in the actual process plans.

Bücker & Seybold [6] incorporated stochasticity in their graph based mesoscopic model by treating initial delays as random variables, described with suitable distribution functions, and applying analytical methods to compute delay propagation. However, the large-scale character of the model does not allow precise modeling of train interactions and the resulting variability in running times.

Hansen et al. [7] presented a macroscopic model for prediction of train running times using historical track occupation data. The dependencies of dwell times as well as running times on the level of open track sections (line segment between two stations) were captured and used to compute robust estimates of arrival and departure times. We extend this approach to the microscopic level in order to accurately model train behavior and interactions on open track sections.

A microscopic online prediction tool, based on a directed acyclic graph with dynamic weight computation using train motion equation has been implemented in the Swiss traffic control system RCS-DISPO [8].

In an earlier work, Kecman & Goverde [9] presented an approach based on computing the arc weights of the microscopic graph model using historical data. Dependence of process times on current delays is determined and incorporated in a dynamic arc weight assignment. Train interactions are modeled with high accuracy by including the main operational constraints and relying on actually realized minimum headway times obtained from the historical data rather than on theoretical values. The depth-first search based algorithm for computing the predicted event times over a graph with dynamic arc weights gives predictions for all reachable events within the horizon.

This approach is extended here by precise modeling of route conflicts and incorporating time losses, due to braking and re-accelerating of hindered trains, in the predictions.

*This paper is a result of the research project funded by the Dutch Technology Foundation STW: “Model-Predictive Railway Traffic Management” (project no. 11025)

¹Department of Transport & Planning, Delft University of Technology, 2628 CN Delft, The Netherlands {p.kecman,r.m.p.goverde}@tudelft.nl
Moreover, we present an adaptive component that exploits the feedback information about the actually realized blocking times of running trains. Trains with process times that continuously deviate from computed estimates in a certain pattern are detected and downstream process times can be adjusted to minimize the expected prediction error.

The next section gives the general setting of the online prediction tool. Section II presents the predictive component that adjusts the estimated running times to include the impact of route conflicts. Adaptive adjustment of estimated running time based on the feedback of realized running times from the running train is described in Section III. Finally, Section IV presents the validation of the model in a case study and Section V application possibilities and directions for future research are given.

II. ONLINE TRAFFIC PREDICTION TOOL

The main components of the tool and the flow of data between them are depicted in Fig. 1. The online prediction tool is based on a timed event graph with dynamic arc weights. The graph topology is built and updated based on the actual timetable, route and connection plan, and current positions of trains on the network on the level of block sections. We assume that the actual route and connection plans are continuously provided by the traffic control for the next 30 min. Each change of the actual plans or new information from the real-time operations results in an update of the graph topology, i.e., adding new trains, modifying train routes, updating connections and removing passed events.

Arc weights represent the predicted process times. The weight of an arc is time-dependent and assigned in a dynamic way, depending on the (estimated) starting time of the modeled process. Arc weights depend on the actual delays (difference between realized and scheduled event times), and predicted delays (difference between predicted and scheduled event times). That way the dependence of running and dwell times on current delays is incorporated in the model. Moreover, the graph weights are adjusted dynamically to incorporate the effects of predicted route conflicts and to minimize the prediction error for running trains based on the already realized running times.

The tool receives a stream of data from the train describer system (log file). In the Dutch train describer system, the train steps are recorded on the level of track sections (a block section consists of one or more track sections), with a message when a new track section is occupied by a train and when a track section is released by a train. Signal aspect changes are also recorded and can be coupled to a running train that caused the aspect change [10]. Train positions are monitored on the level of signal passages. After every signal passage by a running train, the graph is updated with the new information, and a prediction of all event times is executed.

A. Microscopic graph model

The railway traffic is modeled microscopically with a timed event graph (TEG). A TEG is a representation of a discrete-event dynamic system in which events are modeled by nodes and processes by arcs.

