ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
DOI: 10.17559/TV-20131107130926

APPLICATION OF KRONECKER ALGEBRA IN RAILWAY OPERATION

Mark Stefan (Volcic), Johann Blieberger, Andreas Schöbel

Original scientific paper

We present a methodology for dispatching trains which prevents deadlocks and includes possible limitation of the available energy provided by the power supply. Our approach applies Kronecker algebra to manipulate matrices. Generally blocking of trains occurs due to a lack of some resource which can be either infrastructure or energy. Our method can also be used to calculate travel times in a rough way. Thereby blocking time is included in the calculated travel time. To model the movements of trains in a railway system we use graphs, which are represented by adjacency matrices. We assume that the edges in a graph are labelled by elements of a semiring. Usually two or more distinct train route graphs refer to the same track section to model synchronization. Our approach can be used to model a complex railway system. For example, if additional trains have to be scheduled, power stations or interconnection lines fail or are not available due to maintenance, our model can be used to calculate the impact on the travel times of the trains in the system.

Keywords: *deadlock; dispatching; energy-aware; Kronecker algebra; travel time*

Primjena Kroneckerove algebre u poslovanju željeznice

Izvorni znanstveni članak

Predstavljamo metodologiju za otpremu vlakova koja sprečava potpune zastoje i uključuje moguća ograničenja raspoložive energije energetskog sustava. Naš pristup primjenjuje Kronekerovu algebru za manipuliranje matrica. Do blokiranja vlakova općenito dolazi zbog nedostatka nekog resursa, a to može biti ili infrastruktura ili energija. Naša se metoda može primijeniti i za grubi proračun putnih vremena. Vrijeme blokiranja je uključeno u proračunato putno vrijeme. Za prikaz kretanja vlakova u željezničkom sustavu rabimo grafikone, koji su prikazani matricama graničenja. Pretpostavljamo da su granice u grafikonu označene elementima polukruga. Za modeliranje sinhronizacije obično se grafikoni dvaju ili više različitih smjerova vlakova odnose na isti dio kolosijeka. Naš se pristup može primijeniti kod modeliranja složenog željezničkog sustava. Na primjer, ako treba programirati dodatne vlakove, zastoj u energetskoj centrali ili na spojnim pravcima, bilo zbog kvara ili održavanja, našim se modelom može izračunati kako se to odražava na putnom vremenu vlakova u sustavu.

Ključneriječi: *Kroneckerova algebra; otprema; putno vrijeme; vođenje računa o energiji; zastoj*

1 Introduction

We present a graph-based method on a fine-grained level for dispatching trains which prevents deadlocks, calculates the travel time of trains, and includes possible limitations of the available energy provided by the power supply. The routes of trains are modelled by graphs, whereby track sections may be part of routes of several trains. We assume that at the same time only one single train occupies a track section. If an operational standard allows entry in an occupied block ("permissive driving"), our approach can deal with this issue as well by fine-scaling the track section. Our model employs semaphores in the sense of computer science to guarantee that only one train enters a train section. These semaphores are modelled by simple graphs.

Graphs can be represented by matrices. Interestingly, simple matrix operations can be used to model concurrency and synchronization via semaphores [1], [2], [3]. These matrix operations are known as Kronecker sum and Kronecker product. With help of these operations we build a graph describing the overall railway system.

By traversing the resulting graph we compute the travel time of the trains within the network. Blocking among trains occurs due to sharing of train sections, connections, and overtaking.

Blocking time is incorporated into the calculated travel time.

Shared resources other than track sections can also be modelled. In particular, so-called counting semaphores can be used to model discrete power resources. For example, a counting semaphore of size four allows four p-operations before it blocks.

If we discretise energy into standardized packages e.g. 1 MW·h, we can model a power station or substation capable of producing e.g. 20 MW by a counting semaphore of size 20. Of course, a more fine-grained approach is viable too. So we may discretise energy into 100 kW·h or even 10 kW·h steps.

In comparison to simulation tools we determine all possible train movements within the railway system, we find blocking situations and calculate the travel time for each train at once. Thus our approach does not need several simulation runs to find all possible solutions (including blocking situations). Common simulation tools calculate only a single result in one simulation run and so they cannot ensure the detection of blocking situations, which might be present in the given railway system.

In [4] Banker's algorithm is modified such that it can be employed for deadlock analysis in railway systems. Since Banker's algorithm has been designed for standard computer systems it is not well-suited for railway systems. For example it may prohibit allocating a resource (track section) although a potential deadlock can be bypassed. In contrast to our approach both track sections and switches have to be modelled.

In [5] Movement Consequence Analysis (MCA) and Dynamic Route Reservation (DRR) are introduced for deadlock analysis. Both are rule-based methods for which correctness cannot be proved. It delivers false positives.

A comprehensive analysis of methods and algorithms used in railway systems can be found in [6].

2 Basics of Kronecker algebra

This article is based on, [3], [7], [8], and [9], which are concerned with the timing analysis of concurrent programs and the deadlock and travel time analysis for railway systems. Instead of using processors or threads as in [7], we will discuss operations with trains and instead of shared memory, track sections are used for our purpose. For the synchronization of the trains on a track section, semaphores in the sense of computer science (cf. [10]) are used. Thus a train can enter or reserve a track section only if it is not blocked by another train using the semaphore of the track section. Blocking may occur only in succession of semaphore calls.

To model the movements of trains in a railroad system we use graphs, which are represented by adjacency matrices. We assume that the edges in a graph are labelled by elements of a semiring. Definitions and properties of the semiring can be found in [11] and [1]. Our semiring consists of a set of labels $\mathcal{L}$ representing semaphore calls denoted by p_i and v_i (cf. [10]).

Usually two or more distinct train route graphs refer to the same track section to model synchronization.

2.1 Modelling synchronization and interleavings

Kronecker product and Kronecker sum form Kronecker algebra. In the following we define both operations. From now on we use matrices out of $\mathcal{M} = \{M = (m_{i,j}) | m_{i,j} \in \mathcal{L}\}$ only.

