Heuristics for delay management with re-routing

Student:

E. van Wingerden

311228ew

Supervisor:

D. Huisman

Abstract

In delay management the main question is whether trains should wait for a delayed train or should depart on time. In the traditional delay management models passengers always take their originally planned route. In the latest literature they used re-routing of passengers and made significant improvements. However large instances could not be solved with their exact approach. Therefore in this thesis two heuristics are made that use re-routing and simple dispatching rules for the wait-depart decision. At the end experiments based on real-world data from the Netherlands Railway show that simple dispatching heuristics can be a good alternative in order to improve delay management when the optimal solution cannot be determined.

Contents

Chapter 1

Introduction

During peak hours passengers nowadays prefer to travel by train. Railway transport therefore plays an important role in the European mobility. The timetables that most European railway companies tend to use are cyclic in order to ensure a high frequency and an easy to remember timetable. In these timetables each line has to be operated in a cyclic or periodic pattern in which a line, for example, repeats every 30,60 or 120 minutes (see Kroon et al. (2009) for a recent publication on this subject). Often these timetables are constructed in such a way that the inconvenience of changing from train A to B is kept to a minimum. This means that train B departs shortly after train A arrives and preferably with a cross-platform change, i.e. both trains stop at two adjacent tracks of the same platform. Because of the short transfer time the passengers have a larger probability to miss their transfer in case of a delay and therefore the question should be asked whether train B should wait or not. Such decisions are called delay management. The main question in most of the literature so far is whether a train should wait for delayed trains and which trains should depart on time (wait-depart decisions). Since delays are often transferred if a train waits, connections are often not maintained in case of delays. Dollevoet et al. (2010) used re-routing in their model and got significant improves of 2-5% in comparison with the traditional delay management models without re-routing. This model however was not able to solve large data sets. In this thesis we will look at two heuristics that use simple dispatching rules to determine the wait-depart decisions and re-routing of passengers in case a transfer is missed. The data we used consists of parts of the Netherlands railway network (NS), the largest passenger operator on the Dutch Railways network. We will show that these heuristics can improve the solution compared to a no wait policy but can still be far from the optimal solution. In Chapter 2 we will describe the problem, describe re-routing, what the objective of this thesis is and what data we used in this thesis. Chapter 3 contains information about the relevant literature on this subject. In Chapter 4 we will describe the two different heuristics used and which wait-depart decisions we look at. Then we will give an overview of the cases we considered and discuss the results in Chapter 5. At the end we give a conclusion in Chapter 6.

Chapter 2

Problem description

2.1 General problem description

Passengers traveling by train sometimes have to, depending on their location and destination, transfer from train to train at certain stations in order to reach their planned destination. Most of the time these transfer times are scheduled in such a way that the transfer time is reduced to a minimum. In this way the total amount of time to travel from A to B is reduced to a minimum. For example the passengers from Zwolle that travel to Amsterdam have to transfer at Amersfoort. The intercity from Amersfoort to Amsterdam departs five minutes later than the intercity from Zwolle arrives in Amersfoort. In most of the times this reduces the travel time for the passengers however when the train from Zwolle has a delay of more than 5 minutes the passengers cannot make their transfer to the other train to Amsterdam and have to wait for one hour to get the next intercity from Amersfoort assuming they take the next intercity to Amsterdam from Amersfoort. Because NS wants to minimize the total delay of passengers it could be a good idea to let the on-time trains wait for the delayed trains. In this way the passengers can still get on the other train and therefore are less delayed. However this makes an unnecessary delay for other passengers and therefore there is a trade-off between delaying the on-time train and letting the passengers of the delayed train wait.

2.2 Re-routing

Assuming that passengers wait for the next intercity is most of the time not correct. When they have to wait for an hour until the next intercity departs it would probably be better to take another train. In the previous example the passengers can take the regional train at Amersfoort or transfer at Utrecht, depending on the delay time. It is thus not reasonable to assume that passengers wait for the next train. This shows the importance of re-routing and at the same time it shows how sensitive it is because a delay of a few minutes can change the optimal route for the passengers.

