Delft University of Technology

Delft Center for Systems and Control

Technical report 15-016

Improved distributed model predictive control for rescheduling of railway traffic by manipulation of the cost functions*

B. Kersbergen, T. van den Boom, and B. De Schutter

If you want to cite this report, please use the following reference instead:

B. Kersbergen, T. van den Boom, and B. De Schutter, "Improved distributed model predictive control for rescheduling of railway traffic by manipulation of the cost functions," *Proceedings of the 6th International Conference on Railway Operations Modelling and Analysis (RailTokyo2015)*, Narashino, Japan, 13 pp., Mar. 2015. Paper 025.

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.24.73 (secretary)
URL: `https://www.dcsc.tudelft.nl`

*This report can also be downloaded via `https://pub.deschutter.info/abs/15_016.html`

Improved distributed model predictive control for rescheduling of railway traffic by manipulation of the cost functions

Bart Kersbergen [a,1], Ton van den Boom [a], Bart De Schutter [a]
[a] Delft Center for Systems and Control, Delft University of Technology
Mekelweg 2, 2628 CD Delft, The Netherlands
[1] Email: b.kersbergen@tudelft.nl, Phone: +31 (0) 15 27 85331

Abstract
In this paper we introduce two distributed model predictive control (DMPC) approaches that significantly improve the quality of the solutions found compared to the DMPC approaches that were introduced by Kersbergen et al. [2014b] for the rescheduling of railway traffic, while the computation time only increased by a small fraction. In DMPC the global rescheduling problem is split up into several local problems that are solved by local model predictive controllers that communicate with each other to achieve a solution for the global rescheduling problem. We improve the solution found by the DMPC approaches by adjusting the weights in the local problems such that the delay propagation through the network is reduced. We compare the performance in terms of computation time and delay reduction of the different DMPC approaches with the global model predictive control approach for different lengths of the prediction horizon.

Keywords
Model predictive control, Distributed model predictive control, Rescheduling, Dispatching support system

1 Introduction

In the Netherlands, Switzerland, China, Japan, and several other countries the capacity used is nearing the maximum recommended capacity use or may even be more than the recommended one. Because of this small delays often propagate through large parts of the network and cause many other trains to be delayed. To minimize the propagation and total delay dispatchers reschedule and reroute trains, break connections, or in case of large delays even cancel trains. Many researchers have been developing decision support systems for the dispatchers [Dessouky et al., 2006, Corman et al., 2010, 2014, Törnquist Krasemann, 2012, Caimi et al., 2012, Meng and Zhou, 2014] that help them determine the best dispatching actions.

In this paper we build on the work of Kersbergen et al. [2014a,b]. Kersbergen et al. [2014a] model the railway traffic as a discrete-event system and the dispatching problem is solved using a centralized model predictive controller. Kersbergen et al. [2014b] introduce two distributed model predictive control (DMPC) approaches and the performance in delay reduction and computation time are compared to the centralized model predictive control (MPC) approach. We continue this work by introducing two new DMPC approaches, based

on one of the DMPC approaches of Kersbergen et al. [2014b]. By choosing appropriate weights for the cost functions of the distributed model predictive controllers the delay reduction achieved with the new DMPC approaches will be better than the delay reduction achieved with the DMPC approaches of Kersbergen et al. [2014b]. In this paper we describe how these weights can be determined based on properties of the model of the railway traffic and network.

2 Railway Model and Control

In order to determine a new conflict-free schedule and to reduce the delays we need to be able to predict the effects of the dispatching actions on the delay propagation and on the arrival and departure times of the trains and we need to be able to determine dispatching actions that reduce the delays. To predict the effects of the dispatching actions we need to model the railway traffic and network. To determine the dispatching actions that reduce the delays as much as possible we need a controller that can efficiently assess a large number of different combinations of dispatching actions and in a smart way reduce the number of combinations such that it determines the desired dispatching actions and a new conflict free schedule in a very short time. First we will describe the model used to predict the state of the railway traffic in the future. After that we will describe the centralized MPC approach of Kersbergen et al. [2014a] and the DMPC approaches we have developed before [Kersbergen et al., 2014b].