Definition 1 (Kronecker product) Given a m-by-n matrix A and a p-by-q matrix B, their Kronecker product denoted by $A \otimes B$ is a mp-by-nq block matrix defined by

$$A \otimes B = \begin{pmatrix} a_{1,1}B & \cdots & a_{1,n}B \\ \vdots & \ddots & \vdots \\ a_{m,1}B & \cdots & a_{m,n}B \end{pmatrix} \tag{1}$$

Kronecker product allows to model synchronization (cf. [12], [1], and [3]).

Definition 2 (Kronecker sum) Given a matrix A of order m and a matrix B of order n, their Kronecker sum denoted by $A \oplus B$ is a matrix of order mn defined by

$$A \oplus B = A \otimes I_n + I_m \otimes B, \tag{2}$$

where I_m and I_n denote identity matrices of order m and n respectively.

2.2 Representation of railway systems

In general, a railroad system consists of a finite number of trains and track sections which are represented by graphs. As already mentioned the graphs are represented by adjacency matrices and the track sections are represented by binary semaphores in the sense of computer science. The matrices have entries which are referred to as labels $l \in \mathcal{L}$.

Formally, the system model consists of the tuple $\langle \mathcal{T}, \mathcal{S}, \mathcal{L} \rangle$, where $\mathcal{T}$ is the set of graph adjacency matrices describing routes of trains, $\mathcal{S}$ refers to the set of adjacency matrices describing track sections (semaphores) and the labels in $T \in \mathcal{T}$ and $S \in \mathcal{S}$ are elements of $\mathcal{L}$, respectively. The matrices are manipulated by using Kronecker algebra.

As already mentioned, there is a correspondence between matrices and graphs. In general a directed labelled graph $C = \langle V, E, n_e \rangle$ consists of a set of labelled nodes$V$, a set of labelled edges $E \subseteq V \times V$ and an entry node $n_e \in V$. The sets V and E are constructed out of the elements of $\langle \mathcal{T}, \mathcal{S}, \mathcal{L} \rangle$.

In this article we label graph nodes simply by positive integers. Correspondence between graphs and matrices – frequently called *adjacency matrices* – is as follows: If there exists an edge labelled$a$ from node i to node j, then the corresponding adjacency matrix M has $m_{i,j} = a$. If there is no edge between node i and node j, then $m_{i,j} = 0$.

The following example illustrates some interleavings of two train routes in a railway system and how Kronecker sum handles it.

2.3 Synchronization and interleavings – an example

Let the matrices

$$C = \begin{pmatrix} 0 & a & 0 \\ 0 & 0 & b \\ 0 & 0 & 0 \end{pmatrix} \text{ and } D = \begin{pmatrix} 0 & c & 0 \\ 0 & 0 & d \\ 0 & 0 & 0 \end{pmatrix}.$$

The graphs of matrices C and D are shown in Fig. 1. Now interpret C and D as being train routes and a,b, c, and d as being actions of the trains with the following meaning:

- a denotes train C enters track section T_a
- b means C has left track section T_a
- c denotes train D enters track section T_b
- d means train D has left track section T_b

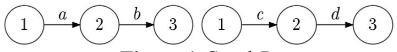

Figure 1 *C* and *D*

Table 1 Timing interleavings

Interleavings
a b c d
a c b d
a c d b
c a b d
c a d b
c d a b

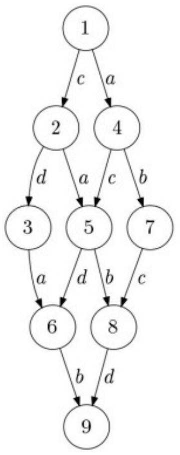

Figure 2 $C \oplus D$

All possible timing interleavings by executing C and D are shown in Tab. 1 and the graph represented by the adjacency matrix $C \oplus D$ is depicted in Fig. 2.

Now assume that T_a and T_b denote the same track section. It is clear that in this case the temporal interleavings of Table 1 are no more valid. The trains have to synchronize in order to perform their actions correctly. This can be modelled by Kronecker product and an additional matrix of the form

$$S = \begin{pmatrix} 0 & p \\ v & 0 \end{pmatrix} \tag{3}$$

where p denotes the action *"Enter the track section"* and v means *"Train has left the track section"*. The correct system behaviour can be described by the matrix $R = (C \oplus D) \otimes S$. As a result the graph will be decomposed into sub-graphs (Figure 3). Clearly only the part reachable from the entry node is responsible for the system behaviour, the others can safely be ignored. Since node 1 is the entry node, we see that now the trains enter the track section one after the other. Note that the two paths in the sub-graph correctly mirror the two cases where C enters the track section before D and vice versa. A proof that Kronecker product models synchronization correctly can be found in [1].

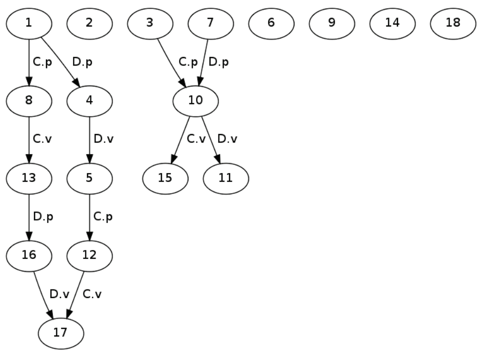

Figure 3 $(C \oplus D) \otimes S$

3 System model

We model a general railway system S by a set of track sections $T = \{T_i \mid 1 \le i \le r\}$. Each track section T_i is modelled by matrix

$$T_i = \begin{pmatrix} 0 & p_i \\ v_i & 0 \end{pmatrix}. \tag{4}$$

In addition, a railway system consists of a set of trains $L = \{L_j \mid 1 \le j \le t\}$. The route R_j of train L_j is a sequence of track sections $T_{l_1}, \dots, T_{l_s}$ for $1 \le l_n \le r$ and $1 \le n \le s$. Each route is modelled by a $2s \times 2s$-matrix. The set of routes is denoted by $R = \{R_j \mid 1 \le j \le t\}$. The behaviour of railway system $S\langle T, L, R\rangle$ is modelled by

$$S = (\oplus_{j=1}^{t} R_j) \otimes (\oplus_{i=1}^{r} T_i), \tag{5}$$

where during the evaluation of the Kronecker product we let $p_i = p_i \cdot p_i$ and $v_i = v_i \cdot v_i$. The different paths in the graph corresponding to matrix S mirror all possible behaviours of the railway system in terms of temporal interleavings of the actions of trains, namely entering and leaving track sections.