We distinguish between signal events (passing of a signal by a running train) and station events (arrival and departure at and from a platform track).

Events of a train are connected by running and dwelling arcs. Interactions between trains are modeled by headway and connection arcs. Connection arcs are introduced for modeling commercial constraints (passenger transfers), or logistic constraints (rolling-stock and crew connections). They connect the arrival event of a feeder train and the departure event of a connecting train in the same station.

Headway arcs separate successive occupations of an infrastructure element by different trains. Route plan for a train contains a list of block objects in the planned route. Every block object is described by the track sections comprised in the block. For each track section, a headway arc is added between the corresponding events (signal passages) of every two successive trains that are planned to traverse the section. Headway arcs are constructed in a way that fully reflects the microscopic operational constraints of railway traffic, described by blocking time theory on open track segments (between two stations) [11] and the sectional release route setting principle in station areas [12].

The graph structure and topology is fully defined based on the actual route and connection plan, and the timetable.

B. Computation of arc weights

Weights of running and dwelling arcs in the graph are equal to the predicted process times, and weights of headway and connection arcs represent the minimum process times of the modeled processes. In order to accurately estimate an arc weight that models a running or dwelling process, we assume that delayed trains typically run in the time-optimal regime and have minimal dwell times, thus exploiting the time supplements to (partially) recover from delays. Similarly, trains running on time or ahead of their schedule aim to run in a lower regime to avoid early arrivals and decrease energy consumption. In that context, a time-dependent, dynamic computation of arc weights [13] is added to the timed event graph described in the previous section. The basic idea behind this approach is that running and dwell times
depend on the previously noted delays [7]. The dependence of running and dwell times has been determined for each relevant combination of block sections and train lines [9].

The weights of headway arcs represent the minimum headway time between two trains on the same infrastructure element. The weight of a connection arc is equal to the minimum transfer time for passenger connections or the minimum time needed to perform activities that enable planned rolling-stock and crew circulations, for logistic connections.

Connection times do not depend on the current delay of trains and the possible effect of delays on headway times was not considered in this work. Therefore, these values are computed offline from the track occupation data and the corresponding arc weights are fixed.

C. On-line prediction of event times

When an update about the realization of an event becomes available (the corresponding message is logged by the train describer system), the information is further propagated through the reachable set of the realized event. The prediction algorithm based on the depth-first search [9] derives estimates of event times in the graph. Since the prediction can be performed in linear time, this methodology is applicable for real-time predictions of train traffic in large and busy networks.

III. ROUTE CONFLICTS PREDICTION

In this section we consider a standard three aspect signaling system where each block is reserved exclusively for one train at a time [12].

The prediction of route conflicts is simple after execution of the prediction algorithm since the estimates of all train speed dependent times (approaching, running, clearing) are known. After including sight and reaction, and setup and release time, taken as constant values for all trains, the blocking times are determined and a route conflict is identified by the overlapping blocking times [14] (Fig. 2).

A route conflict has occurred when the train driver of the approaching train sees a yellow signal aspect indicating that he needs to be prepared to stop at the next signal. Typically, the running time over a block before the red signal aspect increases due to breaking and possible waiting time in front of the red signal. The resulting time loss depends on the duration of the conflict [15].

We define the duration of a route conflict as the time difference between the passing time of a 'yellow' signal and the release time of the subsequent signal to a permissive aspect (overlap in blocking times indicated in red in Fig. 2).

A. Impact of route conflicts on running times

The impact of a route conflict on the running time of the hindered train over the subsequent block depends on the conflict duration and the route and running time of the hindering train. The typical situation that occurs in practice when the two conflicting trains follow the same route is the ‘conflict wave’, where the hindered train keeps passing signals that show yellow aspect and is thus unable to re-accelerate to the full speed. We therefore consider the time loss due to re-acceleration only after the hindered train passes a signal showing a green aspect.

