



**University of
Zurich** ^{UZH}

Department of Geography

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Modeling Carpooling for Multimodal Routing

GEO 620 Master's Thesis

Author

Kissling Julian

11-733-680

Supervised by

Dr. Haosheng Huang

Dominik Bucher (ETHZ)

Faculty representative

Prof. Dr. Robert Weibel

Date of Submission: 21.04.2017

Department of Geography, University of Zurich

Contact

Julian Kissling

Ulmenweg 4
CH-4600 Olten, Switzerland
ju.kissling@gmail.com

Prof. Dr. Robert Weibel

Geographic Information Science (GIS)
Department of Geography
University of Zurich - Irchel
Winterthurerstr. 190
CH-8057 Zurich, Switzerland
robert.weibel@geo.uzh.ch

Dr. Haosheng Huang

Geographic Information Science (GIS)
Department of Geography
University of Zurich - Irchel
Winterthurerstr. 190
CH-8057 Zurich, Switzerland
haosheng.huang@geo.uzh.ch

Dominik Bucher

Institute of Cartography and Geoinformation
Dept. of Civil, Environmental and Geomatic Engineering
ETH Zurich - HIL G 23.1
Stefano-Franscini-Platz 5
CH-8093 Zürich, Switzerland
dobucher@ethz.ch

Abstract

Increasing mobility raises the pressure on existing transportation networks. The private car as the most used means of transportation has a particularly strong environmental impact and produces congestion. The concept of ridesharing, where drivers and riders form a carpool, can help to address this issue by increasing the number of persons per car. Nevertheless, carpooling is still a niche mode of transportation with a difficult access to offers. Often, carpooling providers do not depart from the desired origin which, thus, prevents users from sharing rides. A solution to this problem is multimodal routing, where journeys consist of different modes of transportation. Current systems, however, only combine traditional modes with fixed stop locations and routes by linking the closest stations, hence solving the *Nearest Neighbor Problem*. Carpooling, in contrast, entails imprecise stop locations and can lead to detours. The current state of the art approach is therefore inadequate. Consequently, the question arises as to how carpooling can be integrated into a multimodal system.

In this thesis, a conceptual model of a merging and linking technique, which can represent fuzzy locations and retain flexibility, is presented. In detail, the proposed linking technique relies on drive time areas around considered stable public transportation stops. With the use of these drive time areas, imprecise carpooling stops can be allocated to multiple potential stations. Further, the proposed technique entails stop exploiting along routes in order to maximize flexibility. The concept of Points of Action is introduced in order to find intersections between drive time areas and carpooling routes. Points of Action furthermore guarantee the reachability of a public transportation stop.

The conceptual model is proven in an experiment with the Swiss railway network and real-life carpooling offers derived from an online rideshare platform. It is shown that the proposed merging & linking technique creates a high interconnectivity between carpooling and the railway network. In addition, our system can be queried for meaningful multimodal shortest paths containing carpooling.

The contribution of this thesis is a linking approach for fuzzy and flexible means of transportation in a multimodal routing system, based on well-known modeling approaches. This research can furthermore be implemented in real-life systems of online rideshare platforms or public transportation agencies to provide multimodal trips.

Keywords: Carpooling, Multimodal Routing, Network Linking, Stop Allocation

Acknowledgements

Now, after a year of research in GIScience and Multimodal Routing, I would like to express special thanks to:

- Dr. Kai-Florian Richter, for the initialization of this Master’s thesis.
- Dr. Haosheng Huang and Dominik Bucher, for the proficient supervision and many lively discussions during the process of researching and writing.
- Joanna Schuurman, for the thorough and professional proofreading

Table of Contents

ABSTRACT.....	III
ACKNOWLEDGEMENTS	IV
TABLE OF CONTENTS.....	V
FIGURES.....	VII
TABLES.....	IX
ABBREVIATIONS.....	X
1 INTRODUCTION.....	1
1.1 CONTEXT AND MOTIVATION.....	1
1.2 STATEMENT OF THE PROBLEM & AIM	5
1.3 STRUCTURE OF THESIS	6
2 RELATED WORK.....	7
2.1 ROUTING	7
2.1.1 <i>Graph Theory</i>	7
2.1.2 <i>Shortest Path Algorithms</i>	9
2.1.3 <i>Models & Dynamic Networks</i>	16
2.1.4 <i>Public Transportation Networks</i>	19
2.2 MULTIMODAL ROUTING	26
2.2.1 <i>Network Merging and Linking</i>	26
2.2.2 <i>Routing Techniques</i>	27
2.2.3 <i>Carpooling and Multimodal Routing</i>	28
2.3 RESEARCH GAP	31
3 RESEARCH OBJECTIVES AND OVERALL METHODOLOGY.....	33
3.1 RO 1: MERGING & LINKING CARPOOLING WITH PUBLIC TRANSPORTATION	33
3.1.1 <i>RO 1.1 Characteristics of Real-Life Carpooling Offers</i>	33
3.1.2 <i>RO 1.2 Modeling Carpooling</i>	33
3.1.3 <i>RO 1.3 Merging & Linking</i>	33
3.2 WORKFLOW.....	34
4 DATA MODELING AND GRAPH MERGING & LINKING	36
4.1 PUBLIC TRANSPORTATION (RAILWAY) MODEL.....	36
4.2 CARPOOLING MODEL	40
4.2.1 <i>Carpooling as a Road Network</i>	41
4.2.2 <i>Carpooling as a Timetable</i>	42
4.3 MODEL MERGING & LINKING.....	44
4.3.1 <i>Spatial Allocation of Carpooling</i>	45
4.3.2 <i>Creating Transits</i>	47
4.3.3 <i>Representing Fuzziness and Linking Along Carpooling Routes</i>	47

5	EXPERIMENT	51
5.1	DATA.....	51
5.1.1	<i>Railway Data</i>	51
5.1.2	<i>Carpooling Data</i>	53
5.1.3	<i>Road Network Dataset</i>	59
5.2	DATA STRUCTURES & PREPROCESSING STABLE COMPONENTS	60
5.2.1	<i>General Transit Feed Specification (GTFS)</i>	60
5.2.2	<i>Converting Carpooling Offers into a GTFS Feed</i>	65
5.2.3	<i>Calculating Drive Time Areas</i>	70
5.3	NETWORK GRAPH GENERATION.....	72
5.3.1	<i>Public Transportation Network Graph</i>	74
5.3.2	<i>Carpooling Network Graph</i>	79
5.4	NETWORK MERGING & LINKING	82
5.5	QUERYING	86
6	EVALUATION & RESULTS.....	91
6.1	RO 1.1: CHARACTERISTICS OF REAL-LIFE CARPOOLING OFFERS	91
6.1.1	<i>General Characteristics of Carpooling Offers</i>	92
6.1.2	<i>Spatial Characteristics of Carpooling Offers</i>	94
6.2	RO 1.3: MERGING & LINKING.....	97
6.2.1	<i>Multimodal Graph</i>	98
6.2.2	<i>Stop Allocation & Exploiting</i>	99
6.2.3	<i>Transfers</i>	101
6.2.4	<i>Network Improvement</i>	103
6.2.5	<i>Queries</i>	107
7	DISCUSSION	116
7.1	RO 1.1: CHARACTERISTICS OF REAL-LIFE CARPOOLING OFFERS	118
7.1.1	<i>Carpooling in Switzerland</i>	118
7.2	RO 1.2: MODELLING CARPOOLING	121
7.2.1	<i>Summary of RO 1.1 & RO 1.2</i>	122
7.3	RO 1.3: MERGING & LINKING.....	123
7.3.1	<i>Fuzziness and Flexibility</i>	123
7.3.2	<i>Network Improvement</i>	127
7.3.3	<i>Carpooling in Multimodal Journeys</i>	128
7.3.4	<i>Summary of RO 1.3</i>	130
8	CONCLUSION.....	131
8.1	ACHIEVEMENTS & IMPLICATIONS.....	131
8.2	FUTURE WORK.....	132
	BIBLIOGRAPHY	133
	APPENDIX	139

Figures

Figure 1 Usage pattern of driving, public transportation, bicycling and walking.....	1
Figure 2 Average prices for carpooling offers in comparison to the pricing scheme of the SBB.....	3
Figure 3 Travel distances of carpooling offers.....	4
Figure 4 Situation plan of Königsberg and the schematic graph model.....	8
Figure 5 Schematic examples of a digraph, a network graph and a multidigraph.....	9
Figure 6 Search spaces of a standard Dijkstra's, a bidirectional and an A* search algorithm	10
Figure 7 Search spaces of a Dijkstra's, A* and ALT algorithms.....	12
Figure 8 Average query times compared to the used preprocessing time	15
Figure 9 Schematic representation of a condensed model.....	21
Figure 10 Time-expanded model and time-dependent model	23
Figure 11 Schematic representation of variable transfer times in a time-dependent model	24
Figure 12 A multimodal path and possible substitutions with carpooling.....	29
Figure 13 Five sub paths decomposed from the 2SPSPP	30
Figure 14 Schematic representation of a public transportation trip.....	39
Figure 15 Schematic representation of static transit times and variable transits.....	40
Figure 16 Schematic representation of a carpooling trip.....	43
Figure 17 A carpooling stop contained by two drive time areas of public transportation stops ..	46
Figure 18 Schematic representation of a carpooling stop linked to meta-stops.....	46
Figure 19 Representation of a transfer between carpooling and public transportation.....	47
Figure 20 A carpooling route r crosses a drive time area of a public transportation stop.....	48
Figure 21 Schematic representation of a carpooling connection after exploiting a new stop.....	49
Figure 22 The railway network of Switzerland.....	52
Figure 23 An offer advertised on BlaBlaCar.de	53
Figure 24 Locations of the 17 largest cities in Switzerland.....	55
Figure 25 Number of crawled carpooling offers within an 8 month period	57
Figure 26 Schematic representation of a GTFS feed and its relations	61
Figure 27 Drive time areas of train stations served by the SBB	71
Figure 28 Structure and relations of a GTFS feed.....	73
Figure 29 Schematic representation of the meta graph of the carpooling network.	81
Figure 30 Schematic representation of two interconnected stops	86
Figure 31 Schematic representation of valid routes in terms of the number of hops.....	87
Figure 32 Schematic representation of direct routes (no transfers).....	87
Figure 33 Schematic representation of the Range Problem.....	88
Figure 34 Schematic representation of the inverse RP	89
Figure 35 Number of offers on each weekday	92
Figure 36 Number of offers in a one hour interval of each weekday	93
Figure 37 Distribution of offers per driver.....	93
Figure 38 Number of stop locations per country across Europe	94
Figure 39 Stop locations of carpooling offers and train stations	95
Figure 40 Carpooling routes and a line density across Europe/Switzerland	96

Figure 41 Normalized number of possible transfers along a route	102
Figure 42 Percentile improvement of connections based on Degree Centrality.....	105
Figure 43 Heat map of the top 25% of stop locations improved more than 5%.....	105
Figure 44 Heat maps of stop locations weighted by their PageRank score	107
Figure 45 Direct routes from Olten to Basel using carpooling.....	108
Figure 46 Direct routes from Olten to Basel by train.....	109
Figure 47 Carpooling and railway routes from Olten to Basel.....	109
Figure 48 Carpooling and train route from Olten to Rho Fiera Milano	112
Figure 49 Multimodal route from Bern Wankdorf via Egerkingen to Olten	113
Figure 50 Multimodal route from Chur via Sargans and Zurich to Olten.	114
Figure 51 Schematic representation of routing two consecutive transfer edges.....	115

Tables

Table 1 Structure of the agency file within a GTFS feed	62
Table 2 Structure of the routes file within a GTFS feed	62
Table 3 Structure of the trips file within a GTFS feed.....	63
Table 4 Structure of the stop times file within a GTFS feed	63
Table 5 Structure of the stops file within a GTFS feed.....	64
Table 6 Structure of the calendar file within a GTFS feed.....	64
Table 7 Structure of the calendar dates file within a GTFS feed.....	64
Table 8 Structure of the transfers file within a GTFS feed	64
Table 9 Structure of a carpooling offer divided into steps	66
Table 10 Number of drivers and offers and the percentage of the total amount	93
Table 11 Top 10 most important countries according to stop locations.....	95
Table 12 The number of connections not covered by the railway network of Switzerland	97
Table 13 Number of nodes of different types for all contained MOT	98
Table 14 Number of relationships of different types for all contained MOT	98
Table 15 Number of connected carpooling stops and established links.....	100
Table 16 Number of main and POA stops and the number of relations to train stations	100
Table 17 Number of links to train stations per 1, 10, 25 and 50km of trips of different lengths	101
Table 18 Number of transfer edges created between carpooling and train trips.....	102
Table 19 Schedule of the carpooling routes from Olten to Basel.....	108
Table 20 Schedule of the train routes from Olten to Basel.....	109
Table 21 A train route from Olten to Rho Fiera Milano	110
Table 22 Schedule of a carpooling trip from Olten to Rho Fiera Milano.....	111
Table 23 Schedule of a multimodal journey from Olten to Rho Fiera Milano.....	111
Table 24 Schedule of a multimodal journey from Bern Wankdorf to Olten	112
Table 25 Schedule of a multimodal journey from Chur to Olten.....	113
Table 26 Requirements of the discussed modeling approaches for carpooling.....	121

Abbreviations

2SPSPP:	Two Synchronization Point Shortest Path Problem
ALT:	A*, Landmarks and Triangle inequality
API:	Application Programming Interface
CH:	Contraction Hierarchies
EAP:	Earliest Arrival Problem
FIFO:	First-In-First-Out
GTFS:	General Transit Feed Specification
HH:	Highway Hierarchies
JSON:	JavaScript Object Notation
LCSP:	Label-Constrained Shortest Path Problem
MCP:	Multicriteria Problem
MOT:	Mode of Transportation
NNP:	Nearest Neighbor Problem
PHAST:	Parallel Hardware-Accelerated Shortest Path
POA:	Point of Action
RP:	Range Problem
SBB:	Schweizerische Bundesbahnen
TDD:	Time-Dependent Dijkstra
TED:	Time-Expanded Dijkstra
TNR:	Transit Node Routing
URL:	Uniform Resource Locator
WKT:	Well-Known Text

1 Introduction

1.1 Context and Motivation

Mobility has become more important over the past few decades and may continue to do so in the future (Pajor, 2009). With growing populations, more people are using transportation networks. Estimates say an inhabitant of a larger settlement makes up to 20 trips a day (Celiski et al. 2014 In: Sierpiński et al. 2014). Considering the fact that, today, cars are the most frequently used mode of transportation (MOT) (Sierpiński, 2013) despite being regarded as the least environmentally friendly, the environmental impact as well as the pressure on road networks are huge (Calvo, de Luigi, Haastrup, & Maniezzo, 2004; Sierpiński et al., 2014). As an example, Figure 1 below shows the usage of different MOT in 318 European cities between the years 2001 and 2011.

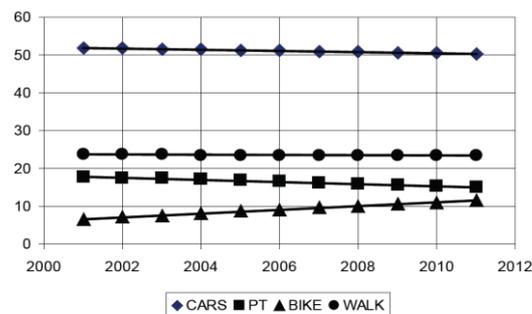


Figure 1 Usage pattern of driving, public transportation, bicycling and walking for 318 European cities between the years 2001 and 2011 (Sierpiński, 2013).

Public transportation is used not even half as frequently as driving. Furthermore, according to the Tagesanzeiger¹, only 1.6 persons occupy a car on average (Tages-Anzeiger, 2013). Thus, high congestion in cities and on major road axes is common. According to Banister (2008), the trend of using a car as the primary mode of transportation can be ascribed to the fact that cycling and walking have become less attractive. People continuously try to minimize their generalized travel costs. Travel costs are a combination of the financial costs and the time taken (Banister, 2008). The optimization of travel costs can be seen mostly when transport is used as a derived demand. This means that people do not travel because of the journey itself, but because of the activity at the destination. With changing travel patterns, leisure-based travel increases (Loo & Chow, 2006; Schlich, Schönfelder, Hanson, & Axhausen, 2004). According to Heinze (2000), leisure mobility is a way of getting away from one's everyday environment and, therefore, increases quality of life. This hypothesis is also known as the Escape Theory (Heinze, 2000).

Even more, travelers usually do not reconsider their MOT because it is bound to their habits, and, however good other forms of MOT are, people will always have a reason to still use their cars

¹ www.tagesanzeiger.ch – The Tagesanzeiger is an interregional newspaper in Switzerland and belongs to the most influential newspapers in Switzerland.

(Banister, 2008; Kenyon & Lyons, 2003). Transport policy measures are therefore needed to achieve a modal shift. These policy measures shall promote the use of different MOT and lead to the development of new transport hierarchies. One approach is to facilitate the use of public transportation (Banister, 2008). Thus, travel planner systems can help in facilitating the access to different MOT. Especially if such a system supports multimodality, the pressure can be distributed among different transportation networks. However, in travel planner systems, reliable information and easy to understand directions are absolutely essential to create a stimulus for using them (Sierpiński et al., 2014). Thus, such a system should, for example, consider Geocoding to identify start- and endpoints, spatial (and time) algorithms to optimize routing between A and B, and communicate with the user in an understandable way (Sierpiński et al., 2014).

As stated above, people are strongly bound to their cars. It can therefore be argued that trying to substitute travelling by car with other MOT is critical and may not succeed. Hence, a MOT which combines the flexibility of cars with the efficiency of public transportation may be a reasonable transport policy measure. One MOT which has the attributes outlined above is carpooling. In carpooling, any driver who is planning a journey can advertise his/her trip on a specific platform such as BlaBlaCar.de². Other users can join the carpool for a small amount of money. Thus, they build a carpool between driver and rider. Most drivers are also willing to make a short detour, which increases flexibility. Therefore, the pick-up and drop-off locations can be at many different places. Carpooling is thus a way of making private cars part of a transportation network (Bit-Monnot, Artigues, Hugué, & Killijian, 2013).

Flexibility is not the only great advantage of carpooling. By comparing the average price per kilometer of carpooling with public transportation, it becomes apparent that it is around 10 times less expensive than the public transportation agency SBB (Schweizerische Bundesbahnen). For both MOT, a flattening can be seen with increasing travelling distance. This phenomenon seems to be crucial as prices may reach exorbitant values on long haul travels. Figure 2 below shows the average price per kilometer in Swiss Francs for roughly 2000 carpooling offers and the pricing scheme of the SBB. The prices of carpooling offers have been converted from Euros to Swiss Francs with an exchange rate of 1:1.06814123³. The average price of the SBB has been calculated for a one-way second class ticket. The average price for a first class ticket is by rule of thumb twice as expensive. Since carpooling does not have a specific pricing scheme, fluctuations are common occurrences. Therefore, a floating average has been calculated. Offers above 1000 km in length are neglected, since the SBB has no comparable offers. Furthermore, the prices of carpooling offers

² <https://www.blablacar.de/> - BlaBlaCar is an online ridesharing community with 40M users in 22 countries worldwide.

³ Exchange rate on February 3rd 2017

longer than 1000 km stagnate at around 0.02 CHF/km. Considering the price, carpooling is a lucrative way to travel. Longer journeys are, compared to public transportation, particularly good value.