Special cases like overtaking or waiting for other trains, which need some additional semaphores are discussed in the later examples. From the discussion above and from [1] it is clear that deadlocks appear as purely structural properties of the underlying graph. Deadlocks manifest themselves as non-final nodes with no successors [2]. A final node corresponds to the destination of a route. A final node of a railway system corresponds to the state, where all trains have reached their destinations.

4 Travel time analysis

Travel time is discussed in detail in [3]. Each node of the graph is assigned a variable and an equation is setup based on the predecessors of the node. A variable is represented by a vector, where each component of the vector corresponds to a train. The equations are used to calculate the travel time.

Let the vector $\mathcal{X} = (X_1, .., X_l, \dots, X_p)^{\mathrm{T}}$. We write $\mathcal{X}^{(l)} = X_l$ to denote the lth component of vector $\mathcal{X}$.

Definition 3 Let $\mathcal{X} = (X_1, \dots, X_p)^{\mathrm{T}}$ and $\mathfrak{Y} = (Y_1, \dots, Y_p)^{T}$. Then we define

$$\max(\mathcal{X}, \mathfrak{Y}) := \left(\max(X_1, Y_1), \dots, \max(X_p, Y_p)\right)^{\mathrm{T}}. \tag{6}$$

Synchronizing nodes are nodes where blocking occurs. These nodes have an incoming edge labelled by a semaphore v-operation, an outgoing edge labeled by a p-operation of the same semaphore, and these edges are part of different trains. In this case the train with the p-operation has to wait until the other train's v-operation is finished.

Definition 4 A *synchronizing node* is a node s such that

- there exists an edge $e_{in} = (i, s)$ with label v_k and
- there exists an edge $e_{out} = (s, j)$ with label p_k,

where k denotes the same semaphore and e_{in} and e_{out} are mapped to different trains.

Definition 5 (Setting up equations) If n is a non-synchronizing node, then

$$\mathcal{X}_n = \max_{k \in Pred(n)}\left(\mathcal{X}_k + t(k \to n)\right) \tag{7}$$

where the lth component of vector $t(k \to n)$ is the time assigned to edge $k \to n$ and edge $k \to n$ is mapped to train l. The other components of $t(k \to n)$ are zero. The set of predecessor nodes of node n is referred to as $Pred(n)$.

Let s be a synchronizing node. In addition, let λ_i and λ_j be the trains where the edges $i \to s$ and $s \to j$ are mapped to. Then for $l \ne \lambda$

$$\mathcal{X}_s^{(l)} = \max_{k \in Pred(s)}\left(\mathcal{X}_k^{(l)} + t(k \to s)^{(l)}\right) \tag{8}$$

and

$$\mathcal{X}_s^{(\lambda_j)} = \max\left(\mathcal{X}_i^{(\lambda_i)} + t(i \rightarrow s)^{(\lambda_i)}, \mathcal{X}_k^{(\lambda_j)} + t(k \rightarrow s)^{(\lambda_j)}\right) \quad (9)$$

where the first term considers the incoming v-edge and the second one the incoming edge of train λ_j.

The system of equations can be solved easily by inserting one equation into another.

5 Example 1 (Travel time analysis)

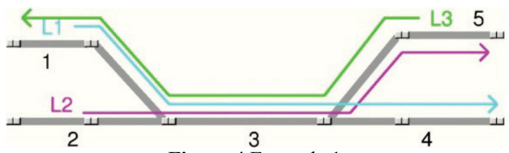

Figure 4 Example 1

In this section we give a small example on creating a graph and the equations for the calculation of the travel time. For the example the railroad system in Figure 4 is used with three trains L_1, L_2 and L_3. The routes of the three involved trains are given in Table 2. The travel time for each operation is given in brackets.

Table 2 Trains, routes, and travel time

Train	Routes (time Value)	Travel time
L_1	$p_3(0), v_1(5), p_4(0), v_3(3), v_4(4)$	12
L_2	$p_3(0), v_2(4), p_5(0), v_3(3), v_5(5)$	28
L_3	$p_3(0), v_5(5), p_1(0), v_3(3), v_1(5)$	21

The matrices for the three routes are set up as follows:

$$R_1 = \begin{pmatrix} 0 & p_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & v_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & v_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & v_4 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$R_2 = \begin{pmatrix} 0 & p_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & v_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_5 & 0 & 0 \\ 0 & 0 & 0 & 0 & v_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & v_5 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$R_3 = \begin{pmatrix} 0 & p_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & v_5 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_1 & 0 & 0 \\ 0 & 0 & 0 & 0 & v_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & v_1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

The matrices for the five track sections have the form $T_i = \begin{pmatrix} 0 & p_i \\ v_i & 0 \end{pmatrix}$ for $1 \leq i \leq 5$.

5.1 Resulting Matrix and Graph

The order of the resulting matrix can be computed by multiplying the order of the involved matrices, i.e., the order of the matrices R_j and the order of the matrices T_i.

In our example the resulting matrix will have order $6912 = 6 \cdot 6 \cdot 6 \cdot 2^5$. Due to synchronization only a small part of the resulting graph is reachable from the entry node. Figure 5 illustrates the resulting graph of the given example. In the example we use the time values in Table 2 for entering and leaving the track sections. To keep the example and the graph simple, we have assigned the travel time values to the v operations. The travel times for the three trains are shown in the right column of Table 2.

