Delay Management with Re-Routing of Passengers

Twan Dollevoet[1,2], Dennis Huisman[1,2], Marie Schmidt[3], and Anita Schöbel[3]

[1] Econometric Institute and ECOPT, Erasmus University Rotterdam
P.O. Box 1738, NL-3000 DR Rotterdam, the Netherlands.
{dollevoet,huisman}@ese.eur.nl

[2] Department of Logistics, Netherlands Railways
P.O. Box 2025, NL-3500 HA Utrecht, the Netherlands

[3] Institute for Numerical and Applied Mathematics,
Georg-August University, Göttingen, Germany.
{m.schmidt,schoebel}@math.uni-goettingen.de

Abstract. Trains often arrive delayed at stations where passengers have to change to other trains. The question of delay management is whether these trains should wait for the original train or depart on time. In traditional delay management models passengers always take their originally planned route. This means, they are in case of a missed connection always delayed with the cycle time of the timetable. In this paper, we propose a model where re-routing of passengers is incorporated.
To describe the problem we represent it as an event-activity network similar to the one used in traditional delay management, with some additional events to incorporate origin and destination of the passengers. We prove NP-hardness of this problem, and we present an integer programming formulation for which we report the first numerical results. Furthermore, we discuss the variant in which we assume fixed costs for maintaining transfers and we present a polynomial algorithm for the special case of only one origin-destination pair.

Key words: Public Transportation, Delay Management, Re-Routing, OD-pairs

1 Introduction and Motivation

Delay management is an important issue in the daily operations of railway companies. It deals with (small) source delays of a railway system as they occur in the daily operational business of any public transportation company. In case of such delays, the scheduled timetable is not feasible any more and has to be updated to a *disposition timetable*. Since delays can also be transferred if a connecting train waits for a delayed feeder train such connections are often not maintained in case of delays. These *wait-depart decisions* are important decisions for the passengers. In order to ensure safe operations and to take the limited capacity of the track system into account, also *priority decisions* are necessary. They determine the order in which trains are allowed to pass a specific piece of track.

There exist various models and solution approaches for delay management. The main question which has been treated in the literature so far is to decide which trains should wait for delayed feeder trains and which trains better depart on time (*wait-depart decisions*). It neglects the limited capacity of the tracks. A first integer programming formulation for this problem has been given in [Sch01] and has been further developed in [GHL08,Sch07], see also [Sch06] for an overview about various models. The complexity of the problem has been investigated in [GJPS05,GGJ+04] where it turns out that the problem is NP-hard even in very special cases. The online version of the problem has been studied in [GJPW07,Gat07]. In [BHLS07], it was shown that the online version of the uncapacitated delay management problem is PSPACE-hard. Further publications about delay management include a model in the context of max-plus-algebra [RdVM98,Gov98], a formulation as discrete time-cost tradeoff problem [GS07] and simulation approaches [SM99,SMBG01].

J. Clausen, G. Di Stefano (Eds): ATMOS 2009
9th Workshop on Algorithmic Approaches for Transportation Modeling,
Optimization, and Systems
http://drops.dagstuhl.de/opus/volltexte/2009/2143

Recently, the limited capacity of the track system is taken into account. This has been done heuristically in a real-world application studied within the project *DisKon* supported by Deutsche Bahn (see [BGJ+05]). Some first ideas on how to model these constraints in the context of delay management have been presented in [Sch09], heuristics and properties of the models including the never-meet property of uncapacitated delay management are presented in [SS08,SS09].

What has been neglected so far are the aspects of re-routing. In the available models it is assumed that passengers take exactly the lines they planned, i.e. if they miss a connection they have to wait a complete period of time until the same connection takes place again. This assumption is usually not valid in practice. Often there is an earlier connection using another line or even changing the path of the trip. A real-world example of a situation where re-routing passengers in case of delays is beneficial is given next.

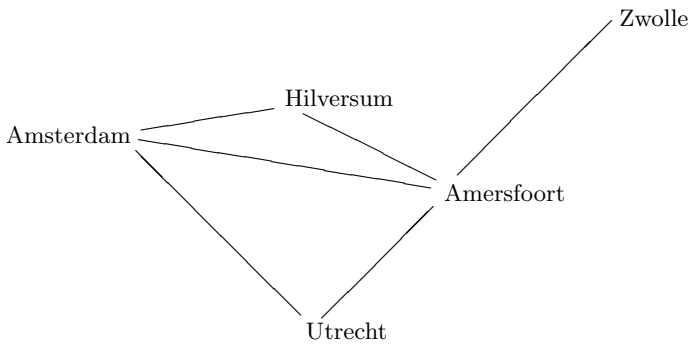

Fig. 1. A small part of the railway network in the Netherlands. A regional train runs from Amersfoort to Hilversum and further to Amsterdam. An intercity service runs from Zwolle to Utrecht and stops at station Amersfoort. All other trains are intercities as well.

Consider the network in Figure 1. An intercity service runs from Zwolle to Utrecht via Amersfoort. There are also intercities from Utrecht to Amsterdam and from Amersfoort to Amsterdam. Finally, a regional train runs via Hilversum from Amersfoort to Amsterdam. A large number of passengers want to travel from Zwolle to Amsterdam, and thus have a transfer at Amersfoort. In the current timetable, the intercity to Amsterdam departs from Amersfoort 5 minutes after the intercity from Zwolle has arrived. Therefore, if the intercity from Zwolle has a small delay, these passengers will miss the connecting intercity to Amsterdam. If the possibility of re-routing the passengers is not taken into account, the decision to delay the intercity from Amersfoort to Amsterdam assumes that the passengers that miss the connection at Amersfoort have to wait for one hour for the next intercity. However, these passengers will probably take the regional train via Hilversum, that departs a few minutes after the intercity has left. As the regional train stops at more locations, the travel time of the regional train is larger than that of the intercity, but the difference is only several minutes. The delay of the passengers will then be far less than one hour. If the delay is so large that the regional train has left as well, the passengers could stay in the delayed train and travel via Utrecht instead. The transfer time in Utrecht is much larger than in Amersfoort. This small example shows that the delay of passengers that miss a connection is often much smaller than one hour. To find the optimal wait-depart decisions, re-routing passengers should therefore be taken into account.

In our paper we will investigate how such a re-routing of passengers can be incorporated into the delay management problem. We denote the resulting model by *delay management with re-routing decisions (DMwRR)*. To the best of our knowledge a re-routing of passengers has never been treated before.

The remainder of the paper is structured as follows. In Section 2 we show how the re-routing of passengers can be modeled in the event-activity network and that delay management with re-routing is NP-hard. An integer program based on the event-activity network is formulated in Section 3. In Section 4 we present a polynomially solvable case in which we show how optimal wait-depart decisions can be made if only one origin-destination pair is present. We furthermore discuss another simplified variant in which we assume fixed delay costs for each maintained changing activity. We finally conclude the paper mentioning ideas for further research.

2 Model

We will make use of an event-activity network to model the delay management problem with re-routing. Event-activity networks were first introduced by [Nac98] for timetabling problems and were used for the classical delay management problems by [Sch06]. The event-activity network will be extended to take re-routing of passengers into account.
We assume that the number of passengers that want to travel from a given origin to a destination at a certain time is known. For example, 200 passengers want to travel from Zwolle to Amsterdam at 8 o'clock in the morning. We denote such an origin-destination pair by $p = \{u, v, s_{uv}\}$, where u is the origin, v is the destination and s_{uv} is the planned starting time of the trip. $\mathcal{P}$ denotes the set of all such origin-destination pairs. From now on, we will abbreviate an origin-destination pair as an OD-pair. We denote w_p for the number of passengers associated to an OD-pair $p \in \mathcal{P}$.
The event-activity network $\mathcal{N} = (\mathcal{E}, \mathcal{A})$ is a directed graph, where $\mathcal{E}$ denotes the set of events and the set $\mathcal{A}$ consists of the activities. The departure or the arrival of a train g at a station v, denoted by $(g - v - Dep)$ or $(g - v - Arr)$ respectively, are the most important events in the network. To incorporate the routes of the passengers, we introduce for every OD-pair $p = \{u, v, s_{uv}\} \in \mathcal{P}$ an origin event $(p - Org)$ and a destination event $(p - Dest)$. Note that besides the origin and the destination, the OD-pairs also contain the time at which passengers want to start their journey. In summary, the set of events in the network, denoted by $\mathcal{E}$, consists of the departure events of the trains, the arrival events of the trains and the origin and destination events for the passengers for a given OD-pair.

$$\mathcal{E} = \mathcal{E}_{\text{dep}} \cup \mathcal{E}_{\text{arr}} \cup \mathcal{E}_{\text{org}} \cup \mathcal{E}_{\text{dest}}.$$

