Heuristics for Delay Management in Utrecht, Gelderland and Noord-Brabant

Max van den Helder

July 6, 2015

Abstract

Since standard integer programming techniques do not succeed in tackling large-scale real-world instances, I apply several heuristics for delay management. The first heuristic I implement is the Waiting Time Rule (WTR), where the decision whether or not to delay a connecting train, in order to maintain a connection, is based on a predetermined threshold, varying between 0 and 5 minutes. In contrast to the WTR, the Ratio of Transferring Passengers (RTP), which is the second heuristic I implement, actually takes into account the number of passengers that plan to use a connection. This could be beneficial for the results since now the usage of a specific connection is considered in the decision. Finally, the third heuristic is based on the classical delay management model without passenger rerouting from Schöbel (2007). This model can be solved with the assumption that passengers who miss a connection will wait for an estimate of additional delay (D) or for the timetable cycle time (the time after which the timetable repeats itself). For each heuristic, the goal is to find the parameter value for which the heuristic performs best (d_{max} for the WTR, ρ_{min} for the RTP and D for the classical model). I evaluate the outcomes for these heuristics separately for the case where only long-distance trains and corresponding stations are taken into account and for the case where all trains and stations are considered. To this end, I execute numerical experiments on real-world instances from Netherlands Railways. I discuss the results and evaluate and compare the performance of the different heuristics. Finally, some improvements and directions for further research are listed.

Supervisor: Dr. T.A.B. Dollevoet
Co-reader: Dr. D. Huisman

Contents

1 Introduction

Netherlands Railways is the main carrier on the Dutch railway network and the management team would like to know how to act in case of a delayed train, regarding to possible transfers on the next station. Is it best to delay a train for a few minutes to maintain a connection between that train and a delayed feeder train (a train which is, given the time, 'connected' to another train at a station they have in common), or is it better to let the train depart on time to prevent delay for the passengers in this train? To answer this question, I will implement three heuristics. Initially, passenger rerouting is not taken into account in the heuristics, but it is considered when evaluating the passenger routing based on the disposition timetable. Passenger rerouting can be seen as traveling by a train via another route to arrive at a destination quicker than by waiting on the next train via the same route. The decisions that have to be made for each connection, can be described as delay management.

One can think of several consequences that play a role in the decision whether or not to delay a connecting train. At first, especially in low-frequency railway systems, the waiting time for passengers can be quite long when they miss a connection. For this reason, delaying a connecting train can be beneficial, especially if the delay of the feeder train is small. In this case, the delay can be made up in a short time and only a few passengers, if any, suffer from this delay. However, if the feeder train arrives at a station with reasonable delay, delaying a connecting train is most of the time that bad for many other passengers, that it does not compensate the benefit for the passengers whose connection would be maintained.

Secondly, punctuality can play a role, since this is the main performance indicator for European railway operators, see Dollvoet and Huisman (2014). When a train is delayed, this punctuality decreases, which is actually measured by the delay on a train at the moment of entering a station. Passenger punctuality is a new performance indicator recently introduced by Netherlands Railways. This indicator measures the percentage of passengers who arrive at their destination with a delay below a certain threshold value. By using the passenger punctuality as a performance indicator, it can be sometimes beneficial to artificially delay a train to maintain its connection with another train. When only using the punctuality performance indicator, there is no purpose in applying delay management, since this indicator only measures the total delay on trains, which would never decrease in delay management.

The remainder of this paper is organized in the following way. In section 2, I explain the structure of the available data and in section 3 I make some (simplifying) assumptions which are needed to solve the problem in a mathematical manner. In section 4, I introduce the delay management model in a more formal way and I explain several concepts which are used throughout the paper. Section 5 actually explains the heuristics that I use and in section 6 the results are presented, including a comparison in section 6.4. In section 7, conclusions are drawn based on the results of the heuristics and directions for further research are listed.

2 Data

The area of focus for my research project consists of parts from the provinces Utrecht, Gelderland and Noord-Brabant. This includes the following six big stations: Utrecht Centraal, 's-Hertogenbosch, Tilburg, Eindhoven, Nijmegen and Arnhem. Besides these stations, there are a lot of smaller stations in between, of which the majority is only considered in certain parts of the numerical experiments. The timetable and data on which I am going to apply the delay management model and heuristics originates from the year 2013. Figure 1 shows the relevant part of the railway network and associated trains in that year.

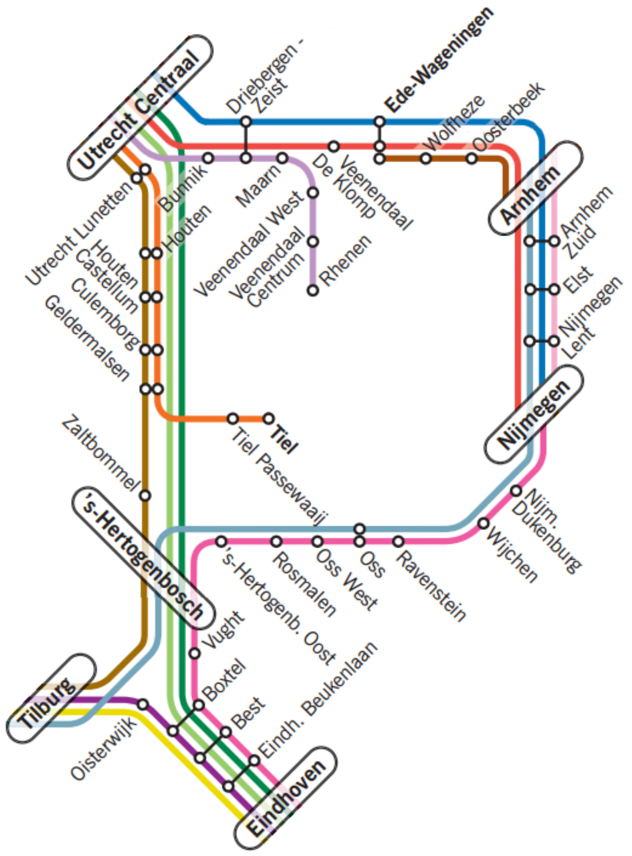


Figure 1: Part of the Dutch Railway Network in Utrecht, Gelderland and Noord-Brabant

Since I am doing this research project for Netherlands Railways, (transfers to) trains from other carriers like Arriva or Veolia are not taken into account and those trains and stations are therefore not displayed in this figure. It is also important to note that not all of the displayed trains end at Utrecht Centraal, Tilburg, Eindhoven, Nijmegen or Arnhem. In fact, trains cover bigger parts of the Netherlands, for instance the green and light green lines are trains which continue in the direction of Amsterdam past Utrecht Centraal and in the direction of Weert past Eindhoven. However, since I will come up with decisions in delay management which are directly related to transferring passengers, possible transfers on stations outside this area are not important and thus not taken into account.

