

The Netherlands  
Research School for  
Transport, Infrastructure  
and Logistics

**TRAIL**



studies  
transport

Seamless Multimodal Mobility

# **Railway Capacity Assessment, an Algebraic Approach**

Ir. R.J. van Egmond



011

727096  
W

Railway capacity assessment,  
an algebraic approach

Bibliotheek TU Delft



C 3035958

**8511  
528G**



# Railway capacity assessment, an algebraic approach

The Netherlands TRAIL Research School, Delft

**Author:**

**Robert-Jan van Egmond**

Faculty of Information Technology and Systems, Delft University of Technology

TRAIL Studies in Transportation Science Series N°. S99/2

Editors : Prof.dr.ir. P.H.L. Bovy  
Prof.dr. G.J. Olsder

Editorial Board: Prof.dr. I.A. Hansen  
Prof.dr.ir. R.E.C.M. van der Heijden  
Prof.ir. F.M. Sanders

**The Netherlands TRAIL Research School**  
Delft University of Technology /  
Erasmus University Rotterdam

TRAIL-office  
Kluyverweg 4  
P.O. Box 5017  
2600 GA Delft  
The Netherlands

Telephone : +31 (0) 15 278 60 46  
Telefax : +31 (0) 15 278 43 33  
E-mail : [mailbox@TRAIL.tudelft.nl](mailto:mailbox@TRAIL.tudelft.nl)  
Internet : <http://TRAIL.tudelft.nl>

**Sales and distribution:**

Delft University Press  
P.O. Box  
2600 MG Delft  
Telephone: +31 (0)15 278 32 54  
Telefax: +31 (0)15 278 16 61

ISBN: 90-407-1935-7

© Copyright by The Netherlands TRAIL Research School. No part of this book may be reproduced in any form by print, photoprint, microfilm or any other means without written permission of the publisher: The Netherlands TRAIL Research School.

---

TRAIL is accredited by the Royal Netherlands Academy of Arts and Sciences. It is a cooperation of the **Delft University of Technology** and **Erasmus University Rotterdam**, in which the following faculties participate: Civil Engineering & Geosciences, Design, Engineering & Production, Architecture, Information Technology & Systems, Aerospace Engineering, Technology, Policy & Management, OTB Research Institute for Housing, Urban & Mobility studies, Economic Sciences, Business Administration, Social Sciences and Law.

# Contents

<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
1.1	Simulation . . . . .	2
1.2	Queueing theoretical approaches . . . . .	2
1.3	An algebraic approach . . . . .	3
1.4	Overview of this report . . . . .	5
<b>2</b>	<b>MAXPLUS AUTOMATA THEORY</b>	<b>7</b>
2.1	Maxplus algebra . . . . .	7
2.1.1	Notations . . . . .	7
2.1.2	Event graphs . . . . .	8
2.2	An open jobshop . . . . .	9
2.3	Maxplus automaton . . . . .	11
2.4	Performance analysis . . . . .	14
2.5	Random schedules . . . . .	16
<b>3</b>	<b>TYPICAL RAILWAY ISSUES</b>	<b>19</b>
3.1	Lower and upper contours of train movements . . . . .	19
3.2	Robustness with respect to delays . . . . .	20
3.2.1	Without buffer times . . . . .	20
3.2.2	With buffer times . . . . .	21
3.2.3	Delay propagation during operations . . . . .	22
3.2.4	Stability margin . . . . .	22
3.3	Parallelism . . . . .	23
3.4	Passenger transfers . . . . .	25
3.5	Moving block system . . . . .	26

---

<b>4 CASE STUDY</b>	<b>29</b>
4.1 Building the model . . . . .	29
4.2 Results . . . . .	31
<b>5 CONCLUSIONS AND FURTHER RESEARCH</b>	<b>35</b>
<b>BIBLIOGRAPHY</b>	<b>37</b>
<b>A ROBUST SCHEDULING WITH EVENT GRAPHS</b>	<b>41</b>
<b>B DELAY PROPAGATION MATRICES OF CASE STUDY</b>	<b>47</b>

## PREFACE

This publication is a result of the research program "Seamless Multimodal Mobility", carried out within The Netherlands TRAIL Research School for Transport, Infrastructure and Logistics, and financed by the Delft University of Technology.

The "Seamless Multimodal Mobility" research program will provide tools for the design and operation of attractive and efficient multimodal passenger transport services. The study presented by this report is part of project 3, "Dependable Scheduling". The main objective of this SMM-project is to develop models for the analysis and prediction of delay propagation in service networks. In this report we restrict ourselves to delay propagation in railway transport systems, which can however be seen as a typical example of a service network.

The objective of this study is to assess the value of the maxplus algebra approach for capacity assessment of railway stations. Hence, this report can be seen as a follow up of [13], where Wakob's approach has been investigated for the same problem. This report should be understandable for both researchers and interested railway companies, represented in the Netherlands by the organizations Railned (capacity management), NS Infrastructuur (maintenance of railway infrastructure), NS Verkeersleiding (traffic control) and Holland Railconsult (railway consultancy).

The author would like to thank Bernd Heidergott, Geert-Jan Olsder and Piet Bovy for our fruitful discussions and editorial suggestions.

## SUMMARY

Train movements can be represented as staircase-shaped blocks in a distance/time diagram. When assessing the capacity of the railway infrastructure, these train movements are scheduled as fast as possible after each other. This can be visualized as piling up the staircase-shaped blocks in the distance/time diagram. Since capacity is also related to the quality of operations, buffer times are inserted between the train movements.

Piling up the train movements and buffer times in the distance/time diagram can be described mathematically by matrix multiplication in the maxplus algebra. Hence, basic results of event graph theory, to which the maxplus algebra is related, can be applied to scheduling train movements: it provides the cycletime of a given schedule and the critical circuit. With the aid of critical circuits the bottlenecks of the infrastructure can be found.

Besides finding the cycletime and the critical circuit, the maxplus approach turns out to be useful to assess delay propagation: given two arbitrary train movements of the schedule, an elegant formula arises for the maximum delay of the first train that does not disturb the second train. This leads to the delay propagation matrix, which, after weighting each entry of this matrix properly, results in a measure for the robustness of the system.

A case study has been worked out for the railway station The Hague HS as an example. The results are compared to another case study which applies a queueing theoretical model to the same station. The outcomes of the case study together with the mentioned theoretical results makes the maxplus approach a promising tool for railway capacity assessment.



# Chapter 1

## INTRODUCTION

Assessing the capacity of a railway system, a trade off has to be made between throughput and quality of operations (cf. [2],[11]). The throughput of a railway system is the number of trains that is served by the network per time unit. Quality of operations is a broad concept, it incorporates safety, reliability and punctuality. We define railway capacity as the maximum throughput under given quality conditions. In this report we focus on the punctuality aspect of the quality.

The evaluation of the capacity of railway systems is of key importance for planning the future infrastructure. We can distinguish infrastructure planning on medium term (about 5 years ahead) and long term (10 years or more ahead). Here the problem arises that the future timetable is unknown, especially for the long term planning. According to Weits (cf. [22]), methods to solve this problem can be divided into two approaches:

- Sampling timetables plus simulation,
- Constructing performance indicators using queueing theory or statistical prediction.

A natural way to deal with the uncertainty about the future timetable is to evaluate a large number of sampled timetables (cf. [17]) with the aid of simulation. This is however a time-consuming procedure. A less time-consuming procedure is to use analytical models like queueing theory or statistical prediction that incorporate the uncertainty about the timetable within the model. However, the degree of details and clarity of these approaches is less compared to simulation. In this report we introduce a third approach:

- Sampling timetables plus constructing performance indicators using algebraic analysis.

The idea is that the approach of sampling timetables can be made less time-consuming by using algebraic analysis instead of simulation, while the degree of details and clarity is largely preserved.

In the next sections, simulation, queueing theoretical approaches and the algebraic approach is described in more detail.

## 1.1 Simulation

One way to assess the capacity of a railway system is by means of simulation. In recent years much effort is made in building simulation tools for railway capacity assessment. Some examples can be found in [3] and [12]. In [12] a simulation tool is described based on the decision support system DONS (Designer of Network Schedules) which generates timetables.

With a simulation model reality can be described in more detail than with analytical models. The amount of detail is a trade off between accuracy and effort. The more detail is required, the more effort is needed to build the model and the more time simulation runs will take.

Simulation is a strong tool to evaluate timetables. However, available computation time, needed for these evaluations, is limited. Furthermore, it provides less insight than analytical tools. Hence, it is less suitable for optimizations (for instance to assess optimal buffer times).

It should be mentioned here that the algebraic approach presented in this report is also suitable for use as a simulation model.

## 1.2 Queueing theoretical approaches

Another approach to assess railway capacity is by means of queueing theory. In a queueing system, customers enter the system and wait in a queue until they are served by the server(s). The analysis of a queueing system mainly focuses on the stability and waiting times, given distributions of the interarrival times of the customers and distributions of the service times of the servers. If the system is stable (i.e. if the number of customers in the queues will not grow to infinity while time elapses), the throughput is given by the arrival rate of the customers. The waiting times are used as a measure of the quality of operations. A railway system can be modeled as a queueing system by interpreting trains as customers and infra elements as servers.

Many results have been achieved on the single server queue (cf. [5]). Many attempts have been made to extend the theory of the single server queues to networks. For specific queueing networks analytical results are known, for instance

for Jackson networks (cf. [10]), where both the interarrival times as well as the server times are assumed to be exponentially distributed.

Often queueing systems in real live are too complex to be fit in a purely analytical model and approximations are needed. This is also the case when applying queueing theory to a railway system. One way to approximate a queueing network is to consider the network as a composition of single server queues and to neglect their mutual dependencies. Such an approach for railway capacity assessment is done by Wakob (cf. [21], [13]).

Applying queueing theory to railway capacity has a number of drawbacks. One drawback is that it is necessary to make rough approximations to obtain analytical results, which decreases the accuracy substantially. Another drawback is that we are not satisfied with the way quality of the system is measured; in a queueing model it is assumed that interarrival times at which the customers enter the system are independent and one measures the time customers have to wait in the queues. However, we are interested in punctuality, which makes the timetable an important aspect. Hence, we should not focus on interarrival times but on the disturbances with respect to the timetable.

### 1.3 An algebraic approach

In this report we present an algebraic approach. In contrast to queueing theoretical models, the timetable plays an important role in the algebraic approach; the order of train movements is fixed.

Besides throughput we focus on the punctuality aspect; we investigate how robust the system is with respect to delays. This analysis is done in a rather deterministic way: given initial delays, we investigate how they propagate through the network if the processing times are deterministic. Robustness with respect to delays and throughput are related to each other by buffer times; inserting buffer times to a schedule will improve the robustness and worsen the throughput. Hence, assessing the capacity of a network can be seen as an optimization problem: find a set of buffer times that obeys robustness constraints and maximizes the throughput.

The algebraic approach presented in this report is based on maxplus automata theory. A specialization of a maxplus automaton is a so-called *heaps-of-pieces* model. The heaps-of-pieces model was first introduced by Viennot in [20] and has been related to timed Petri nets by Gaubert and Mairesse in [7]. This model has triggered us to use it for railway capacity assessment, but then we discovered that the more general theory of maxplus automata was needed.

Let us give an intuitive view on the heaps-of-pieces model from a railway perspective. The capacity of a railway network is mainly restricted by the safety

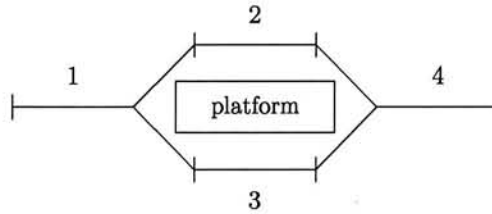


Figure 1.1: A railway network example

system. The railway safety system (as it is currently in use) divides a network into disjunct infra elements. Figure 1.1 shows an example of a network which has been divided into four infra elements. When a train uses such an infra element, this infra element is not accessible for other trains during some time. This time duration does not only take the physical occupation into account, but also the braking distance, reaction time, safety margin and the time needed to turn a switch. Figure 1.2 shows how four different trains occupy the different infra ele-

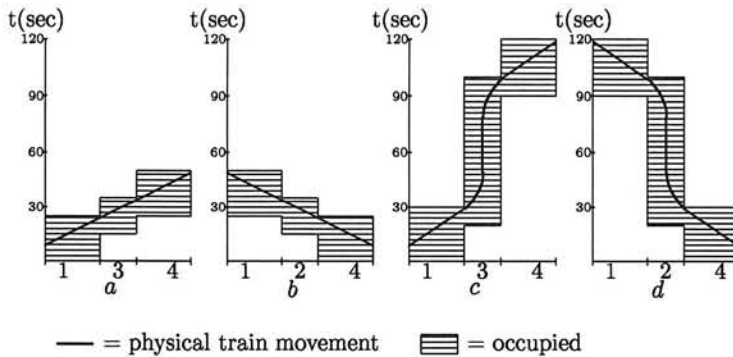


Figure 1.2: Occupations of the network by four different trains

ments of the network example. The position of the trains on the infra elements has been set out horizontally, whereas time has been set out vertically. The staircase-shaped blocks in these diagrams indicate when each infra element of the network is not accessible for other trains. The four different trains consist of an express train in each direction (trains *a* and *b*) and a stop train in each direction (trains *c* and *d*). From Figure 1.2 one can read that the stop trains dwell at the platform and that trains traveling from left to right use infra element 3 whereas trains traveling from right to left use infra element 2.