The activities are the arcs in the directed graph $\mathcal{N}$. Similar to the event-activity network used by [Sch06] for the delay management problem without re-routing, there are driving arcs, waiting arcs and changing arcs. The driving and waiting arcs represent driving from one station to the next and waiting at a station to let the passengers get on and off the train. The changing activities are used by the passengers. They represent the possibility for passengers to transfer from a train that arrives at a certain station to a train that departs at the same station some time later. It should be noted that the driving and waiting arcs impose operational restrictions on the vehicles. On the contrary, a changing arc does not imply that a train has to wait in case of a delay of another train, although it would be convenient for the transferring passengers.
To take the rerouting of passengers into account, we also introduce origin and destination arcs. Let an origin event $e = (p - Org) \in \mathcal{E}_{\text{org}}$ be given, where $p = \{u, v, s_{uv}\}$ represents the passengers that want to travel from station u to station v at time s_{uv}. This event e is connected to all the departure events that depart from u not earlier than the time s_{uv}. It remains to connect the arrival events to the destination events. Consider therefore a destination event $(p - Dest) \in \mathcal{E}_{\text{dest}}$, where again $p = \{u, v, s_{uv}\}$. Denote SP_p for the arrival time of the passengers if there are no delays and denote n_p for the number of transfers needed for this trip. SP_p is clearly a lower bound for the arrival time of the passengers. To derive an upper bound on the arrival time, note that in the worst case all n_p connections are missed. We conclude that an arrival event e should be connected to $(p - Dest)$ if e is an arrival event at station v and if the planned time π_e satisfies $\pi_e \in [SP_p, SP_p + n_p T]$, where T is the cycle time of the original timetable. This concludes the description of the arcs in the event activity network. Summarizing,

$$\mathcal{A} = \mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}} \cup \mathcal{A}_{\text{change}} \cup \mathcal{A}_{\text{org}} \cup \mathcal{A}_{\text{dest}}.$$

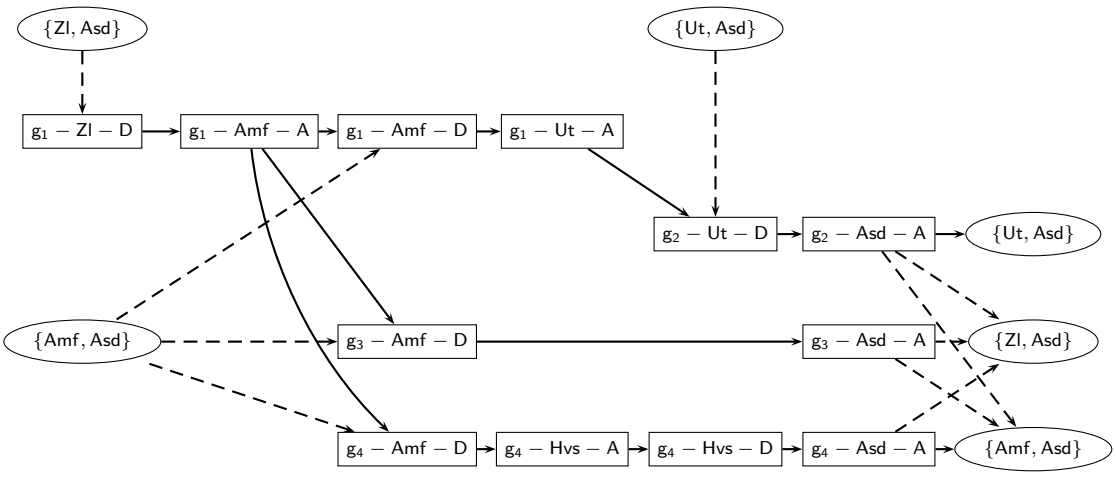

Fig. 2. The event activity network for the situation depicted in Figure 1. The square nodes are the departure and arrival events where "D" stands for departure and "A" stands for arrival. The origin and destination events are represented by ovals omitting the add-ons "Org" or 'Dest" as this is obvious in the picture. As we only consider one possible departure time for each origin-destination pair, we did not include the starting time in the origin and destination nodes. The dashed arcs are the origin and destination arcs, that are introduced to be able to state the shortest path problem for the passengers. The solid lines represent driving, waiting and changing activities.

An example of an event-activity network is given in Figure 2. This event-activity network corresponds to the railway network in Figure 1. The oval nodes represent the origin and destination events, that are introduced to model the behavior of passengers when delays occur. The dashed arcs, that depict the origin and destination arcs, are needed only to take re-routing of passengers into account. Recall that transfer arcs do not impose any operational constraints. It is therefore possible not to maintain a connection in case of delays, which would imply that passengers cannot use such a connection.

For every activity $a \in \mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}} \cup \mathcal{A}_{\text{change}}$ a length L_a is given that represents the technically minimal time that is needed to perform the activity. As the origin and destination activities are not activities in the original sense and thus they are not time consuming, their lengths can be set to 0 or they can just be omitted.

For every event $e \in \mathcal{E}_{\text{arr}} \cup \mathcal{E}_{\text{dep}}$, the planned time is denoted by π_e, i.e. π corresponds to the timetable as it is planned to be operated. For an origin event $e = (p - Org) \in \mathcal{E}_{\text{org}}$ with $p = \{u, v, s_{uv}\}$ we set $\pi_e = s_{uv}$ (which can be interpreted as the time at which a passenger of OD-pair p arrives at his or her departure station). For destination events we have to determine the time when the passengers reach their last station, hence π_e is not known beforehand.

Given a timetable, for every OD-pair a route through the network has to be found, so that the travel time is minimized. To this end, let P be a directed path from e_1 to e_2 in the network $\mathcal{N}$.

- First, assume that $e_2 \in \mathcal{E}_{\text{dep}} \cup \mathcal{E}_{\text{arr}}$. We define $l(P) = \pi_{e_2} - \pi_{e_1}$ to be the travel time or distance between e_1, e_2 in $\mathcal{N}$.
- We now extend this definition to nodes $e_2 \in \mathcal{E}_{\text{dest}}$. Let $pre(e_2, P)$ be the predecessor of e_2 in path P. Then we define $l(P) = \pi_{pre(e_2,P)} - \pi_{e_1}$.

For a path P connecting an OD-pair $p = \{u, v, s_{uv}\}$ we hence obtain $l(P) = \pi_{pre(e_2,P)} - s_{uv}$. As we assume that passengers take the fastest paths to arrive at their destinations, we set $l(p) = l(\hat{P}_{uvs_{uv}})$ where $\hat{P}_{uvs_{uv}}$ is a shortest path from the origin event $e = (p - Org)$ to the destination event $e = (p - Dest)$.

Given a set of source delays d_e associated to some events $e \in \mathcal{E}_{\text{arr}} \cup \mathcal{E}_{\text{dep}}$ the problem is to decide which trains should wait for passengers to arrive from delayed trains and which should depart without waiting. Thus we have to determine which of the connections $a \in \mathcal{A}_{\text{change}}$ will be maintained and which will be removed. We denote the set of maintained connections by $\mathcal{A}_{\text{fix}}$. For the resulting network

$$\mathcal{N}(\mathcal{A}_{\text{fix}}) := (\mathcal{E}, \mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}} \cup \mathcal{A}_{\text{fix}} \cup \mathcal{A}_{\text{org}} \cup \mathcal{A}_{\text{dest}})$$

in which the set of changing arcs has been replaced by $\mathcal{A}_{\text{fix}}$ a new timetable can be constructed using the critical path method (see [Sch07]). The event times for the events $e \in \mathcal{E}_{\text{dep}} \cup \mathcal{E}_{\text{arr}}$ in this new timetable will be denoted by x_e. For an OD-pair p we define $t_{\mathcal{A}_{\text{fix}}}(p) = x_e$ where e is the predecessor of the destination event $(p - Dest)$ on a shortest path from the origin event $(p - Org)$ to the destination event $(p - Dest)$ in the network $\mathcal{N}(\mathcal{A}_{\text{fix}})$.
In $\mathcal{N}(\mathcal{A}_{\text{fix}})$ the travel time of an OD-pair $p = \{u, v, s_{uv}\}$ is analogously defined as

$$l_{\mathcal{A}_{\text{fix}}}(p) = t_{\mathcal{A}_{\text{fix}}}(p) - s_{uv}.$$

In the delay management problem we want to minimize the sum of all delays of the OD-pairs. The delay of an OD-pair $p = \{u, v, s_{uv}\}$ is given as

$$l_{\mathcal{A}_{\text{fix}}}(p) - l(p) = t_{\mathcal{A}_{\text{fix}}}(p) - s_{uv} - l(p).$$

Since s_{uv} and $l(p)$ are constants we can equivalently minimize $t_{\mathcal{A}_{\text{fix}}}(p)$, hence the objective of delay management with re-routing is to find a subset $\mathcal{A}_{\text{fix}} \subset \mathcal{A}_{\text{change}}$ so that we minimize:

$$\min_{\mathcal{A}_{\text{fix}} \subset \mathcal{A}_{\text{change}}} \sum_{p \in \mathcal{P}} w_p \cdot t_{\mathcal{A}_{\text{fix}}}(p).$$

Our first result is to clarify the the complexity status of delay management problem with re-routing and show that it is NP-hard. This is not surprising, because the delay management without re-routing is NP-hard as well ([GJPS05]).