A planning day for trains runs from 4.00 in the morning until 4.00 in the morning of the next day. The timetable for passenger trains is based on an hourly frequency, which can be defined formally as the cycle time (T). This means that a train from (for example) Utrecht Centraal to 's-Hertogenbosch departs 8 minutes past the hour for every hour in a certain time window. The precise time window differs since the timetable for some passenger trains is less dense in the late evening and nights. Of course, in the Netherlands there are more trains from Utrecht Centraal to 's-Hertogenbosch in an hour, e.g. 23, 38 and 53 minutes past the hour, but they are considered separately since the timetable cycle time is 60 minutes. The reason for this lies in the fact that the regional train from Ede-Wageningen via Wolfheze and Oosterbeek to Arnhem (and vice versa) has an hourly frequency.

The period on the day I will consider starts at 10 AM and ends at 3 PM. The reason for this is that there are some additional trains in rush hours and the timetable for trains changes slightly after 9 o'clock in the evening. Some trains are not scheduled anymore after 9 PM and some long distance trains will stop on smaller stations to preserve a decent number of train stops on these stations. Besides, a planning horizon of 5 hours is OK to guarantee that the models run in a reasonable amount of time and that there are still enough delay management decisions to make.

The available data consists of two files, of which one file contains traveling information for all passengers on a given day, i.e. the departure time, origin and destination for each passenger. The last column is dedicated to keep track of the number of passengers with the same traveling information. The data in this file can be seen as the Origin-Destination pairs (OD-pairs). An OD-pair gives the origin and destination of a passenger, without fixing the train(s) the passenger has to travel by. These OD-pairs are important to later determine which connections have to be maintained between a feeder train and connecting train(s) since it can be calculated which connections, if any, passengers have planned to use. It is straightforward to assume that a passenger will always plan to take the route with the lowest travel time from its departure station to its destination station. All passenger data is acquired through the use of the 'OV-chipkaart' (electronic traveling tickets in the Netherlands used for check in and check out), but since passenger data is confidential, some scaling has been applied to the data I make use of.

The other file consists of train series and its characteristics, where for the train numbers there is made use of odd numbers for one direction and even numbers for the opposite direction. For both regional and long distance trains, there are multiple entries in the file, namely the waiting activities on some stations and the driving activities to all stations where the train has to stop. The characteristics include the station of arrival (departure), the time of arrival (departure), the time of departure (arrival) after the waiting (driving) activity of the train, the station of departure (arrival) and the platform from which passengers can get into the train or out of the train. It is possible that there are stations where no waiting activity takes place. This is the case when the arrival time equals the departure time on a station. Of course, passengers have to be able to leave and enter the train, so there is some time available for that, but this is not visible in the published timetable. Besides that, there is a number for each entry, measured on one decimal point, that states the slack time on a waiting or driving activity. This slack time will later be used to make up delay when a train is (artificially) delayed.

3 Assumptions

In order to apply the heuristics and acquire results, several assumptions need to be made regarding the data. Since the passenger data originates from one specific working day, an assumption that needs to be made is that the data would be more or less the same when another working day would have been measured. It does not really matter which working day was measured, but since the timetable is different for some of the trains on Saturdays and Sundays, the passenger data does not match the timetable in a weekend. Therefore, my focus will lay on the timetable of a working day between 10 AM and 3 PM, as explained in the previous section.

A feeder train is connected to another train when the difference between the departure time of that train and the arrival time of the feeder train is greater than or equal to 2 minutes. On big stations, like Utrecht Centraal in the Netherlands, it is possible that there is more transfer time needed for passengers to get from one train to another train, but for simplicity I assume 2 minutes on each station to be fine to make a transfer. This assumption could be relaxed since this better resembles reality and there is additional data available with the minimal transfer time that has to be taken into account for each specific station.

When a certain train is delayed, the delay could actually be in favor of passengers that would not have been on time to travel by that train but are on time now. Of course, they will get into that train and will not wait for the train they had planned to travel by. This assumption is crucial for setting up the routes of all passengers, since in this way the simulated source delays, of which the details will be explained later, can be used for all passengers and not only for those that are transferring from a delayed feeder train to a connecting train. This could also influence the route they had initially planned to use, since in this world with real-time information, passengers will travel via the route that is fastest to their destination station at the moment they arrive at the station. If this has changed due to delays (which could sometimes be in favor as already mentioned), they will choose for this faster route.

4 Problem Formulation

Now I will introduce the commonly used delay management model formally by defining an event-activity network $\mathcal{N} = (\mathcal{E}, \mathcal{A})$, where $\mathcal{N}$ is a directed graph. In this graph, the set $\mathcal{E}$ consists of a set of arrival events and departure events of trains, thus $\mathcal{E} = \mathcal{E}_{arr} \cup \mathcal{E}_{dep}$. In the same way, the set $\mathcal{A}$ consists of a set of waiting activities, driving activities and connections, of which the latter one is represented by activities that can be removed from the network: $\mathcal{A} = \mathcal{A}_{wait} \cup \mathcal{A}_{drive} \cup \mathcal{A}_{change}$.

Waiting activities are used when the arrival time of a train is unequal to its departure time, which will usually be the case on bigger stations and/or when the train driver has to walk to the other side of the train to continue in the opposite direction, and driving activities are used for movements between stations. Connections (or changing activities) represent the transfers that are used by passengers who change from one train to another train. To this end, the arrival event of a feeder train and the departure event of another train have to be 'connected' at a station in such a way that there is enough time to make a transfer (2 minutes in my case). Connections that are maintained pose restrictions on the departure time of connecting trains.

As input for this model, the original timetable consisting of a set of arrival and departure times (π_e), a set of source delays (d_e), minimal required time per activity (L_a) and passenger data (w_p) is used. Then a so called disposition timetable is created; a timetable with (slightly) altered departure and arrival times since the delay (source delay and propagated delay) and the composition of optimal departure and arrival times is done. Thus the disposition timetable consists of a set of new arrival and departure times (x_e), based on the initial timetable, source delays and the minimal required time on activities. Additionally, the binary variables z_a indicate which connections are maintained.