Train movements can thus be visualized by staircase-shaped blocks (see also [23]), which are the *pieces* of the model. Scheduling multiple trains after each other can be visualized by piling up one train movement above the other in the distance/time diagram, resulting in a *heap of pieces* (see Figure 1.3). This kind

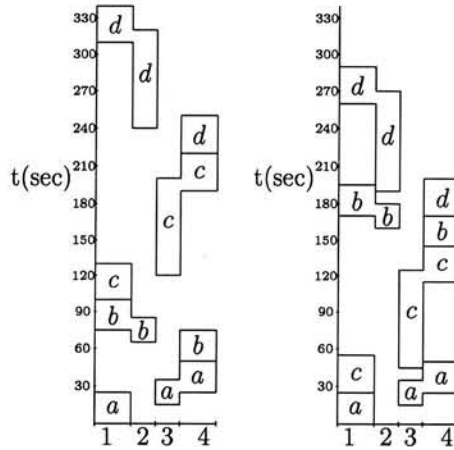


Figure 1.3: Schedules *abcd* and *acbd*

of visualization is very similar to the well known Tetris game. 'Dropping' train movements on top of each other represents processing these train movements as fast as possible after each other, which is a natural thing to do for capacity assessment. The time needed to process a schedule of train movements can be obtained by calculating the height of the *heap* in the distance/time diagram. There appears to be an elegant way to describe this kind of 'dropping' operations by mathematical equations, namely by multiplying matrices in some 'exotic' algebra: the maxplus algebra.

## 1.4 Overview of this report

This report is built up as follows. Section 2 introduces the maxplus automata theory. This theory provides tools for analyzing a certain type of jobshops, namely the jobshops that are linear in the maxplus algebra. Section 3 deals with some typical railway issues that appear when applying the theory to railway capacity: delay propagation, moving block systems, passenger transfers, etc. To assess the value of the algebraic approach, a case study has been worked out in Section 4. In this case study the railway station The Hague HS (The Netherlands) is taken as an example. Section 5 contains the conclusions of this report.



# Chapter 2

## MAXPLUS AUTOMATA THEORY

### 2.1 Maxplus algebra

In this section we give an overview of basic definitions and theorems of maxplus algebra. This structure was introduced in [6]. For an extensive discussion of the maxplus algebra and similar structures we refer to [1].

#### 2.1.1 Notations

Let  $\epsilon = -\infty$  and denote by  $\mathbb{IR}_\epsilon$  the set  $\mathbb{IR} \cup \{\epsilon\}$ . For elements  $a, b \in \mathbb{IR}_\epsilon$  we define the operations  $\oplus$  and  $\otimes$  by

$$a \oplus b = \max(a, b) \tag{2.1}$$

and

$$a \otimes b = a + b, \tag{2.2}$$

where we adopt the convention that for all  $a \in \mathbb{IR}$  we have  $\max(a, -\infty) = \max(-\infty, a) = a$  and  $a + (-\infty) = -\infty + a = -\infty$ . The structure  $\mathbb{IR}_\epsilon$  together with the operations  $\oplus$  and  $\otimes$  is called the maxplus algebra and is denoted by  $\mathbb{IR}_{\max}$ . In particular,  $\epsilon$  is the neutral element for the operation  $\oplus$  and absorbing for  $\otimes$ , that is, for all  $a \in \mathbb{IR}_\epsilon$   $a \otimes \epsilon = \epsilon$ . The neutral element for  $\otimes$  is 0, which is therefore also denoted by  $e$ .

We can extend the maxplus algebra operations to matrices. We denote matrices by capitals and their entries by small letters. We define the following operators on matrices: If  $A, B \in \mathbb{IR}_\epsilon^{m \times n}$  then

$$(A \oplus B)_{ij} = a_{ij} \oplus b_{ij}, \quad i = 1, \dots, m; j = 1, \dots, n. \tag{2.3}$$

If  $A \in \mathbb{R}_\epsilon^{m \times p}$  and  $B \in \mathbb{R}_\epsilon^{p \times n}$  then

$$(A \otimes B)_{ij} = \bigoplus_{k=1}^p a_{ik} \otimes b_{kj} = \max_k (a_{ik} + b_{kj}), \quad i = 1, \dots, m; j = 1, \dots, n. \quad (2.4)$$

For submatrices we use the notation  $A_{I \times J}$ , denoting the submatrix of  $A$  consisting of the rows corresponding to index set  $I$  and the columns corresponding to index set  $J$ .

### 2.1.2 Event graphs

Maxplus algebra finds its application in (timed) event graphs. In the theory of event graphs we investigate systems which are driven by events, for instance the behaviour of a railway system can be described by the arrivals and departures of trains at stations. The order of events in such systems is prescribed by a graph: an arc  $(i, j)$  prescribes that event  $i$  may not occur before event  $j$  has occurred. In timed event graphs we furthermore assign weights  $a_{ij}$  to the arcs of the graph, indicating that event  $i$  occurs at earliest  $a_{ij}$  time units after event  $j$  has occurred.

There is a one to one relation between timed event graphs and maxplus matrices, as the following definitions show:

**Definition 1** (*Graph corresponding to a matrix, arc weight*) The directed graph  $\mathcal{G}(A)$  corresponding to an  $n \times n$  matrix  $A$  is a pair  $(V, E)$  where  $V$  is a the set of nodes numbered from 1 to  $n$  and  $E$  is a set of arcs such that  $(j, i) \in E$  if and only if  $a_{ij} > \epsilon$ . The value  $a_{ij}$  is the weight of arc  $(j, i)$ .

Conversely, we can obtain a matrix from a directed graph.

**Definition 2** (*Matrix corresponding to a graph*) For a given directed graph  $\mathcal{G} = (V, E)$ , the corresponding matrix  $A$  is obtained as the  $|V| \times |V|$  matrix having entry  $a_{ij}$  equal to the weight of the arc  $(j, i) \in E$ . If no arc is present from node  $j$  to node  $i$ , then  $a_{ij} \stackrel{\text{def}}{=} \epsilon$ .

Eigenvalues and eigenvectors of a matrix in the maxplus algebra are defined in a similar way as in conventional algebra:

**Definition 3** (*Eigenvalue, eigenvector*) If for an  $n \times n$  matrix  $A$  a non-trivial vector  $v$  (i.e.  $v_i \neq \epsilon$ , for at least one  $i \in \{1, \dots, n\}$ ) and a number  $\lambda$  exist such that

$$A \otimes v = \lambda \otimes v$$

then  $v$  is called an eigenvector and  $\lambda$  is called an eigenvalue of matrix  $A$ .



The eigenvalue can be interpreted as the cycletime of the system, whereas an eigenvector can be interpreted as an initial condition for which the system has a regular behaviour. For instance, if the events correspond to departures of trains in a railway system then starting with an eigenvector  $v$  results in the cyclic timetable  $\lambda^k \otimes v$  (see also [4]).

**Definition 4** (*Average weight, circuit mean*) The average weight of a path  $\rho = (i_1 \rightarrow i_2 \rightarrow \dots \rightarrow i_k)$  is defined as the sum of the weights of the individual weights divided by the number of arcs, i.e.  $(a_{i_2 i_1} + a_{i_3 i_2} + \dots + a_{i_k i_{k-1}})/(k-1)$ . The average weight of a circuit  $\rho = (i_1 \rightarrow i_2 \rightarrow \dots \rightarrow i_k \rightarrow i_1)$  is called circuit mean.

**Definition 5** (*Strongly connected graph*) A graph is called strongly connected if a path exists from any node to any other node. If a graph is strongly connected, the corresponding matrix is called irreducible.

**Theorem 1** An irreducible matrix has a unique eigenvalue equal to the maximum circuit mean of the corresponding (strongly connected) graph.

**Definition 6** (*Critical circuit*) Any circuit with circuit mean equal to the maximum circuit mean is called a critical circuit.

Although a critical circuit is not necessarily unique, it is often unique in practice. Since the critical circuit determines the eigenvalue (i.e. the cycletime) it can be interpreted as the bottleneck of the system: the system can only be made faster through changes on the critical circuit.

Event graphs can be used to analyze delay propagation. Since we focus in this publication on the automaton approach, we have put the theory of delay propagation in event graphs apart in Appendix I.

## 2.2 An open jobshop

Before we give a formal definition of a maxplus automaton, we first show that certain jobshop processes that are related to this kind of automata can be described by maxplus-linear equations.

Consider a shop with  $m$  machines and  $n$  different jobs. We define a schedule as a sequence  $(l_k, s(l_k), f(l_k))$ . Here,  $l_k$  refers to the  $k^{\text{th}}$  job in the schedule. Vectors  $s(l_k)$  and  $f(l_k)$  indicate the starting and finishing moments of  $l_k$  on the machines, more precisely:  $s_i(l_k)$  is the starting moment of job  $l_k$  on machine  $i$  and  $f_i(l_k)$  is the finishing moment of job  $l_k$  on machine  $i$ . The synchronization of these

starting and finishing moments is of main importance for the kind of shops we are investigating. The machines have to wait for each other when starting as well as when finishing the job. Several reasons can be considered for synchronization, which can be split up into synchronizations within the jobs and synchronizations between the jobs. Examples of *within* synchronizations are the manufacture in a chemical plant which may not cool down during production, a train which must keep running in a railway system. Examples of *between* synchronizations are trains which have to wait for one another when using common infrastructure, a processor in a computer which needs results from other processors. We may think of this synchronization as if the machines sent messages to each other indicating that the machine receiving the message is allowed to start or to finish its job. A machine actually starts (finishes) its job as soon as it has received a starting (finishing) message from each machine. Suppose that machine  $j$  sends a starting message to machine  $i$  at the moment machine  $j$  finishes on job  $l_{k-1}$  plus  $a_{ij}(l_k)$  time units and that machine  $j$  sends a finishing message to machine  $i$  at the moment machine  $j$  starts on job  $l_k$  plus  $b_{ij}(l_k)$  time units. It should be noticed that a machine also sends messages to itself and that  $a_{ij}(l_k)$  and  $b_{ij}(l_k)$  may be negative, i.e. a machine may send starting (finishing) messages before it has finished (started) its own job. The starting and finishing moments can then be obtained from the following equations:

$$s_j(l_k) = \max_i \{f_i(l_{k-1}) + a_{ij}(l_k)\} \quad (2.5)$$

$$f_j(l_k) = \max_i \{s_i(l_k) + b_{ij}(l_k)\}. \quad (2.6)$$

In most practical examples the length of the intervals  $[s_i(l_k), f_i(l_k)]$  are fixed, i.e. the time needed for job  $l_k$  on machine  $i$  does not depend on  $f(l_{k-1})$ . In that case we can set  $b_{ij}(l_k) = \varepsilon, i \neq j$  and we have  $f_i(l_k) = s_i(l_k) + b_{ii}(l_k)$ . However, in Section 3.3 we shall give an example in which the lengths of these intervals are not fixed.

Equations (2.5) and (2.6) are linear in the maxplus algebra and by using maxplus notations we obtain:

$$s(l_k) = f(l_{k-1}) \otimes A(l_k) \quad (2.7)$$

$$f(l_k) = s(l_k) \otimes B(l_k) \quad (2.8)$$

$$\Rightarrow f(l_k) = f(l_{k-1}) \otimes A(l_k) \otimes B(l_k) \quad (2.9)$$

There is a nice way to visualize this kind of processes (although this is only possible for certain types of jobs as we will explain further on). When we put the machines along the horizontal axis and time along the vertical axis then jobs can be visualized by solid blocks in the machine/time diagram. The time evolution of processing a schedule of jobs can be visualized by piling up these blocks. In

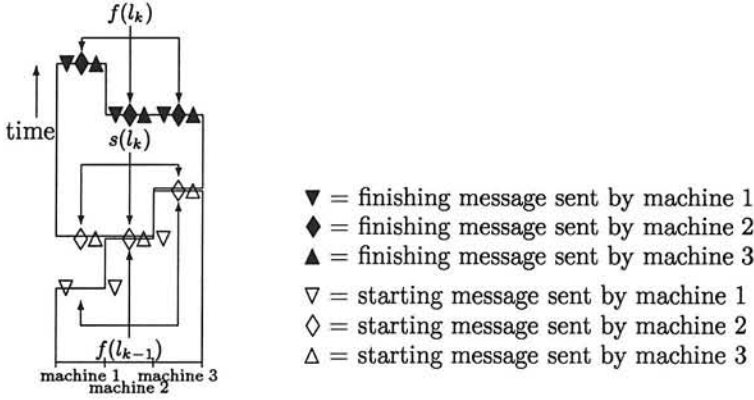


Figure 2.1: Visualization of processing job  $l_k$

Figure 2.1 an example with three machines is shown. The moments at which these machines finish on job  $l_{k-1}$  are indicated by  $f(l_{k-1})$ . Job  $l_k$  is visualized by a solid block of which the lower contour is given by  $s(l_k)$  and the upper contour is given by  $f(l_k)$ . Besides we indicated the moments at which the messages are sent. These moments correspond to the shape of the solid block.