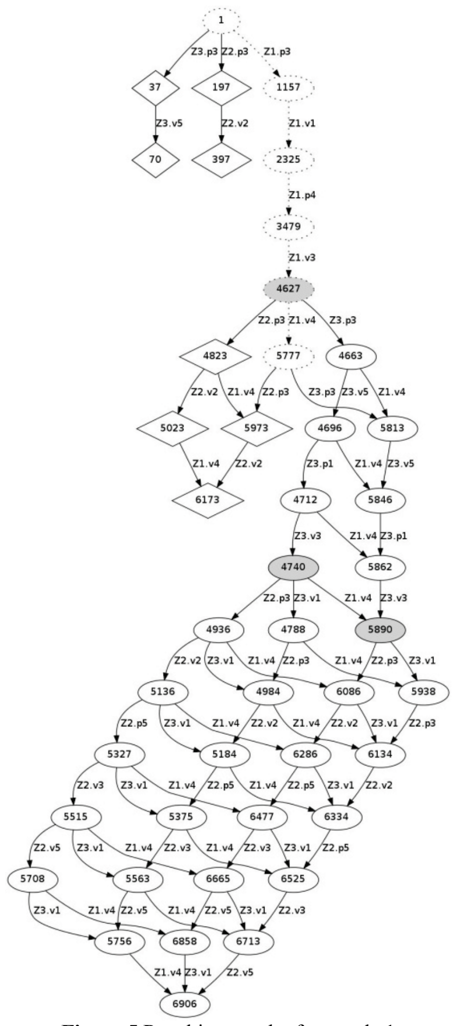

Figure 5 Resulting graph of example 1

To increase readability we distinguish the following node types:

- Diamond nodes denote deadlocks or nodes from which only deadlocks can be reached. Deadlock analysis for railway systems via our approach is studied in [2].
- Solid nodes denote *safe* states. A state is *safe* if all trains can perform their actions without having to take into account the moves of the other trains in the system, provided that the track section which they are to enter is not occupied by another train. If a track section is occupied by another train, the movement of the train wanting to enter may be delayed (blocked) but no deadlock can occur.
- From dotted nodes both diamond and solid nodes can be reached.

- Filled nodes are *synchronizing* nodes (at least two trains must be synchronized).

When calculating the travel time, an equation for each node is set up. As there are a lot of nodes in our example, we present equations for one non-synchronizing node (5846) and for one synchronizing node (5890) only:

$$\mathcal{X}_{5846} = \max\left(\mathcal{X}_{4696} + \begin{pmatrix} v_4 \\ 0 \\ 0 \end{pmatrix}, \mathcal{X}_{5813} + \begin{pmatrix} 0 \\ 0 \\ v_5 \end{pmatrix}\right) \quad (10)$$

$$\mathcal{X}_{5890} = \begin{pmatrix} \max\left(\mathcal{X}_{4740}^{(1)} + v_4, \mathcal{X}_{5862}^{(1)}\right) \\ \mathcal{X}_{5862}^{(3)} + v_3 \\ \max\left(\mathcal{X}_{5862}^{(3)} + v_3, \mathcal{X}_{4740}^{(3)}\right) \end{pmatrix} \quad (11)$$

5.2 Explanation of the equations

$\mathcal{X}_{5846}$**:** To get the travel time for node $\mathcal{X}_{5846}$, the travel time of each predecessor plus the travel time of the connecting-edge are computed. So we get the travel time for each incoming edge. The maximum of these values is taken as the travel time for node $\mathcal{X}_{5846}$. For example node $\mathcal{X}_{4696}$is connected to $\mathcal{X}_{5846}$by an edge labeled as$L_1.v_4$, which means that train L_1executes v_4. As a result v_4 is used in the first vector of the equation for L_1. Because there are no other trains involved in the transition from $\mathcal{X}_{4696}$ to $\mathcal{X}_{5846}$, the other values of the vector are set to 0. The same procedure is done for node $\mathcal{X}_{5813}$, which is connected to $\mathcal{X}_{5846}$by an edge labelled with $L_3.v_5$and thus the vector will have value 0 for the first and second train, and v_5 for the third one.

$\mathcal{X}_{5890}$**:** As described in Eq. (8) and (9), there is a difference between trains which should be synchronized with others and trains which act independently (at the current node). For independent trains, the equation is set up as for non-synchronizing nodes (first and third line in the vector for node $\mathcal{X}_{5890}$). The second line, which describes the travel time for train L_2 will use the travel time value of the third train L_3 becauseL_2 has to be synchronized with L_3 and thus train L_2 has to wait for L_3. If there are any other incoming edges with an action from train L_2, then their travel time values would also be considered in the equation and the maximum of these values would be taken as the train's travel time at the current node.

For this example Kronecker algebra calculations produce a graph with 44 nodes and thus 44 equations must be set up. After creating the equations for each node, the travel time for each node can be computed by inserting one equation into another.

6 Energy awareness

6.1 Modelling with counting semaphores

As explained in the previous sections, for shared resources like track sections, where only one train is allowed to enter or reserve the section, binary semaphores are used. Shared resources other than track sections can also be modelled. In particular, so-called counting semaphores can be used to model discrete power resources. For example, a counting semaphore of size four allows four p-operations before it blocks, e.g. the following trace is allowed for such a semaphore

$p, p, p, v, p, p, v, v, p, p, v, v, v, v$

while the trace

$p, p, p, v, p, p, v, v, p, p, p$

leads to blocking.

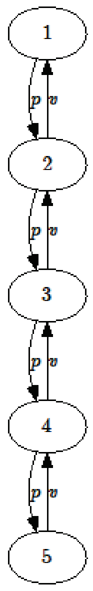

Figure 6 Counting semaphore of size four

A counting semaphore of size four can be found in Figure 6, its matrix representation is

$$S = \begin{pmatrix} 0 & p & 0 & 0 & 0 \\ v & 0 & p & 0 & 0 \\ 0 & v & 0 & p & 0 \\ 0 & 0 & v & 0 & p \\ 0 & 0 & 0 & v & 0 \end{pmatrix}$$

If we quantise energy into standardised packages e.g. 1 MW·h, we can model a power station or substation capable of producing e.g. 20 MW by a counting semaphore of size 20. Of course, a more fine-grained approach is viable too. So we may quantise energy into 100 kW·h or even 10 kW·h steps.