Since the running time estimates are computed based on the free running times, arc weights that model the running processes over affected blocks need to be adjusted to take into account braking (and possible waiting time in front of the signal), running at a lower speed and re-acceleration for every predicted route conflict.

In order to estimate the effects of route conflicts on train running times, all route conflicts in six days of traffic on the busy corridor Leiden–Dordrecht in the Netherlands were filtered out. The impact of the conflict duration on time loss after passing a yellow signal was analyzed (Fig. 3).

The duration of a conflict is computed using the process mining conflict identification tool [10] and the resulting time loss is obtained as a difference between the realized running time over a block and the predicted running time derived depending on the current train delay.

<table>
<thead>
<tr>
<th>Conflict duration [s]</th>
<th>Increase in running time [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-100</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>200</td>
<td>100</td>
</tr>
<tr>
<td>300</td>
<td>200</td>
</tr>
<tr>
<td>400</td>
<td>300</td>
</tr>
<tr>
<td>500</td>
<td>400</td>
</tr>
<tr>
<td>600</td>
<td>500</td>
</tr>
<tr>
<td>700</td>
<td>600</td>
</tr>
<tr>
<td>800</td>
<td>700</td>
</tr>
<tr>
<td>900</td>
<td>800</td>
</tr>
</tbody>
</table>

Fig. 2: Route conflict visualisation

Fig. 3: Dependence of time loss on conflict duration

A regression analysis was performed based on 1330 data points. A robust quadratic fit resisting 25% of the outliers
showed the best performance in terms of coefficient of determination $R^2 = 0.83$. Even though the data points are scarce for conflict duration longer than 400 seconds, the slope of the curve can be interpreted easily as the waiting time in front of the signal. Short conflicts result in a slight running time increase since the signal of conflict changes to a permissive aspect before the hindered train comes to a standstill.

The time loss due to re-acceleration (after passing the green aspect signal) was also analyzed but no correlation with conflict duration was found. This can be explained by the fact that a train starts re-accelerating before it passes the green signal aspect, independent of the conflict duration at the previous signal.

B. Adjusting the running time estimates

Running time adjustment is incorporated in the prediction algorithm that sweeps once through the reachable set of nodes and computes the predicted event times for the corresponding events. For each predicted route conflict, the increase of running time of the hindered train is computed depending on the conflict duration.

We denote an event $i$ by $e_i$ and its realization time by $t(e_i)$. The set of direct predecessors of $e_i$ is denoted by $P(e_i)$, the set of direct predecessors connected with headway arcs by $P_h(e_i) \subset P(e_i)$ and the preceding signal passing event of the same train by $p(e_i) \in P(e_i)$. Finally, $w_{j,i}$ represents the weight of the arc $(e_j,e_i)$. The following procedure shows the implementation of the route conflict prediction and the resulting adjustment of predicted running time.

**Algorithm 1 RouteConflictPrediction**

1. if $t(p(e_i)) > \max_{j \in \{j|e_j \neq p(e_i)\}}(t(e_j) + w_{j,i})$ then
2. $d \leftarrow t(p(e_i)) - \max_{j \in \{j|e_j \in P_h(e_i)\}}(t(e_j) + w(j,i))$
3. $\Delta \leftarrow f(d)$
4. $w_{j,i} \leftarrow w_{j,i} + \Delta$, for $j = \{j|e_j = p(e_i)\}$
5. $t(e_i) \leftarrow \max_{j \in \{j|e_j \in P(e_i)\}}(t(e_j) + w_{j,i})$

If a route conflict has been predicted (line 1) the duration of the conflict $d$ is computed in line 2 and the predicted time loss $\Delta$ as a function of $d$ determined from historical data as explained in Section III-A in line 3. After updating the running time estimate in line 4, the predicted event time is computed in line 5.