Most of the concepts explained here are used throughout the paper. However, the dispatching rules do not make use of the event-activity network and therefore can not be modeled in software like AIMMS. Creating the disposition timetable for these dispatching rules is done in MATLAB.

5 Heuristics

Since the delay management model first introduced by Dollevoet *et al.* (2012) is not applicable for delay management in a small amount of time, I will make use of several fast heuristics proposed by Dollevoet and Huisman (2014). In each heuristic, the first step is to create a disposition timetable, in contrast to the (more advanced) delay management model with passenger rerouting, where the delay management decisions as well as the optimal routing for passengers is solved simultaneously. Secondly, I make use of a script I have written to calculate the passenger routing and travel times based on the disposition timetable and arriving times of passengers on their departure station. This script uses the arrival and departure times of trains from the disposition timetable and calculates, based on just the driving times between stations, the k shortest paths for each OD-pair, see JY Yen (1971). Thereafter, the actual shortest path is calculated by taking into account connections and associated waiting times when transfers at stations are required.

To understand why all of these steps are required, consider the following example. Take the path from Oss to Utrecht Centraal, which could be shortest via 's-Hertogenbosch when looking at the sum of travel times from Oss to 's-Hertogenbosch and from 's-Hertogenbosch to Utrecht Centraal. However, when actually taking into account the fact that passengers have relatively much spare time on their transfer at 's-Hertogenbosch, it can work out that traveling from Oss via Nijmegen and Arnhem to Utrecht Centraal is faster in the end. For this reason, it is important to make a difference between the travel time from station to station and the spare waiting times at stations when needing a transfer.

As a benchmark, a so called no-wait policy is considered, which does not take into account connections at all. Of course, delays are propagated in waiting and driving activities and delay is decreased by using the slack time. However, if a train arrives a few minutes late and therefore the transfer time for passengers to another train is only 1 minute, the connecting train does not wait and the passengers miss their connection.

5.1 Waiting Time Rule

The first heuristic I implement is based on a simple dispatching rule; the Waiting Time Rule (WTR). Under a WTR policy, a maximal waiting time is determined for each connection. This maximal waiting time has to be chosen in such a way that the total travel time of all passengers is minimized. This is influenced by transferring passengers that want to reach their next train in time and passengers already on board of the train, which suffer from delay when this train is artificially delayed in order to maintain a connection for other passengers. It is possible (and could be beneficial) to determine a maximal waiting time for each connection separately, but for simplicity I assume a constant time d_{max} for all connections. In mathematical notation it follows that a connection will be maintained if $d_a \leq d_{max}$, where $d_a = \pi_e + d_e + L_a - \pi_{e'}$, thus d_a resembles the time that a connecting train has to be delayed to maintain the connection with the feeder train. Note that π_e is the time when departure event e is planned and $\pi_{e'}$ is the time when arrival event e' is planned.

For determining the optimal maximal waiting time d_{max}, I will look into values ranging from 0 to 5 minutes, where the setting of 0 minutes corresponds to the no-wait policy. Waiting for more than 5 minutes is very unlikely to be optimal, since all of the passengers in the connecting train will be delayed for that amount of time, which is harmful in the first place, but secondly the risk of missing connections for other passengers on following stations arises.

5.2 Ratio of Transferring Passengers

To take the number of passengers into account that actually want to make use of a connection, I will look into the RTP policy, which defines a threshold (ρ_{min}) with respect to the number of passengers that want to make use of a connection and the number of passengers in the train that would be delayed when the train waits for a few minutes after departure time (and thus the connection is maintained). The actual ratio for each chancing activity a can be calculated in the following way:

$$\rho_a = \frac{\text{Number of passengers that planned to use connection } (e', e)}{\text{Number of passengers that planned to use driving activity } (e, f)} \tag{1}$$

In this equation, $(e', e) \in \mathcal{A}_{change}$ denotes the connection between an arrival event e' from the feeder train and a departure event e from the connecting train and $(e, f) \in \mathcal{A}_{drive}$ denotes a driving activity, where e again resembles the departure event from the connecting train and f resembles the arrival event of this train at a subsequent station.

Under this policy, a connection will be maintained when the number of transferring passengers is high compared to the number of passengers in the connecting train; they who suffer when the train waits for the passengers from the feeder train. Thus in mathematical notation, a connection will be maintained if $\rho_a \geq \rho_{min}$. For determining the optimal threshold value (ρ_{min}), I will experiment with values ranging from 0% to 100% and a value bigger than 100% (110 % will do), since the latter corresponds to the 'no-wait' policy.

Note: Since the RTP is based on the number of passengers that plan to use a certain train or connection, in reality it could sometimes be the case that the number of passengers does not match the actual number of passengers in a train or using a connection. Even though I only execute the heuristics on the timetable between 10 AM and 3 PM, it is important for the RTP policy to consider passengers on other moments of the day as well. Since ρ_a is based on the number of passengers that use a certain connection and driving activity, passengers that start their travel before 10 AM or still travel after 3 PM, have to be counted in the trains where they travel by and the connections they make use of.

5.3 Classical Model

The last heuristic I implement is the classical delay model from Schöbel (2007). To start with, the assumption can be made that passengers who miss a connection have to wait for one cycle time of the timetable (60 minutes for this timetable). After waiting for this long, the next train arrives and passengers will travel by that train. Another approach is the consideration of an estimate of additional delay. In this case, passengers are allowed to travel by another train, generally 15 or 30 minutes after the initial connection. The only difference between the two models is the value of D, which can be chosen somewhere between 0 and 60. The optimal value for D will depend on the timetable characteristics, where the density of the railway network plays a big role.

Making use of the number of passengers that plan to end their journey when arriving at a certain station and the number of passengers that use a certain connection, the model formally shown on the next page can be modeled in AIMMS and then solved by a CPLEX-solver. As a result of solving this model, the disposition timetable is calculated in terms of the variable x_e. Additionally, z_a shows which connections are maintained for passengers that want to transfer from one train to another train. Making use of the general script in MATLAB, as explained in the beginning of this section, I obtain the passenger routing and travel times for each OD-pair. Subsequently, the total travel time of all passengers, which serves as the objective value, can be calculated from the number of passengers in each OD-pair.