Theorem 1. *Delay management with re-routing is NP-hard.*

Proof. The proof will be done by reduction to the "Uncapacitated Facility Location" (UFL) problem. An instance of UFL consists of a set of potential facilities $J = \{1, \ldots, n\}$ and a set of customers $I = \{1, \ldots, m\}$ which have to be served by the facilities. A customer can only be served by a facility if it is opened. Let f_j be the cost for opening facility j and c_{ij} be the transportation cost for serving customer i from facility j. The objective of UFL is to find a subset $Q \subseteq J$ and an assignment of the customers to the facilities so that the total cost consisting of the opening cost of the facilities and the transportation cost is minimized. The objective function is:

$$\tilde{f}(Q) := \min \left(\sum_{i \in I} \min_{j \in Q} c_{ij} + \sum_{j \in Q} f_j \right).$$

For a given instance of UFL we define the following instance of delay management with re-routing (see Figure 3).

- We consider a transportation system with $2 + m + n$ stations, namely two fixed stations u and $\tilde{u}$ and stations v_i for all $i \in I$ and $\tilde{v}_j$ for all $j \in J$.
- As trains we consider one train g running from u to $\tilde{u}$, trains h_j running from $\tilde{u}$ to the stations $\tilde{v}_j$ and trains k_{ij} linking each pair of stations $(\tilde{v}_j, v_i)$. Altogether we hence have $1 + n + mn$ trains each of them driving between one pair of stations only.
- We use the event-activity network based on this transportation system with a departure event, a driving activity of length 1 and an arrival event for each of the $1 + m + nm$ trains. There are no waiting activities. We have the following set $C_1 \cup C_2$ of changing activities consisting of

- transfers from the train g to each of the trains h_j, $j = 1, \ldots, n$. These are the changing activites $c_j = \{(g - \tilde{u} - Arr, h_j - \tilde{u} - Dep)\}, j = 1, \ldots, n$ with length 1. We define

$$C_1 = \{c_j : j \in J\}$$

- transfers from a train h_j to a train k_{ij}, $j = 1, \ldots, n, i = 1, \ldots, m$, i.e.

$$C_2 = \{(h_j - v_j - Arr, k_{ij} - \tilde{v}_j - Dep) : j \in J, i \in I\}.$$

– Furthermore, we need OD-pairs $\mathcal{P}$ given as

$$\mathcal{P} = \{p_i = \{u, v_i, 0\} \forall i \in I\} \cup \{\tilde{p}_j = \{\tilde{u}, \tilde{v}_j, 2\} \forall j \in J\}.$$

We set the number of passengers wanting to travel between the corresponding origin and destination events as $w_{p_i} = 1$ for every $p_i = \{u, v_i, 0\}$ and $w_{\tilde{p}_j} = f_j$ for every $\tilde{p}_j = \{\tilde{u}, \tilde{v}_j, 2\}$.

– Finally, as source delay we assume that the departure event of train g is delayed by $d = 1$ minute.

First we note that maintaining the connection between the trains h_j and k_{ij} does not cause additional delay for any OD-pair. So we can assume that all changing activitities in C_2 are maintained and will in the following only consider such solutions.
Now let $Q \subseteq J$ be a subset of opened facilities. We define a relation between such opened facilities and maintained connections which is only based on the maintained connections in C_1:

$$\mathcal{A}_{\text{fix}}^Q := \{c_j \in C_1 : j \in Q\} \cup C_2.$$

Vice versa for a given subset $\mathcal{A}_{\text{fix}} \subseteq C_1 \cup C_2$ we set

$$Q^{\mathcal{A}_{\text{fix}}} = \{j \in J : c_j \in \mathcal{A}_{\text{fix}}\}.$$

Thus we have a bijection between subsets $Q \subset J$ and subsets $\mathcal{A}_{\text{fix}} \subset \mathcal{A}_{\text{change}}$. It holds:

1. Q is feasible for (UFL) if and only if all passengers reach their destinations if $\mathcal{A}_{\text{fix}}^Q$ is chosen as set of maintained connections.
2. The objective values of (UFL) and delay management with passenger re-routing coincide up to an additive constant, i.e. $f(\mathcal{A}_{\text{fix}}^Q) = \tilde{f}(Q) + const.$

ad 1: A solution Q to an instance of UFL is feasible if and only if there is at least one opened facility. Similarly, all passengers will reach their final destinations if and only if the set of maintained connections within C_1 is not empty.

ad 2: For a given feasible solution Q to an instance of UFL the objective value is

$$\tilde{f}(Q) = \sum_{i \in I} \min_{j \in Q} c_{ij} + \sum_{j \in Q} f_j.$$

In the associated solution network $\mathcal{N}(\mathcal{A}_{\text{fix}}^Q)$ for every OD-pair $p_i = \{u, v_i, 0\}$ the arrival time $t_{\mathcal{A}_{\text{fix}}}(p_i)$ can be calculated depending on the chosen train k_{ij} by adding the lengths L_a of the activities on the path in the event-activity network and the delay $d = 1$. Furthermore, for every OD-pair $\tilde{p}_j = \{\tilde{u}, \tilde{v}_j, 2\}$ the arrival time $t_{\mathcal{A}_{\text{fix}}}(\tilde{p}_j)$ is $t_{\mathcal{A}_{\text{fix}}}(\tilde{p}_j) = s_{\tilde{u}\tilde{v}_j} + 1 + d = 4$ if the connection $(g - \tilde{u} - Arr, h_j, \tilde{u}, Dep)$ is kept alive and $t_{\mathcal{A}_{\text{fix}}}(\tilde{p}_j) = s_{\tilde{u}\tilde{v}_j} + 1 = 3$ otherwise. Thus the associated solution has solution value:

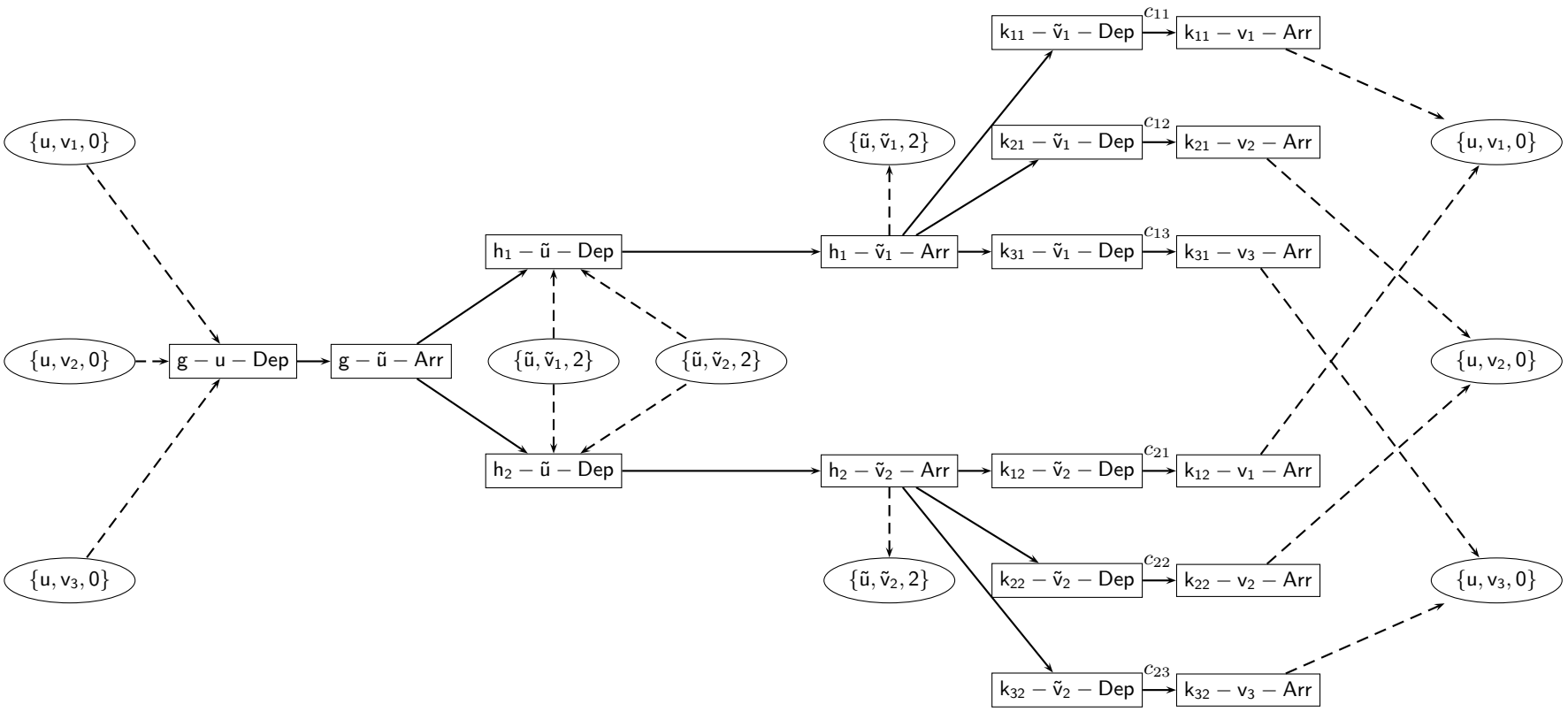

Fig. 3. The event activity network for the instance of the delay management problem with re-routing constructed from an instance of UFL with $m = 3$ customers and $n = 2$ facilities. The square nodes are the departure and arrival events. The origin and destination events are represented by ovals omitting the add-ons "Org" or 'Dest" as this is obvious in the picture. The dashed arcs are the origin and destination arcs, the solid lines represent driving and changing activities.