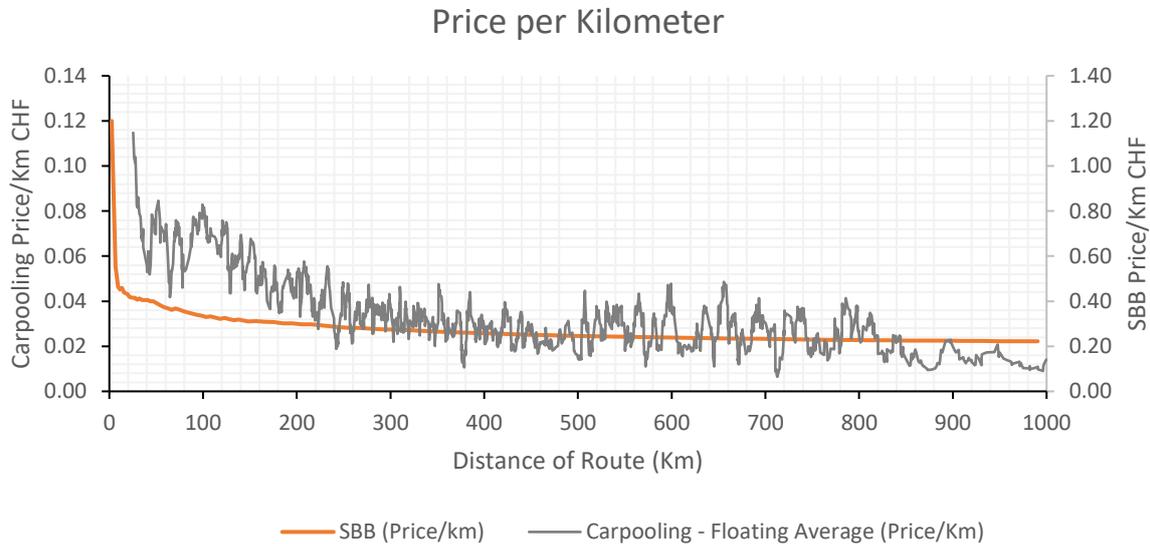


Figure 2 Average prices for roughly 2000 carpooling offers in comparison to the pricing scheme of the SBB.

When having a look at average travelling distances of carpooling, it is visible that carpooling is mostly offered on longer journeys. Offers with a length of more than 1000 km have been excluded in the chart. Roughly 5.5% of all offers are excluded. An accumulation can be seen between routes ranging between 250 km and 800 km, as shown in Figure 3. The total average of a carpooling travel is around 480 km and the median 420 km. In contrast, the average length of travels with the SBB lies around 130 km.

Based on Figure 3, it can be assumed that public transportation (in Switzerland) is, compared to carpooling, mostly used for travelling between regions, whereas carpooling is used on an international scale. Hence, carpooling is not in competition but rather a supplement to public transportation.

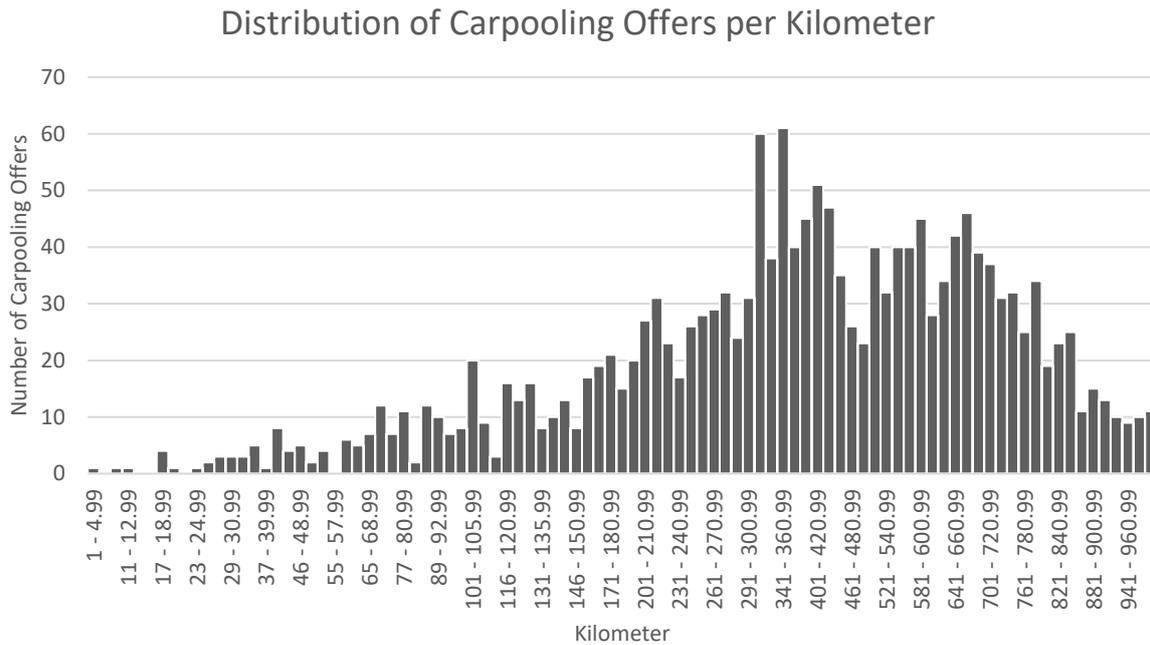


Figure 3 Travel distances of roughly 2000 carpooling offers. The average distance is approximately 480km and the median distance 420km. Journeys >1000km (5.5%) have been excluded from the chart.

However, in Switzerland, carpooling is still a niche MOT and not many offers exist. They often do not depart from a user's desired origin. However, contrary to public transportation, carpooling offers are very dynamic and do not have a fixed schedule nor fixed stops. Each carpooler can define his own stops where he/she could pick up riders. Furthermore, carpooling offers may pass a location directly or with a small detour to a user's desired destination. Unfortunately, the carpooler may not mention these locations, but both the driver and the passenger could profit from stopping there. Hence, an additional effort is needed to search suitable offers, which can be exhausting (Furuhata et al., 2013; Geisberger, Luxen, Neubauer, Sanders, & Volker, 2009). A solution to this inconvenience of searching for offers could be the integration of carpooling into a travel planner system. If the travel planner system supports multimodality, access to carpooling is greater and journeys can be optimized.

In general, multimodal routing combines multiple MOT. For example, a single journey can consist of driving by car and public transportation (cf. Park & Ride). The focus of multimodal transportation is particularly directed at public transportation. This leads to less traffic on the roads and more ecological travel (Grotenhuis, Wiegman, & Rietveld, 2007; Prandtstetter, Straub, & Puchinger, 2013). However, carpooling can also lead to more ecological travel (Ghoseiri, Haghani, & Hamed, 2011). According to BlaBlaCar.de (n.d.), 500k tons of oil could be saved and the emission of CO₂ could be reduced by 1M tons with rideshares made on their platform within the last 12 months alone. As said before, public transportation and carpooling complement each other very well.

As stated in several EU White Papers, multimodal trips are expected to increase. In 2001, the EU has recognized the need for multimodal freight transportation (Commission of the European Communities, 2001). Later in 2011, they expanded this need for general travel between cities (Communities Commission of the European, 2011). Thus, more people should travel in a multimodal way.

Current systems such as Google Maps do indeed support multimodality and are easy to use, but they do not support all available transportation modes. Newer transportation modes such as car sharing and carpooling are yet to be implemented and only a few studies considering their integration into a multimodal system exist (Aissat & Varone, 2015a; Bit-Monnot et al., 2013; Varone & Aissat, 2015). The fact that carpooling has not been integrated yet can be ascribed to several reasons. On the one hand, carpooling is still a niche market and therefore not attractive financially. On the other hand, the dynamism and fuzziness of carpooling is highly problematic for multimodal routing systems. Normally, multimodal routing networks are built upon static networks (Bast et al., 2015). In static networks, fixed stops exist, e.g. public transportation stops or a crossroad in the road network. The base principle of multimodal networks is network merging, where different transportation networks are merged into one single routable graph. This is usually done by applying spatial methods such as a nearest neighbor algorithm. The crucial point is that this cannot be done easily or efficiently with carpooling, since no fixed stops exist and carpooling is of a fuzzy nature, which means detours can be made and consequently new stops can be exploited. New methods must be developed to efficiently merge static and dynamic/fuzzy networks, while retaining the desired flexibility component.

1.2 Statement of the Problem & Aim

This thesis aims to evaluate the technical feasibility of incorporating carpooling into multimodal routing applications. The object of study is the combination of carpooling with public transportation. Both MOT are paragons for different network types. Carpooling as a dynamic/fuzzy network brings the benefit of flexibility, while public transportation as a static network provides reliability and continuity.

According to the current state of the art, different data models for timetable-based networks, such as public transportation, exist. The focus lies on time-expanded and time-dependent networks, which both are adequate solutions for representing timetable-based networks (Bast et al., 2015; Delling, Pajor, & Wagner, 2009b; Goczyła & Cielątkowski, 1995; Müller-Hannemann, Schulz, Wagner, & Zaroliagis, 2007; Pajor, 2009; Pyrga, Schulz, Wagner, & Zaroliagis, 2008; Schulz, 2005). Concerning carpooling, not many studies exist. When modeling carpooling as an independent MOT, time-independent road networks may be used. Keeping the motivation in mind, carpooling

might be an adequate complement to a multimodal routing system (Aissat & Varone, 2015a). Unfortunately, at this point, only two different studies exist (Bit-Monnot et al., 2013; Varone & Aissat, 2015). Both of them follow innovative approaches (cf. 2.2.3 Carpooling and Multimodal Routing). However, in their approaches, carpooling is treated as a substitution to an existing multimodal journey. Hence, they do not consider different modeling approaches for carpooling, nor merging techniques for carpooling with another MOT. Further, the qualification of carpooling offers as a transportation network in general and the applicability as part of a multimodal network have been scarcely evaluated.

Therefore, this thesis aims to:

- Develop a modeling approach for carpooling offers in the light of future integration into a multimodal network.
- Propose a linking technique to combine carpooling and public transportation networks into a multimodal network, respecting fuzziness and flexibility.
- Evaluate the suitability of carpooling offers in terms of spatial and temporal scales, and the benefits of their integration into a multimodal transportation network, with respect to creating better transport planning systems.

1.3 Structure of Thesis

The structure of this thesis follows the work stages that have been undertaken. The following chapter will portray the Related Work in the areas of routing and multimodal routing. Furthermore, the research gap and the research objectives are elucidated. Afterwards, the methodology will be defined and explained on a conceptual basis. In the following experiment, the proposed concepts are tested with real-life data, explaining how the data can be derived, pre-, processed and post-processed. Further, the use of external data models will be justified and extended. The implementation will then demonstrate the generation of transportation graphs for both carpooling and public transportation data. The conceptual methods are needed to merge the resulting transportation graphs. The practical part will be complemented with a results section. The thesis concludes with a discussion of all the previously generated results, highlighting the importance of carpooling in a multimodal network and the benefits of the proposed merging.

2 Related Work

This chapter aims to summarize current research in the areas of routing and multimodal routing. Fundamental principles are elucidated and explained based on state of the art examples of relevant transportation networks, public transportation and carpooling. Finally, the research gap is defined.

2.1 Routing

Routing can be described as the problem of finding a path through a graph (Hart, Nilsson, & Raphael, 1968). Finding such a path is not only bound to transportation networks, but can be applied to any kind of network: telephone traffic, social media, and more (Hart et al., 1968). Crucially, routing is a widely researched topic across disciplines because of its relevance to real-life applications (Pajor, 2009). A system that provides capabilities of routing in a network must be modeled in an adequate manner to provide correct results within an appropriate time. Typically, processing time should not exceed several seconds on a standard device (Goczyła & Cielątkowski, 1995). Hence, fast and reliable algorithms are needed to ensure the above. With increasing mobility (Pajor, 2009; Sierpiński et al., 2014), routing for optimal paths is becoming more important. Traditionally, an optimal path is defined as either shortest in distance or duration. However, as some studies have shown, routing can be also used for defining the most scenic route, the safest path, or other applications as well (Golledge, 1995; Huang et al., 2014). For the former, routing algorithms for shortest paths were developed as early as the 1950s and 60s by Dijkstra (1959), Bellman and Ford (1958) and Hart et al. (1968) (Pajor, 2009). Their application remains widespread in research and in practice. Newer algorithms and approaches, such as *ArcFlags* (Hilger, Köhler, Möhring, & Schilling, 2009) or *Contraction Hierarchies* (Geisberger, Sanders, Schultes, & Vetter, 2012), are usually built upon these concepts. Routing algorithms are consequently the backbone of the whole art of routing. Nevertheless, data models also play a crucial role as they represent the characteristics of a network. Even though different models (time-independent, time-expanded, time-independent) exist, they all share the mathematical concept of the graph theory.

This chapter therefore elucidates different types of routing algorithms and data models. Finally, the current state of the art is demonstrated with an example of public transportation.

2.1.1 Graph Theory

The foundation of routing is *graph theory*. *Graph theory* was first mentioned by Euler (1741). In his paper about the *Königsberg Bridge Problem*, Euler tried to figure out if it was possible to visit every landmass of Königsberg by crossing each connecting bridge only once. Formulating this problem in an abstract way resulted in a graph representation (cf. Figure 4). Even though Euler

did not use the terms vertex and edge, landmass and bridge can be interpreted as the same. Vertices (landmasses) are connected by edges (bridges), hence representing pairwise relations.

It is crucial that many disciplines allow for the representation of pair-relations as mathematical structures. In Social Sciences, graphs can be used to model and reveal relations between people. An organization chart, for example, represents workers and their relationships: (A) -supervises- \rightarrow (B) . In physical geography, graphs can be used to analyze the topological structure of stream networks, where streams are represented as edges and confluences as nodes (Foulds, 1992).

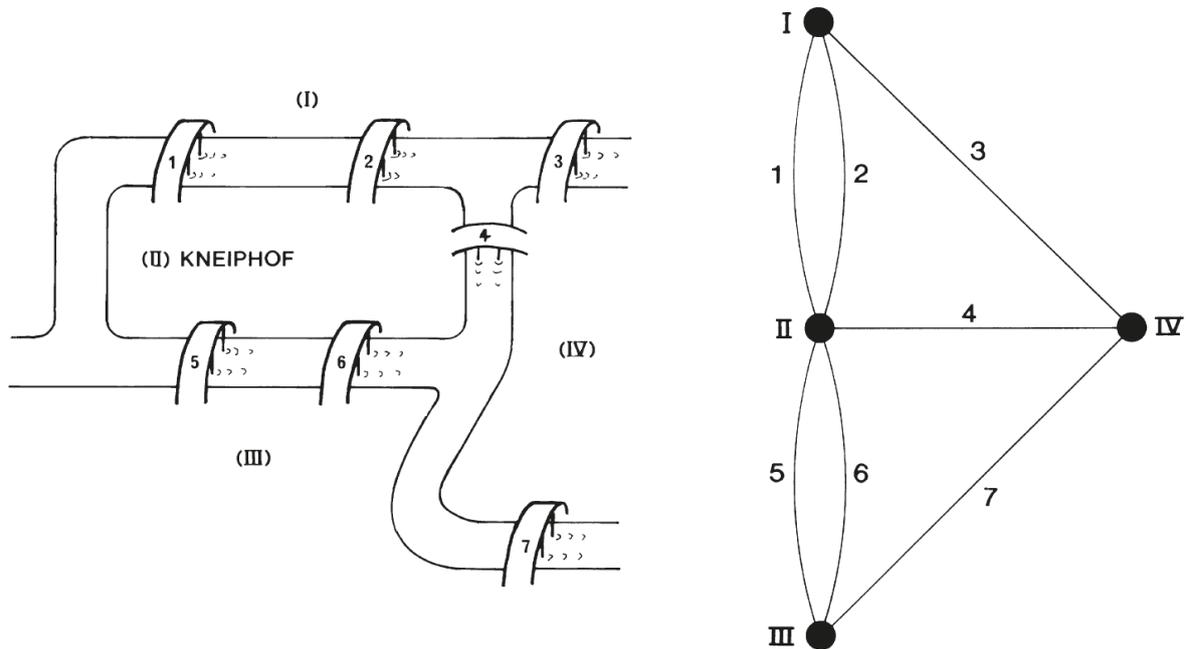


Figure 4 Situation plan of Königsberg (left) and the schematic graph model by Euler. Landmasses (I - IV) are connected by seven bridges (1-7) (Foulds, 1992).

Put more scientifically, a graph G consist of a finite, non-empty set of vertices (aka. nodes) V and a set E of unordered, distinct pairs of vertices called edges (or arcs). Consequently, G is defined by $G = (V, E)$ (Pajor, 2009). An edge $e \in E$ is represented by two adjacent vertices $u \in V$ and $v \in V$, resulting in $e = (u, v) \in E$. Further, an additional edge $e_2 = (v, w)$ is adjacent to e as they share vertex v (Foulds, 1992). While edges are basically undirected, many applications depend on directed edges. Hence, in the before edge (u, v) , v must be the successor of u . A directed graph is also termed as *digraph*. In a *digraph*, edges are usually called arcs and are ordered (Foulds, 1992). An example of a *digraph* is shown in Figure 5.

Networks form a special case of *digraphs*. A *network* is a *digraph* containing a single *source* and a single *sink* node. *Source* nodes are vertices which have only arcs directed away from them,

whereas *sink* nodes only have arcs directed towards them (Foulds, 1992). An example of a network is shown in Figure 5. This definition may be misleading for transportation networks, as they might have any *source* or *sink* nodes. Usually, public transportation routes are offered in both directions.

Standard *digraphs* allow only one arc between two vertices. However, allowing multiple arcs, and thus repetitions, in a *digraph* leads to a *multidigraph*, in which a pair of nodes can have multiple relations (Foulds, 1992). An example of a *multidigraph* is shown in Figure 5.

Furthermore, edges can have weights, denoting the cost it takes to move from u to v (Hart et al., 1968; Pajor, 2009). Edge weights are usually represented by a real number or a function assigning a real number to the edge (Foulds, 1992). Weighted graphs are the baseline for routing purposes, as they allow the use of path algorithms, minimizing the summarized cost of a path p from u to v .

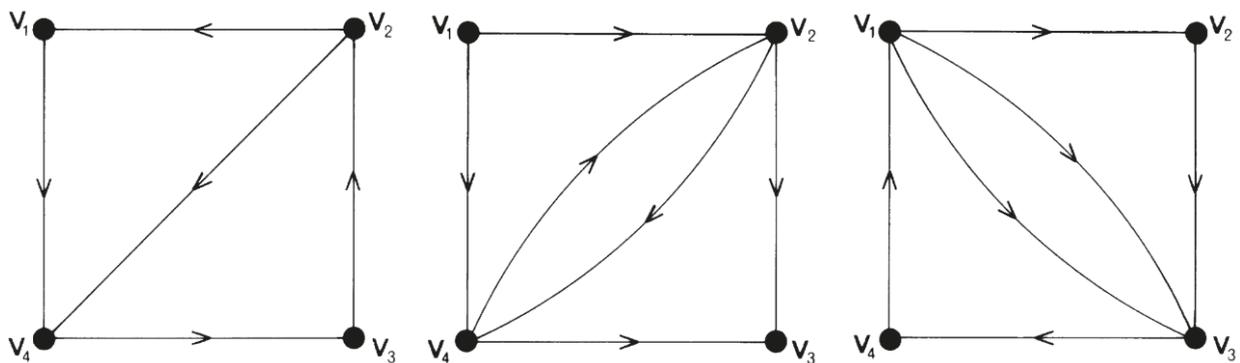


Figure 5 Schematic examples of a digraph (left), a network graph (middle) and a multidigraph (right) (Foulds, 1992).

2.1.2 Shortest Path Algorithms

The goal of routing is to find a shortest route or path in a network. Depending on the size of a network, finding its shortest path can be exhausting and requires high computational power in order to reduce query times (Bast et al., 2015). Therefore, research focuses on different techniques to improve the finding of shortest paths. Five main techniques (Basic, Goal-Directed, Separator-Based, Hierarchical and Bounded-Hop) exist. All of them feature different algorithms which can either be applied to a raw graph or enhance the graph/model to shorten query times. In this section, a brief overview of the main techniques and some of the according algorithms are discussed.

2.1.2.1 Basic Techniques

Basic techniques outline algorithms which can be applied on a basic graph $G = (N, E)$, where N is a set of nodes and E a set of edges. An edge is defined by $(m, n) \in E$. Edges are weighted, e.g. with a distance or a duration d . A shortest path in G from a starting node s to a target node t is defined by $\min(d(s, t))$, thus the path with the minimum length (Sanders & Schultes, 2007).

Basic techniques do not require augmentations of a model and further do not rely on additions such as labels. Hence, they may be applied to any graph with consistent edge weights.