We assume that it is known a priori how much energy each train needs for each track section. Now in our model, a train acquires amounts of power from the power station before it enters a certain track section by issuing exactly the number of p-operations that correspond to the amount of power it will need. On leaving the track section, the train will issue the same number of v-operations to release its power needs. Modelling available energy and required energy in this way ensures that a train needing more energy than can be delivered by the power station, is blocked, i.e., it stops and will continue its travel when enough energy is available (e.g. because another train has released its energy needs). The same approach can be used to model limited interconnection capacities and to avoid accumulating current peaks due to simultaneous train departures.

6.2 Kronecker algebra based dispatching

Due to the information calculated by using Kronecker algebra the dispatching can be planned or modified, based on

- the avoidance of deadlocks,
- the avoidance of headway-conflicts,
- taking the travel time into account, and
- the availability and the consumption of energy.

Of course, avoidance of headway-conflicts reduces consumption of energy (cf. [13] and [14]).

7 Example 2 (energy awareness)

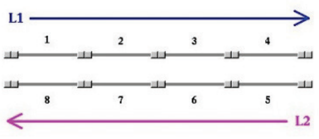

Figure 7 Example 2

7.1 Example 2a (without energy modelling)

We give an example consisting of a simple railroad system with two trains which is illustrated in Fig. 7. The routes of the two trains read as follows:

$$R_1 = p_2, v_1, p_3, v_2, p_4, v_3, v_4$$
$$R_2 = p_6, v_5, p_7, v_6, p_8, v_7, v_8$$

The travel times for the two trains are shown in Tab. 3. Fig. 8 illustrates the resulting graph of the example after applying Kronecker algebra.

Table 3 Travel times for train L_1 and L_2

Track section	Time value		Track section	Time value	
	L_1	L_2		L_1	L_2
p_1	0	0	v_1	2	0
p_2	0	0	v_2	4	0
p_3	0	0	v_3	2	0
p_4	0	0	v_4	4	0
p_5	0	0	v_5	0	1
p_6	0	0	v_6	0	2
p_7	0	0	v_7	0	1
p_8	0	0	v_8	0	2

7.2 Example 2b (Modelling Energy Consumption)

Now we add energy consumption to the trains, where train L_1 needs two energy units for track section p_1 and p_2 and one for p_3 and p_4. L_1 needs two units for track section p_5 and p_6 and one for p_7 and p_8.

So the extended routes for the two trains will be:

$$R_1 = p_9, p_9, p_2, v_1, p_3, v_2, v_9, p_4, v_3, v_4, v_9$$
$$R_2 = p_9, p_9, p_6, v_5, p_7, v_6, v_9, p_8, v_7, v_8, v_9$$

where p_9 models reservation of one single energy unit and v_9 its release.

Fig. 9 illustrates the resulting graph when a maximum capacity of four energy units is available and thus both trains can travel along their routes without constraints.

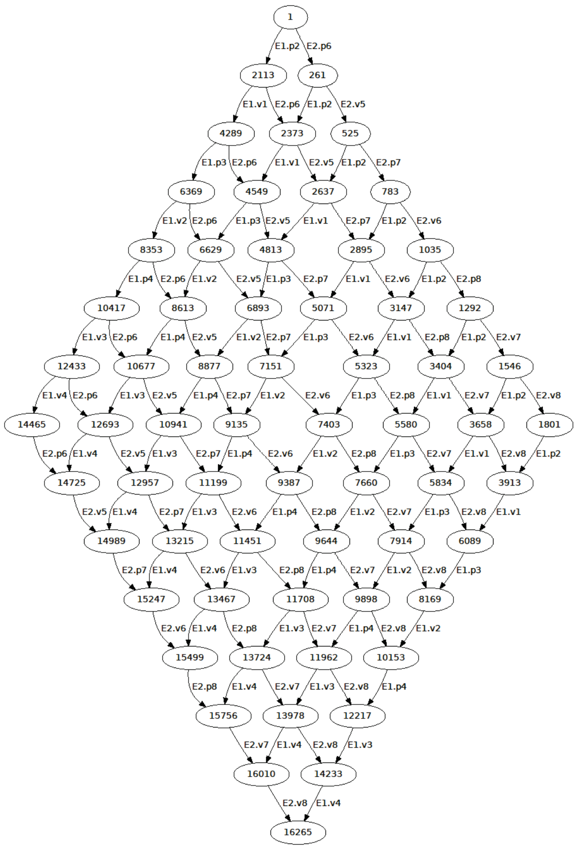

Figure 8 Resulting graph of example 2a

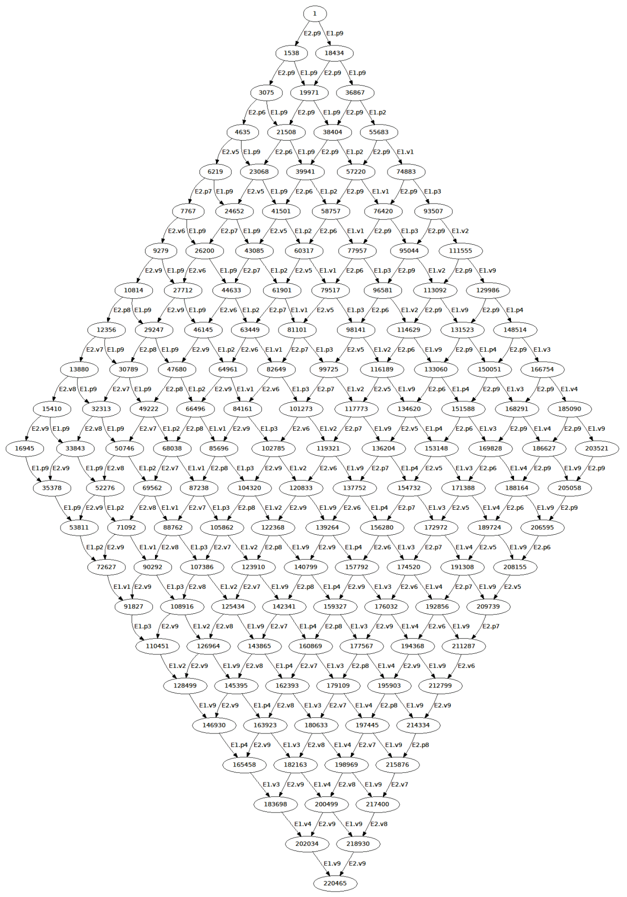

Figure 9 Resulting graph of example 2b (4 units)

Figure 10 shows the result if only three energy units are available and thus only one train can start travelling over its first two track sections and after the release of one energy unit, the second one can start.