2.1 Railway Model

The model used in this paper is a discrete-event system. The events of the system are the arrival and departures of the trains at stations and at junctions and intersections outside the interlocking area of the stations. We assume that the routing through the stations and their interlocking areas is managed locally and therefore conflict-free. This allows us to simplify the model and describe the stations and their interlocking area as single points. Tracks between stations and/or junctions are modeled as a single segment; block sections are not modeled. Because block sections are not modeled the signaling system cannot be modeled explicitly, instead we ensure a safe distance between trains by enforcing a time difference on the arrival and departure times between trains on the same track. This time difference is modeled by *headway constraints* and *separation constraints*. Headway constraints do not only ensure a safe distance between trains on the same track, they also define the order in which the trains traverse the track. A train traversing a track is modeled by a *running time constraints*. A train dwelling at a station is modeled by a *continuity constraints*. Transfers at stations that are guaranteed by the train operators are modeled by *connection constraints*.

The general form of these four constraints is:

$$x_i \geq x_j + \tau_{ij}, \tag{1}$$

where $x_i, x_j \in \mathbb{R}$ are the event times and $\tau_{ij} \in \mathbb{R}$ is the minimum process time (dwell, running, headway, separation, or connection time).

To ensure trains do not depart before their scheduled departure time *timetable constraints* are used. Timetable constraints have the following general form:

$$x_i \geq r_i, \tag{2}$$

where $r_i \in \mathbb{R}$ is the scheduled departure time. The trains depart and arrive as soon as all constraints are satisfied.

To be able to break connections and change the order of the trains on tracks we need to be able to "enable" and "disable" constraints. This can be done by adding binary variables to the constraints. Constraints with binary variables are described by one of the following two general forms:

$$x_i \geq x_j + \tau_{ij} + u_{ij}\beta \tag{3}$$

$$x_i \geq x_j + \tau_{ij} + (1 - u_{ij})\beta, , \tag{4}$$

where $u_{ij} \in \{0, 1\}$ is the binary control variable and $\beta \ll 0$ is a very large negative value. The very large negative value in combination with the binary variable allows us to "enable" and "disable" constraints. These equations should be seen as (1) with an added term $u_{ij}\beta$ or $(1 - u_{ij})\beta$. If $u_{ij} = 0$, then (4) always holds due to the added term. If $u_{ij} = 1$, then 3) always holds due to the added term. The resulting model is a macroscopic model. We will not describe the model in more detail in this paper since it has been described in full detail before [Kersbergen et al., 2014a].

2.2 Model Predictive Control

With the model we can predict the future state of the railway traffic as well as the effects of adjusting the binary variables. The model predictive controller is used to determine the binary variables that reduce the delays as much as possible within a given prediction horizon of length T. The prediction horizon is limited in length since determining the binary variables is a computationally complex task, and choosing a smaller prediction horizon reduces the computational complexity. Furthermore, the situation on the railway network is constantly changing and predicting the future state of the railway traffic perfectly is impossible. As a result the model predictive controller has to recompute the binary variables at discrete time instants $t(k)$ for $k \in \mathbb{N} \cup \{0\}$. At each time instant $t(k)$ the controller receives new information on the current state of the network and the railway traffic, and new estimates for the future process times. It uses this information to determine the binary variables that reduce the delays as much as possible for the prediction horizon from $[t(k+1),\ t(k+1)+T)$. For the events that occur between $[t(k + n + 1),\ t(k + n + 2))$ the binary variables associated to these events are implemented. Here $n \in \mathbb{N} \cup \{0\}$ depends on the time needed to implement the binary variables and the time between two time instants. This process is repeated at every time instant $t(k)$.

To determine the binary variables at each time instant $t(k)$ the model predictive controller has to solve a Mixed Integer Linear Programming (MILP) problem. The MILP cost function is chosen such that it is a measure of the total delay in the network. By solving this MILP problem we find the desired binary variables for that time instant $t(k)$. In Rudan et al. [2013] it is explained how this MILP problem can be determined.

Let us define x as a vector consisting of the event times of the events that occur between $[t(k),\ t(k + p))$, u consists of the binary variables associated to these events, c_x and c_u are the weights on x and u respectively, row vector $c = \begin{bmatrix} c_x & c_u \end{bmatrix}$ and $z = \begin{bmatrix} x^\top & u^\top \end{bmatrix}^\top$. The

cost function of the MILP problem can then be written as $c\,z$ and the MILP problem as:

$$\begin{aligned} &\min_{z} c\,z \\ &\text{s.t. } A\,z \leq b \end{aligned}$$

where $z \in \mathbb{R}^{n_x} \times \{0,1\}^{n_u}$, with n_x the number of continuous variables and n_u the number of binary variables, and where $A\,z \leq b$ contains the constraints that describe the railway network and traffic model.