2.3 Objective

During the thesis the objective is to minimize the total delay of all passengers by making wait-depart decisions. A model to get the optimal solution already exists and is described in Dollevoet et al. (2010). The only problem is that this model does not work on large data sets. CPlex runs out of memory very early in the solution process and quits before a first feasible solution is found. In order to solve these larger cases a possible solution is to build heuristics that can handle large datasets and use simple dispatching rules to approach the optimal solution as good as possible. In this thesis the main focus will be to build these heuristics and try out different dispatching rules. For example to let a train wait if the delay is less than a certain amount of minutes or criteria in which the amount of passengers will also be taken into account to increase or decrease the maximum amount of minutes a train will wait for a delayed train. During the thesis the train always waits whenever a passengers otherwise would not be able to reach their destination.

2.4 Graph representation

The data we used in this thesis can be represented as a graph. Each case consists of a number of stations. For these stations a certain number of arrivals and departures with their corresponding time is given, which represent the nodes. A number of trains travel a route which goes from one node to another node. The arcs represent the time that is needed from one node to another node and determines the route for the trains. In Figure 1 a small part of the railway network is given, which represents case I. In this figure a regional train runs from Amersfoort to Amsterdam via Hilversum, all other trains are intercities.
For this part of the railway an example of the activities is given in Figure 2 to give a good view about the graph.

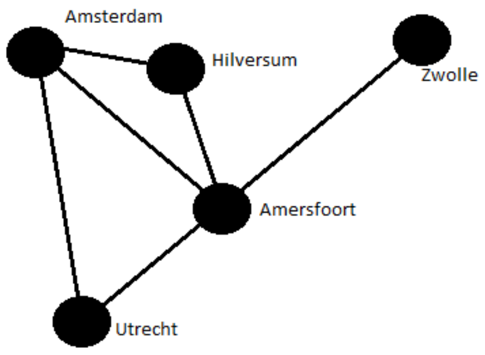

Figure 1: A small graph representation of a small part of the railway network in the Netherlands. In this figure a route from Zwolle to Utrecht is represented.

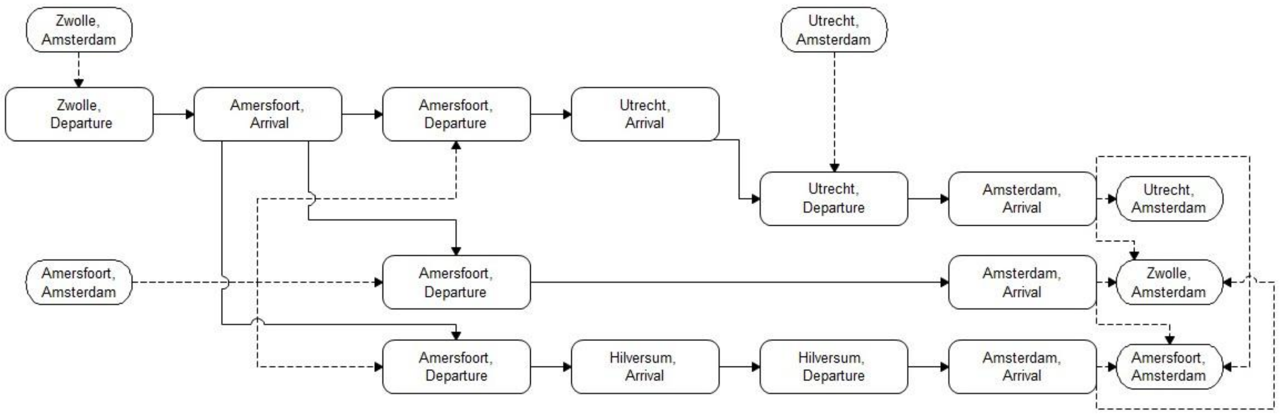

Figure 2: The square nodes are the departure and arrival events for the trains. Each line of nodes represents a different train. The solid lines represent the driving, dwelling and transfer arcs. The origin and destination pairs are represented by the ovals giving the start and end locations of the passengers. The dashed lines represent the origin destination arcs that are introduced to be able to find the shortest path problem.