Sets
$\mathcal{E}_{arr}$ = Set of arrival events
$\mathcal{E}_{dep}$ = Set of departure events
$\mathcal{E} = \mathcal{E}_{arr} \cup \mathcal{E}_{dep}$ = Set of events
$\mathcal{A}_{wait}$ = Set of waiting activities
$\mathcal{A}_{drive}$ = Set of driving activities
$\mathcal{A}_{change}$ = Set of connections
$\mathcal{A} = \mathcal{A}_{wait} \cup \mathcal{A}_{drive} \cup \mathcal{A}_{change}$ = Set of activities

Parameters
π_e = Time when event e is planned
L_a = Minimal required time for activity a
d_e = Delay on event e
$w_e = \sum_{p \in \mathcal{P}(e)} w_p$ = Number of passengers that plan to end their journey at event $e \in \mathcal{E}_{arr}$
$w_a = \sum_{p \in \mathcal{P}(a)} w_p$ = Number of passengers that use a changing activity $a \in \mathcal{A}_{change}$

Constants
$M = \max_{e \in \mathcal{E}}(d_e)$
D = Estimate of additional delay when missing a connection

Decision variables
x_e = New time when event e takes place

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

Mathematical formulation

$$\text{minimize} \sum_{e \in \mathcal{E}_{arr}} w_e x_e + \sum_{a \in \mathcal{A}_{change}} w_a D(1 - z_a) \tag{2}$$

subject to

$$x_e \geq \pi_e + d_e, \ \forall e \in \mathcal{E}, \tag{3}$$

$$x_e \geq x_{e'} + L_a, \quad \forall a = (e', e) \in \mathcal{A}_{wait} \cup \mathcal{A}_{drive}, \tag{4}$$

$$M(1 - z_a) + x_e \geq x_{e'} + L_a, \quad \forall a = (e', e) \in \mathcal{A}_{change}, \tag{5}$$

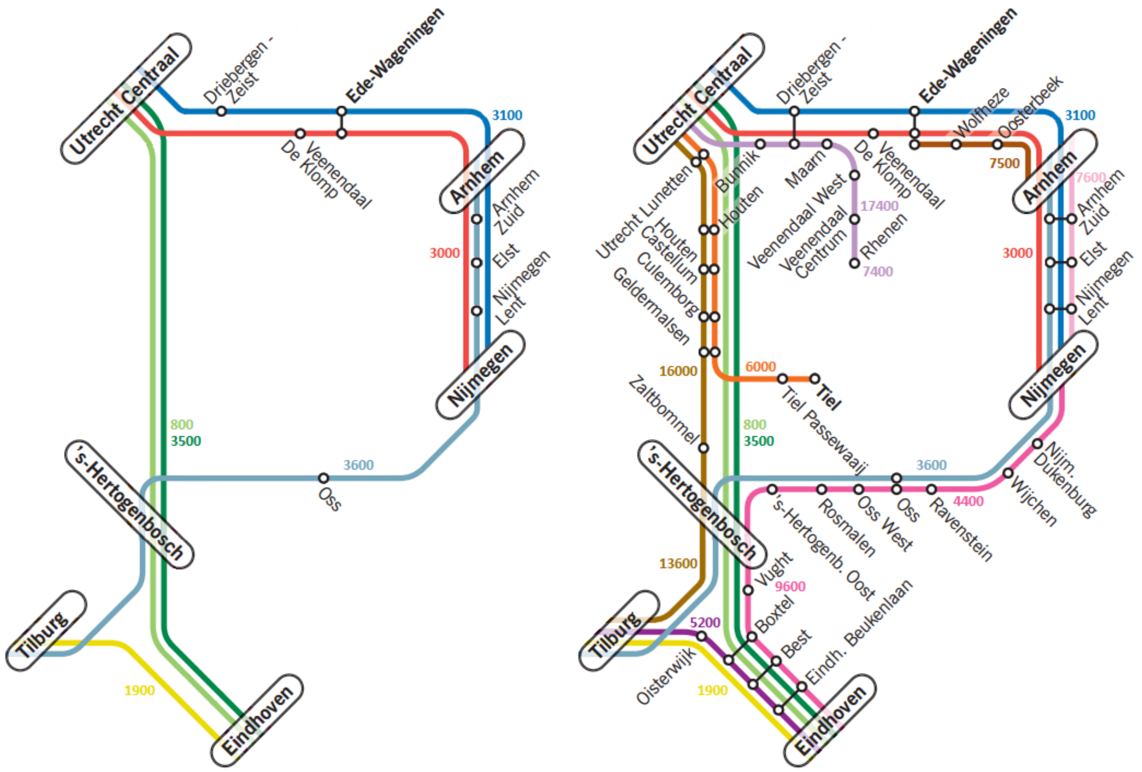
$$x_e \in N, \quad \forall e \in \mathcal{E}, \tag{6}$$

$$z_a \in \{0, 1\}, \quad \forall a \in \mathcal{A}_{change} \tag{7}$$

Explanation of the mathematical formulation
The objective function (2) minimizes the arrival time at their destination for all passengers, while taking into account the suffering from connections which are not maintained. In this model, if a connection is not maintained, the additional travel time is equal to the estimate of additional delay (D). Restriction set (3) makes sure that an event takes place after the planning time of that event plus a possible source delay. Finally, restriction sets (4) and (5) are needed for the propagation of delay in the timetable along waiting/driving activities and maintained changing activities respectively, in such a way that only the minimal required time for each activity is used until there is no delay left. In this way, slack time on activities will be used to make up possible delay. M is a sufficiently large number, which is equal to the maximum delay in the set of source delays, as proposed by Dollevoet *et al.* (2012).

6 Results

I have applied the heuristics to two different cases, of which the first case consists of the long distance trains only (and associated railway stations) and of which the second case includes all trains and stations. The trains and stations used for the two cases are shown in Figure 2a and Figure 2b respectively. In the same color as the line of a train, the number of a train series is depicted.



(a) Case I (b) Case II

Figure 2: Railway network in each of the two considered cases

In Table 1 some characteristics of the cases are listed, which include the number of stations, the number of trains, the number of OD-pairs and the number of passengers in each of the cases. Additionally, in the columns with the number of passengers and the number of OD-pairs, I have indicated in brackets the percentage of passengers that make use of a transfer during their trip. The percentage of passengers that transfer, is about $\frac{1}{3}$th of the OD-pairs with a transfer. This shows that, although there are relatively many OD-pairs with a transfer, not many passengers are involved. The number of passengers in the OD-pairs without a transfer is therefore much higher compared to the number of passengers in the OD-pairs with a transfer.