### 2.3 Maxplus automaton

From the previous section we have got an idea of the kind of processes that may favourably be modeled by maxplus automata. Next we give a formal definition of a maxplus automaton. Here we use the terms resources and tasks instead of machines and jobs.

**Definition 7 (Maxplus automaton)** A maxplus automaton is a triple  $\mathcal{H} = (T, \mathcal{R}, \mathcal{M})$ , where

- $T$  is a finite set (alphabet) whose elements are called tasks. We denote by  $T^n$  the set of words of length  $n$  on  $T$  and by  $T^*$  the set of finite words equipped with concatenation.
- $\mathcal{R}$  is a finite set whose elements are called resources.
- $\mathcal{M}$  is a morphism  $T^* \rightarrow \mathbb{R}_{\max}^{\mathcal{R} \times \mathcal{R}}$  which is uniquely specified by the finite family of matrices  $\mathcal{M}(l), l \in T$ . For a word  $w = l_1 \cdots l_n$ , we have  $\mathcal{M}(w) = \mathcal{M}(l_1 \cdots l_n) = \mathcal{M}(l_1) \otimes \cdots \otimes \mathcal{M}(l_n)$ .

Since a word is a concatenation of tasks, we also use the term *schedule* instead of word.

**Definition 8** (*Upper contour*) We define the upper contour of a schedule  $w$  as the  $\mathcal{R}$ -dimensional row vector  $x(w)$  which satisfies

$$x(w) = x(e) \otimes \mathcal{M}(w),$$

where  $x(e)$ , the upper contour of the empty schedule (word), is a given  $\mathcal{R}$ -dimensional row vector.

The value  $x_i(w)$  must be interpreted as the moment of releasing resource  $i$  after processing schedule  $w$ , whereas the initial condition  $x(e)$  must be interpreted as the moments at which the resources are available to process their first task. Often we assume that all resources are available at  $t = 0$ , i.e.  $x(e) = [e, \dots, e]$ , which is also indicated as 'the horizontal ground assumption'.

Since we have that  $\mathcal{M}(wl) = \mathcal{M}(w) \otimes \mathcal{M}(l)$  the evolution of upper contours can be written as

$$x(wl) = x(w) \otimes \mathcal{M}(l). \quad (2.10)$$

It should be noticed that in most literature about maxplus algebra, dynamical systems of the form  $x(k) = A(k) \otimes x(k-1)$  appear. Equation (2.10) can also be written in this form by the following transformation:

$$A(k) := (\mathcal{M}(l_k))^T \quad (2.11)$$

$$x(k) := x(l_1 \cdots l_k)^T. \quad (2.12)$$

However, the form of (2.10) has the advantage that we can concatenate schedules (words) which can be read from left to right.

**Definition 9** Given a task  $l \in \mathcal{T}$  we define  $R(l)$  as the subset of resources used by task  $l$ .

Since task  $l$  leaves the releases of resources  $i \notin R(l)$  unchanged, for each  $i \notin R(l)$  it holds that  $\mathcal{M}_{ii}(l) = e$  and  $\mathcal{M}_{ij}(l) = \varepsilon, j \neq i$ . As a consequence, it is possible to obtain  $R(l)$  from the matrix  $\mathcal{M}(l)$ .

Next we define elementary tasks. These are the tasks that can be visualized as solid blocks. It should be noticed that in the definition of a heaps-of-pieces model in [7], the pieces are defined as what we call elementary tasks. The reason that we introduce different definitions is that we need also non-elementary tasks for our traffic application.

**Definition 10** (*Elementary task*) A task  $l$  is called an elementary task if  $s(l)$  and  $f(l)$ ,  $\mathcal{R}$ -dimensional row vectors in  $\mathbb{R}_{\max}$ , exist such that  $s(l) \leq f(l)$  and

$$\mathcal{M}_{ij}(l) = \begin{cases} e & \text{if } i = j \text{ and } i \notin R(l) \\ f_j(l) - s_i(l) & \text{if } i \in R(l) \text{ and } j \in R(l) \\ \varepsilon & \text{otherwise.} \end{cases}$$

We refer to  $s(l)$  and  $f(l)$  as the lower contour and the upper contour of task  $l$ . By convention we choose  $s_i(l) = f_i(l) = \varepsilon$  if  $i \notin R(l)$  and  $\min_{i \in R(l)} s_i(l) = e$ , which specifies  $s(l)$  and  $f(l)$  uniquely.

The reader should not confuse solid blocks with 'connected' blocks. Elementary tasks represent solid blocks which are not necessarily 'connected'. The blocks visualized in Figure 1.3 are examples of solid blocks which are not connected.

**Definition 11 (Rank)** We define the rank of a matrix  $A$  as the number of additively independent columns (resp. rows) of  $A$ . More precisely, a subset of columns  $A_S, S \subset \{1, \dots, n\}$ , is called additively dependent if

$$A_{.i} = \bigoplus_{j \in S, j \neq i} \alpha_j \otimes A_{.j}$$

for some  $i \in S$  and  $\alpha_j \in \mathbb{R}_{\max}$ . Else the columns are called additively independent. The rank of a matrix is the maximum cardinality of all subsets consisting of additively independent columns.

**Lemma 1** Given an  $n \times n$  matrix  $A$ , let  $I$  be the set of indices  $i$  with  $a_{ii} = e$  and  $a_{ji} = a_{ij} = \varepsilon$ ,  $j \neq i$ . Let  $R = \{1, \dots, n\}/I$ . Matrix  $A$  represents an elementary task if and only if  $A_{R \times R}$  has rank 1.

**Example 1** Consider the staircase-shaped blocks in Figure 1.2 as tasks in a max-plus automaton with four resources. These are elementary tasks with the following upper and lower contours:

$$\begin{aligned} f(a) &= [25, \varepsilon, 35, 50] & f(b) &= [50, 35, \varepsilon, 25] \\ s(a) &= [0, \varepsilon, 15, 25] & s(b) &= [25, 15, \varepsilon, 0] \\ f(c) &= [30, \varepsilon, 100, 120] & f(d) &= [120, 100, \varepsilon, 30] \\ s(c) &= [0, \varepsilon, 20, 90] & s(d) &= [90, 20, \varepsilon, 0]. \end{aligned}$$

The matrices corresponding to these tasks can be found with the aid of definition 10:

$$\begin{aligned} \mathcal{M}(a) &= \begin{bmatrix} 25 & \varepsilon & 35 & 50 \\ \varepsilon & e & \varepsilon & \varepsilon \\ 10 & \varepsilon & 20 & 35 \\ 0 & \varepsilon & 10 & 25 \end{bmatrix} & \mathcal{M}(b) &= \begin{bmatrix} 25 & 10 & \varepsilon & 0 \\ 35 & 20 & \varepsilon & 10 \\ \varepsilon & \varepsilon & e & \varepsilon \\ 50 & 35 & \varepsilon & 25 \end{bmatrix} \\ \mathcal{M}(c) &= \begin{bmatrix} 30 & \varepsilon & 100 & 120 \\ \varepsilon & e & \varepsilon & \varepsilon \\ 10 & \varepsilon & 80 & 100 \\ -60 & \varepsilon & 10 & 30 \end{bmatrix} & \mathcal{M}(d) &= \begin{bmatrix} 30 & 10 & \varepsilon & -60 \\ 100 & 80 & \varepsilon & 10 \\ \varepsilon & \varepsilon & e & \varepsilon \\ 120 & 100 & \varepsilon & 30 \end{bmatrix}. \end{aligned}$$

In Figure 2.2 the visualization of schedule  $aa$  and schedule  $ab$  are shown. The

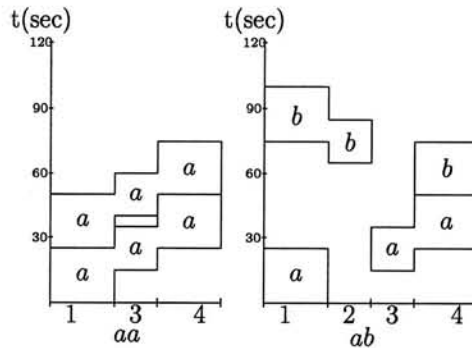


Figure 2.2: Visualization of schedules  $aa$  and  $ab$

matrices that describe these schedules can be found by matrix multiplication:

$$\mathcal{M}(aa) = \mathcal{M}(a) \otimes \mathcal{M}(a) = \begin{bmatrix} 50 & \varepsilon & 60 & 75 \\ \varepsilon & e & \varepsilon & \varepsilon \\ 35 & \varepsilon & 45 & 60 \\ 25 & \varepsilon & 35 & 50 \end{bmatrix}$$

$$\mathcal{M}(ab) = \mathcal{M}(a) \otimes \mathcal{M}(b) = \begin{bmatrix} 100 & 85 & 35 & 75 \\ 35 & 20 & \varepsilon & 10 \\ 85 & 70 & 20 & 60 \\ 75 & 60 & 10 & 50 \end{bmatrix}$$

The task corresponding to  $\mathcal{M}(aa)$  is again an elementary task with

$$f(aa) = [50, \varepsilon, 60, 75]$$

$$s(aa) = [0, \varepsilon, 15, 25]$$

but the task corresponding to  $\mathcal{M}(ab)$  is not an elementary task. The upper contour of this schedule (under the horizontal ground assumption) equals  $x(e)\mathcal{M}(ab) = [100, 85, 35, 75]$ .

## 2.4 Performance analysis

In this section we investigate periodical schedules. In a periodical schedule a given pattern  $w = l_1 \cdots l_k$  is repeated over and over again. We investigate how many times such a pattern can be repeated per time unit. To answer this question, we solve the eigenvalue problem:

$$v \otimes \mathcal{M}(w) = v \otimes \lambda. \quad (2.13)$$

We may interpret eigenvector  $v$  as an initial condition which results in a regular completion and eigenvalue  $\lambda$  as the cycletime of the pattern (see [1]). It is well known that if  $\mathcal{M}(w)$  is an irreducible matrix, any (finite) initial condition leads to the same average cycletime (see [1]). Thus, the maximum frequency of pattern  $w$  equals  $1/\lambda$ .

**Example 2** Consider again the tasks of Example 1. From Figure 2.2 it can be obtained that the cycletime of task  $a$  is 25. The matrix  $\mathcal{M}(a)$  is reducible: it can be split into a part which consists of resources 1,3 and 4 and a part which consist of resource 2 (i.e., the set of resources not used by  $a$ ). The eigenvalue of the first part is 25, whereas the second part has eigenvalue  $e$ .

The cycletime of the schedule  $abcd$  equals 340 (see left picture of Figure 1.3). A faster schedule is  $acbd$  which has cycletime 290 (see right picture of Figure 1.3). So, if the time unit is seconds, the schedule  $abcd$  can be repeated 10 times per hour (40 trains per hour) whereas the schedule  $acbd$  can be repeated 12 times per hour (48 trains per hour).

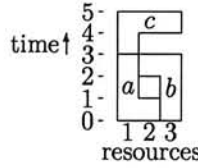
In [7] a comparison is made between the automata approach and the event graph approach. It is shown there that a given schedule can be modeled by an event graph as well. The complexity of calculating the eigenvalue with the automaton approach is of the same order as with the event graph approach, if the complexity of building the event graph is excluded. This illustrates the strength of the automaton approach: each schedule can easily be analyzed by multiplying the appropriate matrices, whereas in the event graph approach a new model has to be build for each schedule. Moreover, the size of the event graph grows with the length of the pattern, whereas the size of the automaton remains constant.

In order to identify bottlenecks of the system, we look for critical tasks of pattern  $w$ . A subpattern of  $w$  is called critical if its cycletime equals the cycletime of  $w$  and the cycletime decreases as soon as one task is left out of the subpattern. A task  $l$  is called critical if a critical subpattern containing  $l$  exists. Critical tasks are thus those tasks that determine the cycletime of the whole pattern.