In Figure 2 we can see the route passengers can take by following the arcs. For example the passengers that travels from Zwolle to Amsterdam. They can take the intercity from Zwolle to Utrecht, transfer at Amersfoort and take the intercity to Amsterdam. Another option is that they can stay in the intercity from Zwolle to Utrecht, transfer at Utrecht and then take the intercity to Amsterdam. At last they can also take the intercity from Zwolle to Utrecht, transfer at Amersfoort and take the regional train to Amsterdam. To determine the fastest route the shortest path should be found for the passengers. The length of the arcs is measured in minutes. When a train gets delayed some of the transfers given in Figure 2 would no longer be available and the shortest path should be calculated again.

2.5 Data

We use the same data as used in Dollevoet et al. (2010). These data consist of the following information:

- The set of stations where trains can arrive and depart from.
- The expected number of passengers travelling from a station to each other station within the set of stations.
- Simulated delay scenario’s to determine whether a certain train is delayed or not.
- Information about arrivals and departures for a certain train number and the corresponding planned time.
- Arcs consisting of detailed information about the planned travel-, dwell- and transfer-time for each part of the route within the set of stations.

Chapter 3

Literature

In this chapter, we will discuss the relevant literature about delay management. We will describe the current methods that have been developed and some future points of research that are mentioned in the papers.

3.1 Papers on delay management

Delay management deals with small delays of a railway system that occur during daily operations. When such delay(s) occur, the current schedule is often no longer feasible and has to be updated to a disposition timetable. Due to the use of cyclic timetables the transfer times for passengers to switch trains are limited. However letting the on-time train wait for the feeder the delay often gets transferred and therefore the connections are often not maintained. During the last years papers have been written with various programming formulations and different restrictions in order to deal with this problem. The first integer programming formulation was described in Schöbel (2001). This formulation uses the concept described in Nachtigall (1998) and makes use of the following activities:

-Driving activities which model the driving of a train between two consecutive stations.

-Waiting activities which represent the waiting of a train in order to let passengers get on and to get off the train.

-Changing activities representing the activities that allow passengers to transfer from one train to another.

In Schöbel (2007) this model has been further developed. She added a never-meet property to make sure the objective function does not count some delay twice, which can occur in certain circumstances described in the article. However in these papers they neglect the fact that some parts of the railway consist of single-track lines.

In Schöbel (2009) algorithms are made that take these capacity constraints into account. In order to do this more activities have been added:

-Headway activities that model the limited capacity of the track system.

However these papers do not adapt the use of re-routing in their formulations.

In Dollevoet et al. (2010) an integer programming formulation has been formulated that uses re-routing. In order to do this they used origin and destination arcs as a set of possible routes in order of time and the best possible route can be picked at any time based on the delays and current time. The last formulation unfortunately only works for small and medium instances because with large data sets CPlex runs out of memory and quits before the first feasible solution is found.

In Biederbick and Suhl (2007) they tried to use simulation and (simple) dispatching to get a solution with fourteen different strategies containing for example:

-Do not wait at all

-Wait until every feeder train has arrived

-Wait *t* minutes

-Wait only if the feeder train has 'many' transfer passengers

They have tested their strategies both with and without passenger routing. The passenger-oriented strategies seem to perform much better than the regular waiting time of Deutsche Bahn AG. However because they did not have actual information about the amount of passengers travelling they could not guarantee that these strategies would also perform best in reality.

In Ginkel and Schöbel (2007) they used a bicriteria in order to solve the problem. Instead of looking at merely the total amount of delay over all passengers they also tried to minimize the weighted number of missed connections. In order to do this they used fixed connections. This method seems to have potential to be used as an online decision support procedure and can be improved by using branch and bound instead of fixing the possible combinations.

Chapter 4

Heuristics for delay management

In this chapter, we will describe two different heuristics, an online heuristic and an offline heuristic. We will first describe both heuristics and then discuss the differences. After that we describe the dispatching rules we tried.