Case	Stations	Trains	OD-pairs	Passengers
I	13	145	1104 (19%)	19164 (6%)
II	39	349	2669 (30%)	27898 (10%)

Table 1: Some characteristics of the instances

For evaluating the heuristics, I have generated a set of delays for both of the cases I consider in the following way. Each arrival event has a probability of 10% to be delayed. This delay only bears the incurred delay from that specific moment, i.e. delay incurred at one or more stations before is not directly present here but will be taken into account by the heuristic when calculating the disposition timetable. The waiting activities, which are present at some of the stations, and the slack time will be used to make up delay. If an event is delayed, its delay is uniformly distributed between 1 and 15 minutes. The arise of a delay of more than 15 minutes at once could happen, but since it is not really common, 15 minutes is the highest delay simulated per arrival event. In this way, a train can still be delayed for more than 15 minutes when there arises delay at multiple arrival events on a route.

The results for each heuristic are averaged over 5 different instances of delays to prevent randomness having too much influence on the results. Of course, a sample of 5 instances is not that large, but since some scripts/functions can be running for over an hour, it is impossible to achieve a much better sample size in such a limited time window. The results per policy for both case are displayed and discussed in the following sections.

6.1 Waiting Time Rule

In a policy based on the Waiting Time Rule, a connection is maintained when a connecting train has to wait for at most d_{max} minutes on a delayed feeder train. In Table 2 and Figure 3, the results are displayed for the different values of d_{max}. Remember that $d_{max} = 0$ corresponds to a no-wait policy.

d_{max}	Case I	Case II
0	598650	770224
1	598506	769598
2	<u>598293</u>	<u>768963</u>
3	598403	769883
4	598519	772176
5	599131	775187

Table 2: Results of the WTR policy (best result underlined for each case)

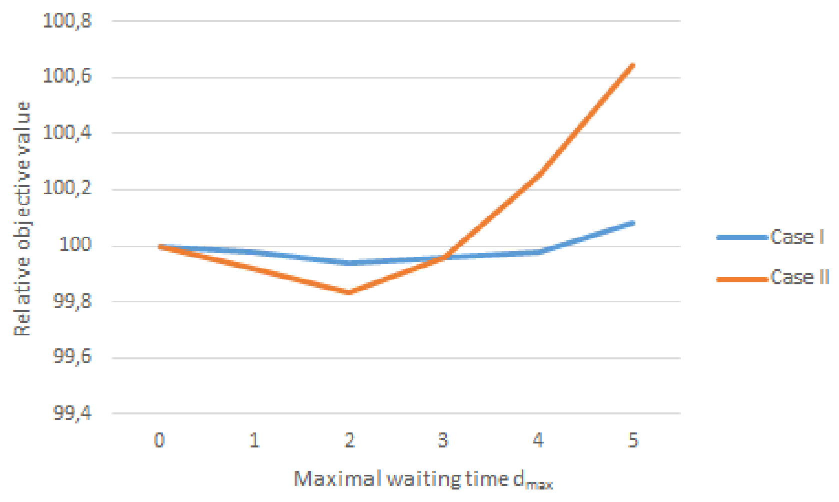


Figure 3: Results of the WTR policy

For both cases $d_{max} = 2$ results in the lowest objective value. As expected, for both cases a high value for d_{max} results in a solution worse than a no-wait policy, which is true for $d_{max} = 5$. For Case II, $d_{max} = 4$ also results in an objective value worse than those of the no-wait policy. The performance of $d_{max} \in \{1, 2, 3\}$ is better than a no-wait policy for both cases and for Case I, even $d_{max} = 4$ gives a better solution.

For Case II, the differences between applying a WTR policy with $d_{max} > 0$ and applying the current no-wait policy are most clear. For Case I, a WTR policy still gives a better solution in terms of the objective value, thus the total travel time of all passengers, but it is less beneficial than for Case II. This difference can be caused by the fact that Case II contains a lot more trains than Case I, which results in more artificial delays (10% of the arrival events incurs a delay) and therefore creates more opportunity for the WTR to work out well (compared to a no-wait policy).

On average, delay will be reduced by 0.1%, which does not sound as a big improvement in performance. A possible explanation for this low improvement is that we have to deal with circumstances where passengers have that much spare time to make a transfer, that the WTR policy does not grant very much improvement compared to the no-wait policy. To give an example: If most of the transfers have to be finished in 2 to 3 minutes, a WTR policy will have a heavy impact on the total travel time of all passengers when there is some delay present. However, if most of the transfers have to be finished in 5 to 10 minutes, there is enough spare time left (besides the 2 minutes required for a transfer) to maintain a connection anyhow when a feeder train arrives with delay. A WTR policy could still sometimes carry improvements in this case, but the benefits will be significantly lower than in the first scenario.

To validate this hypothesis, I have calculated the mean of all spare minutes on transfers that are used, corrected for the 2 minutes which are needed to make the transfer. The mean of these spare minutes works out to be just above 13 minutes for Case I and a little under 8 minutes for Case II. These are actually quite high numbers in a timetable where all trains (except one regional train) have a frequency of 2 or 4 times per hour. For this reason the spare time at transfers is more than enough to still have a connection maintained in case of a delayed feeder train while a no-wait policy is used. On the other hand, I can argue that the current timetable is not really optimal at some stations, since earlier transfers are just missed quite often in this timetable without delays. It would be better for certain passengers to have just a few minutes more at some of the stations ('s-Hertogenbosch for example) to give them the opportunity to make use of one connection earlier. However, this is not something to consider in delay management, but it could be something for Netherlands Railways to look into for next years timetables.