$$\begin{aligned}
\sum_{p\in\mathcal{P}} w_p \cdot t_{\mathcal{A}_{\text{fix}}}(p) =& \sum_{i\in I} w_{p_i} \cdot t_{\mathcal{A}_{\text{fix}}}(p_i) + \sum_{j\in J} w_{\tilde{p}_j} \cdot t_{\mathcal{A}_{\text{fix}}}(\tilde{p}_j) \\
=& \sum_{i\in I} t_{\mathcal{A}_{\text{fix}}}(p_i) + \sum_{j\in J} f_j \cdot t_{\mathcal{A}_{\text{fix}}}(\tilde{p}_j) \\
=& \sum_{i\in I} \min_{j\in Q}(4 + c_{ij} + d) + \sum_{j\in Q} f_j \cdot 4 + \sum_{j\notin Q} f_j \cdot 3 \\
=& \sum_{i\in I} \left(5 + \min_{j\in Q} c_{ij}\right) + \sum_{j\in Q} f_j \cdot 4 + \sum_{j\notin Q} f_j \cdot 3 \\
=& \sum_{i\in I} \min_{j\in Q} c_{ij} + 5 \cdot |I| + \sum_{j\in Q} f_j + \sum_{j\in J} f_j \cdot 3 \\
=& f(Q) + \left(5 \cdot |I| + \sum_{j\in J} f_j \cdot 3\right)
\end{aligned}$$

□

We remark that NP hardness of a similar model also dealing with delay management with re-routing of passengers has been shown in [GGJ+04].

3 Integer Programming Formulation

In this section we will give an integer programming formulation that takes the routing decisions for the passengers into account explicitly. The model is based on the classical delay management model as it was introduced in [Sch01]. We will refer to this classical delay management model as the original model.
The event activity network is a directed graph. We denote $\delta^{\text{in}}(e)$ and $\delta^{\text{out}}(e)$ for the set of arcs into e and out of e, respectively, for every event $e \in \mathcal{E}$.

3.1 Variables

The most important decision is which connections need to be kept alive. For each changing activity $a \in \mathcal{A}_{\text{change}}$ we thus introduce a binary decision variable z_a, which is defined as follows.

$$z_a = \begin{cases} 1 & \text{if connection } a \text{ is maintained,} \\ 0 & \text{otherwise.} \end{cases}$$

The times that the arrival and departure events take place are the next set of decision variables. For each event $e \in \mathcal{E}_{\text{arr}} \cup \mathcal{E}_{\text{dep}}$, we define $x_e \in \mathbb{N}$ as the rescheduled time that event e takes place. The variables $x = (x_e)$ therefore define the disposition timetable. These decision variables are the same as in the original model.
The new aspect that we have to model are the routes that the passengers take. First note that a route has to be determined for every origin-destination pair. Recall that the set $\mathcal{P}$ is defined as the set of all origin-destination pairs. To model the routing decisions for a given pair $p \in \mathcal{P}$, we introduce binary decision variables q_{ap}, which indicate whether arc $a \in \mathcal{A}$ is used in the path that is chosen for origin-destination pair $p \in \mathcal{P}$. Formally, the variables q_{ap} are defined as follows.

$$q_{ap} = \begin{cases} 1 & \text{if connection } a \text{ is used by passengers in } p, \\ 0 & \text{otherwise.} \end{cases}$$

The arrival time for p now depends both on the path that is chosen, and on the disposition timetable x. To be able to incorporate the arrival time of these passengers in a linear model, we introduce a variable $t_p \in \mathbb{N}$, which will represent the arrival time for pair $p \in \mathcal{P}$.

3.2 Integer programming formulation

We first present our integer programming formulation for (DMwRR) and then discuss its meaning.

$$\min \sum_{p \in \mathcal{P}} w_p t_p \tag{1}$$

such that

$$\begin{aligned}
x_e &\geq \pi_e + d_e && \forall e \in \mathcal{E}_{\text{arr}} \cup \mathcal{E}_{\text{dep}}, && (2)\\
x_e &\geq x_{e'} + L_a && \forall a = (e', e) \in \mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}}, && (3)\\
x_e &\geq x_{e'} + L_a - M_1(1 - z_a) && \forall a = (e', e) \in \mathcal{A}_{\text{change}}, && (4)\\
q_{ap} &\leq z_a && \forall p \in \mathcal{P}, a \in \mathcal{A}_{\text{change}}, && (5)\\
\sum_{a \in \delta^{\text{out}}(e)} q_{ap} &= 1 && \forall e = (p - Org) \in \mathcal{E}_{\text{org}}, && (6)\\
\sum_{a \in \delta^{\text{out}}(e)} q_{ap} &= \sum_{a \in \delta^{\text{in}}(e)} q_{ap} && \forall p \in \mathcal{P}, e \in \mathcal{E}_{\text{arr}} \cup \mathcal{E}_{\text{dep}}, && (7)\\
1 &= \sum_{a \in \delta^{\text{in}}(e)} q_{ap} && \forall e = (p - Dest) \in \mathcal{E}_{\text{dest}}, && (8)\\
t_p &\geq x_e - M_2(1 - q_{pa}) && \forall e = (p - Dest) \in \mathcal{E}_{\text{dest}}, a \in \delta^{\text{in}}(e), && (9)\\
z_a &\in \{0, 1\} && \forall a \in \mathcal{A}_{\text{change}}, && (10)\\
q_{ap} &\in \{0, 1\} && \forall p \in \mathcal{P}, a \in \mathcal{A}, && (11)\\
x_e &\in \mathbb{N} && \forall e \in \mathcal{E}_{\text{arr}} \cup \mathcal{E}_{\text{dep}}, && (12)\\
t_p &\in \mathbb{N} && \forall p \in \mathcal{P}. && (13)
\end{aligned}$$

The objective function (1) minimizes the arrival times of all passengers. This is equivalent to minimizing the overall or average delay of the passengers. Constraints (2) imply that events cannot take place earlier than in the original timetable and that source delays are taken into account. To make sure that delays are propagated through the network correctly, constraints (3) transfer the delay from the start of activity a to its end. For maintained connections, that is connections for which $z_a = 1$, constraints (4) transfer delays from the feeder train to the connecting train. The value of M_1 should be chosen large enough for these constraints to be correct. In [Sch06] it has been shown that $M_1 = \max_{e \in \mathcal{E}} d_e$ is large enough. Constraints (2 - 4) are also present in the original model.

Constraints (5 - 9) take the routing decisions into account. First of all, constraints (5) make sure that changing activities can only be used if the connection is kept alive. Constraints (6 - 8) define a shortest path problem for each origin-destination pair p. For every pair, a path is selected from the origin to the destination. The last constraint defines the arrival time for trip p, where M_2 is again a large number. For the arrival event e that is selected and the driving activity a into this event, $q_{pa} = 1$, showing that $t_p \geq x_e$ for this particular event. All other path variables q_{pa} are equal to zero, therefore putting no restriction on the value of t_p.

To find the minimal value of M_2 for which (9) is correct, consider an arbitrary OD-pair $p \in \mathcal{P}$. It was shown in Section 2 that only arrival events that arrive within n_p periods after the planned arrival of the passengers should be connected to the destination event $(p, \text{destination})$, where n_p is the number of transfers for these passengers if the timetable is operated as planned. The maximal delay for the OD-pair p is therefore equal to $n_p T + \max_e d_e$. Assuming that no passenger has more than two transfers, it follows that $M_2 = 2T + \max_{e \in \mathcal{E}} d_e$ is large enough. Indeed, as

$$x_e - M_2 \leq \pi_e + \max_{e \in \mathcal{E}} d_e - M_2 \leq SP_p + 2T + \max_{e \in \mathcal{E}} d_e - M_2 = SP_p,$$

where SP_p is the planned arrival time, $t_p \geq x_e - M_2$ does not pose a restriction.

We remark that the variables z_a are not needed in the above model, since constraints (4) and (5) can be replaced by the constraint

$$x_e \geq x_{e'} + L_a - M(1 - q_{ap}) \quad \forall a = (e', e) \in \mathcal{A}_{\text{change}} \forall p \in \mathcal{P}$$

leading to an equivalent model. Nevertheless, we have chosen to leave these variables in the model to show the similarity with earlier models. Furthermore, the variables z_a could be used to guide the solution process.

3.3 Some preliminary numerical results

We have implemented the integer program for a small part of the railway network in the Netherlands. This small sample consists of 10 stations in the center of the Netherlands, including the stations in Figure 1. The timetable and the passenger figures are obtained from Netherlands Railways. We consider 184 trips and 141 OD-pairs during a planning period of 5 hours in the evening. The sample under consideration contains many OD-pairs for which different routes are possible, especially near Amsterdam.
The resulting event-activity network contains 502 nodes and 1475 arcs. The number of changing activities is equal to 542. The integer program was solved using CPLEX 10.1 on an Intel Core 2 Duo PC (2.33 GHz) with 3 GB of memory. For randomly selected delays, the problem can be solved to optimality within 30 seconds. If only the train from Zwolle to Amersfoort is delayed, as in our motivating example in Section 1, we indeed see that passengers are re-routed via Utrecht. It should be noted that in all our tests, the optimal solution is found in less than 5 seconds, although it takes about 30 seconds to prove optimality of the solution.