2.3 Distributed Model Predictive Control

With DMPC the system to be controlled is split up into several interacting smaller systems. Each of these smaller systems is controlled locally by a model predictive controller. These local model predictive controllers coordinate with the other local model predictive controllers to reach a good control for the global system.

As a starting point for the DMPC approach in this paper we use the second method of Kersbergen et al. [2014b]. This method was based on a reordering of the constraint matrix such that the MILP problem that needs to be solved at every step of the global model predictive controller has the following structure:

$$\min_{z} \begin{bmatrix} \tilde{c}_1 & \tilde{c}_2 & \dots & \tilde{c}_n \end{bmatrix} \begin{bmatrix} \tilde{z}_1^\top & \tilde{z}_2^\top & \dots & \tilde{z}_n^\top \end{bmatrix}^\top \tag{5}$$

$$\text{s.t. } \begin{bmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,n} \\ A_{2,1} & \ddots & & A_{2,n} \\ \vdots & & \ddots & \vdots \\ A_{n,1} & A_{n,2} & \dots & A_{n,n} \end{bmatrix} \begin{bmatrix} \tilde{z}_1 \\ \tilde{z}_2 \\ \vdots \\ \tilde{z}_n \end{bmatrix} \leq \begin{bmatrix} \tilde{b}_1 \\ \tilde{b}_2 \\ \vdots \\ \tilde{b}_n \end{bmatrix} \tag{6}$$

where $\tilde{c}_i$, $\tilde{z}_i$, $\tilde{b}_i$ for $i \in \{1,\dots,n\}$ are vectors of appropriate size, $A_{i,j}$ for $i \in \{1,\dots,n\}$, $j \in \{1,\dots,n\}$ are matrices of appropriate size, and

$$\begin{aligned} \tilde{z}_i &= \begin{bmatrix} \tilde{x}_i^\top & \tilde{u}_i^\top \end{bmatrix}^\top \\ A_{i,i} &= \begin{bmatrix} A_{i,i,x} & A_{i,i,u} \end{bmatrix} \\ A_{i,j} &= \begin{bmatrix} A_{i,j,x} & \mathbf{0} \end{bmatrix} \end{aligned}$$

for $i \in \{1,\dots,n\}$ and $j \in \{1,\dots,n\}/\{i\}$, and where $\tilde{x}_i$ and $\tilde{u}_i$ are vectors of appropriate size. The matrices $A_{i,i}$ are split up into a part $A_{i,i,x}$ that is multiplied by $\tilde{x}_i$, and a part $A_{i,i,u}$ that is multiplied by $\tilde{u}_i$ in (6). The matrices $A_{i,j}$ are also split up into two parts. One part $A_{i,j,x}$ is multiplied by $\tilde{x}_j$ in (6). The part that is multiplied by $\tilde{u}_j$ in (6) is guaranteed to be a zero matrix.

The second method of Kersbergen et al. [2014b] is repeated here for convenience:

Each step of a local model predictive controller can be written as a MILP problem:

$$\min_{\tilde{z}_i} \tilde{c}_i\,\tilde{z}_i \tag{7}$$

$$\text{s.t. } A_{i,i}\tilde{z}_i \leq b_i - \sum_{j \in \{1,\dots,n\}/\{i\}} A_{i,j}\tilde{z}_j \tag{8}$$

where $\tilde{z}_j$, for $j \in \{1, \ldots, n\}/\{i\}$ is determined by the other local model predictive controllers.

The steps to determine a near-optimal feasible global solution are:

1) Define an initial estimate for z denoted by $\hat{z}$
2) For subproblem i, denoted by (7) and (8), assume $\tilde{z}_j$ for $j \in \{1, \ldots, n\}/\{i\}$ is known and equal to $\hat{\tilde{z}}_j$ and solve subproblem i for $i = 1$. Denote the solution as $\hat{\tilde{z}}_i^i$.
3) Update the estimate $\hat{\tilde{z}}_i = \hat{\tilde{z}}_i^i$.
4) Repeat steps 2 and 3 for $i = 2, \ldots, n$.
5) Repeat steps 2, 3, and 4 until $||\hat{\tilde{z}}_i - \hat{\tilde{z}}_i^i|| < \Delta$ for $i = 1, \ldots, n$, where Δ is a very small value.