6.2 Ratio of Transferring Passengers

In a policy based on the Ratio of Transferring Passengers, a connection is maintained when the number of passengers that want to transfer from a feeder train to a connecting train is high enough compared to the number of passengers that travel by the connecting train, at least to the next station. In Table 3 and Figure 4, the results are displayed for the different values of ρ_{min}. The objective values for $\rho_{min} \in \{80, 90, 100\}$ are omitted from the table since from $\rho_{min} = 70$, it is a straight line. Remember that $\rho_{min} = 0$ corresponds to a policy where every connection is maintained, no matter how many passengers want to make use of that connection, and $\rho_{min} = 110$ corresponds to a policy where a connection is never maintained in case of a delayed feeder train (a no-wait policy).

p_{min}	Case I	Case II
0	599858	780356
10	598157	770793
20	<u>598100</u>	769930
30	598595	770042
40	598650	<u>769789</u>
50	598650	770100
60	598650	770190
70	598650	770224
110	598650	770224

Table 3: Results of the RTP policy (best result underlined for each case)

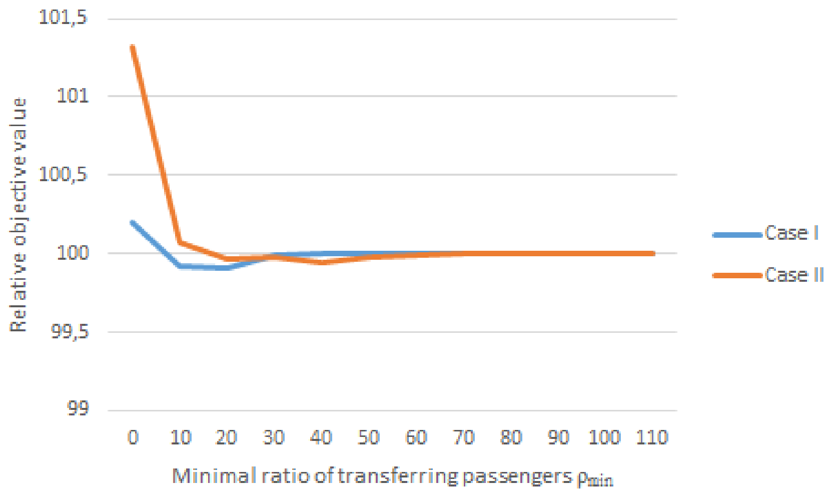


Figure 4: Results of the RTP policy

For Case I, $\rho_{min} = 20$ and for Case II, $\rho_{min} = 40$ results in the best objective value. For both cases, the RTP for $\rho_{min} = 0$ is even performing worse than a no-wait policy and for Case II for $\rho_{min} = 10$ as well. It holds that the performance for $\rho_{min} \in \{10, 20, 30\}$ is better than a no-wait policy for Case I, and the same holds for the values $\rho_{min} \in \{20, 30, 40, 50, 60\}$ for Case II. From $\rho_{min} = 70$, the objective value is equal to the objective value in the no-wait policy for Case II. The same applies to Case I, but there the objective value is yet equal to the no-wait policy from $\rho_{min} = 40$.

For Case II, the amplitude of the objective value is higher than for Case I. The performance of $\rho_{min} = 0$ is not that good for both cases, but it is worst for Case II in comparison with the other values of ρ_{min}. This difference is most likely caused by the fact that there are a lot more trains in Case II than in Case I. Since 10% of the arrival events is artificially delayed, more delay will be present in Case II. With this RTP policy where $\rho_{min} = 0$, every train will wait, independently of the number of passengers that want to make use of the connection. In this way, it could even happen that nobody wants to make a specific transfer, but that connection is still maintained caused by the low value of ρ_{min}.

On average, the performance of the RTP is about 0.1% better than the no-wait policy, which again does not sound as a big improvement. More or less the same explanation as for the low improvement under the WTR policy holds here. Although the decision whether or not to wait in the RTP policy is not based on a maximal waiting time, the characteristics of the timetable are the same. So if the spare time at transfers happens to be that high that delay will not very often break a connection, there is also not much benefit to gain in applying the RTP policy. We have already seen that the mean of all spare minutes on transfers that are used, corrected for the 2 minutes which are needed to make the transfer, is high, so it is likely that the low improvement is caused by the characteristics of the timetable, especially the spare time on transfers.

By comparing the best objective value among the two cases, it is clear that for Case II, where all trains and stations are considered, the optimal ratio ρ_{min} should be higher than for Case I. A possible explanation for this observation is the fact that passengers getting on board of the train at stations $n+1, n+2, n+3$, and so on, are not considered for the decision that has to be made for the connection at station n, see Dollevoet and Huisman (2014). However, these passengers will suffer from delay if the train is delayed at station n and has not made up the delay in the meanwhile, which is plausible since regional trains stop at all stations and there is usually not that many distance between these stations. This implies that the RTP policy underestimates delay from maintaining a connection and choosing a higher value for ρ_{min} will compensate this underestimation and therefore in the end works out to be a better threshold.

6.3 Classical Model

In the classical delay management model, an integer programming formulation on an event-activity network is used to determine the disposition timetable. Thereafter, the routing of all passengers can be determined and the total travel time is calculated. In Table 4 and Figure 5, the results are displayed for different values of D, where $D = 60$ corresponds to waiting for the cycle time of the timetable.

D	Case I	Case II
0	592808	775346
5	592808	772467
10	592770	772004
15	$\underline{592758}$	771872
20	592763	$\underline{771569}$
25	592795	771693
30	592879	771623
35	592917	771749
40	592968	771827
45	593085	771996
50	593098	772164
55	593128	772624
60	593194	772919

Table 4: Results for the classical delay management model (best result underlined for each case)

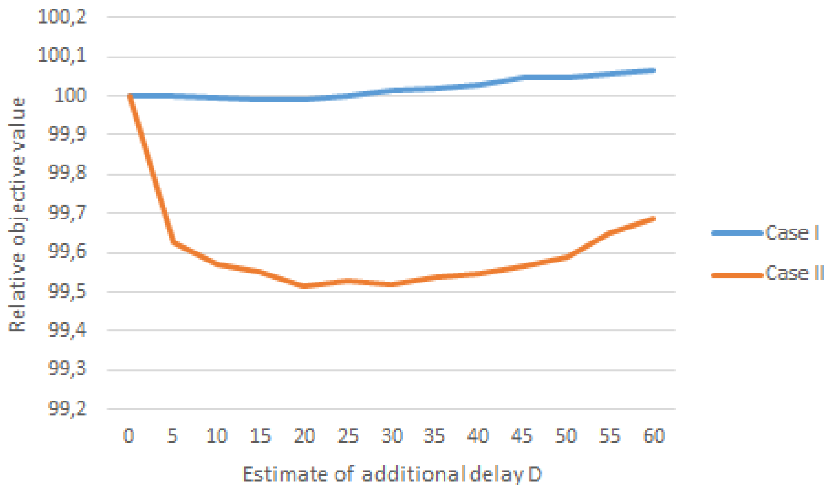


Figure 5: Results for the classical delay management model

For Case I, a pattern in the figure is not really obvious to see. This is a result of the low improvement in objective value between $D = 15$, which is the best value for the additional delay, and other values of D. This implies that the additional delay for passengers who miss a connection is about 15 minutes for this case, which is plausible since most of the considered stations are served 4 times per hour.