4 Special Cases of Delay Management with Re-Routing

In the precedent section we gave an integer programming formulation for the general problem (DMwRR). Now we will identify simplifications and special cases of (DMwRR) in order to understand the border between still polynomial solvable and already NP-hard variants. The knowledge about the reasons for the NP-hardness as well as polynomial approaches for special cases can later serve to construct good heuristics for the general case.
In this section we will hence examine two special cases of (DMwRR). We first present a polynomial algorithm for the case of delay management with re-routing where the demand is given by only one OD-pair. Then we will consider another variant in which the costs for maintaining a connection are fixed. Although this is a strong simplification of delay management with re-routing, it will turn out to be NP-hard as well.

4.1 Delay management with re-routing for one single OD-pair

This subsection deals with a simplification of delay management with re-routing (DMwRR): We assume that we are given just one OD-pair $p = \{u, v, s_{uv}\}$. To simplify the notation in the following chapter we will identify $(p - Org)$ and u and $(p - Dest)$ and v, so u and v will be regarded as events in the network. In this case the problem is solvable by a modified version of Dijkstra's algorithm for finding a shortest path (see [VC79]). The part of the algorithm that has to be modified is the calculation of the node labels that in Dijkstra's algorithm represent the shortest-path distance to the origin and in the modified algorithm will represent the earliest possible arrival time at a node. In order to calculate the transfer of delays efficiently we define $Tr[e]$ as the train belonging to an event $e \in \mathcal{E}_{\text{dep}} \cup \mathcal{E}_{\text{arr}}$.
Let $\mathcal{N}$ be a network with feasible timetable π, $p = \{u, v, s_{uv}\}$ an OD-pair and $\mathcal{D}$ a set of source delays. Like in the original Dijkstra's algorithm we solve in every step the problem of determining an optimal path for a pair of events $\{u, i\}$ where $u = (p - Org)$ is the origin node of the OD-pair $p = \{u, v, s_{uv}\}$ under consideration and $i \in \mathcal{E}$. In order to do this formally, we need the following slight extension of (DMwRR):

Having in mind the practical application in train re-routing we defined in Section 2 the problem (DMwRR) for a network $\mathcal{N}$ and a set of OD-pairs $\mathcal{P}$ consisting of elements of the form $p = \{u, v, s_{uv}\}$ where u is the origin, v the destination and s_{uv} is the starting time. Now we also want to deal with OD-pairs as elements of the type $p^* = \{u, i, s_{uv}\}$ where $i \in \mathcal{E}$ is an *arbitrary* successor of u in $\mathcal{N}$. From a mathematical point of view we can do this easily by just defining $t_{\mathcal{A}_{\text{fix}}}(p^*) := x_i$ as the (artificial) arrival time of such an OD-pair p^*. We hence extend the problem (DMwRR) to instances consisting of a network $\mathcal{N}$ and a set of OD-pairs $\mathcal{P}$ of type p^*.
Let u be the origin node of the considered OD-pair. Determining an optimal path for a pair of events $\{u, i\}$ can hence be seen as solving (DMwRR) for $\mathcal{N}$ and $\mathcal{P} = \{\{u, i, s_{uv}\} : i \in \mathcal{E}\}$. If the problem (DMwRR) is solved for $\{u, i, s_{uv}\}$ we store:

- $T[i]$: Minimal arrival time for passengers traveling from u to i with starting time s_{uv}.
- $\mathcal{A}_{\text{fix}}[i]$: Changing activities that have to be maintained in the optimal solution of (DMwRR) with OD-pair $\{u, i, s_{uv}\}$.
- $TD[i] = \{j : (e, j) \in \mathcal{A}_{\text{fix}}[i]$ for some $e \in \mathcal{E}\}$: Set of (departure) events that transfer a delay to a new train if the optimal path for OD-pair $\{u, i, s_{uv}\}$ is realized.

Let $PERM$ be the set of events for which (DMwRR) has been solved and the above values have been determined. For every e with a direct predecessor $i \in PERM$ we determine the optimal path by first calculating the time plus the delay transferred to e if the connections that belong to the optimal path to i are fixed:

$$z_i[e] = \begin{cases} \max\{\tilde{\pi}_e, T[j] + \sum_{a \in P_{je}} L_a\} & \text{if there is an event } j \in TD[i] \text{ such that } Tr[j] = Tr[e] \\ \tilde{\pi}_e & \text{otherwise} \end{cases}$$

where P_{je} is the path from j to e containing only events of the same train $Tr[j] = Tr[e]$. Then the delay of e when taking a path via i is $\max\{z_i[e], T[i] + L_{(i,e)}\}$. We consequently choose i so that this expression is minimal and obtain $\tilde{T}[e] = \min_{i \in PERM, (i,e) \in \mathcal{A}}\{z_i[e], T[i] + L_{(i,e)}\}$. As in Dijkstra's algorithm we fix the event $\hat{e}$ with smallest $\tilde{T}[e]$.
In order to calculate $\mathcal{A}_{\text{fix}}[\hat{e}]$ and $TD[\hat{e}]$ we distinguish two cases. Let $i_{\hat{e}}$ be the predecessor of $\hat{e}$ in the solution of (DMwRR) for $\{u, \hat{e}, s_{uv}\}$.

- If $\hat{a} = (i_{\hat{e}}, \hat{e})$ is a changing activity and $T[\hat{e}] > z_{i_{\hat{e}}}[\hat{e}]$ we obtain $\mathcal{A}_{\text{fix}}[\hat{e}] = \mathcal{A}_{\text{fix}}[i_{\hat{e}}] \cup \{(i_{\hat{e}}, \hat{e})\}$ and $TD[\hat{e}] = TD[i_{\hat{e}}] \cup \{\hat{e}\}$.
- Otherwise we simply set $\mathcal{A}_{\text{fix}}[\hat{e}] = \mathcal{A}_{\text{fix}}[i_{\hat{e}}]$ and $TD[\hat{e}] = TD[i_{\hat{e}}]$.

The algorithm is summarized below.

Algorithm: Modified Dijkstra for delay management with re-routing with one OD-pair

Input: Instance of (DMwRR) with network $\mathcal{N}$, feasible timetable π, delays $\mathcal{D}$ and one OD-pair $p = \{u, v, s_{uv}\}$.

Step 1. Generate the timetable $\tilde{\pi}$ where $\tilde{\pi}_e = \max_{(i,e) \in \mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}}}\{\pi_e, \tilde{\pi}_i + L_{(i,e)}\}$ by the critical path method.

Step 2. Set $PERM = \{u\}$, $TEMP = E \setminus \{u\}$, $T[u] = s_{uv}$, $\tilde{T}[e] = \infty$ for every $e \in TEMP$, $TD[u] = \emptyset$, $\mathcal{A}_{\text{fix}}[u] = \emptyset$.

Step 3. For every $e \in TEMP$ and every $i \in PERM$ so that $(i, e) \in \mathcal{A}$ set

- $z_i[e] = \begin{cases} \max\{\tilde{\pi}_e, T[j] + \sum_{a \in P_{je}} L_a\} & \text{if there is an event } j \in TD[i] \text{ such that } Tr[j] = Tr[e] \\ \tilde{\pi}_e & \text{otherwise} \end{cases}$
  where P_{je} is the path from j to e containing only nodes of $Tr[j] = Tr[e]$.
- $\tilde{T}[e] = \min_{i \in PERM, (i,e) \in \mathcal{A}} \max\{z_i[e], T[i] + L_{(i,e)}\}$.

Step 4. Set $\hat{e} = \operatorname{argmin} \tilde{T}[e]$, $i_{\hat{e}} = \operatorname{argmin}_{i \in PERM, (i,\hat{e}) \in \mathcal{A}}\{T[i] + L_{(i,\hat{e})}\}$, $PERM = PERM \cup \{\hat{e}\}$, $TEMP = TEMP \setminus \{\hat{e}\}$, $T[\hat{e}] = \tilde{T}[\hat{e}]$.

Step 5. If $\hat{e} = v$ go to Step 7.

Step 6. If $(i_{\hat{e}}, \hat{e}) \in \mathcal{A}_{\text{change}}$ and $T[\hat{e}] > z_{i_{\hat{e}}}[\hat{e}]$
set $\mathcal{A}_{\text{fix}}[\hat{e}] = \mathcal{A}_{\text{fix}}[i_{\hat{e}}] \cup \{(i_{\hat{e}}, \hat{e})\}$ and $TD[\hat{e}] = \{TD[i_{\hat{e}}] \cup \{\hat{e}\}\} \setminus \{j \in \mathcal{E}_{dep} : Tr[j] = Tr[\hat{e}]\}$.
Otherwise set $TD[\hat{e}] = TD[i_{\hat{e}}]$, $\mathcal{A}_{\text{fix}}[\hat{e}] = \mathcal{A}_{\text{fix}}[i_{\hat{e}}]$.
Go to step 3.
Step 7. Set $\mathcal{A}_{\text{fix}} = \mathcal{A}_{\text{fix}}[v]$
Output: Optimal set $\mathcal{A}_{\text{fix}}$ for the given instance of (DMwRR).