4.1 General description of the heuristics

When a delay occurs the original timetable is no longer useful and therefore a disposition timetable is needed. The heuristic checks whether a delay is occurred when a train arrives and adjusts the nodes appropriately to build the disposition timetable. Besides adjusting the nodes also delay management should be build in the heuristics to determine whenever a train departs whether the on-time train waits or departs on time. In order to do this information about the trains that are delayed and the amount of delay is needed and stored during the process. For the online model also the information about the amount of passengers and where they need to transfer is stored. This all goes chronologically. Using this information the heuristic decides, based on the dispatching rules, whether the train waits or departs. In case a train does not wait passengers cannot maintain their connection and therefore need to be re-routed to find their new route to their destination. The re-routing is solved exactly by using Dijkstra's algorithm. During this process all information about the amount of passengers travelling from A to B and where they need to transfer is stored during this process.

4.2 Differences between the offline heuristic and the online heuristic

In the offline heuristics, the new train schedule will be determined first and after the schedule is known the passengers will be re-routed at the end. In the online heuristic the passengers will be re-routed when they miss a connection. The offline heuristic gives the same or better results than the online heuristic when using the same wait-depart decisions because passengers have more information in the offline heuristic than in the online heuristic. For example a passenger normally goes with train A because train B does not allow a transfer to train C, which could have been the fastest route if the passenger was able to transfer from train B to train C. However after the passenger has board into train A, train C gets delayed which would have made it possible to transfer from train B to C after all. When the route is planned during the process the passenger still takes train A because given the information at the moment the passenger needs to decide, he is not able to transfer to train C when taking train B. When the routes will be determined at the end the passenger would have taken train B instead of A. Because the route at the end can differ from the route that is determined during the process the amount of passengers travelling in each train for the offline method is not accurate because they can take different routes than the heuristic during the process uses. As a result of this, the number of passengers cannot be used to alter the maximum waiting time for the offline heuristic. Also it is not reasonable that passengers know all the delays in advance and therefore the online heuristic is more plausible than the offline heuristic. Nevertheless it is good to compare those two heuristics and the differences in results.

4.3 Dispatching rules

In the heuristics we will use simple dispatching rules that determine the maximum time a train will wait for a delayed train. Because it is also possible that multiple trains need to transfer on the on-time train or the amount of passenger in the delayed train are larger than the amount of passengers in the on-time train it is also good to take this information into account.
We used the following dispatching rules in the heuristics to determine the wait-depart decisions:

- Rule 1: Time rule. In case a feeder train is delayed we let a train only wait if the delay is less than a certain amount of minutes.
- Rule 2: Time rule combined with the number of feeder trains. In case there are more feeder trains the on-time train will wait one minute longer for every feeder train. I.e. if there is only one feeder train the train will wait 1 minute, if there are two feeder trains the train will wait 2 minutes.

$$Maximum\ waiting\ time = WT + FT$$

WT is the variable waiting time. FT is the number of delayed feeder trains, which are trains that have passengers aboard that need to board the on-time train.

- ❖ Rule 3: Time rule combined with the amount of passengers in the feeder train(s). The more passengers the feeder trains have compared to the on-time train, the longer the train will wait. The maximum waiting time is calculated as following.

$$Maximum\ waiting\ time = floor\left(\left(1 + \frac{Delayed\ passengers}{Total\ amount\ of\ passengers}\right)^4\right) + WT$$

The delayed passengers are all passengers that cannot transfer to the on-time train if the train would leave according to schedule. The total amount of passengers consists of the delayed passengers and the passengers that arrived on-time and board the on-time train. WT is the variable waiting time.

Chapter 5

Computational experiments

In this chapter we describe four different cases. We looked at the results and compare the results of the two heuristics with each other and with the results of Dollevoet et al. (2010) to show how far the results are from the optimal solution.