Convergence at step 5 was not proven and after extensive testing it was clear that the method does not always converge; so we update step 5 of the DMPC approach to stop after a finite numbers of, say m, iterations:

5) Repeat steps 2, 3, and 4 until $||\hat{\tilde{z}}_i - \hat{\tilde{z}}_i^i|| < \Delta$ for $i = 1, \ldots, n$, where Δ is a very small value. If there is no convergence within m iterations then lock the binary variables $\tilde{u}_i$ for $i = 1, \ldots, n$ to their last determined values and determine the corresponding continuous variables $\tilde{x}_i$.

Because each local model predictive controller only considers its own part of the railway network and does not consider the propagation of the delays to the other parts of the network the quality of the found solutions left room for improvement. In the next section we will introduce the two new DMPC approaches that. Both of these approaches are based on the second method of Kersbergen et al. [2014b], but their delay reduction is much better.

3 Adjusting Cost Functions

In the second method of Kersbergen et al. [2014b] the local model predictive controllers do not consider the delay propagation to the other parts of the network and as a result the train orders at the border of their area of control tend to be suboptimal for the total network, since a large part of the effects of the order changes are only noticed in the next area. To reduce the delay propagation from one area controlled by a local model predictive controller to another area we will improve the second method by adjusting the weights of the events of the trains leaving to another area. The weights that need to be adjusted can be easily found. For the local model predictive controller i the indices of the weights of the MILP problem in $\tilde{c}_i$ that have to be adjusted can be determined by finding the variables of MILP problem i that affect MILP problem j for $j \in \{1, \ldots, n\}/\{i\}$. If we look at how MILP problem i is defined in (7) and (8) it is clear that the indices of the variables of MILP problem j on which MILP problem i depends coincide with the indices of the non-zero columns in $A_{i,j}$. The indices of $\tilde{c}_i$ that need to be adjusted therefore coincide with the indices of the non-zero columns of $A_{j,i}$ for $i = 1, \ldots, n$ and $j \in \{1, \ldots, n\}/\{i\}$. Denote these weights by $\tilde{c}_i^{\text{out}}$ and denote the corresponding variables as $\tilde{z}_i^{\text{out}}$.

By increasing the weights $\tilde{c}_i^{\text{out}}$ the local controllers are forced to focus on reducing the delays of the corresponding events, and by doing so possibly reduce the delay propagation

to the other areas, at the cost of the total delay in the local area. There are many possible choices for the values of the weights $\tilde{c}_i^{\text{out}}$. In this paper we will look at two different choices:

- Increase the values of weights $\tilde{c}_i^{\text{out}}$ with the same factor
- Increase the values of weights $\tilde{c}_i^{\text{out}}$ based on the number of continuous variables the corresponding trains have in the other MILP problems

The first option is really simple: just choose a factor with which to increase all weights $\tilde{c}_i^{\text{out}}$. Our choice is to double the weights. The reason for this is that every continuous variable corresponding to these weights is connected to at least one continuous variable in one of the other MILP problems, and the dependency between the delays of these two continuous variables is very high.

For the second option we need to determine for every train leaving the area which events are used to model this train in the other areas. This corresponds to the continuous variables that are connected through continuity and running time constraints to the variables $\tilde{z}_i^{\text{out}}$. The weights $\tilde{c}_i^{\text{out}}$ are set to the sum of the weights of the standard cost function of the continuous variables that the events are connected to through continuity and running time constraints. In the next section we will show what the effects of these changes to the cost function are on the quality of the solution found with the DMPC approaches and what influence they have on the computation time.

4 Case Study

The case study will consists of two parts. For the first part we will compare the solution quality of the two approaches introduced by Kersbergen et al. [2014b] with the two improved approaches introduced in this paper for 1000 scenarios. For the two approaches in our previous work [Kersbergen et al., 2014b] we will use the sum of delays cost function: $c\,z = [\mathbb{1}^{1\times n_x}\ \ 0.0001 \cdot \mathbb{1}^{1\times n_u}] \begin{bmatrix} x \\ u \end{bmatrix}$, where $\mathbb{1}^{1\times m}$ is a 1 by m vector containing only ones, n_x is the number of continuous variables, and n_u is the number of binary variables. For the first part we will only look at a single time instant of the controllers, in which the controllers have to determine the optimal new schedule for the next hour based on the current situation of the railway network and traffic.

The DMPC approaches of our previous work [Kersbergen et al., 2014b] will be denoted as DMPC 1 and DMPC 2. The two improved DMPC approaches of this paper will be denoted as DMPC 3 and DMPC 4.