7.3 Example 2c (modelling less energy available)

Now we add another constraint: The two trains should do a synchronization when train L_1 is at track section three and L_2 is a track section seven, respectively. In other words, the two trains have to wait for each other before they go on. To ensure synchronization for connections or overtaking, artificial track section has to be inserted into the routes. Adding the artificial track sections 10 and 11 to our routes, we get the following "routes":

$$R_1 = p_9, p_9, p_2, v_1, p_3, v_2, v_9, v_{10}, p_{11}, p_4, v_3, v_4, v_9$$
$$R_2 = p_9, p_9, p_6, v_5, p_7, v_6, v_9, p_{10}, v_{11}, p_8, v_7, v_8, v_9$$

Fig. 11 and Fig. 12 show the resulting graphs with additional energy constraints (four and three available energy units).

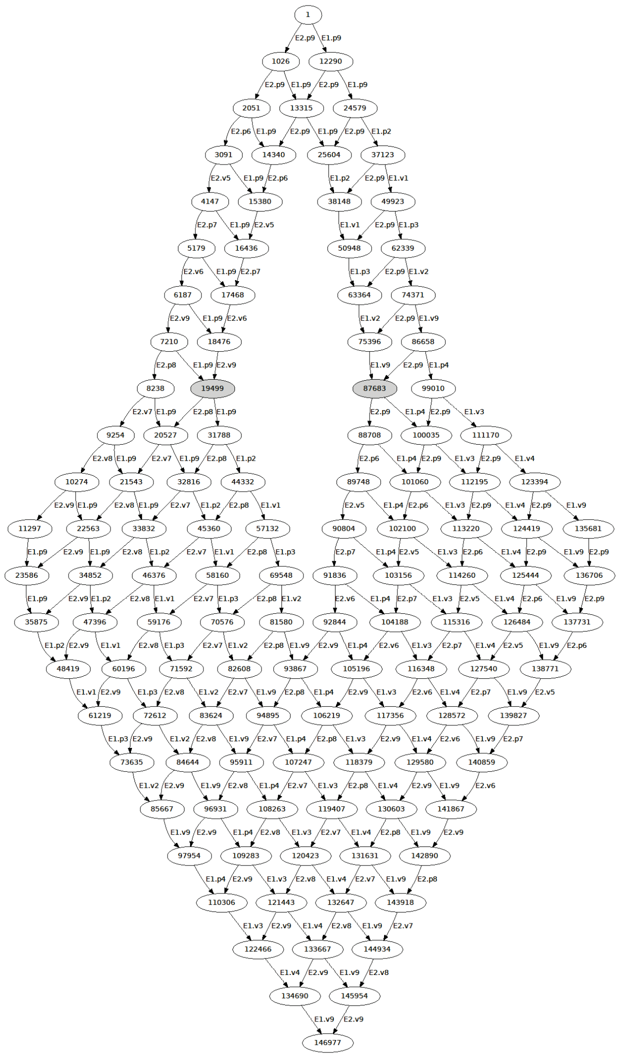
Figure 10 Resulting graph of example 2b (3 units)

In this section, we presented a simple example with two trains, then we added energy awareness to the example and at least, we added a constraint implied by a connection between the two trains.

Tab. 4 shows the results (travel time, number of nodes) for the example. We see that additional constraints (e.g. connections between trains) result in a more accurately defined issue and thus the resulting graphs are smaller.

Table 4 Results of example 2

Example	Travel time		Number of	
	L_1	L_2	nodes	synch. nodes
2a	12	6	64	0
2b (4 units)	12	6	144	0
2b (3 units)	15	12	119	2
2c (4 units)	12	9	103	2
2c (3 units)	15	12	78	4

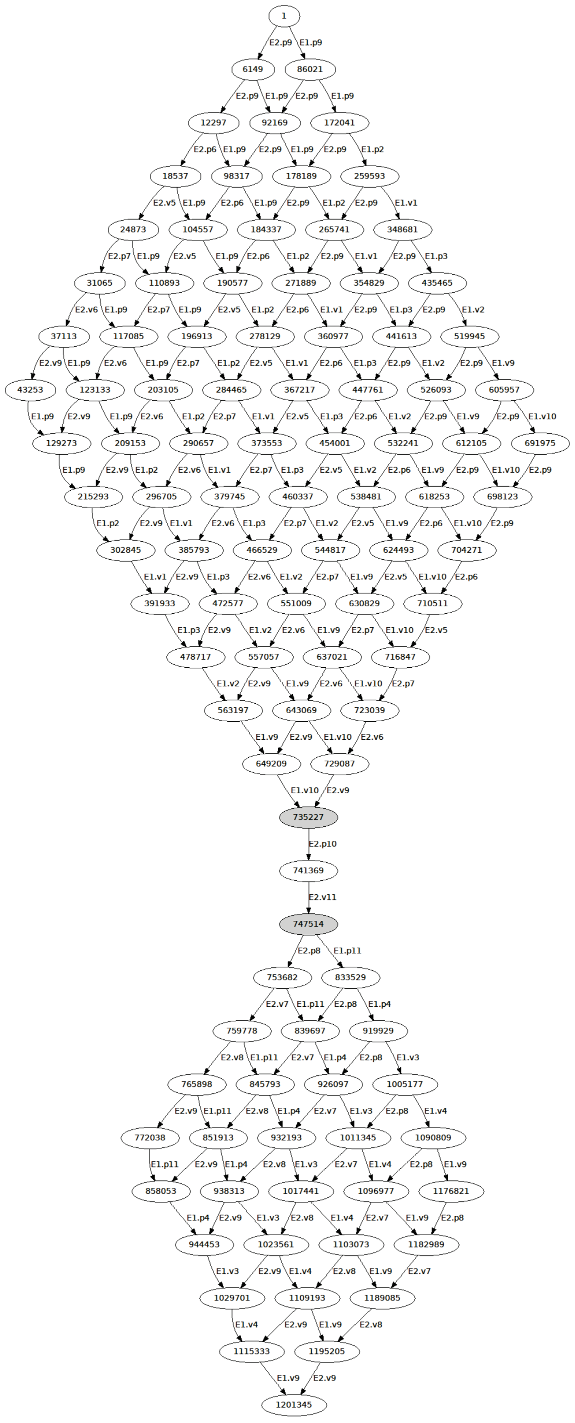
Figure 11 Resulting graph of example 2c (4 units)