Critical subpatterns are the equivalent of critical circuits in event graphs and we can obtain the critical tasks with the aid of critical circuits in event graphs corresponding to the automaton with schedule  $w$ :

**Lemma 2** Consider a pattern  $w = l_1 \cdots l_n$ . Task  $l_k$  is critical if and only if the critical circuit of  $\mathcal{M}(l_k \cdots l_n l_1 \cdots l_{k-1})$  contains at least one resource  $\in R(l_k)$ .

**Example 3** Consider tasks  $a, b$  and  $c$  as in Figure 2.3. To assess the critical tasks in the pattern  $abc$  we proceed as follows: The set of critical nodes of  $\mathcal{M}(abc)$  is  $\{1, 2\}$  and  $R(a) = \{1, 2\}$ , so task  $a$  is critical. The set of critical nodes of  $\mathcal{M}(bca)$

Figure 2.3: Tasks  $a$  and  $c$  are critical

is  $\{1\}$  and  $R(b) = \{2, 3\}$ , so task  $b$  is not critical. Finally, the set of critical nodes of  $\mathcal{M}(cab)$  is  $\{1\}$  and  $R(c) = \{1, 2, 3\}$ , so task  $c$  is critical. The cycletime of the critical subpattern  $ac$  equals the cycletime of the whole pattern  $abc$  which is 5.

## 2.5 Random schedules

In this section we investigate a random schedule  $\{l_k : k \in \mathbb{N}\}$  being an independent and identically distributed sequence where  $l_k$  is the type of the  $k^{\text{th}}$  task which equals  $s_j \in S = \{s_1, s_2, \dots, s_m\}$  with  $P(l_k = s_j) = p_j > 0$ . This leads to the following evolution equation for the upper contour

$$x(k) = x(k-1) \otimes \mathcal{M}(l_k), \quad k = 1, 2, \dots, \quad (2.14)$$

where  $x(k)$  is shorthand for  $x(l_1 \dots l_k)$ . We further assume that we start with a flat ground, i.e.  $x(0) = [e, \dots, e]$ . We are interested in  $\lim_{k \rightarrow \infty} \frac{x(k)}{k}$ . In order to obtain this limit we define

$$z_i(k) = x_i(k) - x_1(k) \quad i = 2, \dots, |\mathcal{R}| \quad (2.15)$$

$$d(k) = x_1(k) - x_1(k-1). \quad (2.16)$$

We refer to  $z(k)$  as the *profile* and to  $d(k)$  as the *growthrate* corresponding to the sequence  $\{x(k)\}$ .

**Theorem 2** Consider the random schedule  $\{l_k\}$  described above. Suppose the following conditions holds:

- i.  $\mathcal{M}(s_1 s_2 \dots s_m)$  is irreducible,
- ii.  $\exists N$  such that  $(\mathcal{M}(s_1 s_2 \dots s_m))^n$  has rank 1 for  $n \geq N$ .

Then

- a.  $z(k)$  converges with strong coupling to a unique stationary regime, with stationary distribution  $\pi$ ,



b.  $\lambda \in \mathbb{R}$  exists such that

$$\lim_{k \rightarrow \infty} \frac{x_i(k)}{k} = \lambda = E_\pi(d(1)) \quad \text{a.s.} \quad i \in \{1, \dots, |\mathcal{R}|\}$$

the expectation of  $d(1)$ , given that the distribution of  $z(0)$  equals  $\pi$ .

Moreover conditions *i.* and *ii.* are necessary conditions for the model to have a unique stationary regime.

*Sketch of the proof:* This theorem can be proven with the aid of two existing theorems in the literature. In [14] sufficient and necessary conditions are given for convergence with strong coupling to a unique stationary regime. For our case, these conditions turn out to be conditions *i.* and *ii.*

Statement *b.* is proven in [19] for the case  $\{z(k)\}$  is an irreducible, positive recurrent Markov chain. Since  $(\mathcal{M}(s_1 s_2 \cdots s_m))^N$  has rank 1, it has a one-dimensional image space which corresponds to a unique profile. The event  $(s_1 s_2 \cdots s_m)^N$  (i.e. the pattern  $s_1 s_2 \cdots s_m$  repeated  $N$  times) has a finite mean recurrence time. Thus the sequence  $\{z(k)\}$  couples within finite time to an irreducible, positive recurrent Markov chain a.s.

□

The value  $\lambda^{-1}$  gives the average number of tasks that can be processed per time unit and is therefore called the throughput. In fact Theorem 2 says that under conditions *i.* and *ii.* the throughput of each resource is the same. Although statement *b.* of Theorem 2 gives an explicit formula for the throughput, it can not be used for calculating the throughput, since it is hard to find the stationary distribution in general. Still, statement *b.* can be useful for approximations of the throughput.

Next we give two examples in which the conditions of Theorem 2 are not fulfilled.

**Example 4** Consider tasks *a* and *b* with

$$\mathcal{M}(a) = \begin{bmatrix} 2 & \varepsilon \\ \varepsilon & e \end{bmatrix} \quad \text{and} \quad \mathcal{M}(b) = \begin{bmatrix} e & \varepsilon \\ \varepsilon & 3 \end{bmatrix}.$$

Thus, task *a* only uses resource 1 (for 2 time units) and task *b* only uses resource 2 (for 3 time units). Let  $p = P(t_k = a) = 1 - P(t_k = b)$ . If  $p \neq \frac{3}{5}$ , each profile is transient since the resources have different average growth rates then. Because condition *i.* of Theorem 2 is not fulfilled we can split the network, namely into a network which consists of resource 1 and a network which consists of resource 2. Then we have for the first network  $\lambda = 2p$  and for the second network  $\lambda = 3(1-p)$ . For  $p = \frac{3}{5}$  both subnetworks have the same throughput, although the stationary regime of  $\{z(k)\}$  depends on the initial condition.

**Example 5** In Section 3.3 parallel tasks will be investigated. Ahead of this section we use the matrix of the parallel processing of tasks  $c$  and  $d$  as an example of a matrix of which all powers have rank higher than 1. This matrix yields:

$$\mathcal{M}(\|\{c, d\}) = \begin{bmatrix} 60 & 40 & 100 & 120 \\ 100 & 80 & 20 & 40 \\ 40 & 20 & 80 & 100 \\ 120 & 100 & 40 & 60 \end{bmatrix}.$$

Let  $P(l_k = \|\{c, d\}) = 1, k \in \mathbb{N}$ . Then each matrix corresponding to a pattern is a power of  $\mathcal{M}(\|\{c, d\})$  which are irreducible. However, all powers of this matrix have rank 2 and condition ii. of Theorem 2 is not fulfilled. Hence, the stationary distribution depends on the initial condition. Indeed, initial condition  $x(0) = [0, 0, 0, 0]$  results in:

$$\begin{aligned} z(0) &= [0, 0, 0] \\ z(1) &= [-20, -20, 0] \\ z(2) &= [-20, -20, 0] \\ z(3) &= [-20, -20, 0] \\ z(4) &= [-20, -20, 0] \\ &\text{etc.} \end{aligned}$$

whereas the initial condition  $x(0) = [0, 0, 0, 20]$  results in:

$$\begin{aligned} z(0) &= [0, 0, 20] \\ z(1) &= [-20, -40, -20] \\ z(2) &= [-20, 0, 20] \\ z(3) &= [-20, -40, -20] \\ z(4) &= [-20, 0, 20] \\ &\text{etc.} \end{aligned}$$

Although this example is degenerated to a deterministic model, it illustrates that the uniqueness of the stationary distribution of  $z$  is not guaranteed when condition ii. of Theorem 2 is not fulfilled.

# Chapter 3

## TYPICAL RAILWAY ISSUES

### 3.1 Lower and upper contours of train movements

The lower and upper contours of tasks that describe train movements are given by the moments at which the resources (infra elements) are occupied and released by a train. Physically, a train occupies a resource from the moment the front of the train enters the resource until the back of the train leaves the resource. The physical occupation by the train is however not the only aspect which determines the occupation and release of a resource. The total time that a train keeps a resource occupied depends on the following:

1. physical occupation,
2. braking distance,
3. reaction time,
4. switch time,
5. safety margin.

Figure 3.1 gives a visualization of the subdivision of the total occupation time. Only the fourth of these aspects, the time needed to turn a switch, depends on the order of trains. So apart from switch times, the occupation of the resources by a train can be described by an upper and a lower contour. Hence, train movements can be modeled by elementary tasks in a maxplus automaton. We can overcome the problem of switch times by always including the switch time, independent of the order of trains. This will result in a conservative approximation of the capacity.

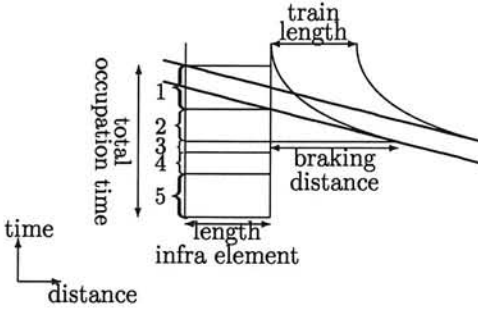


Figure 3.1: Subdivision of total occupation time of an infra element by a train

## 3.2 Robustness with respect to delays

As mentioned in the introduction, railway capacity is the result of a trade off between throughput and quality. Focusing on punctuality, we may define railway capacity as the maximum frequency of a pattern of train movements under the condition that delays do not propagate too much. In order to realize acceptable delay propagation, buffer times are added to the schedule. In this section we investigate how delays propagate in a given schedule with or without buffer times.

We assume that the initial delay is known in advance and that the train movements are shifted to a later time slot. Thus, the propagated delays obtained in this way must be seen as scheduled waiting times.

### 3.2.1 Without buffer times

Suppose task  $a$  has an initial delay of  $\tau$  time units, which means that after processing  $a$ , each resource in  $R(a)$  is released  $\tau$  time units late. This delayed task, denoted by  $a_\tau$ , can be described by the matrix:

$$\mathcal{M}_{ij}(a_\tau) = \begin{cases} \mathcal{M}_{ij}(a) + \tau & \text{if } i \in R(a) \text{ and } j \in R(a) \\ \mathcal{M}_{ij}(a) & \text{otherwise.} \end{cases} \quad (3.1)$$

**Definition 12** Given an upper contour  $x$ , task  $a$  and schedule  $w$ , we denote by  $m_i(x, a, w)$  the maximum delay  $\tau$  of task  $a$  such that the release of resource  $i$  after processing  $a_\tau w$ , starting from  $x$ , is not delayed. In formula:

$$m_i(x, a, w) = \max\{\tau \mid (x\mathcal{M}(a_\tau w))_i \leq (x\mathcal{M}(aw))_i\}$$

Vector  $m$  can be obtained using the following lemma:

**Lemma 3** *Let  $a$  be a task and  $w$  a schedule of tasks. When  $a$  and  $w$  are not equipped with buffer times, then*

$$m(x, a, w) = x_{\mathcal{R}} \mathcal{M}_{\mathcal{R} \times \mathcal{R}}(aw) - x_{R(a)} \mathcal{M}_{R(a) \times \mathcal{R}}(aw).$$

*Proof:*

$$\begin{aligned} m_i(x, a, w) &= \max\{\tau \mid x_{\mathcal{R}} \otimes \mathcal{M}_{\mathcal{R},i}(a_{\tau}w) \leq x_{\mathcal{R}} \otimes \mathcal{M}_{\mathcal{R},i}(aw)\} \\ &= \max\{\tau \mid x_{R(a)} \otimes \mathcal{M}_{R(a),i}(a_{\tau}w) \leq x_{\mathcal{R}} \otimes \mathcal{M}_{\mathcal{R},i}(aw)\} \\ &= \max\{\tau \mid x_{R(a)} \otimes \mathcal{M}_{R(a),i}(aw) + \tau \leq x_{\mathcal{R}} \otimes \mathcal{M}_{\mathcal{R},i}(aw)\} \\ &= x_{\mathcal{R}} \otimes \mathcal{M}_{\mathcal{R},i}(aw) - x_{R(a)} \otimes \mathcal{M}_{R(a),i}(aw) \end{aligned}$$

□

**Example 6** *We illustrate this lemma by the following example. Suppose we have an upper contour  $x$ , task  $a$  and  $b$  satisfying*

$$\begin{array}{lll} x = [e, e, e] & u(a, \cdot) = [2, 1, \varepsilon] & u(b, \cdot) = [\varepsilon, 3, 4] \\ & l(a, \cdot) = [0, 0, \varepsilon] & l(b, \cdot) = [\varepsilon, 2, 0]. \end{array}$$

*In Figure 3.2 the visualization of  $x_{\mathcal{R}} \mathcal{M}_{\mathcal{R} \times \mathcal{R}}(ab)$  (left) as well as the visualization of  $x_{R(a)} \mathcal{M}_{R(a) \times \mathcal{R}}(ab)$  (right) are drawn. We have:*

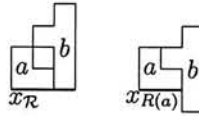


Figure 3.2: Procedure to obtain delay propagation

$$\begin{aligned} x_{\mathcal{R}} \mathcal{M}_{\mathcal{R} \times \mathcal{R}}(ab) &= [2, 3, 4] \\ x_{R(a)} \mathcal{M}_{R(a) \times \mathcal{R}}(ab) &= \underline{[2, 2, 3]} - \\ m(x, a, b) &= [0, 1, 1] \end{aligned}$$

*Indeed, the maximum delay of  $a$  that does not affect the first resource is 0, the maximum delay of  $a$  that does not affect the second or third resource is 1.*

### 3.2.2 With buffer times

Suppose we insert extra tasks in a given schedule  $w$ . These extra tasks represent buffer times meant to catch up delays. We denote the schedule including these buffer times by  $w'$ .