For the second part we will compare the best method found in part one with the global model predictive controller for different lengths of the prediction horizon and repeat the process every minute, by moving the prediction horizon one minute further and computing a new solution. The controller stops once there are no more delays present and all trains run according to the nominal schedule. This is done for 100 scenarios.

For both parts of the case study the model of the railway network is based on the Dutch railway network with the timetable of 2011[1]. The train lines, the line type, and their fre-

[1]The complete timetable is too large to include in this paper and is no longer available online. Since there have been very few major changes in the timetable in the last years the reader can get a general idea of the timetable from the 2015 timetable. The timetable of 2015 can be found (in Dutch) at http://www.ns.nl/reizigers/reisinformatie/informatie/informatie-tijdens-uw-reis/download-dienstregeling-2014-2015.html.

quencies are shown in the Appendix in Table 1. Each line is in both directions and the frequency is per direction.

The part of the network that is strongly connected is considered. This means that the following tracks have been left out: from Leiden via Gouda to Woerden (and back), the tracks from Baarn to Den Dolder (and back), as well as the tracks from Almelo to Zwolle (and back), and the tracks that are not electrified. In the case study only reordering of trains and changing arrival and departure times are considered for dispatching actions. Each hour 326 trains traverse the network and the order of the trains traversing the tracks can be changed at 66 points in the network. For the DMPC approaches the network is split up into four smaller networks based on the method proposed in our previous work [Kersbergen et al., 2014b].

For each scenario we generate delays in the first hour of the railway traffic and the controller will be activated after the first hour. No new delays are introduced after the first hour when the controllers are active. We delay 10% of the trains with a randomly generated delay according to a Weibull distribution with scale parameter 5 and shape parameter 0.8.

All scenarios were solved on a personal computer (PC) with an AMD Phenom II X4 960T (3.0GHz) CPU with 16GB of DDR3-1600 memory. The PC is running Windows 7 64bit. The scenarios were solved using Gurobi 5.6.0 [Gurobi] called through the mex-interface of Matlab R2013b 64bit.

4.1 Case Study: Part 1

The relative increase in delays because of the use of the DMPC approaches is shown in Figure 1. The absolute increase in delays is shown in Figure 1. The computation time for the global model predictive controller and the four DMPC approaches to find their solution is shown in Figure 3.

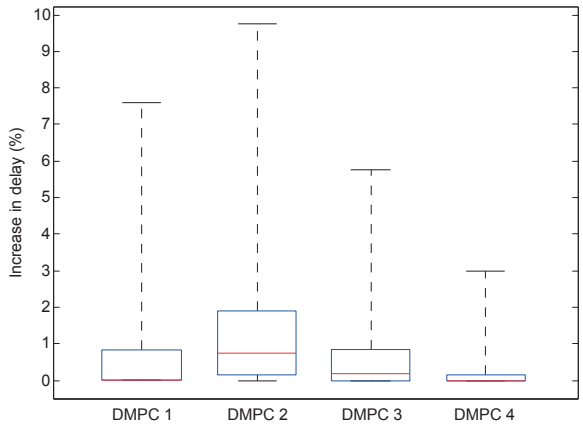

Figure 1: Relative increase in delays (%) compared to the solution of the global MPC, DMPC 1, DMPC 2, DMPC 3, and DMPC 4

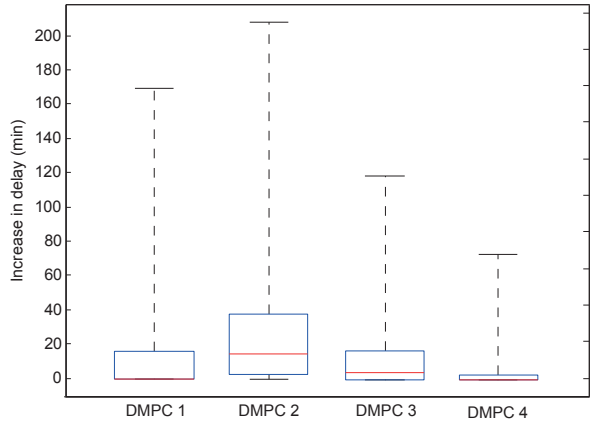

Figure 2: Absolute increase in delays (min) compared to the solution of the global MPC, DMPC 1, DMPC 2, DMPC 3, and DMPC 4