Theorem 2. *The algorithm is correct and finds the optimal solution $\mathcal{A}_{fix}$ to (DMwRR) with one OD-pair in time $O(n^4)$ where n is the number of nodes in the network $\mathcal{N}$.*

Proof. (a). Using induction we see that for a directed path P_{ie} from i to e with $Tr[e] = Tr[i]$ that contains only nodes of the train $Tr[e] = Tr[i]$ where event j precedes event e

$$T[i] + \sum_{a \in P_{ie}} L_a \leq T[j] + L_{(j,e)}.$$

(b). For a given set $A \subset \mathcal{A}_{\text{change}}$ let $x^A[e]$ for $e \in \mathcal{E} \setminus \{v\}$ denote the minimal possible arrival times calculated by the critical path method in $\mathcal{N}(A)$ where $x^A[u] = s_{uv}$. Note that for $A = \emptyset$ $x^{\emptyset}[e] = \tilde{\pi}_e$ for all $e \in \mathcal{E}_{\text{arr}} \cup \mathcal{E}_{\text{dep}}$.

(c). For any solution $\mathcal{A}_{\text{fix}}[e] \subset \mathcal{A}_{\text{change}}$ regarding an OD-pair $\{u, e, s_{uv}\}$ for an $e \in \mathcal{E}$ if we construct $\tilde{\mathcal{A}}_{\text{fix}}[e]$ from $\mathcal{A}_{\text{fix}}[e]$ by removing the edges that are not on a shortest path from u to e in $\mathcal{N}(\mathcal{A}_{\text{fix}}[e])$ it holds that $x^{\tilde{\mathcal{A}}_{\text{fix}}[e]}[e] = x^{\mathcal{A}_{\text{fix}}[e]}[e]$. So we will assume that in the optimal solution to the problem of finding a shortest path from u to e only the connections on the shortest path from u to e are maintained.

(d). For the set $\mathcal{A}_{\text{fix}}[\hat{e}]$ that is constructed in the algorihm in step 6 as solution for the path between u and $\hat{e}$ because of (c) we get

$$x^{\mathcal{A}_{\text{fix}}[\hat{e}]}[e] = \max\{\tilde{\pi}_e, \max_{(i,e) \in \delta^{\text{in}}(e) \cap (\mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}} \cup \mathcal{A}_{\text{fix}})} \{x^{\mathcal{A}_{\text{fix}}[\hat{e}]}[i] + L_{(i,e)}\}\}$$

for all $e \in \mathcal{E}_{\text{arr}} \cup \mathcal{E}_{\text{dep}}$.

(e). Adding changing activities to a set A_1 does not influence the time for events e that happen before the added activities take place. That means for two sets $A_1 \subset A_2 \subset \mathcal{A}_{\text{change}}$ if for all $a = (e_1, e_2) \in A_2 \setminus A_1$ $x^{A_2}(e_1) \geq x^{A_2}(e)$, it holds that $x^{A_1}[e] = x^{A_2}[e]$.

(f). Furthermore we observe that if for a set $\mathcal{A}_{\text{fix}}[\hat{e}]$ for all i such that $(i, e) \in \mathcal{A}_{\text{fix}} \cup \mathcal{A}_{\text{drive}} \cup \mathcal{A}_{\text{wait}}$ $TD[i] \cap Tr[e] = \emptyset$, it holds that $x^{\mathcal{A}_{\text{fix}}[\hat{e}]}[e] = \tilde{\pi}_e$.

First we will show inductively that for every node e with an incoming arc (f, e) it holds that

$$\max\{z_f[e], T[f] + L_{(f,e)}\} = x^{\mathcal{A}_{\text{fix}}[f] \cup \{(f,e)\}}[e] \tag{14}$$

if (f, e) is a changing arc and

$$\max\{z_f[e], T[f] + L_{(f,e)}\} = x^{\mathcal{A}_{\text{fix}}[f]}[e] \tag{15}$$

otherwise.

1. First we regard the edges $(u, e) \in \mathcal{A}_{\text{org}}$. As $\mathcal{A}_{\text{fix}}[u] = \emptyset$ and $s_{uv} \leq \tilde{\pi}_e$ because of (b) it holds that

$$\max\{z_u[e], T[u] + L_{(u,e)}\} = \max\{\tilde{\pi}_e, s_{uv} + 0\} = \tilde{\pi}_e = x^{\mathcal{A}_{\text{fix}}[u]}[e].$$

2. Let e be a node such that its predecessor f in P_{uv} lies on the same train $T[f] = T[e]$. As in $\mathcal{N}(\mathcal{A}_{\text{fix}}[f])$ it holds that $\delta^{\text{in}}[e] = \{(f, e)\}$ (see (c)) and because of (a):

$$\begin{aligned} \max\{z_f[e], T[f] + L_{(f,e)}\} &= \max\{\tilde{\pi}_e, T[f] + L_{(f,e)}\} \\ &= \max\{\tilde{\pi}_e, x^{\mathcal{A}_{\text{fix}}[f]}[f] + L_{(f,e)}\} \\ &= x^{\mathcal{A}_{\text{fix}}[f]}[e]. \end{aligned}$$

3. Let e be a node such that its predecessor f in P_{uv} does not lie on the same train, $T[f] \neq T[e]$. Then $(f,e) \in \mathcal{A}_{\text{change}}$.
   - Suppose that there is no waiting arc terminating in e. Then because of (c) and (e):

$$\begin{aligned}\max\{z_f[e], T[f] + L_{(f,e)}\} &= \max\{\tilde{\pi}_e, T[f] + L_{(f,e)}\} \\ &= \max\{\tilde{\pi}_e, x^{\mathcal{A}_{\text{fix}}[f]}[f] + L_{(f,e)}\} \\ &= \max\{\tilde{\pi}_e, x^{\mathcal{A}_{\text{fix}}[f]\cup\{(f,e)\}}[f] + L_{(f,e)}\} \\ &= x^{\mathcal{A}_{\text{fix}}[f]\cup\{(f,e)\}}[e].\end{aligned}$$

   - Suppose that there is a waiting arc (e_w, e) terminating in e and that $Tr[e] \cup TD[f] = \emptyset$. Thus $\tilde{\pi}_{e_w} = x^{\mathcal{A}_{\text{fix}}[f]}[e_w]$ because of (f) and considering (e) it follows

$$\begin{aligned}\max\{z_f[e], T[f] + L_{(f,e)}\} &= \max\{\tilde{\pi}_e, T[f] + L_{(f,e)}\} \\ &= \max\{\tilde{\pi}_e, T[f] + L_{(f,e)}, \tilde{\pi}_{e_w} + L_{(e_w,e)}\} \\ &= \max\{\tilde{\pi}_e, x^{\mathcal{A}_{\text{fix}}[f]}[f] + L_{(f,e)}, x^{\mathcal{A}_{\text{fix}}[f]}[e_w] + L_{(e_w,e)}\} \\ &= \max\{\tilde{\pi}_e, x^{\mathcal{A}_{\text{fix}}[f]\cup\{(f,e)\}}[f] + L_{(f,e)}, x^{\mathcal{A}_{\text{fix}}[f]\cup\{(f,e)\}}[e_w] + L_{(e_w,e)}\} \\ &= x^{\mathcal{A}_{\text{fix}}[f]\cup\{(f,e)\}}[e].\end{aligned}$$

   - Suppose that there is $(e_w, e) \in \mathcal{A}_{\text{wait}}$ and $Tr[e] \cup TD[f] = \{e_d\}$. Let $\tilde{P}_{e_d e}$ be the path on $Tr[e_d] = Tr[e_w] = Tr[e]$ from e_d to e and $\tilde{P}_{e_d e_w}$ be the path on $Tr[e_d] = Tr[e_w] = Tr[e]$ from e_d to e_w. We see inductively that $x^{\mathcal{A}_{\text{fix}}[f]}[e_w] = \max\{\tilde{\pi}_{e_w}, x^{\mathcal{A}_{\text{fix}}[f]}[e_d] + \sum_{a\in\tilde{P}_{e_d e_w}} L_a\}$. Together with (e) follows:

$$\begin{aligned}&\max\{z_f[e], T[f] + L_{(f,e)}\} \\ =&\max\{\tilde{\pi}_e, T[e_d] + \sum_{a\in\tilde{P}_{e_d e}} L_a, T[f] + L_{(f,e)}\} \\ =&\max\{\tilde{\pi}_e, \tilde{\pi}_{e_w} + L_{(e_w,e)}, T[e_d] + \sum_{a\in\tilde{P}_{e_d e_w}} L_a + L_{(e_w,e)}, T[f] + L_{(f,e)}\} \\ =&\max\{\tilde{\pi}_e, (\max\{\tilde{\pi}_{e_w}, T[e_d] + \sum_{a\in\tilde{P}_{e_d e_w}} L_a\} + L_{(e_w,e)}), T[f] + L_{(f,e)}\} \\ =&\max\{\tilde{\pi}_e, (\max\{\tilde{\pi}_{e_w}, x^{\mathcal{A}_{\text{fix}}[f]}[e_d] + \sum_{a\in\tilde{P}_{e_d e_w}} L_a\} + L_{(e_w,e)}), x^{\mathcal{A}_{\text{fix}}[f]}[f] + L_{(f,e)}\} \\ =&\max\{\tilde{\pi}_e, x^{\mathcal{A}_{\text{fix}}[f]}[e_w] + L_{(e_w,e)}, x^{\mathcal{A}_{\text{fix}}[f]}[f] + L_{(f,e)}\} \\ =&\max\{\tilde{\pi}_e, x^{\mathcal{A}_{\text{fix}}[f]\cup\{(f,e)\}}[e_w] + L_{(e_w,e)}, x^{\mathcal{A}_{\text{fix}}[f]\cup\{(f,e)\}}[f] + L_{(f,e)}\} \\ =&x^{\mathcal{A}_{\text{fix}}[f]\cup\{(f,e)\}}[e].\end{aligned}$$

Thus the assumption given by equations (14) and (15) holds.