**Definition 13** Given an upper contour  $x$ , task  $a$  and schedule  $w'$  including buffer times, we denote by  $m_i(x, a, w')$  the maximum delay  $\tau$  of task  $a$  such that the release of resource  $i$  after processing  $a_\tau w$ , starting from  $x$ , is not delayed. In formula:

$$m_i(x, a, w') = \max\{\tau \mid (x\mathcal{M}(a_\tau w))_i \leq (x\mathcal{M}(aw'))_i\}$$

**Lemma 4**

$$m(x, a, w') = x_{\mathcal{R}}\mathcal{M}_{\mathcal{R} \times \mathcal{R}}(aw') - x_{R(a)}\mathcal{M}_{R(a) \times \mathcal{R}}(aw).$$

The proof of this lemma is similar to the proof of Lemma 3.

The value  $\min_{i \in R(b)} m_i(x, a, w')$  where  $b$  is the final task of  $w'$  equals the maximum delay of task  $a$  that does not disturb task  $b$ .

### 3.2.3 Delay propagation during operations

With the described approach, we obtain scheduled waiting times. For using the same ideas to obtain the true waiting times which occur during operations, we have two remarks:

1. Trains have to decelerate, stop and accelerate if they are disturbed by other trains. The propagated delay will therefore be higher than the initial delay minus the buffer.
2. If we want to use this approach for the true waiting times (i.e. the waiting times which occur during operation) despite of remark 1., we should point resources where trains are allowed to wait. Then waiting trains can be modeled by spitting up the train movements at these resources. If we only allow trains to wait when dwelling at stations we obtain the true waiting times, else we obtain an optimistic estimate of the true waiting times.

### 3.2.4 Stability margin

For capacity assessment we can use the algebraic approach as follows: we add buffers to the schedule such that we are satisfied with the amount of delay propagation. Then the cycletime of the schedule with buffers is a measure for the capacity.

Another way to use the algebraic approach is the following: We question how much buffer time can be added to the schedule such that its cycletime is less than or equal to a prescribed cycletime  $T$ . Of course, there are many ways to add buffer times to the schedule. We restrain ourselves to add the same amount of buffer time after each train movement. This leads us to the following definition:

**Definition 14** (*Stability margin*) Given a schedule of train movements and a prescribed cycletime  $T$ . The stability margin  $\Delta$  is defined as the maximum buffer time that can be scheduled after each train movement such that the cycletime of the schedule does not exceed  $T$ .

It should be remarked that the stability margin might be negative in case the schedule without buffer times has a cycletime larger than  $T$ . Next we give an algorithm for calculating  $\Delta$  (see also [4]).

**Algorithm 1** (*Determining stability margin*)

1. Set  $i = 0$  and  $\Delta_i = 0$ .
2. Obtain the schedule with buffers equal to  $\Delta_i$  and calculate its cycletime  $\lambda_{\Delta_i}$  and its critical circuit  $\zeta_i$ .
3. If  $\lambda_{\Delta_i} = T$ , set  $\Delta = \Delta_i$  and stop. Otherwise goto 4.
4. Set  $\Delta_{i+1} = \Delta_i + \frac{T - \lambda_{\Delta_i}}{|\zeta_i|}$ , where  $|\zeta_i|$  is the number of critical tasks in  $\zeta_i$ . Set  $i$  to  $i + 1$ .
5. Goto 2.

This algorithm always terminates within finite steps, since  $\Delta_i$  is decreasing in  $i$  for  $i > 0$  and the number of circuits is finite.

### 3.3 Parallelism

Sometimes tasks can be processed in parallel although they use common resources. This is typically the case in a railway system: an express train overtakes a stop train, or trains traveling in opposite direction pass each other at a station. In the example of Figure 1.2 for instance, trains  $c$  and  $d$  can pass each other while dwelling at the platform. The parallel process of these trains has been visualized in Figure 3.3. We assume that the tasks to be processed in parallel are elementary tasks.

In the example of the parallel processing of trains  $c$  and  $d$ , there is only one way to process the two tasks in parallel. However, this is not the case in general. For instance, when an express train overtakes a stop train there is probably more than one station to do this. Therefore, we have to specify exactly in which order (parts of) the tasks have to be processed. One way to do this is by assigning

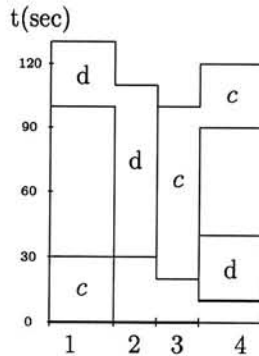


Figure 3.3: Parallel processing of train movements  $c$  and  $d$

starting moments  $h(l)$  to each task  $l$ ,  $h(l) \geq 0$ . These starting moments are not necessarily the moments at which the corresponding tasks start in the schedule, but they only determine the order of (parts of) the tasks.

First we define feasible parallel processing. We call starting moments  $h(l_i)$  feasible with respect to a set of elementary tasks  $\{l_1, \dots, l_n\}$  if for each resource  $r \in \mathcal{R}$  it holds that

$$[s_r(l_i) + h(l_i), f_r(l_i) + h(l_i)] \cap [s_r(l_j) + h(l_j), f_r(l_j) + h(l_j)] = \emptyset \quad i \neq j.$$

In other words, when we visualize the tasks in the distance/time diagram corresponding the starting moments  $h$ , feasibility means that the different tasks do not overlap in the diagram. In the sequel we assume that  $h$  is such that none of the tasks to be processed in parallel is entirely processed as first or as last. In the example of Figure 3.3 we chose  $h(c) = 0$  and  $h(d) = 10$ . In this example different sets  $h$  lead to the same order of (parts of) the tasks.

It is not possible to model tasks to be processed in parallel separately, since the partial ordering of these tasks is lost (none of the train movements in Figure 3.3 is 'above' the other). Therefore, we model the parallel processing of elementary tasks by replacing these tasks by one compound task. We denote this compound task as  $|\{l_1, \dots, l_n\}_h$ . If there is only one way to process the tasks  $l_1, \dots, l_n$  in parallel we omit the  $h$  in the notation. If the parallel processing of tasks is feasible we can obtain a maxplus matrix that describes the corresponding compound task using the following algorithm:

**Algorithm 2** (*Procedure to obtain  $\mathcal{M}(|\{l_1, \dots, l_n\}_h)$* ) Put the tasks in the time/distance diagram at their corresponding starting moments  $h$ . For each resource  $r$  do the following:

Imagine a pencil in the diagram at resource  $r$  and  $t=0$ . Lift the pencil up along the vertical axis and drag all the tasks which are met on the way up. Lift the



pencil until the upper task is about to move. Let  $t$  be the amount of time units the pencil was lifted up. Then  $\mathcal{M}(\|\{l_1, \dots, l_n\}_h)_{ri}$  equals the moment of release of the upper task at resource  $i$  minus  $t$ .

**Example 7** Consider again the parallel processing of tasks  $c$  and  $d$  with  $h(c) = 0$  and  $h(d) = 10$  as in Figure 3.3. This parallel processing is feasible since the tasks do not overlap in the time/distance diagram. We start with resource 1 and we imagine a pencil at resource 1 and  $t = 0$ . The first task this pencil meets is task  $c$ . Next we raise the pencil and drag task  $c$  until it touches task  $d$ , see left picture in Figure 3.4. The final moments of release yields  $[130, 110, 170, 190]$  and we raised

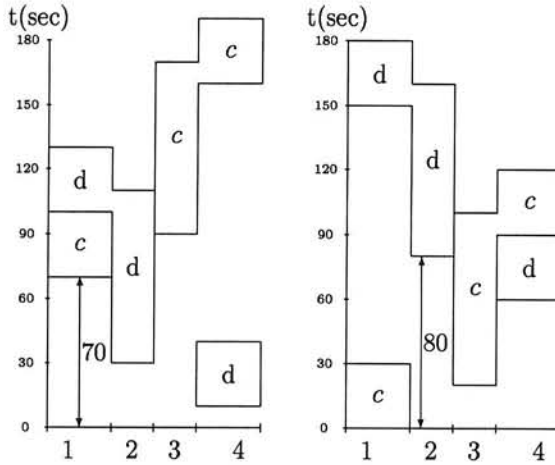


Figure 3.4: Procedure to obtain  $\mathcal{M}(\|\{c, d\})$

the pencil by 70 time units. It follows that  $\mathcal{M}(\|\{c, d\})_1 = [60, 40, 100, 120]$ . Next, we repeat the same procedure with resource 2. Now the pencil first meets task  $d$  and we drag this task until it touches task  $c$ , see the right picture in Figure 3.4. This time the final moments of release yields  $[180, 160, 100, 120]$  and we subtract 80 to obtain  $\mathcal{M}(\|\{c, d\})_2 = [100, 80, 20, 40]$ . Repeating this procedure for resources 3 and 4, we end up with the following matrix:

$$\mathcal{M}(\|\{c, d\}) = \begin{bmatrix} 60 & 40 & 100 & 120 \\ 100 & 80 & 20 & 40 \\ 40 & 20 & 80 & 100 \\ 120 & 100 & 40 & 60 \end{bmatrix}.$$

### 3.4 Passenger transfers

In all previous sections the reason for synchronization of train movements is the common use of the infrastructure: trains have to wait for one another when using

the same infra element. However, we might also incorporate another reason for synchronization: trains have to wait for one another in order to provide a transfer for passengers. In this section we show how passenger transfers can be modeled in a maxplus automaton.

We define a transfer task as follows: Let  $p$  denote a passenger transfer which releases resource  $j$ ,  $t$  time units after resource  $i$  has been released. Task  $p$  leaves the other releases unchanged. Then  $\mathcal{M}(p)$  is the identity matrix, except that  $\mathcal{M}(p)_{ij}$  equals  $t$ . With the aid of task  $p$  we can model a transfer of a train  $a$  which arrives at a platform track corresponding to resource  $i$  to a train  $b$  which departs from a platform track corresponding to resource  $j$ . So, in the schedule  $apb$ , train  $b$  departs at least  $t$  time units after train  $a$  arrives.

**Example 8** Consider the following train movements:

$$\begin{aligned} f(a_1, \cdot) &= [10, 15, \varepsilon, \varepsilon] & f(b_1, \cdot) &= [\varepsilon, \varepsilon, 15, 10] \\ s(a_1, \cdot) &= [0, 5, \varepsilon, \varepsilon] & s(b_1, \cdot) &= [\varepsilon, \varepsilon, 5, 0] \\ \\ f(a_2, \cdot) &= [15, 10, \varepsilon, \varepsilon] & f(b_2, \cdot) &= [\varepsilon, \varepsilon, 10, 15] \\ s(a_2, \cdot) &= [5, 0, \varepsilon, \varepsilon] & s(b_2, \cdot) &= [\varepsilon, \varepsilon, 0, 5] \end{aligned}$$

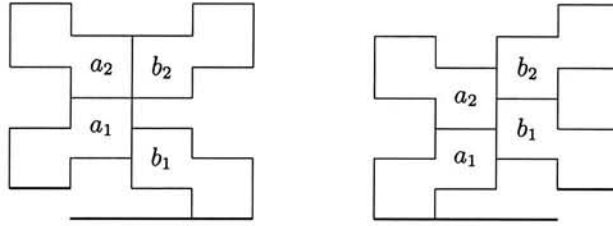
and the schedule  $a_1b_1a_2b_2$ . We now have two possible transfers:  $p_1$ :  $b_2$  may not start before  $a_1$  has finished, such that passengers from  $a_1$  can transfer to  $b_2$  (it is assumed that this transfer takes 0 time units).  $p_2$ :  $a_2$  may not start before  $b_1$  has finished, such that passengers from  $b_1$  can transfer to  $a_2$  (it is assumed that this transfer takes 0 time units). We have:

$$\mathcal{M}(p_1) = \begin{bmatrix} e & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & e & \varepsilon \\ \varepsilon & \varepsilon & e & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & e \end{bmatrix} \quad \mathcal{M}(p_2) = \begin{bmatrix} e & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & e & \varepsilon & \varepsilon \\ \varepsilon & e & e & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & e \end{bmatrix}.$$

Figure 3.5 illustrates the schedule  $a_1b_1p_1a_2b_2$  with initial condition  $x(e) = [5, 0, 0, 0]$  (left picture) and  $x(e) = [0, 0, 0, 5]$  (right picture). The matrices  $\mathcal{M}(p_1)$  and  $\mathcal{M}(p_2)$  do not correspond to elementary tasks. Their product  $\mathcal{M}(p_1p_2)$  does correspond to an elementary task: scheduling both transfers is equivalent of scheduling an elementary task  $l$  with  $f(l) = s(l) = [\varepsilon, e, e, \varepsilon]$ .