Dijkstra's algorithm: The algorithm proposed by Dijkstra (1959) is one of the most famous algorithms for finding the shortest path in a graph. It can be considered as a greedy label-setting algorithm (Sniedovich, 2006). Furthermore, it is a solution to the *one-to-all* shortest path problem, meaning that it computes all distances from a given node n to all nodes in the graph (Bast et al., 2015). In general, a priority queue Q is initialized with all nodes ordered by distance from the starting node s . All distances are infinite, except $s = 0$. The algorithm starts iterating and extracts the node n with the minimal distance from Q . After extracting n , a scan of all edges $e = (n, k)$ incident to n is performed. The distance value of $(s, n) + l(e)$ is determined and if it improves the distance of (s, k) , an edge relaxation is performed. Thus, k with its new distance (s, k) is added to Q (Bast et al., 2015; Sniedovich, 2006). When solving the *one-to-all problem*, the iteration continues until the node furthest away is reached. When solving the *point-to-point problem*, where start and end node are known, the algorithm stops as soon as it scans the target node. (Bast et al., 2015) The performance of Dijkstra's algorithm depends on the search space. Consequently, the larger the graph, the more nodes must be scanned and the slower the algorithm is. The use of a **bidirectional search** can reduce the number of nodes to scan, thus the search space of one direction, by a factor of two. This can be also seen in Figure 6. The idea is to simultaneously start a Dijkstra algorithm from the start to the target and vice versa, with alternating running of forward and reverse search (Hilger et al., 2009). The shortest path is found if a node has been visited from both directions (Sanders & Schultes, 2007). Other studies propose even more speed-up techniques (Schulz, 2005), but this will not be discussed in more detail. Dijkstra's algorithm is very popular for shortest path queries in road networks, as they rely on static edge weights and do not consider a time component. Esri's Network Analyst Extension, for example, relies on Dijkstra's algorithm (Esri, n.d.-a). However, adaptations for time dependent networks exist (Schulz, 2005). Also, depending on the time dependent model, this algorithm can be used out of the box.

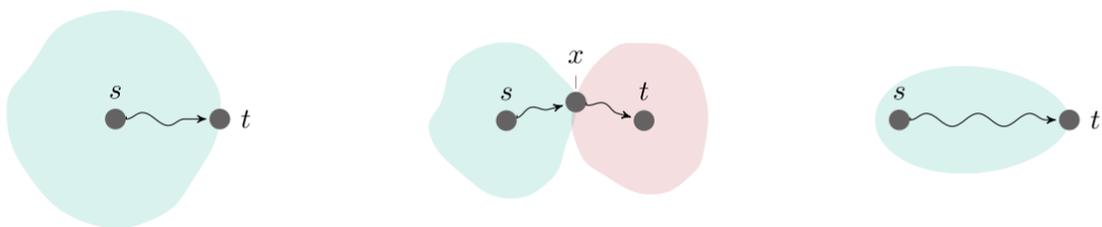


Figure 6 Schematic representation of search spaces of a standard Dijkstra's algorithm (left), a bidirectional search (middle) and an A* search algorithm (right) (Bast et al., 2015).

Bellman-Ford algorithm: In contrast to Dijkstra’s algorithm, Bellman-Ford’s (1958) does not use a priority queue. Instead, a *first-in-first-out (FIFO)* queue is used (Bast et al., 2015). Also, since nodes may be scanned multiple times, it can be defined as a label-correcting algorithm (Bast et al., 2015). The principle of the algorithm is as follows: Like Dijkstra’s algorithm, the start node s is set to $D(s) = 0$, which means zero cost from this node to this node. All other nodes N are set to infinity $D(N) = \infty$. In iterations, edges (m, n) are scanned, if $D(n) > D(m) + l(m, n)$. If so, $D(n)$ is set to $D(m) + l(m, n)$. Therefore, a sequence of relaxation steps is performed. Iterations keep going until the target node is reached (Bannister & Eppstein, 2012).

The Bellman-Ford algorithm is generally slower than Dijkstra’s algorithm, but can in some scenarios be competitive (Bast et al., 2015). An advantage, however, is the possibility of negative edge weights, which are not allowed with Dijkstra’s algorithm (Bannister & Eppstein, 2012). As this thesis works with network datasets where negative edge weights are impossible, this algorithm will not be considered further.

2.1.2.2 Goal-Directed Techniques

Goal-directed algorithms can be defined as guided algorithms. Rather than scanning in all directions, as for example Dijkstra’s algorithm, these algorithms try to avoid scans in the opposite direction of a target node t . Thus, either the geometric embedding of the network or the properties of the graph must be exploited (Bast et al., 2015). The goal is to have an “informed” algorithm, which tries to only scan nodes that are obviously on the optimal path. Consequently, the effort can be reduced (Hart et al., 1968).

A* search algorithm: The A* algorithm is a typical goal directed approach, developed by Hart et al. (1968). This algorithm is basically a modified version of Dijkstra’s algorithm in which the priority of nodes is defined. Prioritizing nodes leads to scanning nodes closer to the target earlier (Bast et al., 2015). However, other than a Dijkstra, an A* is only applicable for *to-one* searches, whereas a Dijkstra can, as mentioned above, find paths *to-all*. A* uses two kinds of information, first it uses the exact cost of a path $l(s, n)$ from the starting point s to a node n and second a heuristic cost function estimated by the cost $h(n, t)$ of n to a target t . During iterations, the algorithm balances l and h when moving through the graph, to find the node n with the smallest cost of $l(s, n) + h(n, t)$ (Amit, n.d.). This algorithm stops as soon as it is about to scan the target node and delivers the correct result (Bast et al., 2015).

Due to its *goal-directed* search, the search space of an A* algorithm is significantly smaller than the one of a Dijkstra algorithm, cf. Figure 6 (Bast et al., 2015). Theoretically, A* should perform well. However, studies show that in a road network where the geographical distance divided by the maximum speed is used as the potential function, no or only small performance gains result (Bast et al., 2015). A common speedup technique is **ALT** (A*, landmarks and triangle inequality).

The principle of *ALT* is to insert a set of landmarks L into the graph. During preprocessing, the distances of all nodes $n \in N$ to L are calculated (Goldberg & Harrelson, 2005). Thus, triangle inequalities involving these landmarks can be used to calculate lower bounds during a *point-to-point* query (Bast et al., 2015). Consequently, the search space is reduced again, which can be seen in Figure 7.

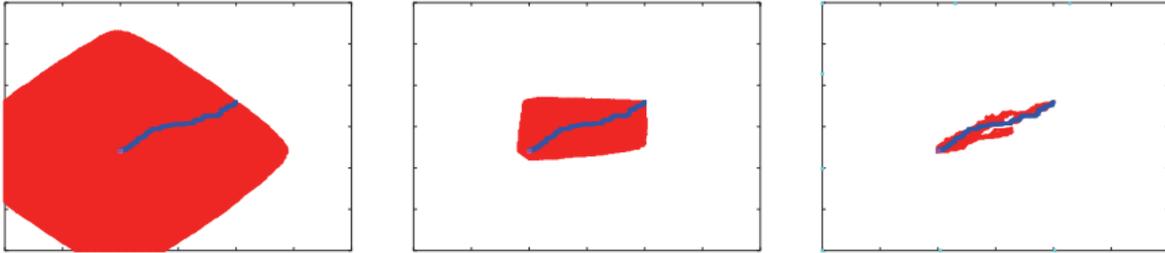


Figure 7 Schematic representation of search spaces of a Dijkstra's algorithm (left), a regular A^* algorithm (middle) and an *ALT* algorithm (right) (Goldberg & Harrelson, 2005).

ArcFlags: The algorithm developed by Hilger et al. (2009) is again based on Dijkstra's algorithm. The idea is, like in A^* , to reduce or more likely to improve the search space for the Dijkstra algorithm. Other than A^* , *ArcFlags* does not use any heuristics, but a subgraph is created during preprocessing. More precisely, the graph is partitioned into K cells of approximately the same size (edges) (Hilger et al., 2009). Every edge has a vector v of K bits, which are called *ArcFlags*. If the edge lies on a shortest path to a node of a cell i , the i -th bit of the vector v is set. Further, the algorithm will scrap edges where the bit for the cell containing the target t is not set (Bast et al., 2015).

According to Bast et al. (2015), *ArcFlags* is currently one of the fastest algorithms in terms of query time for road networks. Unfortunately, preprocessing can be extremely exhausting due to the worst-case time complexity of $O(n^2 \log n)$. As an example from Hilger et al. (2009), preprocessing a graph with 1M nodes and 2.5M edges could take weeks. Although speedup techniques, e.g. the *PHAST* algorithm, for preprocessing exist (Bast et al., 2015), *ArcFlags* is not very well suited for dynamic networks. Nevertheless, some approaches on how to update *ArcFlags* in dynamic graphs have been proposed by Berrettini, D'Angelo and Delling (2009). Since *ArcFlags* is best suited for static, non-dynamic road networks, this thesis will not discuss this algorithm any further.

2.1.2.3 Hierarchical Techniques

Hierarchical techniques try to integrate the hierarchy of the network they represent. As an example, during long travel on the road network, the importance of highways is much higher than rural or suburban streets. Hence, the algorithm can restrict itself to this subnetwork of highways. According to Bast et al. (2015) and others, road categories are a popular heuristic. Unfortunately, these unverifiable input categories may give inexact shortest paths (Bast et al., 2015). Alternatively, preprocessing can be used to compute the importance of nodes and edges without any foreknowledge (Bast et al., 2015).

Contraction Hierarchies (CH): The algorithm proposed by Geisberger, Sanders, Schultes, & Vetter (2012) removes unimportant nodes from a directed and weighted road network. The idea is to implement shortcuts for a temporarily deleted node n and therefore preserve shortest paths. This concept is called *node contraction* (Geisberger et al., 2012). Node contraction is performed by one node at a time until the graph is empty. A set with the original nodes and edges as well as the shortcuts form a *contraction hierarchy*. Nodes are ordered by their importance (Bast et al., 2015). The later a node is removed from the graph, the higher it is in the hierarchy (Geisberger et al., 2012). The contraction process is performed in a preprocessing phase. Geisberger et al. (2012) argue that, for static networks with less change, preprocessing is an adequate approach for speeding up queries. However, networks with an extensive number of shortcuts will also require long preprocessing phases. Thus, *Contraction Hierarchies* is mostly reserved to static road networks, rather than dynamic, timetable based networks. Nevertheless, a resulting graph can be queried using a simple variant of a bidirectional Dijkstra's algorithm. Only edges that lead to a node higher up in the hierarchy are relaxed (Geisberger et al., 2012).

Contraction Hierarchies can be considered as one of the fastest and most successful routing techniques on static road networks. Further, implementations for both PC and mobile devices exist (Geisberger et al., 2012).

2.1.2.4 Bounded-Hop Techniques

Bounded-Hop Techniques require extensive precomputing. In a naïve approach, one could calculate distances between every pair of nodes. Short distances could easily be retrieved with a single table lookup and single-hop paths. Consequently, the idea of bounded-hop techniques is to precompute distances between pairs of nodes and therefore adding "virtual shortcuts", which reduce the number of hops along a path. Queries, only running on this virtual distances, can return a virtual path with only few hops (Bast et al., 2015).

Transit Node Routing (TNR): Other than some of the above routing algorithms, *Transit Node Routing* can be considered as a framework. In *TNR*, three circumstances have to be considered in order to calculate shortest paths: A set of transit nodes, access nodes for all nodes and the fact that not all queries between nearby nodes can be answered using the set of transit nodes (Arz, Luxen, & Sanders, 2013). *TNR* again requires preprocessing to retrieve transit nodes, access nodes, and their distances (Bast et al., 2015). Transit nodes can be calculated using *Contraction Hierarchies*, for example (Arz et al., 2013). Since *CH* builds a hierarchy where nodes that are part in many shortest paths are higher in order, a small set of nodes T can be selected from the top of the hierarchy. Further, distances between all $t \in T$ are calculated (Arz et al., 2013). As using *CH* is already a more sophisticated approach, others use a “base” version of *CH*, namely *Highway Hierarchies (HH)* to retrieve transit nodes (Sanders & Schultes, 2006). However, *HH* also creates a hierarchy which can be used to retrieve transit nodes. Using these transit nodes, a set of access nodes $A(u) \subseteq T$ for every node $u \in T$ can be calculated. If a transit node $v \in T$ is the first transit node in a shortest path P from u , v can be considered as an access node of u . In addition, all distances between u and its access nodes are stored in a distance table (Bast et al., 2015). A query for a *point-to-point* shortest path $s-t$ can thus use the distance table to select paths that minimize the distance $s-a(s)-a(t)-t$ (Bast et al., 2015). Unfortunately, queries can lead to a false result if s and t are too close to each other. In this case, the shortest path may not contain a node from T . Therefore, local searches need a special treatment and thus a *locality filter* needs to be implemented (Sanders & Schultes, 2006). This filter decides whether a query is of global or local (P does not contain a node of T) nature. For local queries, a fallback algorithm is used (Bast et al., 2015).

As mentioned above, *TNR* is a framework rather than an algorithm and therefore needs extensive preprocessing using already extensive algorithms like *CH*. Therefore, *TNR* is mostly used for static road networks.

2.1.2.5 Recap & Performance

In this section, different algorithms and frameworks to retrieve and speedup shortest paths in a transportation network have been presented. As has been shown, some (e.g. *ArcFlags*) require extensive preprocessing. In preprocessing, basic to more sophisticated algorithms (e.g. *CH*) are run on the input graph to reduce the search space, add shortcuts or create virtual paths and lookup tables. The goal of this preprocessing is to improve future shortest path queries. Critically, extensive preprocessing may not be applicable for every type of network. While processing makes sense for static networks that do not change often, it is not an adequate solution for dynamic networks with frequent updates. However, some research in this direction exists (Bast, Brodesser, & Storandt, 2013; Berrettini et al., 2009).

Bast et al. (2015) present performance measures for different algorithms. In their experimental setup, the road network of Western Europe with approximately 18M nodes, 42M edges and 13 road categories was used. An average speed of 10^i km/h was assumed, where i represents a category (Bast et al., 2015). Figure 8 shows the query times for several *point-to-point* queries compared to the preprocessing time. It is important to note that the algorithms only returned the length of the shortest path and not the actual nodes. Subsequently, this fact may have led to a slightly better performance. Most calculations have been performed on a single core machine (3.33. GHz) and the others have been scaled accordingly (Bast et al., 2015). Nevertheless, it can be seen, that there is a trend showing that with less preprocessing time, query time increases. This seems to be crucial, since, for example, an unidirectional Dijkstra’s algorithm scanned approximately 9M nodes, where CH only had to scan 280 nodes (Bast et al., 2015). However, Bast et al. (2015) mention that there is no “best” technique. From a personal view, this makes sense. Depending on e.g. the network type, computational power, and other factors, different algorithms may be chosen. There will always be a trade-off between graph size, preprocessing and query time (Bucher, Jonietz, & Raubal, 2017).

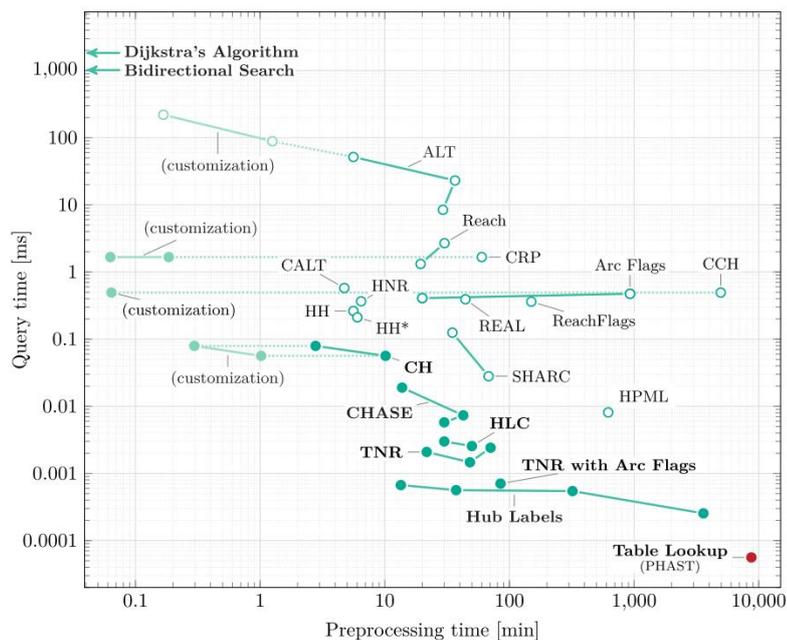


Figure 8 Average query times compared to the used preprocessing time. Results shown are retrieved from an experiment with the road network of Western Europe. Lines represent trade-offs for the same algorithms (Bast et al., 2015).

2.1.3 Models & Dynamic Networks

Many different networks with some special characteristics exist. While road networks can be considered stable, public transportation networks are inherently time dependent. It is crucial that, depending on the input data, different modeling approaches are required. Furthermore, any kind of network may be affected by changes, requiring adjustments to the underlying data structure. This section presents the most prominent models (time-independent, time-expanded and time-dependent) and their applications and characteristics. Lastly, dynamism is discussed and solution approaches are presented.

2.1.3.1 Time-Independent Model

A time-independent network is modeled as a directed time-independent graph. Usually, time-independent graphs are used for static networks as they are well suited for representing geographic distances (Pajor, 2009). Distance is represented as a constant weight of an edge $e = (u, v)$ between two nodes u and v . As for the simple form of a time-independent graph, standard routing algorithms can be applied. Routing on these graphs has been well researched in the past (Delling, Sanders, Schultes, & Wagner, 2009).

For example, road networks are typically modeled as time-independent graphs. In a road network, junctions are represented with a set of nodes N and are connected with edges $e \in E$. An edge exists if and only if a road exists between two junctions. In case of a two-way road, two directed edges are created between the junction nodes (Pajor, 2009). A graph G representing a road network can thus be defined by $G = (N, E)$.

Time-independent models are well suited for representing static networks, but they might lack relations to realism. By considering a road network, congestion may lead to delays, which then increase the duration needed to pass a road segment (Pajor, 2009). Hence, the lack of time-dependency can lead to false results. Moreover, other networks, e.g. public transportation, rely strictly on a timetable. Trains, buses or trams only operate at specific times during a day. It is therefore indispensable that a time-independent model is not applicable for time-dependent MOT.

2.1.3.2 Time-Expanded Model

A time-expanded graph is based on the concept of a time-independent graph (Pajor, 2009). Using discrete time values and creating a respective node for each time, a time-independent graph can be “expanded” to a time-expanded graph (Schulz, 2005). Due to embedding discrete time values in the graph, a time-expanded graph also accounts for arrival and departure times of a trip (Pajor, 2009). Consequently, and stated by multiple authors, a time-expanded model is usually used for timetable-based networks (Bast et al., 2015; Delling, Pajor, & Wagner, 2009a; Müller-Hannemann et al., 2007; Pajor, 2009; Pyrga et al., 2008). Due to its simplicity, this type of graph is widely ac-

cepted and used in current research, as it represents the structure of a network adequately. Besides its simplicity, time-expanded graphs tend to be fairly large in size (Pajor, 2009), since the complexity of an network is stored in the graph itself rather than shifted to the query. Nevertheless, time-expanded models can be routed using basic techniques such as Dijkstra's algorithm. However, they are not sensitive to the *FIFO* property, meaning overtaking of e.g. trains is allowed and not critical. Routing on a time-expanded graph tends to be slower than on a time-dependent graph (Pajor, 2009; Pyrga et al., 2008). Nevertheless, several speedup techniques for well-known routing algorithms have been developed (Bast et al., 2015).