5.1 Cases

The four cases we used are the same as described in Dollevoet et al. (2010) and consists of parts of the railway network in the Netherlands during a period in the late evening. The first case consists of a short period of two hours in the evening and consists of three intercities and one regional train. The following stations are considered in this case:
- Abcoude, Amersfoort, Amersfoort Schotvorst, Amersfoort vathorst, Amsterdam Amstel, Amsterdam bijlmer arena, Amsterdam centraal, Amsterdam lelylaan, Amsterdam muiderpoort, Amsterdam RAI, Amsterdam zuid, Baarn, Bilthoven, Breukelen, Bussum zuid, Den Dolder, Diemen, Diemen zuid, Duivendrecht, Ermelo, Harderwijk, Harde't, Hilversum, Hilversum noord, Hilversum sportpark, Hollandsche rading, Maarssen, Naarden-Bussum, Nijkerk, Nunspeet, Putten, Schiphol, Soest, Soest dijk, Soest zuid, Utrecht centraal, Utrecht overvecht, Utrecht zuilen, Weesp, Wezep, Zwolle.
The second case contains consists of all long distance trains for a time period of four hours and the stations that are considered in this case are:
-Amersfoort, Amsterdam amstel, Amsterdam centraal, Amsterdam zuid, Duivendrecht, Hilversum, Schiphol, Utrecht centraal, Zwolle.
The third case consists of all long distance trains in the Randstad, the Western and most populated part of the Netherlands. The stations considered in this case are all stations of the second case and the following stations:
-Den Haag centraal, Den Haag HS, Gouda, Leiden centraal, Rotterdam Alexander, Rotterdam centraal.
In both the second and third case there are no regional trains in the set. The fourth case contains the same region as the second case but also consists of regional trains and therefore besides the stations of the second case also of the following stations:

-Abcoude, Amsterdam bijlmer arena, Amsterdam lelylaan, Amsterdam muiderpoort, Amsterdam rai, Amsterdam Sloterdijk, Baarn, Bilthoven, Breukelen, Bussum zuid, Den dolder, Diemen, Diemen zuid, Hilversum noord, Hilversum sportpark, Hollandsche rading, Maarssen, Naarden-bussum, Soest, Soest dijk, Soest zuid, Utrecht overvecht, Utrecht zuilen, Weesp.

When the regional trains are taken into account the number of Origin-Destination pairs grow enormously. Therefore only the Origin-Destination pairs with high passenger figures are included in this case.

For the first case only a delay for the train from Zwolle to Amersfoort is interesting, because otherwise no connections are violated. This delay can take a value between 0 and 30 minutes. For all other cases 100 delay scenarios are given. In these delay scenarios each arrival activity has a probability of 10% to be delayed. When a train is delayed the size of the delay is a uniformly distributed integer number between 1 and 15 minutes. Table 1 gives an overview of the number of the origin-destination pairs, the amount of passengers, trains and stations in each case. Take in consideration that the amount of passengers have been scaled for secrecy and does not represent the true number of passengers that travel in the given time period. The percentages represent the amount of origin-destination pairs and passengers that need to transfer. The low number of transfers in case IV is caused by only using origin-destination pairs with high passenger figures.

Case	OD-pairs	Passengers	Trains	Stations
I	111 (15%)	23 (2.3%)	7	41
II	283 (56%)	145 (15%)	117	9
III	675 (48%)	341 (21%)	168	15
IV	239 (4%)	190 (4%)	282	33

Table 1: Overview of cases.

5.2 Computational results

We used Matlab version 7.10.0.499 on an Intel (R) Core™2 CPU (@2.40 GHz) with 2 GB of memory to run the heuristics. We first looked at rule I and case I and tried different waiting times. Figure 3 shows the average objective values for different waiting times. It is clear that waiting for few minutes lowers the objective value but when we let the trains waiting too long the objective values increase rapidly. It also shows that the online values are higher than the offline values as we already predicted before.