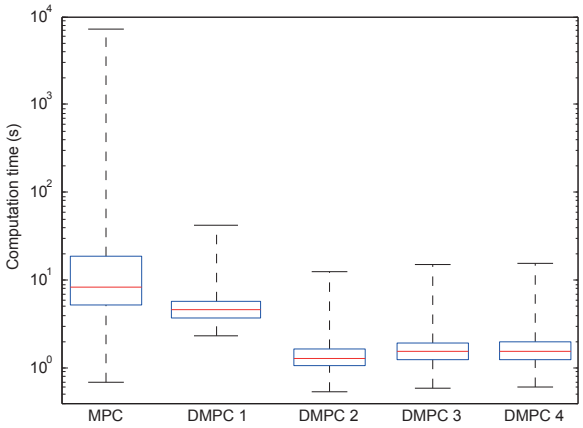

Figure 3: Computation time of the global MPC, DMPC 1, DMPC 2, DMPC 3, and DMPC 4

From Figures 1 and 2 it is clear that the adjustment of the weights has a beneficial effect on the solution quality. Especially DMPC 4 finds very good solutions; the average increase in delays is only 0.14% (2.9 minutes), which is much lower than the average of 0.63% (12.7 minutes) for DMPC 1, and the 1.27% (25.1 minutes) average increase of DMPC 2. From Figure 3 it is clear that the change in cost function does not have a significant effect on the computation time. DMPC 3 and 4 are slightly slower than DMPC 2, but still much faster than DMPC 1 and the global MPC approach.

4.2 Case Study: Part 2

In this part of the case study we will look at the total delay from the start of scenario until all trains run according to their nominal schedule again. The effectiveness of DMPC 2.2

is compared to the global MPC approach for prediction horizons of length 30, 45, 60, 75, and 90 minutes. Since no new delays are introduced the amount of delays will decay when shifting the prediction horizon further. As a result at later time instants the MILP problems to be solved by the controllers will be easier to solve, therefore we will only look at the computation time of the first 10 time instants of each scenario. This results in 1000 time instants for which we compare the computation time for each of the control approaches and prediction horizons. The computation times are shown in Figure 4. Because the computation time needed to solve some time instants for the global MPC approach with a prediction horizon of 75 minutes already takes nearly 20000 seconds, making it infeasible for an online implementation, we have not tested the global approach for a prediction horizon of 90 minutes.

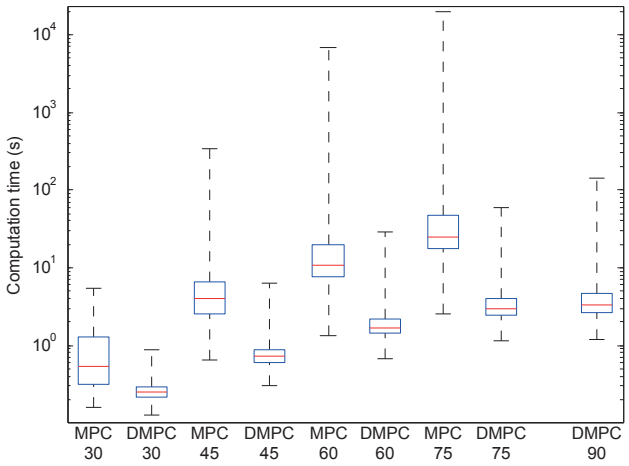

Figure 4: Computation time for DMPC 2.2 and the global MPC approach for prediction horizons of length 30, 45, 60, 75, and 90 minutes.

To compare the performance in terms of delay reduction of the different approaches for the different prediction horizons we have summed up the total delay of the 100 scenarios and compared them to each other and the delays in case no control actions were taken. The results are shown in Figure 5.

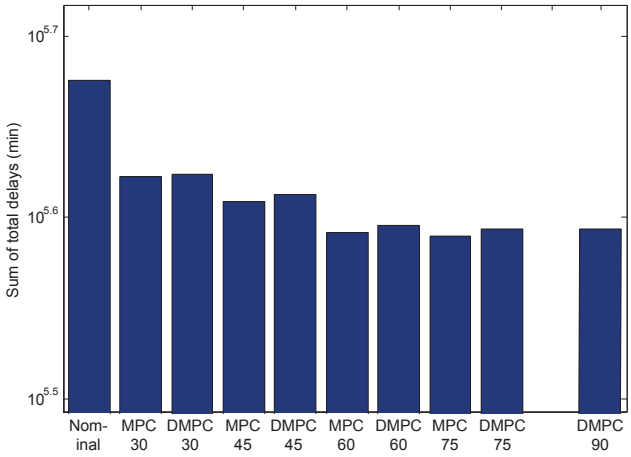

Figure 5: Sum of delays of all scenarios for DMPC 2.2 and the global MPC approach for prediction horizons of length 30, 45, 60, 75, and 90 minutes.