8 Railway analysis by using Kronecker algebra

We have developed an algorithm [6] to reduce the graph and to calculate the travel time, which consists of several steps:

1. **Read data:** The necessary data of each train (e.g. route, time values) is read.
2. **Kronecker algebra:** Apply Kronecker algebra to manipulate the given matrices.
3. **Get the relevant nodes:** The algorithm starts at the first node and traverses the graph downwards until no successors can be reached. All reached nodes are stored in a map. Each node is checked and if it is a synchronizing node, it is stored in a different map including the necessary synchronizing information (involved trains, track section). This information is called synchronizing condition.
4. **Delete unnecessary synchronizing nodes:** The algorithm starts at the final node and traverses the graph upwards until the start node is reached. Some synchronizing nodes are not relevant for the computation of the travel time and thus, these nodes are removed from the synchronizing nodes map and considered as non-synchronizing nodes. In more detail, a synchronizing node can be considered as non-synchronizing node for the travel time calculation, if one of its successors contains the same synchronizing conditions. In this case the synchronization could be done in the successor. After applying Kronecker algebra there may exist chains of synchronizing nodes in the resulting graph with the same synchronizing conditions at each node. Step 4 eliminates all of them but the last one and thus the calculation is much easier and faster. Fig. 13 and Fig. 14 illustrate the elimination of unnecessary synchronizing nodes.

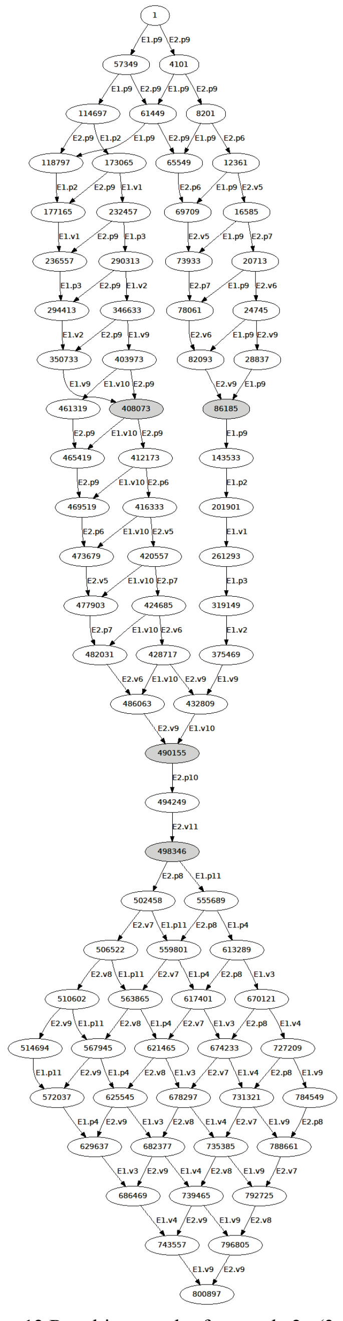
Figure 12 Resulting graph of example 2c (3 units)

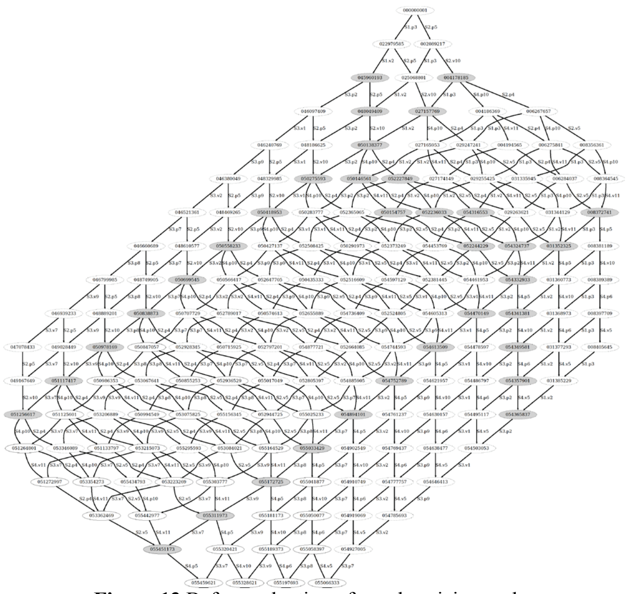
Figure 13 Before reduction of synchornizing nodes

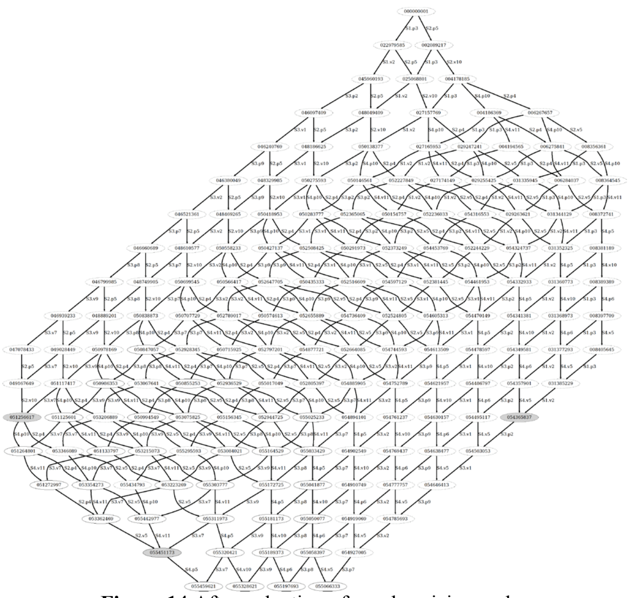
Figure 14 After reduction of synchronizing nodes

5. **Calculate the travel time:** For each synchronizing node, the travel time is calculated by using the time value from the map, inserting it into the equations (see Section 4) and calculating the maximum value. This is done by starting at the entry node and traversing the graph downwards until the final node is reached.