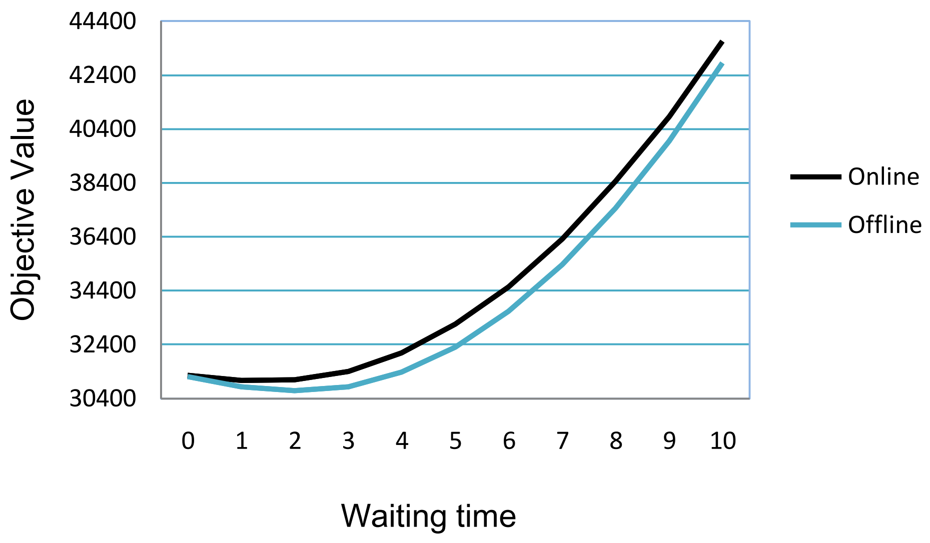

Figure 3: The average objective values for case I using rule I.

Table 2 shows the best solutions for the online heuristic using rule I. We tried waiting times between 0 and 10 minutes for all cases starting at 0. Whenever the objective value increases we stopped running because the objective value only tends to increase rapidly after that point when adding waiting time. The third column represents the best waiting time in order to get the minimum amount of delay. All other columns represent the average value over all delay scenarios. The second column represents the average objective value. The fourth column represents the average percentage of origin-destination pairs that are delayed and the fifth column shows the percentage of passengers that are delayed. The last column shows the average time needed in order to solve the case. The time the heuristics needs to solve the case is determined by finding the shortest path and is therefore almost the same for all rules we used. In the last case the waiting time is much higher than the other cases. This can be explained by the fact that in this case only the origin-destination pairs with a high amount of passengers have been taken into account and therefore it is better to wait longer than in the other cases. Table 3 shows the same table only for the offline heuristic. It is clear that the objective values are lower, and the best values are found by letting trains waiting less long compared to the offline model.

Case	Objective Value	Waiting Time	Delayed OD-pairs	Delayed passengers	Average solution time (s)
I	31062	1 min	9,1 %	7,8 %	0,1 0
II	213181	4 min	22,9 %	17,3 %	2,56
III	681003	4 min	26,2 %	23,4 %	19,45
IV	380302	9 min	29,0 %	22,0 %	135,55

Table 2: Best online values found using rule I.

Case	Objective Value	Waiting Time	Delayed OD-pairs	Delayed passengers
I	30686	2 min	9,8 %	8,8 %
II	198924	3 min	22,5 %	16,8 %
III	644754	3 min	25,8 %	22,9 %
IV	333036	9 min	29,0 %	22,0 %

Table 3: Best offline values found using rule I.

However these simple dispatching criteria do not make use of any information that is stored at all. Therefore we used rule II to take the number of feeder trains also into account. The best offline values using rule II are found in Table 4. The waiting time in the second column represents the time a train will wait if there is no feeder train. Every extra feeder time will increase the maximum time a train will wait with 1 minute.

Case	Objective Value	Waiting Time	Delayed OD-pairs	Delayed passengers
I	30686	1 min	9,8 %	8,8 %
II	198887	3 min	22,9 %	17,3 %
III	644919	2 min	25,9 %	22,9 %
IV	333036	8 min	29 %	22 %

Table 4: Best offline values found using rule II.

The objective values do not differ much of the values found using rule I. For case III the objective value even increases. The small differences could have been expected because the probability multiple trains are delayed which both have passengers that need to transfer to the on-time train is small. Therefore the objective values are almost the same as the objective values found using rule I. Because feeder trains can have just a few passengers that need to transfer or a lot of passengers that need to transfer it could be a good idea to use the number of passengers in order to get a better solution. Therefore we used rule III to use the amount of passenger to make the wait-depart decisions. The results can be found in Table 5 which show the best offline solutions found using rule III.

Case	Objective Value	Waiting Time	Delayed OD-pairs	Delayed passengers
I	30669	1 min	9,9 %	9,1 %
II	192544	2 min	23,0 %	16,9 %
III	636288	1 min	25,9 %	22,8 %
IV	330836	3 min	29,0 %	22,0 %

Table 5: Best offline values found using rule III.