An illustrative example of route conflict prediction is given in Fig. 4. The graph shows three trains $q, r, s$ (the trains are planned to pass signal $S_2$ in that order) with their planned routes over the given subnetwork. If we define by $t(s,2)$ the time when train $s$ passes signal $S_2$ a route conflict can be identified by comparing the passing time of train $s$ at signal $S_1$, $t(s,1)$ with the earliest possible release time of signal $S_2$ due to minimum headway times after passing of trains $q$ and $r$, $\max(t(q,3) + h^{q,s}, t(r,2) + h^{r,s})$. If train $s$ passes signal $S_1$ before release time of $S_2$ the conflict is identified and the running time estimate of train $s$ between signals $S_1$ and $S_2$ can be adjusted as shown in Algorithm 1.

\[
S_1 \leftarrow 0 \quad S_2 \leftarrow 0 \quad S_1 \leftarrow 0
\]

\[
(q, S_1) \quad (q, S_2) \quad (q, S_3)
\]

\[
(r, S_1) \quad (r, S_2) \quad (r, S_4)
\]

\[
(s, S_1) \quad (s, S_2) \quad (s, S_3)
\]

\[
h^{q,s} \quad h^{r,s}
\]

\[
w
\]

Fig. 4: An illustrative example of route conflict prediction

IV. ADAPTIVE ADJUSTMENTS OF RUNNING TIMES PREDICTIONS

The presented prediction model is event driven, i.e., all event times are predicted when an update on the current positions of a running train becomes available. The estimated running times over block sections depend on the departure delay from the last scheduled stop. An adaptive component of the prediction model keeps track of the actually realized running times of a running train and adjusts the predicted running times until the next scheduled stop of the train.

An illustrative schematic example of adaptive prediction is given in Fig. 5. The running train departed from station $A$ and in the situation from the figure has just cleared the $j^{th}$ out of $m$ blocks to station $B$ where it is scheduled to stop. The gray solid line starting at station $A$ represents the predicted running time of the train based on the actually registered departure delay. For the sake of clarity, for subsequent realized signal passages only the predicted running time over the following block is shown.

The prediction error of the running time over block $b_k$ is denoted by $\delta_k$ and computed after each observed signal passage. For all subsequent blocks until the next stop we derive the estimated prediction error $\hat{\delta}$ and adjust the estimates of running times over the remaining blocks by

\[
\hat{\delta}_i = \frac{1}{j} \sum_{k=1}^{j} \delta_k , \forall i = j + 1, ..., m
\]

The red dotted line in Fig. 5 denotes the adjusted prediction of running times to station $B$.

The adaptive component of the prediction model enables online detection of train runs with process times that continuously exceed or fall behind the off-line computed estimates, and adjusts the predictions of future train behavior accordingly. By applying this adaptive prediction strategy, the continuous delay sources of the conflict-free run of a
single train (e.g. due to particular driving style or defective rolling-stock) as well as temporary speed restrictions (due to infrastructure malfunctions or maintenance), will be possible to identify and include in predictions.

V. CASE STUDY

The predictive model has been tested and validated on the busy corridor Leiden–The Hague–Rotterdam–Dordrecht in the Netherlands. The 60 km long corridor is (partially) traversed daily by approximately 300 trains per direction.

The training set of data consists of train describer log files for six days of traffic in two traffic control areas. While sweeping the files with the process mining and conflict identification tool [10], the dependencies of process times on current delays are computed. The predictions are performed on a test set consisting of track occupation data for one day of traffic.

For model validation (and example of application) we simulate the real-time environment by scanning the train describer log file from the test set that contains the chronologically sorted infrastructure and train messages from the train describer log files of two traffic control areas (Rotterdam and The Hague). Traffic control input is included in the form of a list of trains described by train number, timetable, route plan (block sections) and expected entrance time to the observed part of the network (or the first departure times if the train starts within the observed area).

A. Comprehensive evaluation

We tested the model performance by sweeping the test set train describer log file with rolling prediction horizons of different length. The prediction algorithm is initiated and the rolling horizon is moved after receiving a message that reports the realization of each of the 9776 signal and station events during one day of traffic on the corridor. Table I shows the average prediction error, the average number of events that are predicted in each algorithm execution, and the average number of arcs for prediction horizons of 2 hours, 1 hour, 30 minutes, 20 minutes and 10 minutes.