For Case II, a clear pattern is visible in the figure and the objective value is lowest for $D = 20$. This implies that the additional delay for passengers who miss a connection is about 20 minutes. Remember that in this case both long distance trains and regional trains (and all stations) are considered. Therefore, 20 minutes of additional delay when missing a connection seems reasonable since in this timetable, most of the long distance trains serve an associated station 4 times per hour and regional trains serve all stations only 2 times per hour (and only once per hour between Ede-Wageningen and Arnhem).

Comparing the best value of D to the worst value of D for each case separately, the objective value of the classical model decreases about 0.01% for Case I and about 0.5% for Case II. The improvement of the objective value is clearly more significant for Case II. For the relatively low overall improvement of the objective value, the same explanation applies as for the previous heuristics. The used sets of delays are the same for each heuristic and the initial timetable is identical of course, so the low improvement is due to the amount of spare time on a station, when making use of a transfer. This spare time is particularly high compared to the density of the timetable. Especially for Case I, a spare time of 13 minutes while most trains have a frequency of 4 times per hour is quite high and therefore results in a low improvement when varying the value of D. There are simply not many delay management decisions to be made in case of a delayed feeder train. For Case II, the same applies but now the trains (except one regional train) have a frequency of 2 or 4 times per hour and the spare time on changing events is 8 minutes on average. Thus in this case, it does make sense to look at connections when a feeder train is delayed, since it is not straightforward that the connection is maintained when letting the connection train depart on time.

6.4 Comparison

After solving the delay management model with three heuristics, it is time to compare the results. In Table 5, the optimal absolute and relative objective values for each heuristic can be found for both of the two cases, where the no-wait policy is the benchmark to which the objective values are normalized in the second part.

As can be seen in the table, most of the heuristics result in a better objective value, thus a lower total travel time for passengers, than the no-wait policy. For Case I, the performance of the classical delay management model is best, followed by the Ratio of Transferring Passengers and the Waiting Time Rule. For Case II, the Waiting Time Rule leads to the best solution, followed by the Ratio of Transferring Passengers, but the difference in objective values is quite small. On top of that, for Case I the RTP performs better than the WTR and for Case II vice versa. Therefore it is hard to say which dispatching rule performs better.

	Policy	Case I	Case II
Absolute	No-wait	598650	770224
	WTR	598293	768963
	RTP	598100	769789
	Classical	592758	771569
Relative	No-wait	100.00	100.00
	WTR	99.94	99.84
	RTP	99.91	99.94
	Classical	99.02	100.17

Table 5: Comparison of the absolute and relative objective values for each heuristic

While the performance for the heuristic based on the classical delay management model is clearly best for Case I, the performance of this heuristic is even worse than the no-wait policy for Case II. This is quite strange since this implies that applying a delay management heuristic results in a worse policy than doing nothing; i.e. never delay a connecting train when a feeder train is delayed. It is most likely that this strange outcome is the result of something else, namely that the no-wait policy and the other two heuristics provide too optimistic results.

Initially, the WTR and RTP heuristic seemed to work fine; same result for the no-wait policy and improvements in the objective value as expected for different values of the threshold. However, when constructing the classical delay management model in AIMMS, I have encountered some, at the first sight, strange behavior at waiting and driving activities with negative slack time. The classical delay management heuristic implicitly models slack time, since only the minimal time required for an activity is required as input, while the slack time is taken into account when the optimal disposition timetable is calculated. Testing an instance without any delay resulted in a disposition timetable different from the initial timetable. This may sound strange since no delay implies that there are no delay management decisions to be made, thus every train will depart and arrive as planned. However, the activities with negative slack time will be delayed in such a way that there will be more time assigned to these activities than in the initial timetable since otherwise, the model will be infeasible. To provide some extra insight; if a train departs from Nijmegen at 5.35 and arrives at Nijmegen Lent at 5.39 according to the timetable, the listed driving time is 4 minutes. However, the slack time on this activity is -1 minute, which implies that the actual required driving time between Nijmegen and Nijmegen Lent is 5 minutes instead of 4 minutes. The model in AIMMS will correctly solve this (and thus make it feasible) by assigning 5 minutes driving time to this activity, but the created disposition timetable with no delays will therefore not precisely resemble the initial timetable.

What in the end seemed to go wrong is the consideration of negative slack time in the no-wait policy and the previous heuristics. Negative slack time is treated right when constructing the disposition timetable and adding artificial delay for maintaining connections, but the actual outcome will most likely works out to be infeasible. Negative slack times do not only have to be taken into account when adding source delays and artificial delays for delay management purposes, but also have to be considered for constructing a feasible solution. Referring to the example from earlier, in a feasible solution, 5 minutes driving time is used from Nijmegen to Nijmegen Lent. Currently, in the first two heuristics and the no-wait policy, only the 4 minutes driving time as in the timetable is used. In some situations, this will compensate with positive slack time earlier or later on, especially when passengers travel a long distance and therefore encounter most likely more positive slack time than negative slack time. That is often the case on longer journeys since the slack time is relatively not that often negative, as can be seen per case in Table 6.

	Case I	Case II
Positive slack time	74.4%	68.8%
Negative slack time	5.0%	15.8%
No slack time	20.6%	15.4%

Table 6: Type of slack time on activities per case

For Case II, the slack time is negative about three times more often than for Case I. This results in a bigger deviation for the no-wait policy, the WTR heuristic and the RTP heuristic compared to what the results in a feasible solution would have been. Therefore the performance of these policies would have been somewhat worse for Case I and considerably worse for Case II than listed in Table 5. For this reason, the classical delay management model will most likely be the best choice for both of the cases when the negative slack time would have been handled correctly in all heuristics and the no-wait policy.