These values show that the use of passengers decreases the objective values significantly. Especially for the larger cases the third rule performs better for all waiting times as can be found in the appendix.

As predicted these objective values are higher than the optimal values found in Dollevoet et al. (2010). They found an optimal value for case I of 30462 which is close to the solution we found of 30669. However for case II they found an optimal value of 172073 which is much better than the solution we found using the heuristics.

However when using a no-wait policy case I has a objective value of 31217 which is about 2% more than the objective value we found in rule III. Therefore these dispatching criteria could be a good alternative when the optimal solution cannot be calculated. Because all delay management models assume that all passengers have all information at any moment we only compared the offline model because it makes the same assumptions. However the passengers do not know everything as explained in Chapter 4 and it could be better to use the online heuristic instead of the offline heuristic in order to determine the best waiting times. However we cannot compare the results and therefore we will just show the best online objective values we found for all cases using rule III which can be found in Table 6.

Case	Objective Value	Waiting Time	Delayed OD-pairs	Delayed passengers
I	30973	0 min	9,2 %	8,1 %
II	206322	3 min	23,3 %	17,4 %
III	671639	2 min	26,2 %	23,2 %
IV	377019	4 min	29,1 %	22,0 %

Table 6: Best online values found using rule III.

5.3 Overview

Table 7 and Table 8 give an overview of all the objective values for the offline and online heuristics. In Table 7 also the optimal solutions found by Dollevoet et al. (2010) are given.

Case	Rule I	Rule II	Rule III	Optimal solution
I	30686	30686	30669	30462
II	198924	198887	192544	172073
III	644754	644919	636288	523978
IV	333036	333036	330836	No solution

Table 7: Overview of the offline objective values.

Case	Rule I	Rule II	Rule III
I	31062	31062	30973
II	213181	213006	206322
III	681003	680911	671639
IV	380302	380302	377019

Table 8: Overview of the online objective values.

Chapter 6

Conclusions

In this thesis we introduced heuristics which calculate the total amount of delay for all passengers using simple dispatching rules. We looked at an online heuristic in which passengers only have information available until that moment and at an offline heuristic in which passengers know the new route in advance. We predicted that the offline heuristic would have better results and came to the conclusion that this always the case. During the thesis we used three different rules as wait-depart decision and compared the results. We found that using the amount of passengers to determine the waiting time improves the solutions quite well. However, even these results are still not even close to the optimal solution. Only this model assumes that the passengers know the route in advance and this is not reasonable. Therefore it could be a good idea to have a look at building online models that calculate the optimal solution and compare these results to the results we have found in this thesis. Still the use of simple dispatching rules to determine the wait-depart decisions can improve results compared to a no-wait policy and therefore could be used when the optimal solution cannot be determined.

References

C. Biederbick, L. Suhl. Decision support tools for customer-oriented dispatching. *Lecture notes in computer science*, pages 171-183, 2007.

T. Dollevoet, D. Huisman, M. Schmidt, A. Schöbel. Delay management with re-routing of passengers. *Econometric Institute Report EI 2010-31*, 2010.

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

L. Kroon, D. Huisman, E. Abbink, P.-J. Fioole, M. Fischetti, G. Maróti, L. Schrijver, A. Steenbeek and R. Ybema. The new Dutch Timetable: The OR Revolution. Interfaces, 39: 6-17, 2009.

K. Nachtigall. *Periodic Network Optimization and Fixed Interval Timetables.* Deutsches Zentrum für Luf- und Raumfahrt, Institut für Flugführung, Braunschweig, 1998. Habilitationsschrift.

M. Schachtebeck, A. Schöbel. To wait or not to wait and who goes first? Delay management with priority decisions. *Transportation Science, DOI 10.1287/trsc.1100.0318,* 2010.

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

A. Schöbel. Integer programming approaches for solving the delay management problem. *Lecture notes in computer science, 4359 : 145-170,* 2007.

A. Schöbel. Capacity constraints in delay management. *Public transport. Planning and operations, 1(2) : 135-154,* 2009.