### 3.5 Moving block system

One of the recent issues in railway engineering is a moving block system. In such a system tracks are no longer divided into disjunct infra elements which can be

Figure 3.5: Schedule  $a_1b_1p_1a_2b_2$  with different initial conditions

occupied by one train at a time, but each train finds itself in a virtual block which moves along with the train and the safety system provides that virtual blocks of different trains do not overlap.

When dealing with a moving block system we can still use the ideas of a maxplus automaton. A moving block system can be modeled as if the tracks were divided into infinite many infra elements of infinite small length. So we have a continuum of resources:  $\mathcal{R} \subset \mathbb{R}$ . We can describe each (elementary) train movement by an upper and lower contour  $f$  and  $s$  which are functions from  $\mathcal{R}$  to  $\mathbb{R}_{\max}$ . The matrices that describe the train movements in the discrete model are replaced by functions from  $\mathcal{R} \times \mathcal{R}$  to  $\mathbb{R}_{\max}$ . For elementary tasks these functions are given by:

$$\mathcal{M}(l)(x, y) = \begin{cases} e & \text{if } x = y \text{ and } x \notin R(l) \\ f(x) - s(y) & \text{if } x \in R(l) \text{ and } y \in R(l) \\ \varepsilon & \text{otherwise.} \end{cases} \quad (3.2)$$

The evolution of release times, denoted by  $u(x)$ , is given by:

$$u_{k+1}(x) = \bigoplus_y \mathcal{M}(l_k)(x, y) \otimes u_k(y) \quad (3.3)$$

where  $\bigoplus$  denotes taking the maximum over infinitely many numbers. A schedule of trains can be described by 'multiplying' the corresponding  $\mathcal{M}$ -functions:

$$\mathcal{M}(ab)(x, y) = (\mathcal{M}(a) \otimes \mathcal{M}(b))(x, y) \stackrel{\text{def}}{=} \bigoplus_z \mathcal{M}(a)(x, z) \otimes \mathcal{M}(b)(z, y) \quad (3.4)$$

A regular completion can be found by solving the following eigenvalue equation:

$$\bigoplus_y \mathcal{M}(w)(x, y) \otimes v(y) = \lambda \otimes v(x). \quad (3.5)$$

We can translate many concepts from the maxplus algebra to this new structure. However, finding sufficient and necessary conditions for the existence of solutions of the eigenvalue problem is much more complicated than in the finite case. Another issue for this new structure to be solved is how to implement the max operator efficiently.



# Chapter 4

## CASE STUDY

For assessing the value of the algebraic approach, the previously described procedures have been applied in a case study. This case study has been performed for the railway station The Hague Holland Spoor (The Hague HS), one of the two main stations of the city of The Hague (which lies in the western part of the Netherlands).

### 4.1 Building the model

In Figure 4.1 the layout of The Hague HS is schematically drawn. This layout has

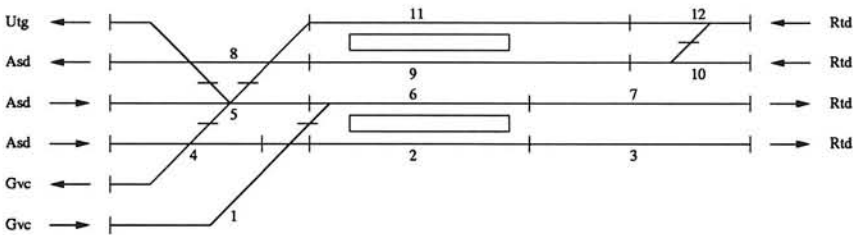


Figure 4.1: Station layout of The Hague HS

been divided into 12 disjunct infra elements which can serve at most one train at a time. In Figure 4.1 also the origins and destinations are shown, using the following abbreviations: Rtd : Rotterdam, Utg : Utrecht, Asd : Amsterdam and Gvc : The Hague CS.

The schedule of trains that visit The Hague HS is based on an hourly pattern which consists of 23 train movements, see Table 4.1. High speed trains (Thalys) are not included in this table since they are not scheduled every hour. For each

arr.time	orig. - dest.	type	serie	route	nr.
03'	Bd - Asd	IR	5400	10 9 8	1
03'	Rtd - Hn	IR	3400	12 11 8	2
11'	Kfh - Hg	FR	A3	12 10 9 8	3
11'	Gvc - Rtd	AR	5100	1 2 3	4
13'	Asd - Brusz	IC	2400	5 6 7	5
15'	Rtd - Gvc	IR	1900	12 10 9 8 5 4	6
15'	Gvc - Rtd	IR	1900	1 2 3	7
17'30"	Gvc - Rtd	AR	5100	1 6 7	8
19'	Kfh - Utg	FR	B1	7 6 5 8	9
19'	Brusz - Asd	IC	2400	12 11 8	10
19'	Rtd - Gvc	AR	5100	10 9 8 5 4	11
25'	Hn - Rtd	IR	3400	4 1 2 3	12
28'	Asd - Bd	IR	5400	5 6 7	13
33'	Ddr - Asd	IR	5400	10 9 8	1
33'	Rtd - Hn	IR	3400	12 11 8	2
37'	Hrl - Gvc	IC	1500	10 9 8 5 4	14
41'	Gvc - Rtd	AR	5100	1 2 3	4
43'	Asd - Vs	IC	2100	5 6 7	5
49'	Vs - Asd	IC	2100	12 11 8	10
50'	Rtd - Gvc	AR	5100	10 9 8 5 4	11
54'	Gvc - Hrl	IC	1500	1 2 3	15
55'	Hn - Rtd	IR	3400	5 6 7	13
58'	Asd - Ddr	IR	5400	4 1 2 3	12

Table 4.1: Hourly pattern of train movements in The Hague HS

train movement one can obtain from Table 4.1 (from left to right) the following: The arrival time according to the time table, the origin and destination of the train, the type of the train: freight train (FR), stop trains (AR = AggloRegional), express trains (IR = InterRegional) or intercity trains (IC), its serial number and the set of resources it uses. Some of the 23 train movements occupy the resources in the same way. Leaving the duplicates out, we end up with 15 different train movements. We have numbered these 15 train movements in the right column of Table 4.1.

Assumptions concerning speed, train length and the like can be found in Table 4.2. When a train enters the station all resources used during arrival are occupied at the same moment (the whole route is 'locked') and the resources are released one after the other. The same procedure is used when a train leaves the station. These rules are incorporated in the contours describing the train movements.

max speed [ $m/s$ ]	FR	AR	IR	IC
acceleration [ $m/s^2$ ]	16	22	22	22
deceleration [ $m/s^2$ ]	.2	.5	.4	.4
dwelt time [s]	.3	.66	.66	.5
train length [m]	0	60	90	120
reaction time [s]	600	200	300	300
reaction time [s]	5			
safety margin [s]	10			
switch time [s]	30			

Table 4.2: Assumptions on the train movements

## 4.2 Results

It turns out that the cycletime of the hourly pattern at The Hague HS equals 2125 seconds, roughly 35 minutes. The critical part of schedule yields:

tasks:	5	8	9	10	11	1	14	5	13
	↓ ↗	↓ ↗	↓ ↗	↓ ↗	↓ ↗	↓ ↗	↓ ↗	↓ ↗	↓ ↗
resources:	6	7	8	8	9	9	5	6	6

Here the critical tasks are indicated by their numbers corresponding to the right column of Table 4.1. The critical resources are those resources where successive critical tasks 'touch'. Notice that the critical part of the schedule is a circuit: the final critical task 13 is followed by the first critical task 5 via resource 6.

When scheduling the same amount of buffer time after each task, we can expect that the cycletime increases by at least the buffer time multiplied by the number of critical train movements. For instance, scheduling 1 minute buffer time after each task leads to an increase of the cycletime by at least 9 minutes. This cycletime turn out to be 2703s, roughly 45 minutes. The cycletime is increased by more than 9 minutes, indicating that another circuit has become critical. Indeed, the critical circuit now consists of 11 critical tasks:

tasks:	4	7	8	9	10	11	1	14	5	11	12
	↓ ↗	↓ ↗	↓ ↗	↓ ↗	↓ ↗	↓ ↗	↓ ↗	↓ ↗	↓ ↗	↓ ↗	↓ ↗
resources:	2	1	7	8	8	9	9	5	5	4	2

Using Algorithm 1 we obtain the stability margin, which equals  $141+6/11$ s. So, if we add buffer times of  $141+6/11$ s (roughly 2 minutes and 22 seconds) we end up with a cycletime of exactly one hour.

In the appendix two delay propagation matrices are shown. In these matrices only the delay propagation within one cycle has been considered. The first one is the delay propagation matrix without buffers. The columns corresponding to critical tasks mainly consist of zeros (the few non-zeros in these columns are due to the fact that only delay propagation within one cycle has been considered).

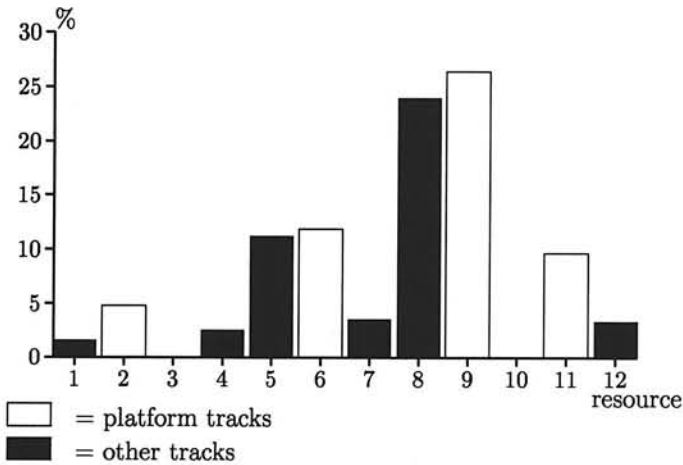


Figure 4.2: Share of resources in critical circuit of random permutations

The second matrix is the delay propagation matrix with 1 minute buffers. The elements of this matrix should be weighted by an appropriate weight function in order to decide whether the delay propagation is acceptable.

We can also incorporate passengers transfers. These transfers yield:

7 waits for 5,  
 11 waits for 10,  
 14 waits for 2,  
 5 waits for 4,  
 11 waits for 10,  
 13 waits for 15.

Scheduling these transfers, it turns out that the cycletime remains the same, indicating that the transfers are not critical. This is not very surprising since the schedule was designed to provide these transfers.

It turns out that relatively many resources become critical in a schedule. Moreover, the set of critical resources heavily depends on the order of train movements. In order to identify the bottleneck of the system we investigate a set of random permutations of the given schedule. For each permutation we calculate the critical circuit and we assess for each resource its share in the total set of these critical circuits. The result of 1000 random permutations has been set out in Figure 4.2. The bars corresponding to the platform tracks are colored white instead of black in order to distinguish them from other resources. Discarding these platform tracks we can point resources 5 and 8 as the bottlenecks of the system, which is in accordance with the results in [13]. However, platform tracks 6 and 9 turn out



to be even more critical. In the model of [13] only junctions and intersections are considered as candidates for bottlenecks. Hence, the result that platform tracks 6 and 9 are even more critical cannot be confirmed nor be denied by [13].



## Chapter 5

# CONCLUSIONS AND FURTHER RESEARCH

We can conclude that a maxplus automaton is a powerful tool for analyzing railway capacity: it provides the maximum frequency and the amount of delay propagation of a given pattern of train movements with or without buffer times. For assessing the bottleneck of given infrastructure, analyzing only one pattern is not enough. The set of critical resources will be relatively large in general and heavily depends on the order of train movements. So, a better view on critical resources can be obtained by investigating random permutations of the pattern.