2.1.3.3 Time-Dependent Model

Time dependency is an important factor in many networks. Public transportation, for example, has fixed lines which operate at set times during the day. Thus an optimal route depends on the departure time, as well as on the arrival time (Bast et al., 2015). In addition, a road network which is congested during rush hours has continuously changing edge weights. If a traveler desires to find a path with the earliest possible arrival, a different modeling approach must be chosen. An adequate solution is to use a time-dependent model. In a time-dependent model, edge weights no longer have constant weights (Pajor, 2009). A travel time function is assigned to edges in the graph which represent the duration it takes to traverse them at a specific time during the day (Bast et al., 2015). Because of the use of travel time functions, a time-independent model only has one relation from a start node to a target node, if at least one elementary connection exists (Bast et al., 2015). Consequently, the size of the graph decreases massively compared to a time-expanded model. A shortest path is therefore inconclusively the shortest path per se, but the fastest. Further, since the optimal route depends on a departure time at a certain node, paths may differ at different departure times (Goczyła & Cielątkowski, 1995; Pajor, 2009). Generally speaking, basic algorithms can also be run on time-dependent models but may be bound to some restrictions. Dijkstra's algorithm, for example, does only work consistently, as long as a later train cannot arrive earlier. Thus, overtaking is problematic, meaning the *FIFO* property must be fulfilled (Bast et al., 2015; Pajor, 2009). Unfortunately, the phenomenon of overtaking is quite common in public transportation. A solution to this is proposed by Pajor (2009). Routes which exhibit overtaking are split into a minimal set of routes, so that there are no more conflicting trains on the same route. This approach solves the problem of overtaking trains, but works against the advantage of time-dependent models, namely the graph size. In addition, time-dependent modeling requires augmentations of query algorithms, since the complexity has been shifted from the graph to the query (Pajor, 2009). Also, additional memory is needed to store the travel time functions (Pajor, 2009). Nevertheless, Pyrga et al. (2008) have shown that these disadvantages do not outweigh the gain from the smaller graph size and the smaller query time. Time-dependent models are a good alter-

native to time-expanded models, since they reduce the size of the graph. However, it must be considered that the complexity of the network is shifted to the query and, therefore, requires some adaptation to basic techniques. It can thus be said that both time-expanded and time-dependent models are adequate solutions for modeling time dependency.

2.1.3.4 Dynamic Networks

Transportation networks, regardless of whether they are time-dependent or not, tend to have a dynamic component. Road networks, for example, are usually affected by congestions, detours and other obstacles. These, sometimes unpredictable, changes have an impact on the routable network. However, if a network can be considered stable, using algorithms that require extensive preprocessing but reduced query times make sense. However, changes in a network would require a rerun of the preprocessing to nevertheless ensure fast and correct query outputs (Bast et al., 2015). This can come at a high cost (reconsider preprocessing times of *ArcFlags*) (Hilger et al., 2009). With this said, research has also focused on how to bypass the problem of extensive preprocessing, so that dynamic networks can also profit from sophisticated routing algorithms. Bast et al. (2015) listed four approaches currently discussed:

Repair: Rather than rerunning preprocessing, the repair approach tries to adjust areas in the graph that are affected by the change. Theoretically, this seems to be possible for different techniques (*ALT*, *ArcFlags*, *CH*, ...), but with strongly varying success.

Bypass wrong parts: This approach aims to bypass wrong parts of the preprocessing and, hence, increase the complexity of the query. Therefore, the query algorithm must be adjusted. Bast et al. (2015) further state that this approach may lead to an increase in query times. By simply flicking over this approach, it could potentially be that with a high number of changes, the graph could fail in some way. If too many changes are made, the query algorithm must bypass many defect parts and may be much slower and even return false results.

Metric-independency: This approach also has an influence on the query algorithm. The idea is to move all metric-dependent components to the query. Thus, preprocessing will be based only on metric-independent components which are less sensitive to changes. However, shifting complexity to the query entails a significant decrease in query times.

Metric Split: This approach aims, similarly as the one above, to split the metric-dependent and -independent components. Rather than shifting the dependent component to the query, the split approach divides preprocessing into two stages. The first stage considers the much more stable topology of a network. In the second stage the much cheaper metric-dependent component is processed. The later stage conceives edge weights which are more affected by changes. As an example, in the *ALT* algorithm, landmarks are kept and only their distances are recalculated.

This section illustrates that even algorithms that require preprocessing can be used on dynamic networks. However, not all of the approaches outlined above can be applied to every algorithm. Furthermore, some algorithms do not even require such techniques as they are not bound to preprocessing at all (cf. Dijkstra’s algorithm, Bellman-Ford algorithm).

2.1.3.5 Discussion

This section introduced three different modeling approaches and the key factors of dynamic networks. When asking the question, “*What model suits best?*”, no correct answers can be given, although it could be argued that for a public transportation network, which is inherently time-dependent, the use of a time-dependent model makes the most sense because of the small graph size and fast queries. However, Pyrga, Schulz, Wagner, & Zaroliagis (2004) have shown that when more realism is added to a time-dependent graph, its size grows significantly. Further, solving e.g. a *Multicriteria Problem* is merely 58% faster on a time-dependent graph. In addition, Müller-Hannemann & Schnee (2007) state that changes in a time-expanded model can be realized with less effort. Hence, the appropriateness of a model is highly influenced by the purpose it has to fulfill and the nature of a transportation network.

2.1.4 Public Transportation Networks

Public transportation networks are quite different to road networks. Road networks can easily be represented with a straightforward graph, where junctions are modeled as nodes u & v and streets consequently as edges $e = (u, v)$. Edges exist only if there is a road in real life and are weighted by their distance (Pajor, 2009). As a road network can be accessed and used on demand, it can be considered time-independent. Thus, finding a shortest path relies consequently only on one criteria, the edge weight (Bast et al., 2013).

In a public transportation network, multi-criteria optimization is far more important, as not only the edge weight is needed for finding an optimal path, but also other criteria such as transits and financial costs (Bast et al., 2015; Pajor, 2009), leading to the *Multicriteria Problem*. Goczyła & Cielątkowski (1995) present a list of optimality criteria and additional constraints that define and influence the finding of an optimal route between two points:

- Minimal journey time (correlates with the road network, if the edge weight represents duration)
- Minimal number of train changes
- Minimal (financial) cost of journey
- Minimal distance covered during the journey

As some of these criteria may conflict—for example distance and duration, i.e. when a route is longer but quicker—not all of them must be fulfilled for an ideal route. The criteria of an optimal route are thus somewhat personal. Regarding the constraints formulated by Goczyła &

Cielątkowski (1995), especially two statements seem important: *earliest time of departure from the starting point* and *latest time of arrival at the ending point*. These seem to be key as a passenger most likely wants to start his journey as soon as possible and arrive as fast as possible in a convenient way. In a time-independent road network, such a journey is simply represented by the shortest path which can be commenced on at any time. This is a main difference to public transportation, where specific connections are only offered at a specific time during the day (Bast et al., 2015), leading to the *Earliest Arrival* and *Range Problem*.

Routing on a public transportation network poses further problems because of the strict time dependency and multiple criteria. Primarily, there is the *Earliest Arrival Problem (EAP)*, aiming to determine an earliest arrival as possible. The *Range Problem (RP)* requests an optimal path from a given range of possible departure times. The *Multicriteria Problem (MCP)* focuses on delivering optimal paths based on multiple criteria rather than just travel time. Lastly, some specialties related to the *Multicriteria Problem*, such as uncertainty, fares, and night trains exist. It is crucial that public transportation must be modeled in an adequate way to be able to solve the above-mentioned problems. However, the model by itself is not necessarily a solution to all problems. Some might need adjustments to the query algorithms as well.

The following sections therefore aim to deliver definitions of the posed problems, a current state of the art on modeling approaches and specialties occurring in public transportation networks.

2.1.4.1 Earliest Arrival Problem (EAP)

The earliest arrival problem describes a circumstance in time-dependent routing. Generally speaking, the goal is to minimize the difference between the arrival time and the given departure time (Pyrga et al., 2008). Therein, Pyrga et al. (2008) define two different variants of this problem, namely the simplified and the realistic version. In the simplified version, one assumes that transfer times are negligible. The realistic version defines transfers as weighted part of a trip and therefore require nonnegative transfer times (Pyrga et al., 2008). Solving the earliest arrival problem is straightforward, according to Bast et al. (2015). On a time-expanded graph, Dijkstra's algorithm, here also known as *TED* (time-expanded Dijkstra), can be used. By starting at the first event of a source stop s that occurs at a defined time τ , the earliest arrival is found as soon as the first event of the target stop t is reached (Bast et al., 2015).

On time-dependent graphs, Dijkstra's algorithm can be used as well, but has to be augmented (*TDD – Time-Dependent Dijkstra*). Also, the cost functions must be nonnegative and the *FIFO* property must be fulfilled (Bast et al., 2015).

2.1.4.2 Range Problem (RP)

In public transportation, a traveler is bound to fixed departure times. By starting at a specific time, an optimal path may not be guaranteed, as for example a connection at exactly this time would require many transfers. Therefore, it seems to be important that one would like to find the optimal

path within a time range (e.g. 8:30 – 9:00). To solve this problem, a set of journeys with a minimum travel time is required (Bast et al., 2015). The *RP* can be best solved using a time-dependent model by implementing a tentative travel time function (Bast et al., 2015) or by using a frequency-based model (Bast & Storandt, 2014).

2.1.4.3 Multicriteria Problem (MCP)

Travelling on a road network is usually only bound to one criterion, the shortest or fastest path. In public transportation networks, however, one could argue that an optimal path is also defined by multiple criterion, for example duration, cost, or number of transfers (Goczyła & Cielątkowski, 1995; Martins, 1984). Müller-Hannemann & Schnee (2007) discuss an algorithm for time-expanded models. The idea is to optimize paths based on travel time, number of interchanges, and the price.

2.1.4.4 Modeling Public Transportation

Generally, a public transportation network can be modeled using the simplest of all, a **condensed model** (Goczyła & Cielątkowski, 1995; Pajor, 2009). In a condensed model, lines connect stop locations with each other. Adding weights to the edges enables the possibility of finding a shortest route between multiple stops x . As is shown in Figure 9, edges are added for one-way and round

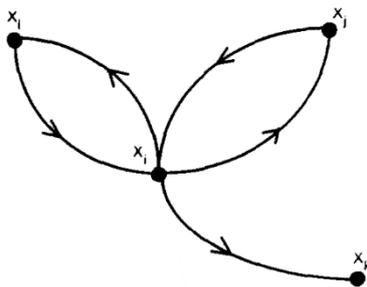


Figure 9 Schematic representation of a condensed model. Stations (nodes) are connected with directed edges representing connections (Goczyła & Cielątkowski, 1995).

trips. Hence, a directed graph is needed (Goczyła & Cielątkowski, 1995). This model can represent the structure of a public transportation network adequately (Pajor, 2009). However, this simple model cannot be used in a real-life scenario, because an important factor is missing, namely departure and arrival times. Hence, shortest route algorithms result only in representing the travel time between two stops, not considering departure or transit times (Pajor, 2009). Subsequently, queries on a condensed model do not solve the *EAP*.

Public transportation can be considered time-dependent, because timetables exist. Certain events, e.g. a train from A to B, happens at a predetermined point in time (Bast et al., 2015). Contrary to a road network, public transportation cannot be accessed with the same flexibility, because passengers have to act according to a timetable. Consequently, a model must be able to represent these events. Current research therefore focuses on two major models, the time-expanded and the time-dependent model, as well as on a third less common model, the frequency-based model (Bast & Storandt, 2014; Dellling, Pajor, et al., 2009b; Pyrga et al., 2004; Schulz, 2005). As a prerequisite, it is important to define a timetable.

Timetable: In public transportation, a timetable consists of time-dependent events (Bast et al., 2015). On a more theoretical level, a timetable stores information about stops, trains/buses/ ... , departure and arrival times and how stations are connected (Müller-Hannemann et al., 2007). Combining this information, a connection $c =: (z, s_1, t_1, s_2, t_2)$ can be defined. This connection c can be considered as a train (z) going from A (s_1) to B (s_2) at a specific time (t_1, t_2) (Müller-Hannemann et al., 2007; Pajor, 2009). It is important to note, that a connection does not represent an actual geographic route, but simply a relation between two stop locations.

Time-Expanded Model: As written above, time-expanded models are well studied for timetable-based networks. They are an expansion of a simple time-independent model (cf. condensed model). In contrast to a condensed model, a time-expanded model represents events, rather than connecting stops (Pajor, 2009). They are therefore also known as event graphs (Bast et al., 2015). A connection can be considered as an event. Hence, in a basic model, a graph contains a node for every departure (t_d) and arrival time (t_a) located at a stop (s). The times t_d & t_a are connected with an edge and thus represent a connection (Bast et al., 2015). Pajor (2009) further discusses two variants of the time-expanded model, which concern transfers. In a simple model, the minimum transfer criterion is omitted, which means that at each stop where the departure time is greater than the arrival time, every connection could be made. Conceptually speaking, this will return a true shortest path, but in practice, catching a different train that leaves one minute after the arrival of another is not always possible. Consequently, a more realistic version of the time-expanded model has been developed. In a realistic model, transfers are enabled by adding an additional transfer nodes and edges between connections (cf. Figure 10) (Pajor, 2009; Pyrga et al., 2008).

Figure 10 shows the higher complexity of a time-expanded graph in comparison to a time-dependent graph. Every elementary connection is represented with nodes and edges plus transfer nodes, if transfers are possible. This consequently leads to increased graph sizes. Nonetheless, this additional complexity makes time-expanded models immune to the *non-overtaking FIFO* property. Basic routing techniques can be applied without any necessary augmentation or preprocessing to solve the *EAP* (Bast et al., 2015; Pajor, 2009; Pyrga et al., 2008). Finding a shortest path solving the *EAP* can be exhausting using basic techniques, since for example Dijkstra's algorithm can only be run in a unidirectional way. Bidirectional running would require the exact target node before querying. However, this target node is not known in advance, since the arrival time is unknown (Pajor, 2009). Research has therefore focused on optimizing and accelerating queries on a time-expanded graph. Delling, Pajor, et al. (2009), for example, have observed, that in a public transportation network, many equal paths between pairs of nodes exist. Thus, they propose a technique that combines *ArcFlags*, *ALT*, and *node blocking* to omit scanning multiple edges of the same route.

Other authors also propose speedup techniques for querying time-expanded graphs by augmenting e.g. *ALT*, *Geometric containers*, and other techniques (Bast et al., 2015; Delling, Pajor, et al., 2009b; Schulz, 2005; Wagner, Willhalm, & Zaroliagis, 2005).

Generally speaking, time-expanded graphs are well suited to model public transportation networks, as they are well studied (Bast et al., 2015; Pyrga et al., 2008; Schulz, 2005), enable solving the *EAP* and *MCP* using basic techniques (Müller-Hannemann & Schnee, 2007; Pyrga et al., 2008), and can be enhanced with multiple speedup techniques (Bast et al., 2013; Delling, Pajor, et al., 2009b; Schulz, 2005; Wagner et al., 2005).

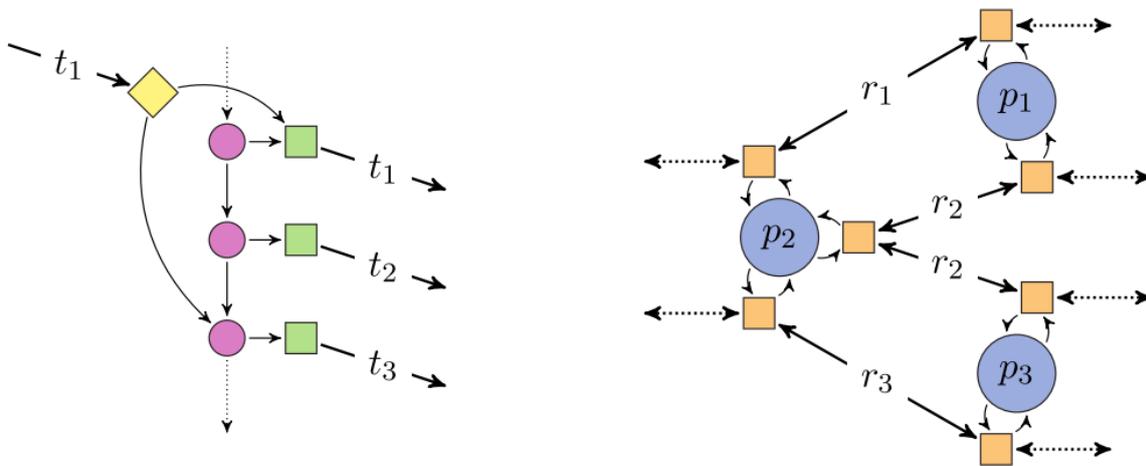


Figure 10 Time-expanded model (left) and time-dependent model (right). On the left, an arrival node (diamond) has a direct edge to a departure node (square) if it is the same train stopping at a stop. An edge to a transfer node (circle) exists, if a transfer to a different train is possible. In the time-dependent model, stops (circles) are directly connected via routes (squares) (Bast et al., 2015).

Time-Dependent Model: Another well-studied model is the time-dependent model. It aims to even more naturally represent a timetable based network (Brodal & Jacob, 2004). Other than the time-expanded model, not every connection in the timetable is represented. An edge between station nodes u and v is implemented if at least one elementary connection exists. Edges are time-dependent, meaning that the exact time information is encoded using piecewise linear functions (Bast et al., 2015; Brodal & Jacob, 2004; Pajor, 2009). For every connection c , an interpolation point $(t_1, \Delta(t_1, t_2))$ needs to be added to the function f of s_1 and s_2 . Evaluating the function at one interpolation point t_i , $f(t_i)$ returns the exact travel time. Further, the evaluation of an earlier point $t < t_i$ results in $f(t_i) + \text{waiting time at } s_1$ (Pajor, 2009). Similar to a simple version of the time-expanded model, the before explained approach does not account for realistic transfers (Pajor, 2009). Pyrga et al. (2008) therefore extended this model by adding additional route nodes. As can be seen in Figure 10, every route node r is connected via two edges to its station node p . The edge (r, p) represents getting off a train which does not require any cost, and is thus weighted by 0. The edge (p, r) however, is weighted by a constant transfer weight

representing the transfer time needed (Pajor, 2009; Pyrga et al., 2008). An even more sophisticated approach explained by Pyrga et al. (2008) and Pajor (2009) enables variable transfer times by interconnecting the route nodes directly. Figure 11 shows that in this case, station nodes are omitted. Additionally, the number of edges grows quadratic (Pajor, 2009). Subsequently, implementing variable transfer times leads to an information loss in the graph. The use of this model in combination with other networks, resulting in a multimodal network is critical, since network merging normally requires stop nodes (Pajor, 2009). Time-dependent models are well suited for public transportation networks as they allow to quickly solve the *EAP* and *RP* (Bast et al., 2015). Further, their space consumption is low, therefore the complexity is moved to the query algorithms (Pajor, 2009).

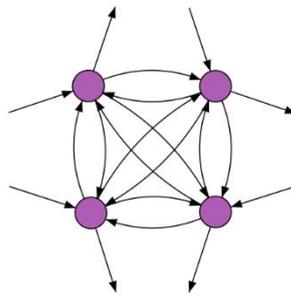


Figure 11 Schematic representation of variable transfer times in a time-dependent model. Route nodes (purple) are directly linked (Pajor, 2009).

Frequency-Based Model: A less common model is the frequency based model proposed by Bast & Storandt (2014). The aim is to represent a certain continuity of a network (Bast et al., 2015). Trams in Zurich, for example, run the same route every 7 minutes during the day and every 20 minutes in the later evening. A frequency-based model therefore compresses tuples of a departure time t , an interval time i and a frequency f into a set. Hence, by computing $t + f_i$ so that $t + f_i \leq t + i$ is true, original departures can be reconstructed (Bast et al., 2015). The base model is similar to a time-dependent graph. Each stop s is represented as a node and if an elementary connection between a pair of stops exist, an edge (s_1, s_1) is added to the graph (Bast & Storandt, 2014). Query times can be enhanced with this model, as many not feasible routes can be pruned (Bast & Storandt, 2014). This modeling approach is a good solution for public transportation networks with stable routes.

2.1.4.5 Specialties of Public Transportation Networks

Uncertainty and Delays: In public transportation, delays and cancellations are a common phenomenon. Delays, for example, can make it impossible to catch a connecting train or bus. It is therefore crucial that a travel planner system must be able to deal with uncertainties. Firmani, Italiano, Laura, & Santaroni (2013) have shown that current timetable routing is not always reli-

able, since they usually do not consider uncertainties such as unpredictable delays. They especially criticize that current systems rely on incorrect estimations of waiting and transfer times. Hence, a method must be found to quickly update network models. An approach on how to allow up-to-date queries is proposed by Müller-Hannemann & Schnee (2009). They developed a prototype which handles a stream of information and adapts a time-expanded model based on the incoming information. The idea is to insert, delete, or update nodes and edges, but also to recompute the necessary ones (Müller-Hannemann & Schnee, 2009). For example, a train that arrives late can make certain transfers impossible. Thus, the affected transfer edges and nodes must be removed from the graph, but also the future departure time of the delayed train should be updated. They further showed that their prototype can update a time-expanded graph extremely quick (Müller-Hannemann & Schnee, 2009). Therefore, time-expanded models seem to be a good solution when dealing with live-stream information.