If follows that the calculation of

$$\begin{aligned}\tilde{T}[e] &= \min_{i\in PERM,(i,e)\in\mathcal{A}} \max\{z_i[e], T[i] + L_{(i,e)}\} \\ &= \min_{i\in PERM}\{\min_{(i,e)\in\mathcal{A}_{\text{change}}}\{x^{\mathcal{A}_{\text{fix}}[f]\cup\{(f,e)\}}[e]\}, \min_{(i,e)\in\mathcal{A}\setminus\mathcal{A}_{\text{change}}}\{x^{\mathcal{A}_{\text{fix}}[f]}[e]\}\}\end{aligned}$$

leads to the optimal path from u to e among the set of paths where an element i from the actual set $PERM$ precedes e.
It remains to show that the set $\mathcal{A}_{\text{fix}}[\hat{e}]$ and the label $T[\hat{e}] = x^{\mathcal{A}_{\text{fix}}[\hat{e}]}[\hat{e}]$ are optimal for the node $\hat{e}$ chosen in step 4 of the algorithm, that means that there is no $A \subset \mathcal{A}_{\text{change}}$ such that there is a path from u to e in $\mathcal{N}(A)$ and

$$x^A[\hat{e}] < x^{\mathcal{A}_{\text{fix}}[\hat{e}]}[\hat{e}].$$

This assumption will be proven inductively, too. For the origin node u setting $\mathcal{A}_{\text{fix}}[u] = \emptyset$ leads to $T[u] = s_{uv}$ which is optimal.
Suppose that in the iterations 1 to $k-1$ of the algorithm the choice of $\mathcal{A}_{\text{fix}}[e]$ and the labels $T[e]$ are optimal for the regarded nodes e.
Now let $\hat{e}$ be the node such that in the k-th iteration $T[\hat{e}] = \tilde{T}[\hat{e}] \leq \tilde{T}[e]$ for every $e \in TEMP$. Suppose that there is a set $A \subset \mathcal{A}_{\text{change}}$ such that there is a path from u to e in $\mathcal{N}(A)$ and

$$x^A[\hat{e}] < x^{\mathcal{A}_{\text{fix}}[\hat{e}]}[\hat{e}]. \tag{16}$$

Let $P^A_{u\hat{e}}$ be the optimal path from u to $\hat{e}$ in $\mathcal{N}(A)$.

(A). If the predecessor e_0 of $\hat{e}$ in $P^A_{u\hat{e}}$ is in $PERM$, because of the assumption that the labels $T[e]$ and chosen sets $\mathcal{A}_{\text{fix}}[e]$ are optimal for all $e \in PERM$

$$\begin{aligned} x^A[\hat{e}] =& x^{\mathcal{A}_{\text{fix}}[e_0] \cup \{(e_0,\hat{e})\}}[\hat{e}] \\ =& \min_{i \in PERM} \{ \min_{(i,\hat{e}) \in \mathcal{A}_{\text{change}}} x^{\mathcal{A}_{\text{fix}}[f] \cup \{(f,\hat{e})\}}[\hat{e}], \min_{(i,\hat{e}) \in \mathcal{A} \setminus \mathcal{A}_{\text{change}}} x^{\mathcal{A}_{\text{fix}}[f]}[\hat{e}] \} \\ =& x^{\mathcal{A}_{\text{fix}}[\hat{e}]}[\hat{e}] \end{aligned}$$

if $(e_0, \hat{e}) \in \mathcal{A}_{\text{change}}$ and

$$\begin{aligned} x^A[\hat{e}] =& x^{\mathcal{A}_{\text{fix}}[e_0]}[\hat{e}] \\ =& \min_{i \in PERM} \{ \min_{(i,\hat{e}) \in \mathcal{A}_{\text{change}}} x^{\mathcal{A}_{\text{fix}}[f] \cup \{(f,\hat{e})\}}[\hat{e}], \min_{(i,\hat{e}) \in \mathcal{A} \setminus \mathcal{A}_{\text{change}}} x^{\mathcal{A}_{\text{fix}}[f]}[\hat{e}] \} \\ =& x^{\mathcal{A}_{\text{fix}}[\hat{e}]}[\hat{e}] \end{aligned}$$

otherwise, which is a contradiction to (16).

(B). If the predecessor e_0 of $\hat{e}$ in $P^A_{u\hat{e}}$ is in $TEMP$ let e_1 denote the last node in $PERM$ on the path $P^A_{u\hat{e}}$ (e_1 exists because $u \in PERM$) and $e_2 \in TEMP$ its successor. So as $T[\hat{e}] = \tilde{T}[\hat{e}] \leq \tilde{T}[e]$ for $e \in TEMP$

$$x^A[\hat{e}] > x^A[e_2] = \max\{z_{e_1}[e_2], T[e_1] + L_{(e_1,e_2)}\} \geq \tilde{T}[e_2] \geq \tilde{T}[\hat{e}] = T[\hat{e}] = x^{\mathcal{A}_{\text{fix}}[\hat{e}]}[\hat{e}]$$

which contradicts (16).

Now it remains to show that $\mathcal{A}_{\text{fix}}[v]$ is the optimal solution to (DMwRR) for the OD-pair $p = \{u, v, s_{uv}\}$. As defined in Section 2, $\mathcal{A}_{\text{fix}}[v]$ is optimal if it minimizes $t_{\mathcal{A}_{\text{fix}}}(p) = x^{\mathcal{A}_{\text{fix}}}[e]$ for the predecessor e of v on a shortest path from u to v in the network $\mathcal{N}(\mathcal{A}_{\text{fix}})$. Suppose that the set $\mathcal{A}_{\text{fix}}[v]$ and the predecessor e calculated by the algorithm are not optimal with regard to an optimal path from u to v. The same considerations as above in (A) and (B) lead to a contradiction. So the set $\mathcal{A}_{\text{fix}}[v]$ as it is set in step 7 of the algorithm indeed is the optimal solution to (DMwRR) for the OD-pair $p = \{u, v, s_{uv}\}$.
The generation of the timetable in step 1 is done in time $O(n^2)$ by the procedure given in [Sch07] as well as step 7. The initialization of the algorithm in step 2 can be done in time $O(n)$. As in each repetition of the steps 3-6 one element is removed from $TEMP$, the number of times the steps 3-6 are repeated is bounded by $n-1$, the number of elements initially contained in $TEMP$. We observe that for given $e \in TEMP$ and $i \in PERM$ with $(i, e) \in \mathcal{A}$ the calculation of $z_i[e]$ and $\tilde{T}[e]$ can be done in time $O(n)$ if for each node $k \in \mathcal{E}_{\text{arr}} \cup \mathcal{E}_{\text{dep}}$ a pointer to the (unique) successor of k on an arc $a \in \mathcal{A}_{\text{wait}} \cup \mathcal{A}_{\text{drive}}$ is stored. So step 3 can be executed in time $O(n^3)$. As the steps 4-6 are done in time $O(n)$ the running time of the modified Dijkstra algorithm is in $O(n^4)$.

□

4.2 Re-routing with fixed costs

The delays that arise in delay management with re-routing for the passengers by the wait-depart decisions for the connections can be divided into two types:

1. A connection is maintained: The waiting train and the passengers on the waiting train are delayed.
2. A connection is not maintained: The passengers that wanted to take this connection have to travel along another, probably longer path.