When considering the computation time in Figure 4 DMPC 4 can compute the solutions for all scenarios and time instants within 150 seconds for all lengths of the prediction horizon. For a length of 75 minutes or less the DMPC approach is even able to keep the computation time below 60 seconds. The global MPC approach is only able to keep the maximum computation time below 150 seconds for a prediction horizon length of 30 minutes. For the prediction horizon of 60 minutes the maximum computation time already comes close to 7000 seconds, which makes it very difficult to use in on-line optimization.

From Figure 5 it is clear that for the same control horizon the global approach consistently performs better than DMPC 4. Furthermore, it is clear that increasing the prediction horizon from 30 to 60 minutes has a huge effect on the amount of delays, but increasing it further than 60 minutes has a much smaller effect on the delays. For DMPC 4 it even appears that there is no benefit from increasing the prediction horizon past 75 minutes. The sum of total delays even goes up a little from 392482 minutes to 392657 minutes. With a longer prediction horizon it is more likely that dispatching actions in other areas change the delay propagation of the trains and therefore the estimated delay propagation that each local controller bases its dispatching actions on becomes less accurate and that may have negative effects on the quality of the global solution.

If we consider both computation time and solution quality, the best choice is DMPC 4 with a prediction horizon of 75 minutes. With a prediction horizon of 75 minutes DMPC 4 performs similarly to the global approach with a prediction horizon of 60 minutes, but the average and perhaps more importantly the maximum computation time is much lower. Making it a more suitable candidate for implementation in an on-line rescheduling tool.

5 Conclusions

In this paper we have introduced an improved distributed model predictive control (DMPC) approach. We have compared the solution quality in terms of delay reduction and computation time of our improved approach with two DMPC methods previously introduced in the literature and the global mode predictive control approach. We have tested the effects

of different lengths of the prediction horizon on the solution quality and computation time. From the case study it can be concluded that the proposed improved DMPC approach can perform better than the global model predictive control approach when considering both computation time and solution quality.

For our future work we will test how splitting the network up into a different number of smaller parts and different methods for splitting up the network affects the performance of the DMPC approaches.

6 Acknowledgments

This research is supported by the Dutch Technology Foundation STW, project 11025 "Model-Predictive Railway Traffic Management; A Framework for Closed-Loop Control of Large-Scale Railway Systems". STW is part of the Netherlands Organisation for Scientific Research (NWO), and which is partly funded by the Ministry of Economic Affairs.

Appendix

Table 1: Train lines and their frequencies for the 2011 timetable of the Dutch Network. InterCity (IC) trains are interregional trains and Sprinters (SP) are local trains.

Train line	Train type	Frequency
Alkmaar - Maastricht CS	IC	every 30 minutes
Arnhem - Ede-Wageningen	SP	every 60 minutes
Arnhem - Zutphen	SP	every 30 minutes
Amsterdam CS - Almere Oostvaarders	SP	every 30 minutes
Amsterdam CS - Amersfoort	SP	every 30 minutes
Amsterdam CS - Breda	SP	every 30 minutes
Amsterdam CS - Den Haag CS	IC	every 30 minutes
Amsterdam CS - Dordrecht	IC	every 60 minutes
Amsterdam CS - Haarlem	SP	every 30 minutes
Amsterdam CS - Hoofddorp	SP	every 30 minutes
Amsterdam CS - Lelystad Centrum	IC	every 30 minutes
Amsterdam CS - Rotterdam CS	IC	every 60 minutes
Amsterdam CS - Roosendaal	IC	every 60 minutes
Amsterdam CS - Uitgeest (Haarlem)	SP	every 30 minutes
Amsterdam CS - Uitgeest (Zaandam)	SP	every 30 minutes
Amsterdam CS - Vlissingen	IC	every 60 minutes
Apeldoorn - Enschede	SP	every 30 minutes
Breukelen - Rhenen	SP	every 30 minutes
Den Haag CS - Breda	SP	every 30 minutes
Den Haag CS - Enschede	IC	every 60 minutes
Den Haag CS - Groningen	IC	every 60 minutes
Den Haag CS - Gouda Goverwelle	SP	every 30 minutes
Den Haag CS - Haarlem	SP	every 30 minutes