The advantage of the automaton approach compared to the event graph approach is its modeling power. The complexity of calculating the throughput with the automaton approach is of the same order as with the event graph approach, if the complexity of building the event graph is excluded. This illustrates the strength of the automaton approach: each schedule can easily be analyzed by multiplying the appropriate matrices, whereas in the event graph approach a new model has to be build for each schedule. Moreover, the size of the event graph grows with the length of the pattern, whereas the size of the automaton remains constant. Modeling train movements separately is however only possible if the contours do not depend on the order of trains. Of all components of the occupation time, only the switch time depends on the order. By introducing an approximation regarding the switch times the automata approach can still be used, for instance by assuming that a switch always has to be turned, independently of the order.

Compared with simulations, the algebraic approach has the advantage that it provides critical train movements, critical resources and delay propagation. Hence, the algebraic approach is more suitable for optimizations than simulations. Furthermore, if the running times are assumed to be stochastic, the algebraic approach is still useful as a simulation tool.

In this paper we were able to obtain the amount of delay propagation for given

buffer times. However, we are interested in solutions of the reverse problem: Given constraints on the amount of delay propagation, find buffer times that meet these constraints. Assessing the capacity, we are interested in buffer times that meet the delay propagation constraints and minimize the cycle time. Future research will focus on the resulting optimization problem. Another issue for further research is the moving block system. Here, we have a theoretical as well as a practical challenge: namely by finding conditions for the existence of a solution of the resulting eigenvalue problem and by implementing the max operator in an efficient way.

# Bibliography

- [1] F BACCELLI, G COHEN, GJ OLSDER AND JP QUADRAT Synchronization and linearity (An algebra for discrete event systems)  
Wiley, Chichester, 1992
- [2] M BÄR, K FISCHER AND G HERTEL Leistungsfähigkeit - Qualität - Zuverlässigkeit  
Transpress VEB Verlag für Verkehrswesen, Berlin, 1987
- [3] R BERGMARK AND B ÖSTLUND Comprehensive modeling produces rational investment  
In: Railway Gazette International, pp. [171] – [174], 1994
- [4] JG BRAKER Algorithms and applications in timed discrete event systems  
Thesis, Delft University of Technology, 1993
- [5] JW COHEN The single server queue  
North Holland, Amsterdam, 1969
- [6] RA CUNINGHAME-GREEN Minimax algebra  
Lecture notes in economics and mathematical systems, vol 166, Springer-Verlag, Berlin, 1979
- [7] S GAUBERT AND J MAIRESSE Modeling and analysis of timed Petri nets using heaps of pieces  
Submitted to IEEE-JAC, 1998
- [8] RJ VAN EGMOND Propagation of delays in public transport  
In: Proceedings of the 6th EURO Working Group on Transportation, Göteborg, 1998
- [9] S. EILENBERG Automata, languages and machines, volume A  
Academic Press, New York, 1974
- [10] E GELENBE AND G PUJOLLE Introduction to queueing networks  
Wiley, Chichester, 1987

- 
- [11] G HERTEL Leistungsfähigkeit und Leistungsverhalten von Eisenbahnbetriebsanlagen - Modelle und Berechnungsmöglichkeiten  
In: Schriftenreihe der Deutschen verkehrswissenschaftlichen Gesellschaft e.V., DVWG, Reihe B, pp. [120] – [156], 1994
- [12] JS HOOGHMSTRA AND MJG TEUNISSE The use of simulation in the planning of the Dutch railway services  
In: Proceedings of the 1998 Winter Simulation Conference, pp. [1139] – [1146], 1998
- [13] AF DE KORT, B HEIDERGOTT, RJ VAN EGMOND AND G HOOGHMSTRA Train movement analysis at railway stations: Procedures & evaluation of Wakob's Approach  
TRAIL Studies in Transportation Science, S99/1, Delft University Press, 1999
- [14] J MAIRESSE Products of irreducible random matrices in the  $(\max, +)$  algebra  
In: Advanced Applied Probability, vol 29, pp. [444] – [477], 1997
- [15] NEDERLANDSE SPOORWEGEN Jaardienst reizigerstreindienst 1996-1997  
NS Reizigers Logistiek, Utrecht, The Netherlands, 1996
- [16] GL O'BRIEN Limit theorems for sums of chain-dependent processes  
In: Journal of Applied Probability, vol 11, pp. [582] – [587], 1974
- [17] MA ODIJK Randomized generation of timetables to facilitate and improve the design of rail infrastructure  
In: Schriftenreihe des Instituts für Verkehrssystemtheorie und Bahnverkehr, vol 2: Eisenbahnnetzwissenschaftliches Kolloquium: Taktfahrplan und Kapazität, pp. [115] – [126], 1996
- [18] GJ OLSDER, JAC RESING, RE DE VRIES, MS KEANE AND G HOOGHMSTRA Discrete event systems with stochastic processing times  
In: IEEE transactions on automatic control, vol 35, no 3, [299] – [302], 1990
- [19] JAC RESING, RE DE VRIES, G HOOGHMSTRA, MS KEANE AND GJ OLSDER Asymptotic behavior of random discrete event systems  
In: Stochastic processes and their applications, vol 36, pp. [195]–[216], 1990
- [20] GX VIENNOT Heaps of pieces, I: Basic definitions and combinatorial lemmas  
Lecture Notes in Mathematics, no 1234, Springer, pp. [321] – [350], 1986
- [21] H WAKOB Ableitung eines generellen Wartemodells zur Ermittlung der planmäßigen Wartezeiten im Eisenbahnbetrieb unter besonderer Berücksichtigung der Aspekte Leistungsfähigkeit und Anlagenbelastung  
Veröffentlichungen des verkehrswissenschaftlichen Institutes der RWTH Aachen, vol 36, 1985

- 
- [22] E WEITS Reliable indicators of the capacity of railway infrastructure  
In: Schriftenreihe des Instituts für Verkehrssystemtheorie und Bahnverkehr,  
vol 2: Eisenbahnnetzwissenschaftliches Kolloquium: Taktfahrplan und  
Kapazität, pp. [20] – [35], 1996
- [23] E WENDLER Weiterentwicklung der Sperrzeitentreppe für moderne Signal-  
systeme  
Signal und Draht, vol 87, no 7-8, pp. [268] – [273], 1995





# Appendix A

## ROBUST SCHEDULING WITH EVENT GRAPHS

In this appendix we show how delay propagation in event graphs can be analyzed. Most of this work has been presented before in [8]. We further show how optimal buffer times can be found.

### Simulation

Starting point is an event graph which is determined by travel time matrix  $A$ . Consider a timetable determined by an eigenvector  $v$  of matrix  $A$ :

$$x(k) = \lambda^k \otimes v \tag{A.1}$$

Suppose that at time  $k = 0$  there are some delays  $d(0)$  which result in a vector of disturbed moments of departure  $\tilde{x}(0) = x(0) + d(0)$ . Because all connections remain the same, the time evolution of  $\tilde{x}$  can be found by applying matrix  $A$  and timetable  $x$ :

$$\tilde{x}(k+1) = A \otimes \tilde{x}(k) \oplus x(k+1). \tag{A.2}$$

The difference between  $\tilde{x}$  and  $x$  determines the propagation of delay  $d(0)$ :

$$d(k) = \tilde{x}(k) - x(k). \tag{A.3}$$

This way of calculating the propagation of delays is thus merely a matter of simulation. Also if matrix  $A$  is replaced by random matrices  $A(k)$ , i.e. if the travel times are considered as stochastic variables, this way of simulation can be used.

## Analysis

In this section we investigate which nodes will be disturbed by an initial delay. For this purpose we introduce the following definition:

**Definition 15**  $m_{ij}$  is the maximum delay at node  $j$  that does not reach node  $i$ .

We denote the matrix containing the entries  $m_{ij}$  by  $M$ . The values  $m_{ij}$  can be obtained using the following lemma:

**Lemma 5** Let  $A$  be an irreducible matrix with eigenvector  $v$  and eigenvalue  $\lambda$ , then  $m_{ij} = v_i - v_j - (A_\lambda^+)^+_{ij}$ .

Here,  $(A_\lambda)_{ij} = a_{ij} - \lambda$  and  $A_\lambda^+ = A_\lambda \oplus A_\lambda^2 \oplus A_\lambda^3 \oplus \dots$ .

*Proof.* Consider a path  $\rho$  from node  $j$  to node  $i$  (such a path exists because the graph is assumed to be strongly connected). Renumber the nodes of this path as  $1, \dots, r$  (node  $j$  becomes node 1 and node  $i$  becomes node  $r$ ). Each pair of successive nodes on this path have non-negative slack, i.e. the departure time at the second node minus the arrival time of a train coming from the first node. Let us calculate the total slack on the whole path:

$$\begin{aligned} \text{total slack of } \rho &= (v_2 + \lambda) - (v_1 + a_{21}) + \\ &\quad (v_3 + \lambda) - (v_2 + a_{32}) + \\ &\quad \dots + \\ &\quad (v_r + \lambda) - (v_{r-1} + a_{rr-1}) \\ &= v_r - v_1 + \lambda - a_{21} + \lambda - a_{32} + \dots + \lambda - a_{rr-1} \end{aligned}$$

Let  $P_{ij}$  denote the set of all possible paths from node  $j$  to node  $i$ . The maximum delay in node  $j$  that does not reach node  $i$  equals the slack between node  $j$  and  $i$ , minimized over  $P_{ij}$ . Thus:

$$m_{ij} = \min_{\rho \in P_{ij}} \left\{ v_i - v_j + \sum_{(k,l) \in \rho} (\lambda - a_{kl}) \right\} \quad (\text{A.4})$$

$$= v_i - v_j - \max_{\rho \in P_{ij}} \left\{ \sum_{(k,l) \in \rho} (a_{kl} - \lambda) \right\} \quad (\text{A.5})$$

$$= v_i - v_j - (A_\lambda^+)^+_{ij} \quad (\text{A.6})$$

The final equality (A.6) is based on the shortest path algorithm, cf. [1].

□

Suppose an initial delay  $d_j$  occurs at node  $j$ . We can immediately obtain the impact of this initial delay: The nodes that will be disturbed by this delay are the nodes  $i$  for which it holds that  $m_{ij} < d_j$ . The maximum delay these nodes receive due to delay  $d_j$  equals  $d_j - m_{ij}$ . Notice that matrix  $M$  can be calculated in advance.

**Example 9** Consider a network and its corresponding  $A$ -matrix as in Figure A.1. One can verify that  $v = [0, 0, \frac{1}{2}]'$  is an eigenvector with eigenvalue  $\lambda = 2\frac{1}{2}$  and

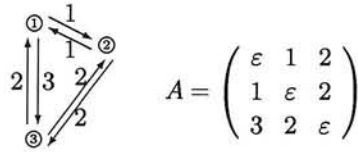


Figure A.1: An example

that the critical circuit is  $\{1, 3\}$ . We have:

$$A_{\lambda}^{\dagger} = \begin{pmatrix} 0 & -1 & -\frac{1}{2} \\ 0 & -1 & -\frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} & 0 \end{pmatrix},$$

and applying Lemma 5 gives:

$$M = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}.$$

In Example 9 the columns of  $M$  belonging to the nodes of the critical circuit turn out to be zero-columns. This means that any delay on the critical circuit eventually disturbs every node. The fact that  $m_{ij} = 0$  whenever both  $i$  and  $j$  are nodes in the critical circuit is obvious: the critical circuit has no slack. The fact that  $m_{ij} = 0$  also holds if only  $j$  is a critical node is due to the fact that the critical circuit is unique as is claimed by the following lemma:

**Lemma 6** *If the critical circuit is unique and  $j$  is a node of the critical circuit, then  $m_{ij} = 0$  for every node  $i$ .*

For the proof of this lemma we refer to [8].

Example 10 shows that this lemma does not hold in general if the critical circuit is not unique.

**Example 10** Consider the following  $A$ -matrix of travel times:

$$A = \begin{pmatrix} \varepsilon & 2 & \varepsilon & \varepsilon \\ 2 & \varepsilon & \varepsilon & 1 \\ 1 & \varepsilon & \varepsilon & 2 \\ \varepsilon & \varepsilon & 2 & \varepsilon \end{pmatrix}$$

This matrix has an eigenvector  $v = [0, 0, 0, 0]'$  and eigenvalue  $\lambda = 2$ . There are two critical circuits:  $\{1, 2\}$  and  $\{3, 4\}$ . Lemma 5 gives:

$$M = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

Although all nodes belong to a critical circuit,  $M$  has non-zero elements.

## Buffer times

In order to increase the punctuality of the transportation system, buffer times  $b_{ij}$  are added to decrease propagation of delays. The system including buffer times is described by a new matrix written as  $\tilde{A}$  with  $\tilde{a}_{ij} = a_{ij} + b_{ij}$ .