Delling, Giannakopoulou, Wagner, & Zaroliagis (2008) criticize that the topology of a time-expanded network has to be changed when it is not required in a time-dependent network. In a time-dependent network, the route can simply be traced to the end and the arrival time increased at every station the train stops (Delling et al., 2008). Consequently, no changes need to be made to the underlying network graph.

This section illustrates that uncertainties can be implemented in both time-dependent and time-expanded models with different amounts of efforts. Although the effort needed in time-expanded models is higher due to topology changes, performance is still extremely good (Müller-Hannemann & Schnee, 2009).

Night trains: Another specialty which presumably only affects train networks are overnight connections in the form of night trains. Arriving too early in the morning is probably not desirable for passengers, because they want to get enough sleep. Gunkel, Schnee, & Müller-Hannemann (2011) therefore undertook a study which integrates sleep time in a multicriteria search, with very promising results.

As night trains are a niche product in public transportation networks, these will be not discussed any further in this thesis.

Fares: According to Bast et al. (2015), optimizing paths based on monetary costs is difficult. Pricing schemes differ strongly between agencies and are generally difficult to represent by a mathematical model. Although Müller-Hannemann & Schnee (2006) propose a system called MOTIS which aims to optimize a route by adding a financial cost factor to a multicriteria search, the solu-

tion using fare zones from Delling, Pajor, & Werneck (2014) may be more suitable for urban networks. The idea is to not explicitly use cost as a weight, but computing Pareto sets of journeys which optimize fare zones.

2.2 Multimodal Routing

Multimodal routing is the art of combining multiple networks of different MOT into one single network. Current literature focuses on merging walking, car travel, public transportation and flight networks. In recent studies, bicycling is considered as a useful network as well (Bast et al., 2015). Also, newer MOT such as car sharing and carpooling recently gained some attention (Aissat & Varone, 2015a; Bucher et al., 2017). The idea is to facilitate journeys which use different MOT, but in an optimized way. According to Bast et al. (2015), multimodality is only given if the combined networks differ. As an example, they state that combining only timetable based networks (trains, trams, and buses) do not lead to a true multimodal network since they can be represented as one single schedule. However, they note that this is their personal definition. This definition is further rather technically driven. If it were assumed that trains and buses were not the same MOT, a trip containing both would be considered a multimodal journey. Others use an even more conceptual definition. Bit-Monnot, Artigues, Huguet, & Killijian (2013), for example, do not consider specific MOT at all: “(...) A multimodal transportation network is modeled with an edge-labeled graph $G = (V, E, \Sigma)$ where V is the set of nodes, Σ the set of modes (for instance foot, car or public transportation) and E is the set of labeled edges” (Bit-Monnot et al., 2013:2). Hence, no universal definition exists. As for this thesis, multimodality is defined as a combination of MOT which use different physical networks, independently of their model.

2.2.1 Network Merging and Linking

A key part of multimodal routing is the merging of different networks into one single large multimodal network graph, in which feasible journeys can be calculated. Sequences of different MOT which are not possible (train -> car -> bus) shall be preempted. Ideally, the user should be able to set preferences (Bast et al., 2015).

In a first step, networks of different MOT are modeled independently, meaning that each network is modeled based on its nature. A public transportation network G_{Pub} , for example, is modeled as a time-expanded or time-dependent graph, whereas a road network G_{Road} may be modeled as a time-independent graph (Bast et al., 2015; Delling, Pajor, et al., 2009a; Dibbelt, Pajor, & Wagner, 2015; Pajor, 2009). In the light of the fact that these networks shall be merged, but still be distinguishable, labeling nodes and edges is mandatory (Dibbelt et al., 2015). Nodes and edges of the road network graph G_{Road} might be labeled *road*. In a second step, all independent graphs are unified into one large graph $G_{Multi} = (G_{Road}, G_{Pub})$ (Delling, Pajor, et al., 2009a). It is crucial that this graph G_{Multi} does not, at the moment, offer any transfers between G_{Road} and G_{Pub} . In a

third step, linking edges between $G_{Road} \subseteq G_{Multi}$ and $G_{Pub} \subseteq G_{Multi}$ must be implemented. A common approach is to solve the *Nearest Neighbor Problem (NNP)* (Pajor, 2009). In order to do so, a station node $s \in G_{Pub}$ is connected with the closest node $j \in G_{Road}$. A link edge $e = (s, j)$ is implemented. Delling, Pajor, et al. (2009a) add a threshold for link edges. They determine that the maximum distance between s and j must be shorter than 5 km. Thus they add restrictions to the linking process, in order to prevent inconvenient journeys. Dibbelt et al. (2015) use the same approach, but they further state that additional information from transport agencies could be used to define link edges. The SBB, for example, defines in their schedule⁴ how different stations are linked. If a station b is reachable from station a , information about the time needed to proceed from a to b is given. Since this information was not present for Dibbelt et al. (2015), they defined an average walking speed to calculate the duration needed which they can use as an edge weight of e . Pajor (2009) further specifies that linking networks are asymmetric. In the provided example with G_{Pub} and G_{Road} , it does not make sense to link all nodes from G_{Road} with G_{Pub} , but vice versa. Linking all nodes from G_{Road} with G_{Pub} would lead to an enormous amount of link edges, which makes no sense. The linking process is therefore an asymmetric, directed process.

2.2.2 Routing Techniques

2.2.2.1 Label-Constrained Shortest Paths

In multimodal routing, labeled graphs are used to distinguish between different MOT. Hence, an approach to finding an optimal path is the *label-constrained shortest path problem (LCSP)*. It is an augmentation of the standard shortest path problem (Pajor, 2009). The idea is to compute journeys that obey constraints on the MOT (Bast et al., 2015). A clear definition of the problem is given by Pajor (2009:52):

“Given an alphabet Σ , a language $L \subset \Sigma^$, a weighted, directed graph $G = (V, E)$ with Σ -labeled edges and source and target nodes $s, t \in V$, we ask for a shortest path P from s to t , where the sequence of labels along the edges of the path forms a word of L . Thus given $P = [v_1, \dots, v_k]$ it has to hold that*

$$label((v_1, v_2)) \, label((v_2, v_3)) \, \dots \, label((v_{k-1}, v_k)) \in L$$

A label-constrained shortest path is the shortest path based on a language L given to the query. As a schematic example, given the language [walking, train, flight], the query shall result in an optimal path where a user has to walk to a train which brings him/her to the airport, where he/she can take the airplane. The sole paths walking, train and airplane shall be optimized.

⁴ Retrieved from the SBB schedule in the form of a GTFS feed, provided by <http://gtfs.geops.ch/> (Accessed: 03.03.2017)

In general, an augmentation of Dijkstra's algorithm is used to solve the *LCSPP* (Bast et al., 2015). The augmentation of different algorithms such as the A* as speedup techniques have been studied as well (Barrett et al., 2008; Holzer, 2008).

2.2.2.2 Multicriteria Optimization

Similar to standard routing, multicriteria optimization is an important factor in multimodal routing as well. Bast et al. (2015) argue that even though *label constraints* deliver feasible journeys, they have some drawbacks. Users need to be aware of the transportation networks in order to avoid launching a query for impossible journeys (train – car – train). In addition, journeys that combine different MOT in a different sequence are not considered. Delling, Dibbelt, Pajor, Wagner, & Werneck (2013) have therefore studied how multimodal journeys can be calculated considering three different optimization criterions. In their research, they use arrival time, costs and convenience. Convenience is defined as the number of transfers, walking duration and financial cost for potential taxi rides. Using these criteria, a Pareto set of possible Journeys is calculated. They state, however, that the more criteria one defines, the larger the Pareto set will be. Nevertheless, with their approach, they can present a set of possible journeys to the user, from which he/she can chose.

2.2.3 Carpooling and Multimodal Routing

Carpooling as a MOT in a multimodal routing system has not been researched well in the near past. Even though traditional transportation may provide new aspects of the mobility (Aissat & Varone, 2015a), only two studies exist.

This section describes two different approaches to integrating carpooling into a multimodal planning system.

2.2.3.1 Carpooling as a Substitution

The approach from Aissat & Varone (2015) tries to sequentially substitute parts of a traditional multimodal journey with carpooling, so that it satisfies a passengers request. On a basic level, they use an existing API to retrieve a traditional multimodal path containing, e.g. bus, train, and walking. In a second step, they try to substitute parts of this trip with carpooling. They use the stops from the calculated traditional path to find carpooling offers which can feasibly substitute parts of this trip. A connection from the traditional part is only substituted by carpooling if it is faster, which means the arrival at the next stop the carpooler will stop at must have an earlier arrival time than the traditional connection. Therefore, the waiting time at a pick-up location and the riders' arrival are also part of the time needed for a connection/substitution. Figure 12 from Aissat & Varone (2015a) shows the path from the API (grey) and possible substitutions with carpooling (blue).

Furthermore, they use closeness measures to find suitable matches between users and drivers. By first finding only drivers close to a user based on an orthodromic distance, they can narrow down the set of drivers. Then, they define measures for maximum detours a driver can undertake. They argue that an additional 20% to the quickest path is acceptable. In a third step, they calculate all shortest paths of the matching drivers, also considering the time from the driver's position to the pickup location. Then, they can select the sub path with the most time saving.

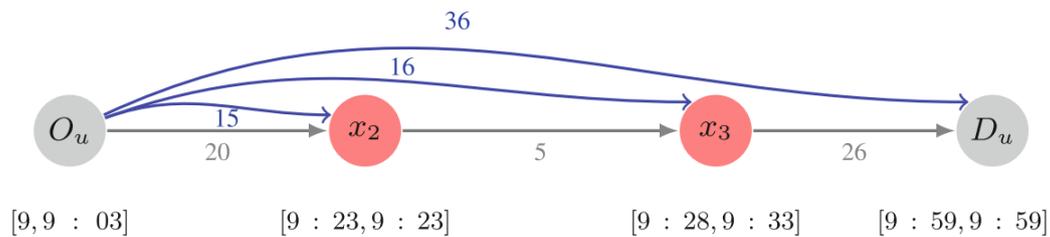


Figure 12 A multimodal path (grey) and possible substitutions with carpooling (blue) (Aissat & Varone, 2015a).

In an experiment, a very small subset of carpooling data with approx. 540 drivers and lengths of carpooling trips between 2 km and 130 km were analyzed (Aissat & Varone, 2015a). As illustrated in the Introduction, this does not correlate with actual carpooling offers retrieved from an extensive international platform such as BlaBlaCar.de. Regarding computational efficiency, the authors show that with an increasing number of available drivers, the computation time increases linearly (Aissat & Varone, 2015a).

The researchers were further able to demonstrate that integrating carpooling into a multimodal routing system, using this approach, is able to reduce the travelling time significantly. Compared to the traditional trips retrieved by the API, 67.3% could be improve using only carpooling and 30.5% could be improved using carpooling as a substitution for a part of the trip. Only in 2.2% of the cases carpooling did not improve the initial trip (Aissat & Varone, 2015a). It is essential to keep in mind, however, that this high match may be somewhat biased since the carpooling offers are spread in a relatively small geographic region, the time of each trip was fixed to a specific time, and depending on the public transportation network of the region of interest, results may differ drastically. As an assumption, in regions or countries where public transportation is highly prevalent, carpooling may be an inferior substitute.

In a theoretical fashion, Aissat & Varone (2015) showed that carpooling can indeed be integrated into a multimodal routing system and may have a large impact, especially for passengers. They also noticed that carpooling is rather a complement to public transportation than a substitution thereof, because carpooling can also access areas where no public transportation is available (Aissat & Varone, 2015a, 2015b; Varone & Aissat, 2015).

2.2.3.2 Two Synchronization Point Shortest Path Problem

The approach from Bit-Monnot, Artigues, Huguet, & Killijian (2013) is based on calculating multiple shortest paths in a multimodal network. They basically use two networks as base for their multimodal graph, namely the public transportation network also containing walking and a road network representing the carpooling network. Rather than explicitly specifying the MOT, they define the first network as the riders' network and the latter as the drivers' network (Bit-Monnot et al., 2013). Their idea is to synchronize both the journeys of the rider and the driver, which is called "the two synchronization point shortest path problem (2SPSPP)". These two synchronized paths can be decomposed into 5 sub paths. The first two paths describe the way from the rider's and driver's individual starting point to a pick-up location, path three is the shared path to a drop-off location and paths four and five are again the individual paths towards the rider and the driver's respective destination. The pick-up (x_{up}) and drop-off (x_{off}) locations are chosen by attempting to minimize the summarized travel cost of both users. Figure 13 from Bit-Monnot et al. (2013) shows the five sub paths.

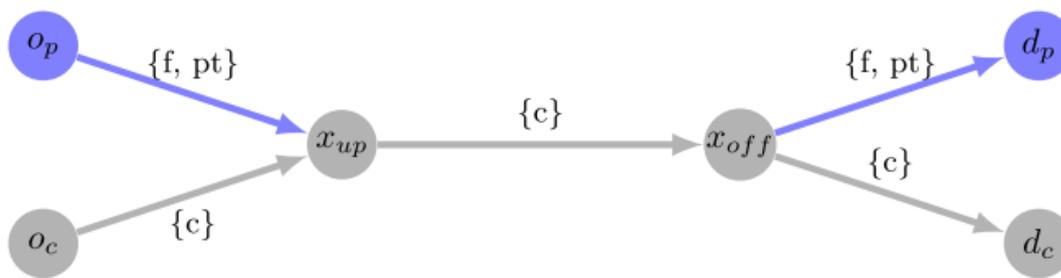


Figure 13 Five sub paths decomposed from the 2SPSPP (Bit-Monnot et al., 2013).

This process shall help reduce the duration of the cumulated path. However, the 2SPSPP approach has some limitations. Bit-Monnot et al. (2013) assume that every driver is willing to drive a detour. Moreover, this detour time is not limited, which can be inconvenient for drivers. Also, the system is not flexible enough, because a specific driver has to be chosen to solve the 2SPSPP. Hence, their system might be based on user needs for a travel planner system, but not be implementable in a practical context (Furuhata et al., 2013; Rehrl, Bruntsch, & Mentz, 2007; Sierpiński et al., 2014).

2.3 Research Gap

Research in routing has progressed significantly since the beginnings in the 1950s with Bellman (1958) and Dijkstra (1959). The application of their algorithms remains of high importance today and they form the foundation for further improvements in routing. With the development of different speedup techniques, queries can be solved in a fraction of time, which makes them viable to solve everyday tasks, such as way finding or finding an optimal train connection.

Not only sophisticated routing algorithms, but also highly developed data models enabled researchers and companies to represent and query any type of transportation network. Based on *graph theory* (cf. Foulds (1992)), three major modeling approaches exist: Time-independent models for road networks, time-expanded (Schulz, 2005) and time-dependent (Pyrga et al., 2008) models for time dependent networks. While the application of time-independent models is clear, the advantages and disadvantages of the latter are still debated, but both are accepted as viable solutions. Nevertheless, the difference between the two time-dependent models is the allocation of complexity. Whereas time-expanded models store complexity inside the graph, time-dependent models shift the complexity to the query, leading to augmentations of the routing algorithms (Pajor, 2009). Consequently, a cookbook for choosing the right model does not exist. Hence, it has to be evaluated which model suits best for a new MOT.

With both highly developed routing algorithms and sophisticated data models, multimodal routing comes to focus. Multimodality enables querying for journeys that combine different MOT in an optimized way. Research agrees on the process of modeling different networks independently and merge/link them afterwards (Bast et al., 2015). Unfortunately, according to different studies, the same approach for linking is used every time (Delling, Pajor, et al., 2009a; Dibbelt et al., 2015). Solving the *Nearest Neighbor Problem (NNP)*, that is to say linking the nearest stops from different networks, seems to be the generic solution. In all conscience, no further merging techniques could be found in previously published literature. Although some authors consider restrictions, for example maximum distance between stops or a static walking speed, the potential flexibility of the contained MOT is not considered (Delling, Pajor, et al., 2009a; Dibbelt et al., 2015).

A reason for the lack of research on merging techniques could be that mostly the same MOT are combined into multimodal networks. Because walking, driving, public transportation and flight networks can be restricted to fixed stop locations, flexible merging techniques are not required.

Considering recent MOT such as carpooling, flexibility seems to be a crucial factor. Generally speaking, carpooling does offer a fixed schedule, but also has the flexibility to drive detours. Consequently, the above-mentioned problem of finding an appropriate modeling approach is present. When considering carpooling as a part of a multimodal network, not only the base model, but also

the merging technique must be chosen adequately. The current state of the art in solving the *NNP* seems to be inappropriate, as it does not consider flexibility on a high level.

Up to this day and in all conscience, only two studies about carpooling in multimodal routing exist (Aissat & Varone, 2015a; Bit-Monnot et al., 2013). Even though they aim to show the usefulness of carpooling as a substitution, they do not discuss carpooling as a full-fledged, integrated part of a multimodal network. The approach chosen is to substitute parts of an already existing multimodal trip with carpooling. Consequently, a multimodal path with carpooling can only be calculated if a traditional multimodal or unimodal journey exists. It does not seem to be possible to bypass areas where no other MOT exist with carpooling. Therefore, carpooling is only considered as substitute and not as a full-fledged part of the multimodal network. This circumstance can potentially lead, which is also implied by the authors, to inefficient queries in large data sets, as they consist of multiple stages (traditional path – carpooling path – finding match). It is not clear why the above quoted researchers consider carpooling only as a substitute. An evaluation of and reasoning with existing data was not presented.

The research gap can be found in the areas of carpooling as a part of a multimodal network. Current merging/linking techniques are not able to fully represent the flexibility of carpooling. Further, the role of carpooling in a multimodal network is not clear. Only studies exist which treat carpooling as a substitute and not as a fully-fledged part.

3 Research Objectives and Overall Methodology

Based on the research gap identified in chapter 2.3, the following major research objective of this master's thesis is defined:

3.1 RO 1: Merging & Linking Carpooling with Public Transportation

An MOT showing characteristics of multiple network types, in this thesis carpooling, shall be investigated. The fuzziness and flexibility of carpooling requires special graph merging techniques.

3.1.1 RO 1.1 Characteristics of Real-Life Carpooling Offers

The characteristics of real-life carpooling offers shall be investigated in terms of quantity, temporal scale, and spatial scale to assess the benefits of carpooling in a multimodal routing system.

3.1.2 RO 1.2 Modeling Carpooling

RO 1.2: In the light of a future integration into a multimodal network, a model which allows retaining flexibility and fuzziness has to be elaborated for carpooling.

3.1.3 RO 1.3 Merging & Linking

RO 1.3: Network merging/linking techniques have to be developed, providing a multimodal network which:

- a. Retains flexibility and fuzziness of the contained MOT networks
- b. Improves the overall network
- c. Is able to calculate useful multimodal journeys using current shortest path algorithms

3.2 Workflow

In this sections, the workflow of this thesis is defined. Three major stages are elucidated in order to answer the above stated research objective.