Calculating the delay of the first type by a heuristic approach motivates the following simplified re-routing problem with fixed costs:
Let $\mathcal{N} = \{\mathcal{E}, \mathcal{A}\}$ be a directed network with edge lengths L_a for all $a \in \mathcal{A}$. Let $\mathcal{A}_{\text{change}} \subset \mathcal{A}$ be a set of connections that can be maintained or removed. We assume that maintaining a connection $a \in \mathcal{A}_{\text{change}}$ yields a fixed delay of d_a for the passengers. Let $\mathcal{P} = \{\{u, v\}\}$ be a set of OD-pairs, given as a subset of $\mathcal{E} \times \mathcal{E}$ with demand w_p for each $p \in \mathcal{P}$. The objective of this variant is to minimize the costs arising as fixed delays for maintaining connections plus the travel costs of the OD-pairs. Hence, the objective function is

$$\min_{\mathcal{A}_{\text{fix}} \subset \mathcal{A}_{\text{change}}} \sum_{p \in \mathcal{P}} w_p \cdot D_{\mathcal{A}_{\text{fix}}}(u, v) + \sum_{a \in \mathcal{A}_{\text{fix}}} d_a$$

where $D_{\mathcal{A}_{\text{fix}}}(u, v)$ is the optimal path distance from u to v in the network in which all connections $a \in \mathcal{A}_{\text{change}} \setminus \mathcal{A}_{\text{fix}}$ are removed.

Like in delay management with re-routing this problem can be solved in polynomial time if there is only one OD-pair (by adding the fixed costs d_a divided by the demand of the OD-pair w_p for a connection a to its length L_a and applying Dijkstra's algorithm) but even this simplified variant is NP-hard in general.

Theorem 3. *Re-routing with fixed costs is NP-hard.*

Proof. Analogously to the proof of NP-hardness for (DMwRR) we can prove this theorem by constructing an equivalent re-routing with fixed costs problem for each instance of UFL. The network $\tilde{\mathcal{N}}$ we construct here differs from the network $\mathcal{N}$ considered in the proof of Theorem 1 only in the absence of the OD-pairs $\{\tilde{u}, \tilde{v}_j\}$ and the associated origin and destination nodes $(\tilde{p}{-}Org)$ and $(\tilde{p}{-}Dest)$ and origin and destination arcs $(\tilde{p}{-}Org, h_{j_2}{-}\tilde{u}{-}Dep)$ and $(h_j{-}\tilde{v}_j{-}Arr, \tilde{p}{-}Dest)$. The fixed costs for $a \in \mathcal{A}_{\text{change}}$ are given by $d_{(g-\tilde{u}-Arr, h_j-\tilde{u}-Dep)} = f_j$ and $d_{(h_j-\tilde{v}_j-Arr, k_{ij}-\tilde{v}_j-Dep)} = 0$. Similar to the proof of Theorem 1 we observe that we can assume the connections $(h_j - \tilde{v}_j - Arr, k_{ij} - \tilde{v}_j - Dep)$ to be maintained because their fixed costs are 0. Like in that proof for a given set of facilities Q we define

$$\mathcal{A}^Q_{\text{fix}} := \{(g - \tilde{u} - Arr, h_j - \tilde{u} - Dep) : j \in Q, i \in I\} \cup \{(h_j - \tilde{v}_j - Arr, k_{ij} - \tilde{v}_j - Dep) : j \in J, i \in I\}.$$

and for a given subset $\mathcal{A}_{\text{fix}} \supset \{(h_j - \tilde{v}_j - Arr, k_{ij} - \tilde{v}_j - Dep) : j \in J, i \in I\}$ we set

$$Q^{\mathcal{A}_{\text{fix}}} = \{j \in J : (g - \tilde{u} - Arr, h_j - \tilde{u} - Dep) \in \mathcal{A}_{\text{fix}}\}. \tag{17}$$

Now a subset $Q \subset J$ and the associated subset $\mathcal{A}_{\text{fix}}$ are both feasible or infeasible and the difference between their objective values is $5 \cdot |I|$ as can be seen analogously to the proof of Theorem 1.

□

5 Conclusion and Further Research

In this paper, we introduced a model that allows to react to delayed trains not only by wait-depart decisions for the following trains but also by re-routing of passengers. For this purpose we introduced the origin and destination of the passengers as events in the event-activity network used in delay management and connected the wait-depart decisions to a shortest path problem in the resulting network. We proved that this problem is NP-hard. Furthermore, we developed an integer programming formulation for the delay management problem with re-routing.

Two main directions for further research on delay management with re-routing can be distinguished. First, special cases of the problem should be considered. For these special cases, faster solution procedures can be developed. For example, if the event-activity network has a special structure, this structure can be exploited to solve the delay management problem more efficiently. The methods to solve these easier problems can be used in the second direction of research: solving the delay management problem. In the paper we have reported some initial computational results on a small instance of the Dutch railway network. However, more experiments are required. In practice, the delay management problem should be solved on a very short notice. Therefore, heuristics should be developed that find a reasonable solution within a short computation time. To evaluate the quality of the solutions found by the heuristics, it is also interesting to investigate exact solution methods. Decomposing the problem in the wait-depart decisions on one hand and the re-routing of the passengers on the other hand could improve the running times of the exact solution methods.
In practice, the limited capacity of the infrastructure has a large impact on the real-time performance of a railway operator. Therefore, the capacity constraints should be integrated in the delay management models. Considering other routing or network location problems under the aspect of demand given as OD-pairs may also lead to interesting problems.

References

[BGJ+05] N. Bissantz, S. Güttler, J. Jacobs, S. Kurby, T. Schaer, A. Schöbel, and S. Scholl. DisKon - Disposition und Konfliktlösungs-management für die beste Bahn. *Eisenbahntechnische Rundschau (ETR)*, 45(12):809–821, 2005. (in German).

[BHLS07] A. Berger, R. Hoffmann, U. Lorenz, and S. Stiller. Online delay management: Pspace hardness and simulation. Technical Report ARRIVAL-TR-0097, ARRIVAL Project, 2007.

[Gat07] M. Gatto. *On the Impact of Uncertainty on Some Optimization Problems: Combinatorial Aspects of Delay Management and Robust Online Scheduling*. PhD thesis, ETH Zürich, 2007.

[GGJ+04] M. Gatto, B. Glaus, R. Jacob, L. Peeters, and P. Widmayer. Railway delay management: Exploring its algorithmic complexity. In *Proc. 9th Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 3111 of *LNCS*, pages 199–211, 2004.

[GHL08] L. De Giovanni, G. Heilporn, and M. Labbé. Optimization models for the single delay management problem in public transportation. *European Journal of Operational Research*, 189(3):762–774, 2008.

[GJPS05] M. Gatto, R. Jacob, L. Peeters, and A. Schöbel. The computational complexity of delay management. In D. Kratsch, editor, *Graph-Theoretic Concepts in Computer Science: 31st International Workshop (WG 2005)*, volume 3787 of *Lecture Notes in Computer Science*, 2005.

[GJPW07] M. Gatto, R. Jacob, L. Peeters, and P. Widmayer. On-line delay management on a single train line. In *Algorithmic Methods for Railway Optimization*, number 4359 in Lecture Notes in Computer Science, pages 306–320. Springer, 2007.

[Gov98] R.M.P. Goverde. The max-plus algebra approach to railway timetable design. In *Computers in Railways VI: Proceedings of the 6th international conference on computer aided design, manufacture and operations in the railway and other advanced mass transit systems, Lisbon, 1998*, pages 339–350, 1998.

[GS07] A. Ginkel and A. Schöbel. To wait or not to wait? The bicriteria delay management problem in public transportation. *Transportation Science*, 41(4):527–538, 2007.

[Nac98] K. Nachtigall. *Periodic Network Optimization and Fixed Interval Timetables*. Deutsches Zentrum für Luft– und Raumfahrt, Institut für Flugführung, Braunschweig, 1998. Habilitationsschrift.

[RdVM98] B. De Schutter R. de Vries and B. De Moor. On max-algebraic models for transportation networks. In *Proceedings of the International Workshop on Discrete Event Systems*, pages 457–462, Cagliari, Italy, 1998.

[Sch01] A. Schöbel. A model for the delay management problem based on mixed-integer programming. *Electronic Notes in Theoretical Computer Science*, 50(1), 2001.

[Sch06] A. Schöbel. *Customer-oriented optimization in public transportation*. Optimization and Its Applications. Springer, New York, 2006.

[Sch07] A. Schöbel. Integer programming approaches for solving the delay management problem. In *Algorithmic Methods for Railway Optimization*, number 4359 in Lecture Notes in Computer Science, pages 145–170. Springer, 2007.

[Sch09] A. Schöbel. Capacity constraints in delay management. *Public Transport*, 2009. to appear.

[SM99] L. Suhl and T. Mellouli. Requirements for, and design of, an operations control system for railways. In *Computer-Aided Transit Scheduling*. Springer, 1999.

[SMBG01] L. Suhl, T. Mellouli, C. Biederbick, and J. Goecke. Managing and preventing delays in railway traffic by simulation and optimization. In *Mathematical methods on Optimization in Transportation Systems*, pages 3–16. Kluwer, 2001.

[SS08] M. Schachtebeck and A. Schöbel. IP-based techniques for delay management with priority decisions. In Matteo Fischetti and Peter Widmayer, editors, *ATMOS 2008 - 8th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems*, Dagstuhl Seminar proceedings, 2008.

[SS09] M. Schachtebeck and A. Schöbel. To wait or not to wait and who goes first? Delay management with priority decisions. Technical report, Institut für Numerische und Angewandte Mathematik, Georg-August Universität Göttingen, 2009. NAM Report.

[VC79] J.M. Moore V. Chachra, P.M. Ghare. *Applications of Graph Theory Algorithms*. Elsevier North-Holland, New York, 1979.