In order to apply off-line delay management heuristics to an on-line delay management problem, a short computation time is required. Constructing the disposition timetable was managed in a few seconds for the dispatching rules (WTR and RTP) and even in under a second for the classical delay management model. However, determining the actual routing for every passenger by using the general script, took a lot longer. MATLAB is not particularity fast when using if-loops and for-loops (which is sometimes unavoidable) compared to using matrix calculations. Dependent on the number of trains and OD-pairs and the type of heuristic, the scripts ran between 2.5 minutes and 99.5 minutes for constructing the disposition timetable and determining the optimal routing, see Table 7 for the running times per heuristic and per case.

		Case I	Case II
WTR	Disposition Timetable	0.60	3.43
	Optimal Routing	89.39	2217.00
	Total Running Time	89.99	2220.43
RTP	Disposition Timetable	0.49	1.14
	Optimal Routing	171.34	4425.59
	Total Running Time	171.83	4426.73
Classical	Disposition Timetable	0.02	0.09
	Optimal Routing	173.85	5976.71
	Total Running Time	173.87	5976.80

Table 7: Running time of MATLAB/AIMMS in seconds per case for each heuristic

As shown in the table, the running time for making the disposition timetable is often negligible. However, for all heuristics hold that the optimal routing script for Case II runs about 25 to 35 times as long as for Case I. The number of considered stations, trains and OD-pairs is respectively 3, 2.4 and 2.4 times as high for Case II in comparison with Case I, see Table 1. This shows that the complexity of the problem increases drastically when the size of the case extends. This is not strange at all since the addition of regional trains directly influences the number of travel possibilities, even for traveling between two stations where long distance trains stop. Since if a regional train departs at a better point in time regarding the arrival time on the departure station of some passengers, it could be beneficial for them to take the regional train instead of waiting for the next long distance train. The increase in running time is therefore not linearly related to the increase in case size.

Comparing the running time of creating the disposition timetable among the heuristics, the classical delay management model is fastest, followed in turn by the RTP and the WTR. The disposition timetable for the classical delay management model is modeled in AIMMS, which is designed to tackle this kind of problems, so it is not strange that this running time is lowest. The results for the dispatching rules are acquired in MATLAB and the running time differs somewhat dependent on the dispatching rule.

For the running time of calculating the optimal routing, it is important to note that the WTR has to calculate for all passengers only 6 routings ($d_{max} \in \{0, 1, 2, 3, 4, 5\}$), the RTP has to calculate 12 routings ($\rho_{min} \in \{0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110\}$) and the classical delay management model has to calculate 13 routings ($D \in \{0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60\}$). This results in considerable longer running times for the optimal routing for the RTP and classical delay management model compared to the WTR.

By looking at the total running time, it is clear that I have to conclude that solving these heuristics in MATLAB is far too slow for applying them to the on-line delay management problem. In a programming language like Java or C++, the heuristics would have possibly ran faster. The major problem lies in the script for the passenger routing, where different possibilities are checked in order to choose the shortest route in terms of the travel time.

7 Conclusion

In this paper I have described and executed several heuristic methods to solve the off-line delay management problem. I have compared the results of these heuristics among each other and to the no-wait policy. For the case where only long distance trains and associated stations are taken into account, the classical delay management model leads to the best solution, with a decrease of 1% in the total travel time of the passengers compared to the no-wait policy. For the case where all trains and stations are considered, the classical delay management model does not provide the best solution. This is not as expected and will likely be caused by a too optimistic result from the other heuristics. While the classical delay management model is correctly solved, the script I have used for the other heuristics does not take into account the actual driving time, but the planned driving time. When a lot of negative slack time is present (without compensating positive slack time), the travel time of passengers is not correctly calculated and therefore an unfeasible solution is acquired. Most likely the classical delay management model would have also performed better for Case II when the results of the other heuristics would have been acquired correctly and thus resulted in a worse objective value.

I have also implemented two dispatching rules, of which the Waiting Time Rule is currently used by Netherlands Railways. These dispatching rules do not massively decrease the total travel time of passengers for the timetable I have used. This is caused by the fact that the average spare time on connections is large compared to the density of the timetable; having 13 minutes on average to make a transfer to a train which has a frequency of 4 times per hour will not very often lead to a possible broken connection and thus a situation where delay management can improve the total travel time of the passengers.

In terms of the running time, making the disposition timetable is done in a few seconds for all heuristics. However, calculating the optimal passenger routing takes a lot longer, dependent on the heuristic and the considered case. Especially when calculating the passenger routing for Case II, the number of possibilities is huge and therefore the running time was about 100 minutes for one of the heuristics. This is clearly far too slow for applying these heuristics to an on-line delay management problem, where one has to make decisions in a few seconds. For speed purposes, the programming languages Java or C++ would have been a better choice.

For trying to acquire a lower objective value, the iterative heuristic as described in Dollevoet and Huisman (2014) can be used. This heuristic is based on the classical delay management model, but uses a parameter D_p which differs among the OD-pairs $p \in \mathcal{P}$. With the obtained disposition timetable, new routes can be determined for all passengers and when finding an OD-pair that misses a connection, the actual delay can be used to update the estimate D_p. Repeating this process will find optimal values for D_p and hopefully give better results in terms of the total travel time of the passengers.

Another possible direction for further research is the consideration of a more realistic case, that is the transfer time on each station can be determined separately, dependent on the number of tracks on a station. Distinction can also be made between transfers from one platform to another platform or transfers which are cross-platform. To this end, there is data required which lists the transfer times for all stations and the tracks which are located next to the same platform.

A last relevant aspect to consider is the limited capacity of the railway infrastructure, i.e. the number of platforms and the possibility of a train to pass another train in or just before a station. In general, connections between trains will hold when these trains are not supposed to stop near the same platform. However, if there are multiple trains at a station arriving with some delay, the capacity of the railway infrastructure can influence the possible outcomes of delay management decisions.

8 References

1. Dollevoet, T., Huisman, D., Schmidt, M., & Schöbel, A. (2012). Delay management with rerouting of passengers. Transportation Science, 46(1), 74-89.

2. Dollevoet, T., & Huisman, D. (2014). Fast heuristics for delay management with passenger rerouting. Public Transport, 6(1-2), 67-84.

3. Schöbel, A. (2007). Integer programming approaches for solving the delay management problem (pp. 145-170). Springer Berlin Heidelberg.

4. Yen, J. Y. (1971). Finding the k shortest loopless paths in a network. management Science, 17(11), 712-716.