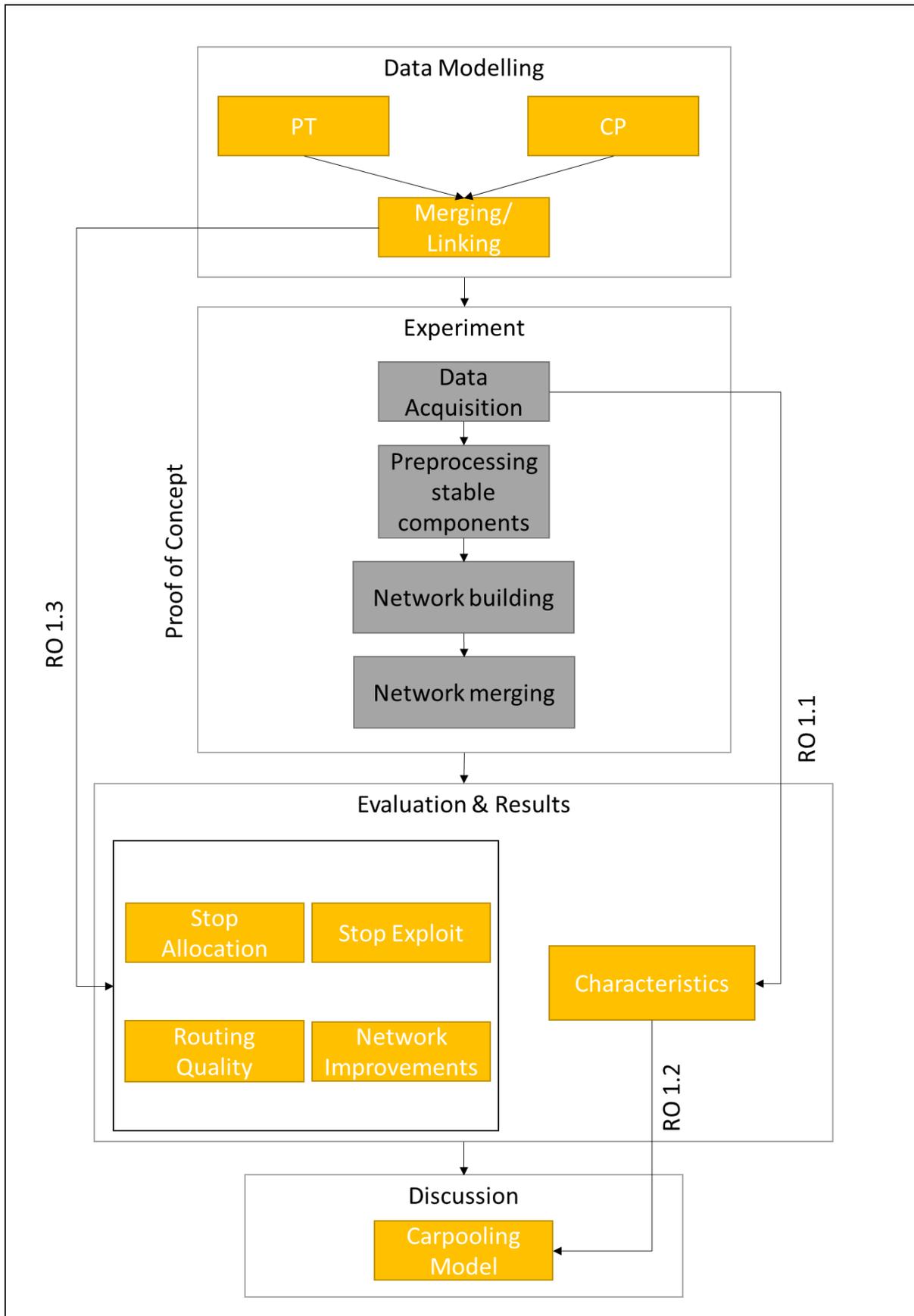
In a first stage, modeling approaches for public transportation and carpooling networks are elaborated on a conceptual level. In consideration of a future multimodal network, adequate models will be distinguished based on the current state of the art. Carpooling, as a novel MOT in multimodal networks, receives additional attention. Further in the first stage, a novel merging technique is proposed, which considers flexibility and fuzziness of an MOT.

In a second stage, an experiment for the modeling and merging techniques proposed in stage one is set up. The experiment encompasses crawling for real-life carpooling offers and the implementation of an experimental multimodal routing system.

The implementation of a multimodal routing system shall be the baseline for delivering proof of concept. The realization of this system follows the current state of the art in multimodal routing. Firstly, stable components of the networks are preprocessed. Secondly, different networks are modeled separately and finally merged and linked using the proposed merging technique.

The third stage aims to evaluate and present results derived from the experimental setup. The crawled carpooling data is investigated in terms of quantity, spatial scale and temporal scale, considering RO 1.1. Further, the resulting multimodal network is evaluated, focusing on RO 1.3. The effectiveness of the proposed merging and linking technique is elucidated based on the stop allocation of carpooling stops, the exploit of new stop locations, the overall network improvement and the ability of querying for meaningful multimodal shortest paths.

Lastly, RO 1.2 regarding the modeling of carpooling is discussed in the discussion chapter based on the characteristics and properties derived from the crawled carpooling data. The flow diagram below shows the workflow in a graphical form:



4 Data Modeling and Graph Merging & Linking

A multimodal routing system combines different modes of transportation in a single application which allows a user to find the shortest routes from A to B using different modes of transportation (MOT) (Bast et al., 2015). A multimodal network thus consists of several connected sub-networks from different MOT. Generally, these sub-networks can have different properties, which leads to quite different modeling approaches and challenges in merging them. This also partly applies to the networks considered in this thesis, public transportation and carpooling. Therefore, the models for both networks are explained separately. Since the model for public transportation is already well-known, existing research with only small adaptations can be relied on, in contrast to carpooling, where no real modeling approach exists. Carpooling has properties from both public transportation and the road network and is therefore bound to some sort of fuzziness and flexibility. On the one hand, carpooling offers also have predefined start and end locations similar to e.g. train stations. However, while train stations are at a fixed geographic location, carpooling offers only specify the origin and destination city location, hence providing fuzzy location information. On the other hand, carpooling uses the road network, which enables changing the route on the fly, hence heading to previously unspecified stops. Thus, carpooling also entails flexibility.

Thus, this thesis aims to elaborate an adequate model for a carpooling network and a meaningful merging technique which considers fuzziness.

In order to create a multimodal network, it is essential to merge/link all MOT in an adequate way. Thus, for consistency reasons, the same metric for edge weights has to be applied. Then, and only then, is it possible query the network for multimodal connections. As already mentioned in the Related Work chapter, current approaches use *Nearest Neighbor Algorithms*, which are well suited for large datasets. However, fuzziness and flexibility cannot be easily considered. Therefore, a different method for merging the public transportation and carpooling networks is presented.

When describing modeling (transportation) networks, the application of the Graph Theory is indispensable. Thus, in the following, models will be explained using graphs.

4.1 Public Transportation (Railway) Model

This section elaborates on the model used for the public transportation network. Although different possible models exist, one specific model is focused on, namely an adaption of the realistic time-expanded modal. A time-expanded model adds complexity to the graph, rather than to the algorithm. This however leads to increased graph sizes, but also to the possibility of using standard routing algorithms such as Dijkstra's shortest path (Bast et al., 2015; Delling, Pajor, et al., 2009a; Pajor, 2009; Pyrga et al., 2008). As this thesis does not focus on either graph size or

speedup techniques for path queries, a time expanded model is useful, since it represents the complete network containing time information as a graph. Although a time expanded model is not natively time dependent, it considers the complete integration of a timetable with its timestamps and fixed edge weights and therefore adapts a certain time dependency. Again, a benefit of a time-expanded model is the simple adaption of Dijkstra’s shortest route algorithm (Dijkstra, 1959). As Pajor (2009) mentions, a disadvantage of this model is the fact that the target node and therefore the arrival time is unknown. It will be demonstrated that this can be bypassed with a slight adaption to the model and the possibility of different query methods (cf. 5.5 Querying). In real time-dependent modeling, edge weights are not static values, but a function. However, transfer times can be static. The size of a graph can therefore be reduced drastically. The decision for a time-expanded model is based on the following arguments:

- The complexity of a transportation network is represented in the graph (cf. Pajor (2009); Schulz (2005)). This fact is later needed for the proposed merging technique.
- Routable with basic routing techniques (cf. Bast et al. (2015))
- Immune to the non-overtaking *FIFO* property (cf. Pajor (2009))
- Efficient updating (cf. Müller-Hannemann & Schnee (2007))

Public transportation in general relies on a timetable that is used to find connections. A timetable contains of a set of Stations s ($s \in S$), a set of trains, buses, etc. z ($z \in Z$), a set of times t ($t \in T$) and a set of connections c ($c \in C$) (Pajor, 2009). In a timetable, t is represented as a timestamp.

A connection c is a tuple of $c =: (z, s_1, t_1, s_2, t_2)$. A train z starting at a specific location s_1 at a specific time t_1 and arriving at a location s_2 at a time t_2 . Thus, the travel time of a connection is simply the difference of $t_2 - t_1$. A connection has no stops in between (Pajor, 2009). A connection c can therefore also be interpreted as a leg of a trip $l \supset c$.

A trip l can be described as a tuple $l =: (c_0, \dots, c_n)$. It is to define, that in a trip l , every connection c has the same train, bus, ... z . Thus, a trip represents the whole travel of a certain train, bus, etc. Conceptually, therefore $z = l$ can be defined.

The existence of a route/line r is not essential for a timetable, but can be represented in the model. In such a case, r consists of multiple equal trips $r = (l, \dots, l_n)$. Hence, a route is time independent and only the contained trips are time dependent.

To clarify the above definitions, we illustrate a fictional example: The public transportation Agency “PubTrans” offers journeys from New York via Washington DC to Miami without any stops in between. Hence they offer the route $r_{NY-DC-MIA}$. There are four trains a day undertaking these

routes. Hence, four trips l exist. These trips l are time-dependent, since they leave at specific times. On each trip, a train stops in DC. Thus, a trip contains of two legs or connections c (NY-DC and DC-MIA). If the first train leaves at 10:00, arrives at 14:00 in DC, leaves there at 14:05, and finally reaches Miami at 23:00, the connections are defined as $c_1 =: (Train1, NY, 10:00, DC, 14:00)$ and $c_2 =: (Train1, DC, 14:04, MIA, 23:00)$.

The time-expanded model is now applied to the above definition of a timetable pair c . At this point, the following nodes and edges are determined, as also shown in Figure 14:

- Station node s : A station node belongs to a set of station nodes S , $s \in S$. S holds information about a specific stop, e.g. name, platforms, geographic location.
- Time node t : A time node t defines both, the departure and arrival time of a train z at a stop s . Thus, t is a tuple $t =: (departure, arrival)$. Further, t is part of a connection c and therefore only valid for a specific z . The property $t_{departure} > t_{arrival}$ is mandatory.
- Edge (s, t) : The time node t is connected to a specific stop s , whereas (s, t) does not store additional information.
- Edge (t_1, t_2) : The edge between two time nodes t_1, t_2 represents the actual connection or a leg of a trip. This edge is weighted by a certain metric. As written above, the duration is used as weight.
- Trip node l : A trip node l represents a set of connections C and can be seen as a train z . From now on, we use l instead of z . Therefore, a trip is time-dependent.
- Edge (t, l) : Marks the affiliation of a time t_n to a trip l . Hence, each trip l has edges with multiple t . The edge itself does not store any additional information.
- (Optional) Route node r : A route node represents a set of trips l .
- Edge (r, l) : Marks the affiliation of a trip l to a route r . Hence, each route r has edges with multiple l . The edge itself does not store any additional information.

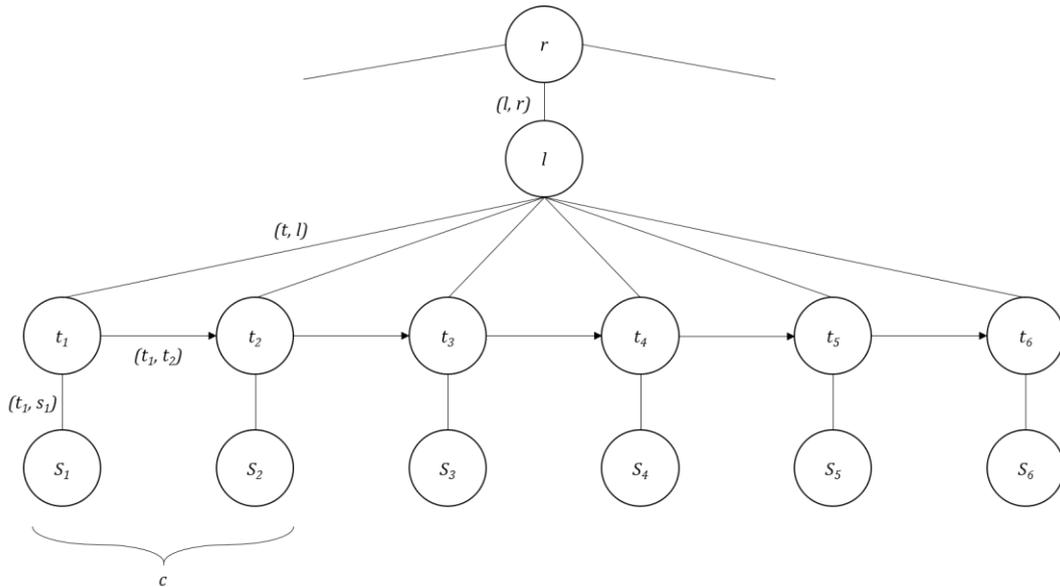


Figure 14 Schematic representation of a trip (l) composed of a sequence of times (c) belonging to a route (r).

With the above nodes and edges, a simple model can be described. This model works well for single ride journeys as trips are presented separately. Furthermore, shortest paths can be easily retrieved using Dijkstra's algorithm. Modelling transits need further adaptations to the model. Other than Pajor (2009), we do not implement an additional transit node, but an additional edge between time nodes of different trips, t_a belonging to l_a and t_b belonging to l_b lead to an edge (t_a, t_b) (cf. Figure 15). Similar to (t_1, t_2) edges, the transit edge is weighted by the duration of t_b departure - t_a arrival. Using the same metric allows the use of Dijkstra's shortest path algorithm. Apart from the advantage of being able to use Dijkstra's algorithm, this model approach also has a disadvantage. Since there is no new node implemented which can be detached from any trip, the problem persists that this transit relation could be interpreted as actual part of a trip. It is therefore crucial that an additional property be applied. The use of a property or labeled graph can help solve this issue by adding a transit `true` property or by labeling this type of edges. Furthermore, threshold values can be defined when creating an edge between t_a and t_b , e.g. create edge (t_a, t_b) if $3min < t_b$ departure - t_a arrival $< 10min$. Thus, valid transits can be defined via a conditional. This conditional also defines transfer times in a static manner; the maximum transfer time will be 10 minutes from platform a to platform b. By splitting up every stop into child stops that define the platform, variable transit times can be implemented. A meta-stop m is implemented and related with child stops s representing platforms. Further, an edge (s_1, s_2) between every child stop s is created representing the variable transfer time (cf. Figure 15). The conditional for creating (t_a, t_b) can consequently be retrieved from (s_1, s_2) . Implementing variable transit times adds much complexity to the model. It is to mention, that the edge (s_1, s_2) must inconclusively be stored in the graph itself. A solution can be to store pairs of child stops and their minimum transfer times in an external database or even a text file.

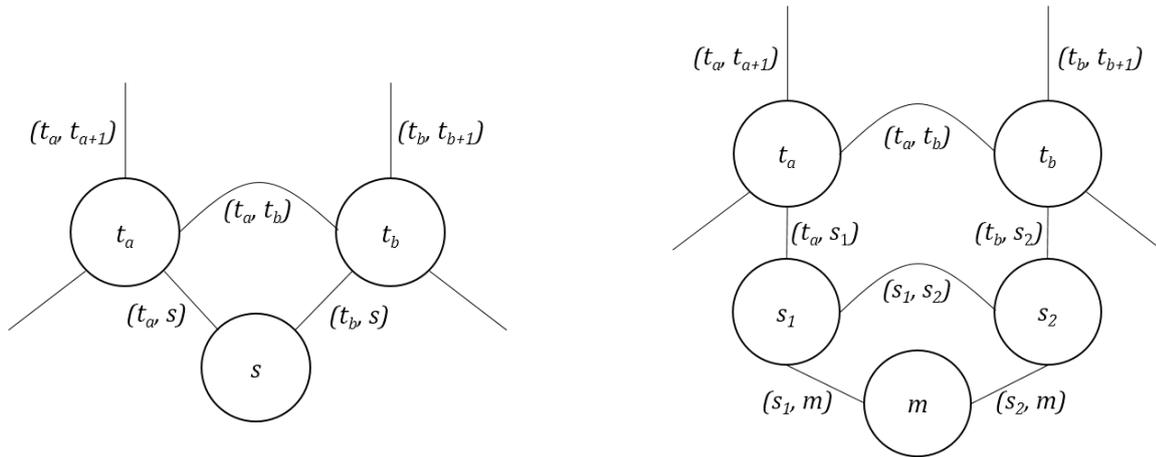


Figure 15 Schematic representation of static transit times (left) and variable transits (right).

Implementing a meta-stop m not only helps representing variable transfer times, it also adds some sort of intuitive simplicity to the graph. Furthermore, larger train stations may have platforms far from each other and thus not share the same geographic location; some platforms may even have a different name (e.g. Bahnhof Löwenstrasse is a sub-station of Zurich HB).

In this section, the model for a public transportation network has been elaborated. It has been shown that the use of a time-expanded model with slight adaptations in terms of transit edges and meta nodes may be an adequate way of representing public transportation as a graph, since it represents the whole schedule as a graph. A time-expanded model further adds the complexity of a transportation network into the graph, rather than shifting it to the query (cf. 2.1.3.2 & 2.1.3.3). This allows the use of basic techniques/shortest path algorithms. Further, the implementation of meta-stops helps differentiating between platforms and sub-stations of a train station.

4.2 Carpooling Model

Modeling carpooling data is difficult since it shows characteristics from both a timetable-based network like public transportation and a “free” road network. Hence, there will be no “right” way to model a carpooling network. In fact, the type of model should be designed according to the nature of the MOT. Carpooling, for example, has a sort of a timetable, but still uses the time-independent road network. In this section, two possible approaches are explained, but the focus will be on a timetable-based model.

Carpooling can be explained similarly to a timetable. Typical carpooling consists of a set of stops $s (s \in S)$, a set of drivers $d (d \in D)$, a set of times $t (t \in T)$, and a set of connections $c (c \in C)$ where $c =: (d, s_1, t_1, s_2)$. It is important to note that carpooling usually has no geographically fixed pick-up and drop-off locations (cf. 5.1.2 Carpooling Data), which is a major difference to public transportation. However, the proposed model disregards this fact, as it is only relevant for future network merging/linking.

In contrast to a typical timetable, there is no arrival time t_2 in c . Without prior routing, t_2 is unknown. Further, a set of c can be described as trip l . The occurrence of routes R is not typical for carpooling, since no fixed schedule exists. However, they might occur if a certain trip reoccurs on a regular basis. Then, $r \in R =: (l_0 \dots l_n)$ can be assumed.

The lack of an arrival time leads to the first approach:

4.2.1 Carpooling as a Road Network

Considering the fact that the exact route is unknown, we also have no information about edge weights and times T . Thus, a legible approach seems to be modeling a road network instead of the carpooling offers itself. Thus, the offers are not represented in the network graph itself, but will be routed on demand. The model for the road network is straightforward. Each junction j is a node. Consequently, (j_a, j_b) represent an actual street. Edges will only be built if a street exists and two way streets will be represented by two edges in opposite directions (Pajor, 2009). The edge weight on (j_a, j_b) can either be distance or duration, whereas duration can be assumed more adequate when looking further to a multimodal network. Moreover, since the aim is to obtain information about T , duration is appropriate. With the above statements, a time-independent road network is formed. As a timetable exists in carpooling, time also need to be represented in the road network. Furthermore, stop locations exist.

Therefore, stop nodes and departure nodes can be placed onto the graph. Edges for (t, s) and (t, j) are created. This however, already requires merging techniques. A spatial location on s and j is required. Then, a closest point or nearest neighbor algorithm can be applied. This results in a graph that can be routed with simple shortest path algorithms. Further, real-time information like the traffic situation or available detour time can be used to adjust the route based on the current situation.

Unfortunately, this approach has some disadvantages as well. Trips cannot be represented in the network graph easily, since the route has to be recalculated with every query when traffic information shall be considered. Caching could be applied, which however is not part of this thesis. Furthermore, using the road network instead of the carpooling's timetable may lead to increased query times. Creating a graph for a road network leads to, based on the scale, a huge graph. Querying for shortest routes can therefore be inefficient. Furthermore, when aiming for a multimodal network, trying to decrease the network graph is desirable. It also prevents a high flexibility in carpooling, as routes can be recalculated on the fly. Even though the graph would be routable for every possible journey, carpooling offers are strictly bound to start, via and end nodes. It is not desirable to change a carpooler's offer. Because of the above reasons, this thesis follows a timetable-based model for carpooling.

4.2.2 Carpooling as a Timetable

As mentioned above, carpooling has characteristics of both a road network and a timetable-based network such as public transportation. Hence, it is possible to model carpooling the same as public transportation. However, if done so, carpooling will lose its flexibility because it will be modeled as a static network. Nevertheless, time-expanded models are well suited for dynamic networks as they can be updated in a straightforward way, retaining correctness (Müller-Hannemann & Schnee, 2009). Using sophisticated spatial merging techniques during the creation of a multimodal network can restore this flexibility and help represent fuzziness. This will be explained in the following sections.