Continued on next page

Table 1 – continued from previous page

Train line	Train type	Frequency
Den Haag CS - Hoorn	SP	every 30 minutes
Den Haag CS - Lelystad Centrum	IC	every 30 minutes
Den Haag CS - Lelystad Centrum	SP	every 30 minutes
Den Haag CS - Utrecht CS	IC	every 30 minutes
Den Haag CS - Utrecht CS	SP	every 30 minutes
Den Haag CS - Venlo	IC	every 30 minutes
Den Haag CS - Roosendaal	SP	every 30 minutes
Den Helder - Nijmegen	IC	every 30 minutes
Enkhuizen - Amersfoort	IC	every 30 minutes
Eindhoven - Weert	SP	every 30 minutes
Eindhoven - Tilburg	SP	every 30 minutes
Groningen - Zwolle	IC	every 60 minutes
's Hertogenbosch - Nijmegen	SP	every 30 minutes
's Hertogenbosch - Deurne	SP	every 30 minutes
Hoorn - Hoofddorp	SP	every 30 minutes
Roermond - Maastricht Randwyck	SP	every 30 minutes
Roosendaal - Vlissingen	IC	every 60 minutes
Roosendaal - Zwolle	SP	every 30 minutes
Rotterdam CS - Amersfoort	IC	every 30 minutes
Rotterdam CS - Deventer	IC	every 60 minutes
Rotterdam CS - Leeuwarden	IC	every 60 minutes
Rotterdam CS - Uitgeest	SP	every 30 minutes
Schiphol - Groningen	IC	every 60 minutes
Schiphol - Eindhoven	IC	every 30 minutes
Schiphol - Enschede	IC	every 60 minutes
Schiphol - Leeuwarden	IC	every 60 minutes
Schiphol - Nijmegen	IC	every 30 minutes
Sittard - Heerlen	IC	every 30 minutes
Sittard - Heerlen	SP	every 30 minutes
Utrecht CS - Almere Centrum	IC	every 30 minutes
Utrecht CS - Breda	SP	every 30 minutes
Utrecht CS - Breukelen	SP	every 30 minutes
Utrecht CS - Hoofddorp	SP	every 30 minutes
Utrecht CS - Tiel	SP	every 30 minutes
Utrecht CS - Zwolle	SP	every 30 minutes

References

G. Caimi, M. Fuchsberger, M. Laumanns, and M. Lüthi. A model predictive control approach for discrete-time rescheduling in complex central railway station areas. *Computers and Operations Research*, 39(11):2578–2593, 2012.

F. Corman, A. D'Ariano, D. Pacciarelli, and M. Pranzo. Centralized versus distributed

systems to reschedule trains in two dispatching areas. *Public Transport*, 2:219–247, 2010.

F. Corman, A. D'Ariano, D. Pacciarelli, and M. Pranzo. Dispatching and coordination in multi-area railway traffic management. *Computers & Operations Research*, 44(1):146 – 160, 2014.

M. M. Dessouky, Q. Lu, J. Zhao, and R. C. Leachman. An exact solution procedure to determine the optimal dispatching times for complex rail networks. *IIE Transactions*, 38 (2):141–152, 2006.

Gurobi. Gurobi optimizer reference manual, 2014. URL `http://www.gurobi.com`.

B. Kersbergen, J. Rudan, T. van den Boom, and B. De Schutter. Towards railway traffic management using switching max-plus-linear systems. *Discrete Event Dynamic Systems*, pages 1–41, 2014a. ISSN 0924-6703. doi: 10.1007/s10626-014-0205-7. URL `http://dx.doi.org/10.1007/s10626-014-0205-7`.

B. Kersbergen, T. J. J. van den Boom, and B. De Schutter. Distributed model predictive control for rescheduling of railway traffic. In *Proceedings of the 17th International IEEE Conference on Intelligent Transportation Systems (ITSC2014)*, pages 2732–2737, Qingdao, China, October 2014b.

L. Meng and X. Zhou. Simultaneous train rerouting and rescheduling on an n-track network: A model reformulation with network-based cumulative flow variables. *Transportation Research Part B: Methodological*, 67(0):208–234, 2014.

J. Rudan, B. Kersbergen, T. van den Boom, and K. Hangos. Performance analysis of MILP based model predictive control algorithms for dynamic railway scheduling. In *Proceedings of the European Control Conference 2013 (ECC)*, pages 4562–4567, Zurich, Switzerland, July 2013.

J. Törnquist Krasemann. Design of an effective algorithm for fast response to the rescheduling of railway traffic during disturbances. *Transportation Research Part C: Emerging Technologies*, 20(1):62–78, 2012.