<table>
<thead>
<tr>
<th>Prediction horizon [min]</th>
<th>120</th>
<th>60</th>
<th>30</th>
<th>20</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. error [s]</td>
<td>58.69</td>
<td>53.80</td>
<td>46.56</td>
<td>39.72</td>
<td>24.07</td>
</tr>
<tr>
<td>Avg. no. events</td>
<td>1040</td>
<td>532</td>
<td>269</td>
<td>180</td>
<td>90</td>
</tr>
<tr>
<td>Avg. no. arcs</td>
<td>2288</td>
<td>1117</td>
<td>590</td>
<td>389</td>
<td>202</td>
</tr>
</tbody>
</table>

Since the prediction algorithm is linear, computational complexity, which depends on the size of the input graph is not considered as a criterion for choosing the most appropriate prediction horizon. Average prediction error, as well as the average number of nodes and arcs are monotonically decreasing as shorter prediction horizons are considered. We obtain an average error shorter than 1 minute for the prediction horizon of 30 minutes. The 10 minutes horizon is included in the analysis to show that in terms of average prediction error, our model outperforms the approach of predicting event times using train motion equations [8].

B. Predictive adjustment of running times of hindered trains

The effect of adapting the predicted running times to include time loss due to route conflicts is illustrated in Fig. 6, which shows the blocking time diagram of 3 trains approaching station Rotterdam (RTD).

A ‘conflict wave’ between local train ST5029 and intercity train IC2129 at station Schiedam (SDM) was captured and indicated in red. It is evident that the running time of train IC2129 over the block between signals SDM827 and SDM38 was adapted and extended to include the time loss due to the restrictive aspect of SDM38 in front of the approaching train. The corresponding increase in running time was computed as a function of the conflict duration. The running time adjustment due to other, shorter conflicts in the change is less significant.

C. Adaptive prediction for running trains

An example of adaptive prediction that minimizes the prediction error for running trains is shown in Fig. 7.
The example considers intercity train IC1919 that does not have scheduled stops between The Hague HS (GV) and Delft (DT). The first prediction, derived at the moment of departure from The Hague HS, is represented by the blue line. After three corrections of running time estimates resulting in predictions shown in black, red and green, the prediction error of less than 1 second was achieved (the final running time estimate given in green practically overlaps the realized running time given in magenta).

The predicted arrival time error is monotonically decreasing as the train progresses towards station Delft. Therefore the propagation of prediction error to connected events of other trains is reduced thus affecting the overall performance of the model.

VI. CONCLUSIONS

This paper presented an approach for predicting train event times which considers time losses resulting from route conflicts and adjusts the process time estimates based on the actually realized event times of the running trains.

The dependence of the running time increase of a hindered train on the duration of a route conflict has been determined. The model is able to predict the occurrence of a route conflict and adjust the estimated running time of the hindered train depending on the predicted conflict duration. Moreover, the concept of adaptive running time adjustment until the next scheduled stop of a running train, based on minimizing the previous prediction errors has been developed and implemented in the online prediction model. The model has been applied in a real-life case study and its performance, as well as the impact of the length of prediction horizon, was analyzed.

The predictive model provides effective decision support to signalers and traffic control and contributes to a better utilization of railway infrastructure, improved reliability of train services, and more reliable and dynamic passenger information. The developed model will be embedded in a closed-loop model-predictive railway traffic control framework where online optimization algorithms will automatically resolve detected conflicts and propose control decisions to traffic controllers together with the predicted conflicts [16]. This way an intelligent railway traffic management system will be obtained that pro-actively monitors the railway traffic and supports traffic controllers with decisions that optimize the traffic on a network level, beyond the traditional local control areas.

REFERENCES