As a short recap of carpooling's structure, there follows a list of nodes and edges:

- Stop node s : A stop (start, via or end location) node belongs to a set of station nodes S , $s \in S$. S holds information about a specific stop, e.g. name, geographic location. The geographic location may be fuzzy.
- Time node t : A time node t defines the departure of a driver d at a stop s . Further, t is part of a connection c and therefore only valid for d . t exists only at the starting location of a carpooling offer.
- Edge (s, t) : The time node t is connected to its starting location s , whereas (s, t) does not store additional information.
- Trip node l : A trip node l represents a set of connections C and can be seen as driver d . From now on, l is used instead of d . Therefore, a trip is timed pendant.
- Edge (t, l) : Marks the affiliation of a time t to a trip l . Hence, each trip l has edges with multiple t . The edge itself does not store any additional information.
- (Optional) Route node r : A route node represents a set of trips l .
- Edge (r, l) : Marks the affiliation of a trip l to a route r . Hence, each route r has edges with multiple l . The edge itself does not store any additional information.

In contrast to public transportation, important information for a complete connection and trips are not available. Carpooling does not directly provide the information of arrival times at via or end locations, which should be derived using routing algorithms on a road network. Although this seems to be crucial for a carpooling offer, because the duration may vary greatly depending on the traffic situation, it is problematic for a transportation network. In order to be able to use shortest path algorithms, edge weights are needed. For a timetable-based network, these edge weights usually represent the duration of a certain connection c . Thus, not only the departure but also the arrival time at a specific stop is necessary. Therefore, for now a fictional node t_f for the arrival time at a stop s is implemented. In this conceptual model the fictional time node t_f holds no time

information and simply acts as a placeholder. When creating a working example, time information for t_f has to be derived by routing the offer on a road-network. Thus, t_f can be fed with time information, transforming it to a non-fictional time node t . Consequently, the tuple for a connection looks as follows: $c =: (d, s_1, t_1, s_2, t_{f2})$. Now, the edge (t_1, t_f) can be created between stop times and represent the duration of a connection: $duration/edge\ weight = t_{f2\ arrival} - t_1\ departure$.

The following new node and edge of the carpooling model are implemented:

- Time node t_f : A time node t_f represents the fictional arrival and departure time at a stop node s . The property $t_{f\ departure} > t_{f\ arrival}$ is mandatory.
- Edge (t_1, t_{f2}) : The edge between two time nodes t_1, t_{f2} represents the actual connection or a leg of a trip. This edge is weighted by a certain metric. As written above, the duration is used as weight, which in this case is $t_{f2\ arrival} - t_1\ departure$.

After introducing the fictional node t_f and the resulting edge (t_1, t_{f2}) , a full carpooling model can be introduced. A typical carpooling journey with one via point s_2 can be seen in Figure 16.

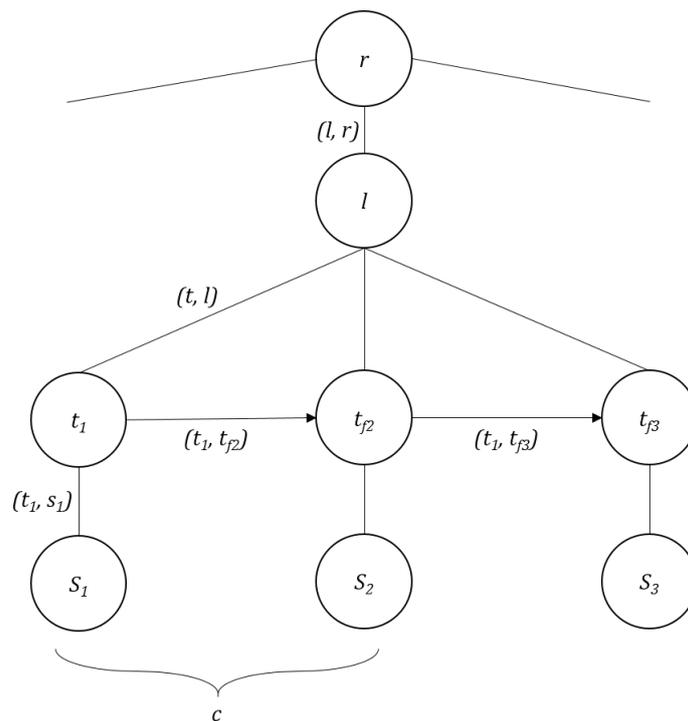


Figure 16 Schematic representation of a carpooling trip (l) composed of a sequence of connections (c) belonging to a route (r). Arrival and departure times at future stops are unknown. Hence a fictional stop time t_f is set.

Considering transit times, the same procedure applies as for public transportation, namely static transit times, restricted by conditionals. An additional edge (t_a, t_b) between two trips a & b is created, if a certain conditional is *true* (cf. Figure 11).

In this section, a routable network model for carpooling was proposed. Fictional nodes were implemented to interpolate the lack of time information along a journey. This network can be queried using standard routing algorithms such as Dijkstra's shortest path. It is crucial that in order to finalize a carpooling model in a real-life scenario, preprocessing has to be done to retrieve information for t_f . Nevertheless, the base model helps to understand the overall nature of a carpooling network. Regarding flexibility and fuzziness, it could not be shown how fuzziness affects the model above, which however is a desired outcome. Fuzziness shall not be included in the network graph on a single network since it would add severe complexity and may be data dependent. Fuzziness on a single network can be represented using a specific query. Hence, the complexity will be moved from the graph to the query. When it comes to multimodal routing, a certain fuzziness in the network may be required, especially if not all MOT have a fuzzy nature. Thus, the following section focuses on network merging and implementing fuzziness in a multimodal network graph.

4.3 Model Merging & Linking

Networks of different MOT need to be connected to each other in order to create a multimodal network. This merging is required in order to query for multimodal journeys (Pajor, 2009). Stops of different networks are connected/linked to each other based on their geographic location. Hence, a stop a of network A is linked to stop b of network B if they are closest to each other. This is also known as the *Nearest Neighbor Problem (NNP)*. Solving the *NNP* is a common approach when connecting different MOT in order to retrieve a multimodal network (Bast et al., 2015; Delling, Pajor, et al., 2009a; Dibbelt et al., 2015; Pajor, 2009). An important advantage is the presence of highly efficient algorithms for solving the *NNP* in big datasets (Pajor, 2009). This, however, requires exact stop locations, which carpooling offers does not specify. A different approach must thus be used in this sense. Furthermore, carpooling offers are of a fuzzy nature. When connecting a public transportation and a carpooling network, not only stops have to be connected, but also routes to stops. Even though a carpooler has not specified a potential stop location, he could still stop at a public transportation stop if he passes it anyway. The basic idea of connecting the two networks is to integrate carpooling into the public transportation network, which means that carpooling will, for example, adapt public transportation stops. Thus, in the following, the models described in the former sections are outlined:

- Public Transportation (Railway) Network: The Railway network is modeled based on its timetable. The use of a time-expanded model enables the use of simple shortest path algorithms. Additional meta-stops have been integrated to differentiate between child stops. The public transportation network will be denoted as N_{PT} .
- Carpooling (timetable-based) network: After considering two approaches, the focus is laid on a timetable-based approach. Similar to public transportation, a time-expanded model has been chosen. Missing stop times have been substituted with fictional time nodes. Flexibility and fuzziness could not have been integrated into the model itself. The carpooling network will be denoted by N_{CP} .

As argued above, the single carpooling network does not consider fuzziness or flexibility. In the following sections, a merging technique which restores flexibility and considers fuzziness around stops and along the actual route is presented.

4.3.1 Spatial Allocation of Carpooling

Carpooling offers do not leave from an exact location. According to several websites offering carpooling (BlaBlaCar.de, e-carpooling.ch, carpoolworld.com), it is common to only mention the city an offer starts from or arrives in. This inaccuracy is problematic for a multimodal routing system. In a multimodal routing system, different modes of transportation are connected in any way. This inaccuracy furthermore aggravates the solving of the *NNP*. Hence, if carpooling does not have an exact stop, it is not clear how and to which public transportation stop a carpooling offer can be connected. In case of connecting a carpooling stop to just one public transportation stop in the same city, the flexibility of carpooling is lost, because theoretically, the driver could drop off the passenger at any stop within the city. However, connecting a carpooling stop to every public transportation stop within the same city does not seem to be ideal, either. It can be inconvenient for the driver if he/she has to drive a longer detour than he/she wants to. As prerequisite, a more or less accurate geographic location of a carpooling stop must be retrieved through geocoding. Further explanations will follow in the Data Enrichment chapter. It is important to further note that the following approach would also work if carpooling stops come with exact locations.

A legible solution to connect carpooling with public transportation is to analyze which public transportation stops could potentially be reached by a driver. On the one hand, drive time areas around a carpooling route could be calculated, in order to check how far a carpooler can deviate from the original route. On the other hand, one could implement drive time areas around public transportation stops and check if a carpooling stop lies within a drive time area. The former way requires the calculation of drive time areas for every route, while the latter calculates drive time areas for considerably stable public transportation stops.

Thus, drive time areas $a \in A$ are implemented around public transportation stops S_{PT} . Rather than just finding the nearest neighbor of a geocoded carpooling stop S_{CP} , a set of $S_{PT} \in S_{PT}$ are determined. When $S_{CP} \subset a$, then $S_{PT} = \{ S_{PT} \in a \}$. Since meta-nodes $m \in S_{PT}$ for public transportation stops were implemented, S_{PT} can be reduced and only the meta-stops m of S_{PT} retained. Then, an edge (m, S_{CP}) for every $m \in S_{PT}$ can be created. Figure 17 shows a case where $S_{CP} \subset a_1$ and $S_{CP} \subset a_2$. Consequently, edges for (m_1, S_{CP}) and (m_2, S_{CP}) are created. This linking results in a unified set of stops $S_G = S_{PT} \cup S_{CP}$.

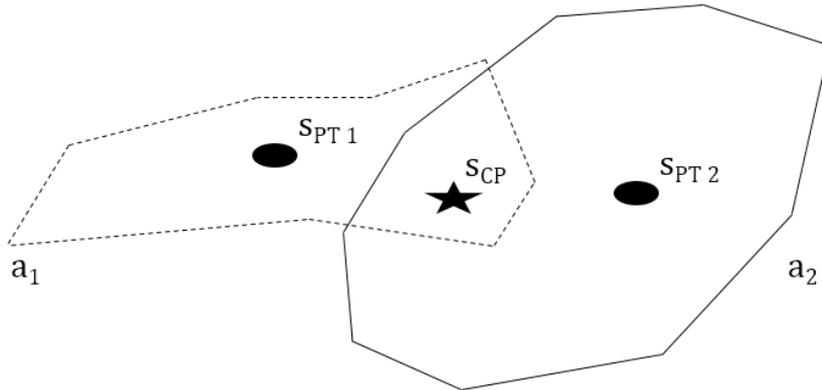


Figure 17 A case, where a carpooling stop is contained by two drive time areas of public transportation stops.

Based on the above statement, a new edge is identified:

- Edge (m, S_{CP}) : The edge between a carpooling stop S_{CP} and a meta-stop from public transportation m is used to connect both the carpooling and the public transportation network. The edge stores no additional properties.

Figure 18 shows the link operation in a graphic form. It can be seen that S_{CP} must have been contained by a_1 from m_1 and a_2 from m_2 . Hence, edges (m_1, S_{CP}) and (m_2, S_{CP}) have been created. S_{PT} show child stops from public transportation, which have already been related to m_1 or m_2 .

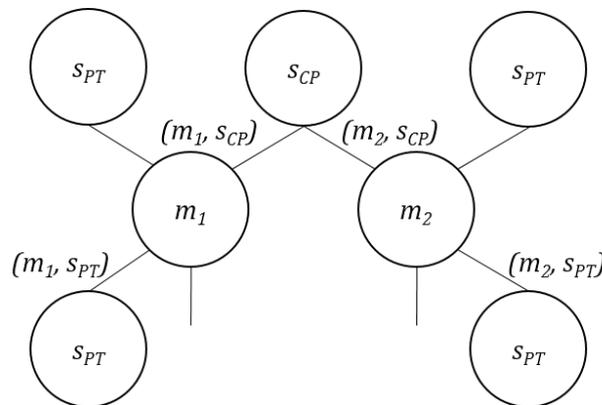


Figure 18 Schematic representation of a carpooling stop S_{CP} linked to meta-stops m_1, m_2 of the public transportation network.

Since carpooling stops can be connected to multiple public transportation stops within a reachable area, the fuzziness of the inaccurate geographic locations of the origin, destination, and via locations defined by the carpooler can be represented. Furthermore, this leads to a more flexible system, since transits from public transportation cannot only be at a specific stop, but at multiple different stops. This opens the possibilities for quicker journeys.

4.3.2 Creating Transits

By merging carpooling and public transportation, unified set of stops $S_{PT} \cup S_{CP} = S_G$ was defined. Similar to a single network, trips of both MOT can be linked to represent mode changes. Using conditionals, it can be defined when a transit from carpooling to public transportation is appropriate. Hence, an edge (t_{CP}, t_{PT}) is created if a certain conditional, e.g. $3\text{min} < \text{transit time} < 10\text{min}$, is *true* (cf. Figure 19).

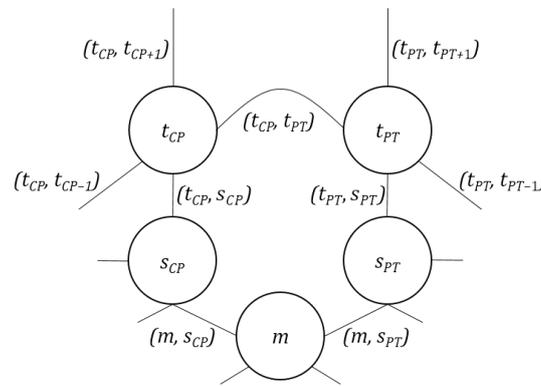


Figure 19 Schematic representation of a transfer between carpooling and public transportation. If a given conditional is true, a transfer edge (t_{CP}, t_{PT}) is created.

The condition of $t_{\text{departure}} > t_{\text{arrival}}$ must be true in every case, since a transfer to a train earlier than an arrival is not possible. Further, the edge weight can be calculated by $t_{PT \text{ departure}} - t_{CP \text{ arrival}}$ and vice versa, depending on the arrival and departure time. Again, variable transit times are not implemented. However, different transit times from carpooling to public transportation and vice versa can be implemented during the linking process. A different conditional than for single networks can be chosen. Additionally, and according to (Bast et al., 2015), penalties for a mode change can be implemented by simply increasing the conditional for a change from carpooling to public transportation.

4.3.3 Representing Fuzziness and Linking Along Carpooling Routes

Carpooling has the ability to be more flexible than public transportation because it is not bound to railway tracks or predefined routes. Hence, a carpooler can drive detours which exploit new stops/destinations that are not directly on the planned route. This flexibility, however, has some drawbacks. Even though the starting point, the destination, and maybe some via points are known, the exact route a carpooler is driving is not mentioned in an offer. Thus, it is inevitable to pre-

calculate a route for a real-life system. Further explanations follow in the Carpooling Data chapter. However, the conceptual basis can be explained. Concerning the fact that drive time areas A have been used to merge carpooling and public transportation stops, A can also be used to exploit potential stops s_f along a carpooling route r . Similar to the stop merging in the previous section, rather than relating s_f to a specific s_{pt} stop, an edge (m, s_f) is created between the meta-stop m of s_{pt} and s_f . Consequently, a new stop on r has been exploited which also belongs to the set of stops S_G of the overall graph. Figure 20 shows a case where r crosses a_2 but not a_1 . Thus, an edge will be created between s_f and the meta-stop m of s_{pt2} .

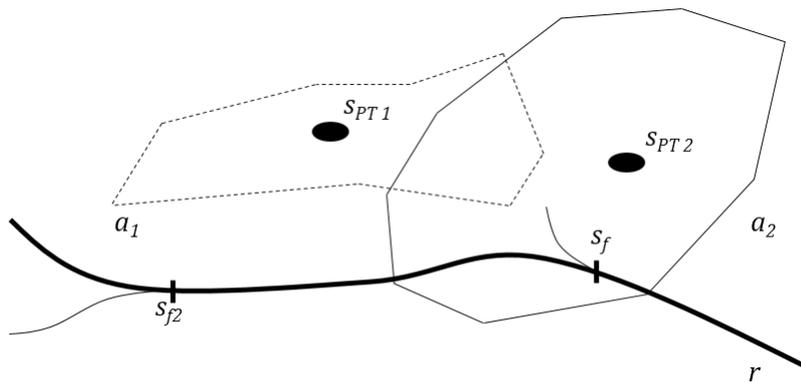


Figure 20 A case where a carpooling route r crosses drive time area a_2 of a public transportation stop s_{PT2} . s_f indicates a point along r where the driver needs to take an action, e.g. leave highway or stay on highway.

It is important to mention that s_f along a route r cannot be chosen randomly or by the closest Euclidean distance between r and s_{PT2} , as s_{PT2} might not be reachable from s_f . For example, if r would be a highway crossing a_2 without a ramp in a_2 , a driver would not be able to reach s_{PT2} . Thus, a way is needed to only exploit s_f which allows a connection to s_{PT2} . A feasible solution is to use *Points of Action (POA)*. In this thesis, *POA* are defined as points on the road network where a user has to take an action. This might be crossings, junctions, or highway ramps. If a user does not have to take an action, there will not be a *POA*. Consequently, the approach is to find intersections between *POA* of a route with drive time areas of public transportation stops. This further guarantees that a specific public transportation stop is reachable. Other *POA* along a route, as shown in Figure 20 as s_{f2} , which do not intersect with a drive time area, can be omitted.

Exploiting new *POA* stops along a route which guarantee a connection to a public transportation stop, require the implementation of time nodes at s_f . Thus, the original connection $c_0 =: (d, s_{CP1}, t_1, s_{CP2}, t_2)$ of a trip are split up into two new connections $c_1 =: (d, s_{CP1}, t_1, s_f, t_f)$ and $c_2 =: (d, s_f, t_f, s_{CP2}, t_2)$. Figure 21 shows the structure of the graph after applying the new connections. Therefore, the following new nodes and edges can be identified:

- *POA* stop node s_f : This node represents a newly exploited *POA* stop along a carpooling route.
- Edge (m, s_f) : This edge represents the connection of s_f to a meta-stop m of public transportation. Hence, $s_f \in S_G$.
- *POA* time node t_f : This node represents the arrival and departure time at the newly exploited *POA* stop s_f .

Missing time information at a *POA* stop s_f can be calculated in a straightforward way. The time at a *POA* stop s_f is simply the departure time at its predecessor plus the duration of the segment (s_{f-1}, s_f) . In this case, s_{f-1} might also be the origin or a via point. Therefore, $t_f = s_f \text{ departure} + \text{duration}(s_{f-1}, s_f)$. As the route of the carpooler is not adjusted, arrival and departure times at later stops do not change because of the insertion of s_f .

The concept of *POA* has further benefits. Since *POA* are located at crossings, junctions, and other points on the road network, different routes may exploit the same *POA* if they are passing the same crossing. Thus, *POA* in the network graph are distinct. Consequently, transits between carpooling routes which intersect at a *POA* can be created using the same approach as mentioned before.

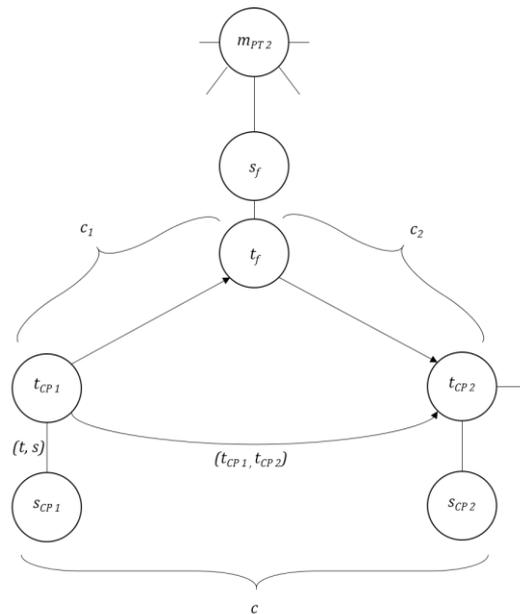


Figure 21 Schematic representation of a carpooling connection c_0 from s_{CP1} to s_{CP2} after exploiting a new stop s_f . New connections c_1 and c_2 are defined using a fictional stop time t_f .