Fig. 15 depicts the reduced graph of the example of Section 5, where only four relevant nodes (entry node, final node and two synchronizing nodes) are remaining for the travel time calculation. Note that calculation of travel time can be redone with different time values without the need to regenerate the reduced graph.

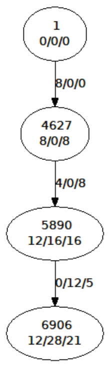

Figure 15 Reduced graph

In Tab. 5 information on several examples is gathered, the number of nodes, the number of relevant nodes (including entry-node and final-node) for travel time calculation and the running time of the algorithm (Processor: Intel Core i7 CPU 870@2.93GHz × 8, Memory: 4 GB). The timing behaviour of our algorithm is linear in the number of nodes in the graph.

Table 5 Examples and their running time

Example	Number of		Algorithm running time ms
	Nodes	Relevant nodes	
Travel time (Section 5)	6 912	4	< 15
2a	16 384	2	< 15
2b (3 units)	65 536	4	< 30
2b (4 units)	98 304	2	< 30
2c (3 units)	262 144	6	< 35
2b (4 units)	393,216	4	< 35
Travel time (11 track sections, 4 trains)	300 000 000	13	< 290

In our examples we have used integer values for travel time values. Extensions of our model, e.g. with decimal values or taking braking and acceleration time into account in case of blocking are possible, but not discussed here.

9 Conclusion

We have presented a graph-based method for calculating travel time of trains, finding deadlocks and taking into account energy considerations within a railway network on a fine-grained level. Kronecker algebra is applied to manipulate matrices and to create graphs, which can be represented by adjacency matrices. Our approach can be used to model complex railway systems including aspects of being deadlock-free, being conflict-free, and being minimal in terms of energy consumption.

Acknowledgements

The authors would like to thank the Austrian Ministry of Transport, Innovation and Technology (bmvit) and the Austrian Research Promotion Agency (FFG) for sponsoring the project EcoRailNet in the frame of the program New Energy 2020 (Project-ID: 834586).

10 References

[1] Mittermayr, R.; Blieberger, J. Shared Memory Concurrent System Verification using Kronecker Algebra. // Automation Systems Group, Technische Universität Wien, Vienna, Austria, 2011.
[2] Mittermayr, R.; Blieberger, J.; Schöbel, A. Kronecker Algebra based Deadlock Analysis for Railway Systems. // Promet - Traffic & Transportation. 24, 5(2012), pp. 359-369. DOI: 10.7307/ptt.v24i5.1171
[3] Volcic, M.; Blieberger, J.; Schöbel, A. Kronecker Algebra based Travel Time Analysis for Railway Systems. // FORMS/FORMAT 2012 - 9th Symposium on Formal Methods for Automation and Safety in Railway and Automotive Systems. (2012), pp. 273-281.
[4] Cui, Y. Simulation-Based Hybrid Model for a Partially-Automatic Dispatching of Railway Operation, Universität Stuttgart, 2010.
[5] Pachl, J. Steuerlogik für Zuglenkanlagen zum Einsatz unter stochastischen Betriebsbedingungen, TU Braunschweig, 1993.
[6] Volcic, M. Energy-efficient Optimization of Railway Operation. An Algorithm Based on Kronecker Algebra. // PhD-Thesis, 2014.
[7] Mittermayr, R.; Blieberger, J. Timing Analysis of Concurrent Programs. // Proc. 12th International Workshop on Worst-Case Execution Time Analysis, 2012.
[8] Volcic, M.; Blieberger, J.; Schöbel, A. Kronecker algebra and its broad applications in railway systems. // EURO-ŽEL 2013: Recent Challenges for European Railways, (2013), pp. 275-282.
[9] Volcic, M.; Blieberger, J.; Schöbel, A. Kronecker algebra as a frame for optimisation of railway operation. // 21st International Scientific Conference - TRANSPORT 2013; Mechanics Transport Communications. 11, 3(2013), pp. 57-63.
[10] Dijkstra, E. W. Over Seinpalen, 1965.
[11] Kuich, W.; Salomaa, A. Semirings, Automata, Languages, Springer, 1986. DOI: 10.1007/978-3-642-69959-7
[12] Buchholz, P.; Kemper, P. Efficient Computation and Representation of Large Reachability Sets for Composed Automata. // Discrete Event Dyn. Systems. 12, 3(2002), pp. 265-286. DOI: 10.1023/A:1015669415634
[13] Montigel, M. Innovatives Bahnleitsystem optimiert den Zugverkehr im Lötschberg-Basistunnel. // Signal + Draht. 9, (2008), pp. 20-22.
[14] Mehta, F.; Rößiger, C.; Montigel, M. Potenzielle Energieersparnis durch Geschwindigkeitsempfehlungen im Bahnverkehr. // Signal + Draht. 9, (2010), pp. 20-26.
[15] Plateau, B. On the Stochastic Structure of Parallelism and Synchronization Models for Distributed Algorithms. // ACM Sigmetrics. 13, 2(1985), pp. 147-154. DOI: 10.1145/317786.317819

Authors' addresses

Mark Stefan (Volcic), Dipl.-Ing.Dr. techn.
Austrian Institute of Technology GmbH
Giefinggasse 2, A-1210 Vienna, Austria
E-mail: mark.stefan@ait.ac.at

Johann Blieberger, Prof. Dipl.-Ing. Dr. techn.
TU Wien
Institute of Computer Aided Automation
Treitlstraße 1-3, A-1040 Vienna, Austria
E-mail: blieb@auto.tuwien.ac.at

Andreas Schöbel, Privatdoz. Dipl.-Ing. Dr. techn.
OpenTrack Railway Technology Ltd.
Kaasgrabengasse 19/8, A-1190 Vienna, Austria
Karlsplatz 13, 1040 Vienna, Austria
E-mail: andreas.schoebel@opentrack.at