The new timetable and cycletime follows from the eigenvector and eigenvalue of  $\tilde{A}$ :

$$\tilde{A} \otimes \tilde{v} = \tilde{\lambda} \otimes \tilde{v}. \quad (\text{A.7})$$

Again, we are interested in the maximum delay in node  $j$  that does not reach node  $i$ , denoted by  $\tilde{m}_{ij}$ . In the new model the slack between two successive nodes equals  $(\tilde{v}_2 + \tilde{\lambda}) - (\tilde{v}_1 + a_{21})$ , resulting in the following expression for  $\tilde{m}_{ij}$ :

$$\tilde{m}_{ij} = \tilde{v}_i - \tilde{v}_j - \left( A_{\tilde{\lambda}}^+ \right)_{ij}. \quad (\text{A.8})$$

Notice that the lacking of the tilde above  $A$  in (A.8) is not a misprint. On the contrary, it is essential that the buffer times disappear in the synchronization constraints when calculating the propagation of delays.

**Example 11** Consider the network given in Example 9 and add buffer times of  $\frac{1}{2}$  to all the travel times:

$$\tilde{A} = A \otimes \frac{1}{2} = \begin{pmatrix} \varepsilon & 1\frac{1}{2} & 2\frac{1}{2} \\ 1\frac{1}{2} & \varepsilon & 2\frac{1}{2} \\ 3\frac{1}{2} & 2\frac{1}{2} & \varepsilon \end{pmatrix}$$

Then  $\bar{\lambda} = 3$ ,  $\bar{v} = v = [0, 0, \frac{1}{2}]'$  and, according to (A.8),

$$\tilde{M} = \begin{pmatrix} 1 & 2 & \frac{1}{2} \\ 1 & 2 & \frac{1}{2} \\ \frac{1}{2} & 1\frac{1}{2} & 1 \end{pmatrix}.$$

## Optimal buffer times

Optimal buffer times can be found by solving the following problem: Given constraints on  $m_{ij}$ , find a timetable  $v$  which obeys these constraints and minimizes the cycletime. This problem can be written as the following optimization problem:

$$\min_{v, \lambda} \lambda \quad (\text{A.9})$$

$$\text{s.t. } \min_{\rho \in P_{ij}} \{v_i - v_j + \sum_{(k,l) \in \rho} (\lambda - a_{kl})\} \geq \underline{m}_{ij}, \quad \forall i, j \quad (\text{A.10})$$

where  $\underline{m}_{ij}$  is a lowerbound for  $m_{ij}$  (cf. (A.4)). It should be noticed that constraints (A.10) incorporate the fact that trains wait for one another by choosing  $\underline{m}_{ij} \geq 0$ ; Then the constraint  $v_i + \lambda \geq v_j + a_{ij}$  is automatically fulfilled for each arc  $(j, i)$  of the graph. Problem (A.9)-(A.10) is a convex optimization problem with a linear object function. Moreover, it can be written as an LP-problem by writing the constraints (A.10) for each path separately. Hence, this problem can be solved by standard optimization techniques.

It remains how to find appropriate constraints on  $m_{ij}$ . For this purpose we consider a static model. Instead describing the  $k^{\text{th}}$  departure from a node as in the dynamic model, the departures are described by separate variables in the static model. In this way we have an explicit expression for the actual departure time  $X_i$  at node  $i$ , given initial delays  $\{D_j\}$ :

$$X_i = v_i + \max_j \{D_j - m_{ij}\}. \quad (\text{A.11})$$

In words: the actual departure time at node  $i$  is the departure time at node  $i$  according to the timetable plus the maximum delay that is 'received' at node  $i$ . Next, given the distributions of initial delays, we are interested in distributions of departure times. Assuming that the initial delays are independent we can obtain from (A.11):

$$P(X_i - v_i \leq x) = \prod_j P(D_j \leq x + m_{ij}) \quad (\text{A.12})$$

Assuming that vehicles never depart earlier than  $v_i$ , is equivalent to assuming that  $P(D_i < 0) = 0$ . By this assumption we can interpret  $X_i - v_i$  as the waiting time of a passenger which planned to depart at  $v_i$  from node  $i$  and has to wait  $X_i - v_i$  time units before the vehicle actually departs. So, we can translate values of  $m_{ij}$  into terms of waiting times.



# Appendix B

## DELAY PROPAGATION MATRICES OF CASE STUDY

The next pages show two delay propagation matrices. These matrices describe the amount of delay propagation within the hourly pattern at The Hague HS, which consists of 23 train movements. These movements are described in Table 4.1. An element  $i, j$  of such a matrix gives the maximum delay of the  $j$ th train movement in the schedule that does not disturb the  $i$ th train movement. Since it is assumed that the schedule is repeated over and over again, the whole matrix is filled. It is however assumed that a delay does not propagate more than one cycle. Table B.1 shows the delay propagation matrix without buffer times and Table B.2 shows the delay propagation matrix if after each train movement 1 minute buffer time is scheduled.

In Table B.1 some of the columns consist for the largest part of zeroes. These are precisely the columns that correspond to the critical train movements: 5, 8, 9, 10, 11, 1, 14, 5 and 13. These zeroes can be explained by the fact that there is no slack on the critical circuit. The non-zeroes in these columns are due to the fact that only delay propagation within one cycle has been considered.

These delay propagation matrices can be used to obtain a performance measure on the robustness of the schedule by weighting these elements in a appropriate way.

	1	2	3	4	5	6	7	8	9	10	11	12	13	1	2	14	4	5	10	11	15	13	12
1	57	57	57	42	0	57	42	0	0	0	0	401	364	0	65	0	Inf	40	65	0	Inf	Inf	Inf
2	0	57	57	42	0	57	42	0	0	0	0	401	364	0	65	0	Inf	40	65	0	Inf	Inf	Inf
3	0	0	57	42	0	57	42	0	0	0	0	401	364	0	65	0	Inf	40	65	0	Inf	Inf	Inf
4	Inf	Inf	Inf	42	0	57	42	0	0	0	0	14	364	0	65	0	14	40	65	0	14	Inf	0
5	Inf	Inf	Inf	Inf	0	57	42	0	0	0	0	401	324	0	145	0	Inf	0	147	82	Inf	0	Inf
6	0	0	0	Inf	277	57	42	0	0	0	0	401	364	0	65	0	650	40	65	0	650	277	636
7	Inf	Inf	Inf	0	Inf	Inf	42	0	0	0	0	14	364	0	65	0	14	40	65	0	14	Inf	0
8	Inf	Inf	Inf	42	0	Inf	42	0	0	0	0	56	324	0	107	0	56	0	107	42	56	0	42
9	57	57	57	42	0	57	42	0	0	0	0	56	324	0	107	0	56	0	107	42	56	0	42
10	57	57	57	42	0	57	42	0	0	0	0	56	324	0	107	0	56	0	107	42	56	0	42
11	57	57	57	42	0	57	42	0	0	0	0	56	324	0	107	0	56	0	107	42	56	0	42
12	57	57	57	42	0	57	42	0	0	0	0	56	324	0	107	0	56	0	107	42	56	0	42
13	57	57	57	42	0	57	42	0	0	0	0	Inf	324	0	107	0	56	0	107	42	56	0	42
1	57	57	57	42	0	57	42	0	0	0	0	Inf	Inf	0	107	0	56	0	107	42	56	0	42
2	57	57	57	42	0	57	42	0	0	0	0	Inf	Inf	0	107	0	56	0	107	42	56	0	42
14	57	57	57	42	0	57	42	0	0	0	0	401	406	0	145	0	56	0	107	42	56	0	42
4	57	57	57	42	0	57	42	0	0	0	0	0	Inf	Inf	Inf	Inf	56	0	107	42	56	0	42
5	57	57	57	42	0	57	42	0	0	0	0	401	324	0	145	0	Inf	0	107	42	56	0	42
10	57	57	57	42	0	57	42	0	0	0	0	417	422	0	0	16	Inf	Inf	107	42	56	0	42
11	57	57	57	42	0	57	42	0	0	0	0	401	364	0	65	0	Inf	40	65	42	56	0	42
15	57	57	57	42	0	57	42	0	0	0	0	0	Inf	Inf	Inf	Inf	0	Inf	Inf	Inf	56	0	42
13	57	57	57	42	0	57	42	0	0	0	0	401	324	0	145	0	Inf	0	147	82	Inf	0	42
12	57	57	57	42	0	57	42	0	0	0	0	14	364	0	65	0	14	40	65	0	14	Inf	42

Table B.1: Delay propagation matrix without buffers



	1	2	3	4	5	6	7	8	9	10	11	12	13	1	2	14	4	5	10	11	15	13	12
1	635	575	515	560	538	455	500	440	380	320	260	601	564	200	205	140	Inf	120	145	60	Inf	Inf	Inf
2	60	635	575	620	598	515	560	500	440	380	320	661	624	260	265	200	Inf	180	205	120	Inf	Inf	Inf
3	120	60	635	680	658	575	620	560	500	440	380	721	684	320	325	260	Inf	240	265	180	Inf	Inf	Inf
4	Inf	Inf	Inf	620	598	515	560	500	440	380	320	274	624	260	265	200	214	180	205	120	154	Inf	60
5	Inf	Inf	Inf	Inf	578	495	540	480	420	360	300	641	564	240	325	180	Inf	120	267	182	Inf	60	Inf
6	180	120	60	Inf	417	635	680	620	560	500	440	781	744	380	385	320	970	300	325	240	910	477	816
7	Inf	Inf	Inf	60	Inf	Inf	620	560	500	440	380	334	684	320	325	260	274	240	265	180	214	Inf	120
8	Inf	Inf	Inf	120	98	Inf	60	578	518	458	398	394	662	338	385	278	334	218	325	240	274	158	180
9	255	195	135	180	158	75	120	60	578	518	458	454	722	398	445	338	394	278	385	300	334	218	240
10	315	255	195	240	218	135	180	120	60	578	518	514	782	458	505	398	454	338	445	360	394	278	300
11	375	315	255	300	278	195	240	180	120	60	578	574	842	518	565	458	514	398	505	420	454	338	360
12	435	375	315	360	338	255	300	240	180	120	60	634	902	578	625	518	574	458	565	480	514	398	420
13	435	375	315	360	338	255	300	240	180	120	60	Inf	902	578	625	518	574	458	565	480	514	398	420
1	435	375	315	360	338	255	300	240	180	120	60	Inf	Inf	578	625	518	574	458	565	480	514	398	420
2	495	435	375	420	398	315	360	300	240	180	120	Inf	Inf	60	685	578	634	518	625	540	574	458	480
14	495	435	375	420	398	315	360	300	240	180	120	461	466	60	145	578	634	518	625	540	574	458	480
4	495	435	375	420	398	315	360	300	240	180	120	60	Inf	Inf	Inf	Inf	634	518	625	540	574	458	480
5	555	495	435	480	458	375	420	360	300	240	180	521	444	120	205	60	Inf	578	685	600	634	518	540
10	555	495	435	480	458	375	420	360	300	240	180	537	542	120	60	76	Inf	Inf	685	600	634	518	540
11	575	515	455	500	478	395	440	380	320	260	200	541	504	140	145	80	Inf	60	85	620	654	538	560
15	555	495	435	480	458	375	420	360	300	240	180	120	Inf	Inf	Inf	Inf	60	Inf	Inf	Inf	634	518	540
13	615	555	495	540	518	435	480	420	360	300	240	581	504	180	265	120	Inf	60	207	122	Inf	578	600
12	635	575	515	560	538	455	500	440	380	320	260	214	564	200	205	140	154	120	145	60	94	Inf	620

Table B.2: Delay propagation matrix with buffers

## **TRAIL Studies in Transportation Science**

A series of The Netherlands TRAIL Research School for fundamental and theoretical studies on transport, infrastructure and logistics.

Kort, A.F. de, B. Heidergott, R.J. van Egmond, Hooghiemstra, G., *Train Movement Analysis at Railway Stations: Procedures & Evaluation of Wakob's Approach*, February 1999, TRAIL Studies in Transportation Science, S99/1, Delft University Press, The Netherlands

Egmond, R.J. van, *Railway capacity assessment, an algebraic approach*, July 1999, TRAIL Studies in Transportation Science, S99/2, Delft University Press, The Netherlands

Forthcoming:

Nes, R. van, *Design of multimodal transport systems. Setting the scene: Review of literature and basic concepts*, 1999, TRAIL Studies in Transportation Science, Delft University Press, The Netherlands

Sales and distribution:

Delft University Press

P.O. Box 98

2600 MG Delft

Telephone: +31 (0)15 278 32 54

Telefax: +31 (0)15 278 16 61









3035958

在 一 二 三 四 五 六 七 八 九 十

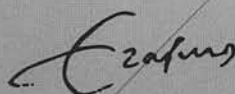


**The Netherlands TRAIL Research School**

Delft University of Technology /  
Erasmus University Rotterdam/  
*University of Groningen*

Kluyverweg 4  
P.O. Box 5017  
2600 GA Delft  
The Netherlands  
Telephone: +31 (0)15 278 60 46  
Telefax : +31 (0)15 278 43 33

 **TU Delft**



**RUG**