Nevertheless, if a driver takes a detour, the actual arrival time at s_{PT} is not equal the arrival time at s_f , since a certain duration is needed to travel from s_f to s_{PT} . As the size of a drive time area d is known, the maximum time needed is $size(d)$. Thus, in a worst-case scenario for a drive time area of 5 minutes, a driver arrives at s_{PT} 5 minutes later. This fact needs to be considered when creating transit edges between carpooling offers and public transportation trips.

Creating transits between a newly exploited *POA* stop along a route requires applying penalties for mode changes. Since the carpooler in the worst-case $size(d)$ arrives later at s_{PT} , $size(d)$ can be added to the lower and upper bound of the conditional introduced in section 4.1. In case a driver would only need $size(d)/2$ to s_{PT} , transits are still only created for $lower\ bound + size(d) < transit > upper\ bound + size(d)$. Consequently, this approach is only an approximation.

In case a driver makes a detour to s_{PT} , arrival times at all later stops will change. For that reason, travel time functions could be assigned to edges. Nevertheless, it would be possible that a driver of a carpooling offer should not have to drive more than one detour, even if the sum of the duration of all detours would lie within its predefined detour range. If multiple detours would be allowed, the characteristics of carpooling would be change. Carpooling usually defines approximately three stops along a route. Allowing an undefined number of detours might lead to a bus- or tram-like behavior, meaning that the driver has to drive many short trips with many of detours.

Consequently, in this thesis, the use of drive time areas of half the size of the maximum allowed detour time is recommended. Hence, if one detour is made, the maximum allowed detour time is exhausted. Further, times at later stops do not need to be adjusted.

Due to merging two similar networks N_{PT} and N_{CP} while retaining the structure of stops and meta-stops, a multimodal network N_G is created, which is still routable by using standard routing algorithms such as Dijkstra's shortest path. Also, the possibility of detours along a carpooling route could be implemented in N_G . Further, these detours are only routed made required. Hence, the optional detours could be represented without changing the carpooler's route.

5 Experiment

In this chapter, the data model and merging techniques elucidated in chapter 4 Data Modeling and Graph Merging & Linking are implemented and evaluated. The experiment shall be built on real-life data as they can be found on online platforms. Thus, in the first section, Data, the three different data sets used are described. The following sections will discuss the actual implementation containing preprocessing steps, network graph generation, model merging and querying.

5.1 Data

This section describes the data used for the further implementation. In general, three different types of data have been gathered: Public transportation schedules, carpooling offers, and street data. As this thesis focuses on a national scale, data has been acquired with a focus on Switzerland. Switzerland has a very dense public transportation network of many different MOT. While trams and buses operate on a city to regional scale, trains serve connections across the entire country. Since carpooling offers tend to operate on an even broader scale, this thesis omits small scale MOT such as trams and buses and focuses on the railway network. In addition, due to limited computational power, the use of a smaller data set was mandatory.

Railway schedules could be retrieved directly via online download, whereas for carpooling offers, no public sources were available. Acquiring carpooling offers therefore required a more expanded process, which is explained in more detail in the respective section. The characteristics of carpooling offers is further analyzed in the following chapter 6.

5.1.1 Railway Data

The railway network of Switzerland is mostly operated by one agency. However, smaller agencies offering regional journeys exist. Nevertheless, they share a coordinated schedule. Therefore, the timetables of the railway network could have been retrieved as a GTFS feed from geOps⁵. GTFS feeds will be discussed further in section 5.2.1.

The timetable contains 1912 train stations and roughly 790k connections, covering the whole of Switzerland (cf. Figure 22). Geographic locations are only given for stop locations, while train routes do not have a geometry. For the purpose of modeling this schedule as a time-expanded graph and merging it to a multimodal network, this information is sufficient. The retrieved data set contains additional information apart from the timetable. As trains are offered by multiple agencies, additional information is given on who is operating certain connections. Furthermore,

⁵ <http://gtfs.geops.ch/> - geOps is a small GIS company located in Switzerland and Germany. geOps converts public transportation schedules in GTFS feeds and offers free downloads.

agencies provide information about the minimum transfer time at a stop. Hence, this definition can be used to correctly model transfers.



Figure 22 The railway network of Switzerland retrieved from the SBB at: <http://www.sbb.ch/freizeit-ferien/allgemeine-informationen/wallpaper/netzkarte.html> (Accessed: 10.03.2017)

In more detail, the retrieved data set contains the following information:

- **Agencies:** A list of 62 agencies offering journeys on the Swiss railway network.
- **Routes:** A list of 28455 time independent routes.
- **Trips:** A list of 69150 time dependent trips.
- **Stops:** A list of 1912 train stations, plus additional entries for every platform at a stop.
- **Stop times:** A list of 789331 stop times describing the arrival and departure times of a trip at stops.
- **Calendar dates:** A list of 5.1M calendar dates defining the availability of trips.
- **Transfers:** A list of 4817 entries defining the minimum transfer time at a train station.

A connection $c =: (z, s_1, t_1, s_2, t_2)$ described in the Data Modeling chapter 4.1 can be made up of two stops and at least two stop times plus a trip representing a train z . The retrieved data is therefore sufficient for modeling a time-expanded network graph.

Fares: According to Müller-Hannemann & Schnee (2006), financial costs are an important criterion when finding an optimal route. Therefore, additional fare information has been provided by the SBB. Unfortunately, the public transportation agency SBB relevant in this thesis has a special pricing system. Other than fixed prices per trip, the SBB calculates prices based on so called “Tarif 52

Kilometer". In essence, the price increases the longer a trip is. Unfortunately, prices do not increase in a linear fashion. Exact "Tarif kilometers" for every SBB trip were not available, but only for the ~75k most important journeys. Since these journeys contain via points, it was not possible to unhook "Tarif Kilometer" for single connections. In this thesis, fares of the railway network are therefore disregarded.

5.1.2 Carpooling Data

Carpooling is a growing market with many million users worldwide. Multiple online platforms exist where drivers can advertise a trip and passengers can find matching offers. On BlaBlaCar.de, a typical offer is divided into three sections of information (cf. Figure 23). The first part covers information about the driver (name, age, rating). After a trip, passengers can rate their drivers. The second part contains the actual information about the trip. Key parameters are, similar to public transportation, date and time. Further, the starting-, via- and end-point of a journey are listed. The last part involves fare information as well as capacity. Since carpooling uses personal cars, the number of seats is limited.

 <p>Steffen P 35 Jahre Botschafter/in</p> <p>★ 4,9/5 - 112 Bewertungen f 133 Freunde/innen</p>	<p>Do. 09. März - 19:00 Hägendorf → Zürich → Medlingen</p> <p>● Hägendorf, Schweiz ● Ankunft: Zürich, Schweiz (Bitte sprechen Sie die Details mit dem Fahrer/der Fahrerin ab.)</p>	<p>6 € pro Mitfahrer/in</p> <p>4 Plätze frei</p>
---	---	--

Figure 23 An offer advertised on BlaBlaCar.de. Typical information includes driver specific (left), journey specific (middle) and fare/availability specific (right) data.

By reconsidering the structure of a timetable, it can be determined that carpooling offers follow a similar scheme. Thus, modeling these offers as a time-expanded network should be possible. In the following, a list presents all information that can be retrieved and for what purpose it can be used:

- **Driver name** (optional): The name of a driver can be considered the same as that of a public transportation agency.
- **Driver rating** (optional): The rating of a driver can reflect reliability as well as convenience. When integrating carpooling into a multimodal routing system, it is of high importance that offers are reliable. Multimodal journeys fail if one sequence fails. Furthermore, convenience may be an optimization criterion of a user (cf. Delling, Dibbelt, Pajor, Wagner, & Werneck (2013)).

- **Date and time** (required): Carpooling offers are inherently time-dependent. An offer is only valid at a discrete point in time. Date and time is required for modeling carpooling as a routable network since time is a component of a connection. As can be seen in Figure 23, only an exact departure time is listed. An arrival time may differ greatly depending on detours, the amount of traffic, and other factors.
- **Stop locations** (required): Stop locations are used to embed an offer in a geographic space. Further, they are needed to merge different transportation networks.
- **Fare information** (optional): Information about the financial costs of a trip are not required to build a routable network. However, costs can be used as an optimization criterion.
- **Available seats** (optional/required): Information about the number of free seats is conceptually optional for a routing system. In practice, however, this information is required, since a carpooling offer can only be taken as long as there are free seats. This circumstance also counts for multimodal trips.
- **Detour time** (optional/required): On BlaBlaCar.de, drivers have the possibility to specify a detour range, which is usually between 0 and 30 min. Detour time is needed to implement a fuzzy merging technique between a carpooling network and a public transportation network.

As can be seen from the list above, carpooling offers on BlaBlaCar.de nearly satisfy all the requirements of a time-expanded model (stops, times, trip). Unfortunately, information about the arrival time at a certain stop is not available or only an approximation is given, since traffic and detours influence these arrival times. A time-expanded model, however, requires exact time information in order to model connections and enable transfers at stops. Consequently, data retrieved from BlaBlaCar.de must be enriched. The data enrichment process is further explained in the last section 5.1.2.2.

5.1.2.1 Acquiring Carpooling Data

The data set of carpooling offers used in the further experiment has been retrieved from BlaBlaCar.de. Since this thesis focuses on Switzerland, only offers which have at least one stop located within the borders of Switzerland were gathered. Unfortunately, no freely available data set could be found. Further, BlaBlaCar.de does not offer a public API to request offers. In addition, carpooling offers are very dynamic and only valid once. Consequently, a crawler was implemented in Java to gather carpooling offers on a daily basis. This section explains how the crawler was implemented and used to retrieve carpooling offers.

Prerequisites: Since no API was available, a crawler was implemented to derive carpooling offers. However, crawling data from an online platform has some drawbacks. On a basic level, a crawler accesses a website in a similar matter as a user would do. When running a crawler, it is necessary to be aware of the fact that potentially a very high number of requests could be made to the servers of the platform owner, which may slow down their service. On behalf of a sustainable crawling, the scope of offers to crawl has been narrowed down to major cities in Switzerland. It is important to mention that with an official API, the number of offers could have been increased. For this experiment, the 17 biggest cities defined by the *Bundesamt für Statistik (BFS)*⁶ have been chosen. In addition, Olten was added to the search set as well, because of its central position in the Swiss transportation network. The cities used are: *Zurich, Geneva, Basel, Berne, Lausanne, Winterthur, St. Gallen, Lucerne, Lugano, Biel/ Bienne, Thun, Olten, Schaffhausen, Chur, Le Grand-Saconnex, Uster,* and *Sion*. Figure 24 shows that these cities are equally spread over Switzerland. In addition, queries on BlaBlaCar.de do not only return offers from the exact defined location, but also from surrounding villages. The offer from Figure 23 appeared as a result for a search with Olten as starting location, which is approximately 5.5 km away. Hence, crawling offers for larger cities also return results for surrounding areas and therefore increases the amount of data.

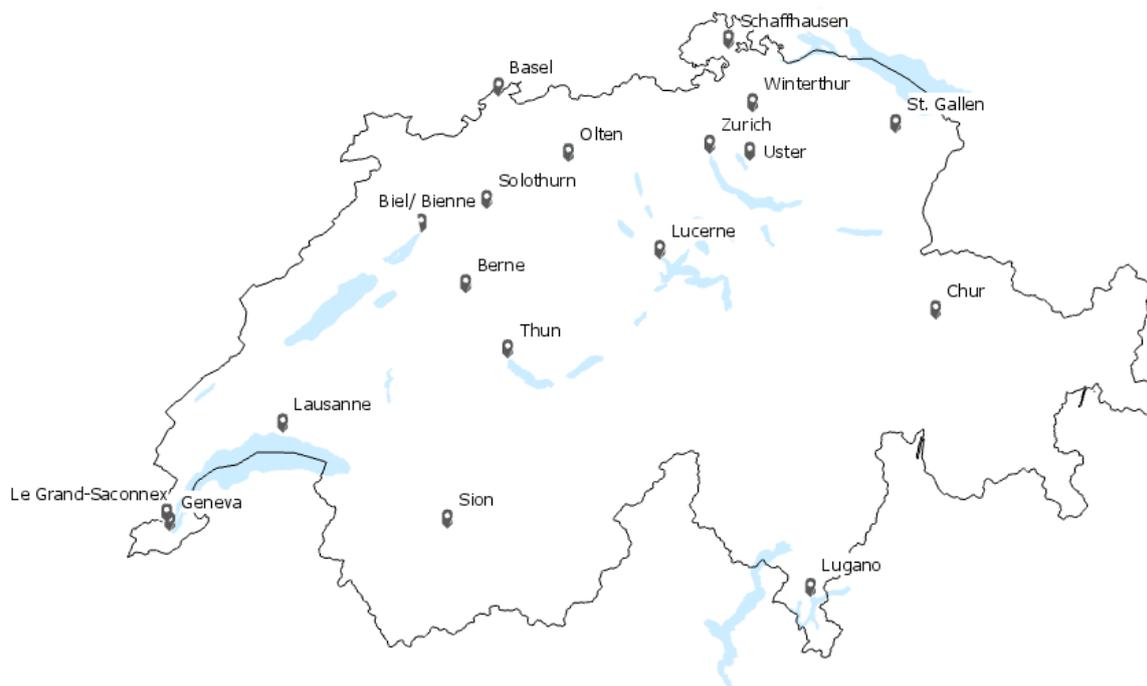


Figure 24 Locations of the 17 largest cities in Switzerland. These cities are used as search criteria for the crawler. Olten does not belong to the largest cities, but has been chosen due to its central position in the Swiss transportation network.

⁶ <https://www.bfs.admin.ch/> - Federal Statistical Office of Switzerland (Accessed: 25.05.2017)

Implementation: The crawler implemented for this experiment was written in Java. The language, however, does not have an influence on the results. In addition, a MySQL database was setup to store the retrieved and parsed data.

The crawler consists of three classes, the main class, a parser class, and an offer class. Generally speaking, the main class takes a list of cities as input. For each city, a HTML request is made to BlaBlaCar.de. The parser then processes the returning HTML element. It creates an Offer Object for every offer contained in the HTML element and pushes it into a list. Finally, the list of processed offers is stored in a MySQL database.

The procedure can be described as follows:

<p>Crawler</p> <p>input: List of cities L_c</p> <p>output: All unique offers for all cities stored in a MySQL database</p> <ol style="list-style-type: none"> 1 foreach $l \in L_c$ do 2 request HTML page element E for a search of l 3 parse E 4 foreach offer HTML element $e_0 \in E$ do 5 new Offer (id, driver, rating, date, time, start, stops, end, price, car, url, query) $\rightarrow o$ 6 push o to List of offer L_0 7 remove duplicates 8 post unique offers to MySQL DB

Since carpooling offers are very dynamic and vary from day to day, a cron job was set up to run the crawler every morning at 11:00. This time was chosen because many offers are posted in the early morning, departing on the same day. Thus, crawling in the very early morning would have also lead to a data loss. Nevertheless, the crawler could have been run multiple times during the day, but in regard of the workload of the BlaBlaCar servers, one crawl per day was deemed adequate.

Result: With the use of the crawler, around 18k offers were crawled within an 8-month period. Figure 25 shows the number of offers crawled per date. It becomes apparent that at the beginning of September 2016 the number of carpooling offers increased drastically. The reason for this is not clear, since no adjustments of the crawler were made. It can be assumed, however, that BlaBlaCar.de augmented their search engine. Entries with zero offers can be explained with the downtime of the server when the crawler was running.

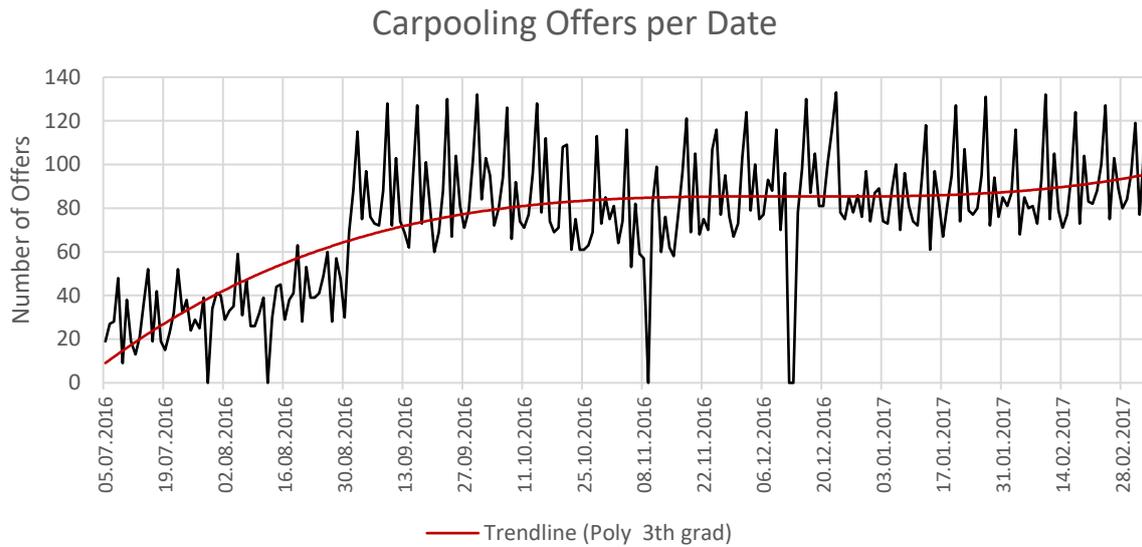


Figure 25 Number of crawled carpooling offers within an 8 month period. Zeros can be explained by the downtime of either the crawling server or BlaBlaCar.de at the moment of the request.

The resulting data set of offers gathered by the crawler were saved into a MySQL database. Each record in the database represents an offer. An offer is a tuple of a unique ID, driver's name, driver's rating, date, time, start, stops, end, price, car, URL, and query (date and city).

It is important to mention that stop locations have no geographical component. The proposed merging/linking technique, however, requires a geographic location. Consequently, the retrieved data had to be enriched and supplemented with a spatial component. The following section therefore explains the enrichment process of carpooling offers.

5.1.2.2 Data Enrichment

In the previous sections the crawling of carpooling offers were elucidated. Unfortunately, these offers, crawled from the Internet, lack certain information. For example, stop locations have no geographic location. Although this is not problematic for a user, it is for creating a multimodal network and the proposed merging/linking technique. Furthermore, without a geographic location, certain goal-directed algorithms such as the A* cannot be applied (cf. 2.1.2.2 Goal-Directed Techniques).

Apart from the missing spatial component, carpooling offers on BlaBlaCar.de do not specify an exact arrival time. A time-expanded model, however, requires this information. Consequently, in chapter 4.2.2 Carpooling as a Timetable, a fictional time node $t_{\mathcal{F}}$ was defined, which acts as a placeholder. Implementing a routable carpooling network demands this node to be filled with discrete time information. Therefore, it is important to enrich the crawled carpooling data. Following information must be supplemented:

- **Geographic locations:** The exact geographic position of a stop is mandatory for the proposed approach of merging the carpooling network with the railway network. Furthermore, the geographic location enables the use of goal-directed algorithms.
- **Duration:** The duration of a journey segment is mandatory in order to calculate arrival and departure times at the destination and intermediate stops and thus enabling transfers.

Enrichment process: The enrichment process covers two stages. The first stage geocodes stop locations of the crawled data, while the second stage calculates a potential route of a carpooling offer. Considering geocoding, the use of an external API is appropriate. Possible solutions are provided by Google⁷, Esri⁸, Geonames⁹ and others. Keeping in mind that potential routes also need to be calculated, Google's Directions API performs both stages in one request. Thus, Google's Directions API as was chosen as the tool to enrich the data.

A JavaScript program was written to automate the enrichment process. In essence, the application sends a request to the Directions API for each crawled offer. The request accepts arguments to personalize the directions results. For this experiment, default values were chosen. However, the departure time of the offer and a traffic model could be used to increase the correctness of the result. In this thesis, this enhancement has been omitted as it would require proof. As no data exists about the actual route a driver has taken, the correctness cannot be validated.

The response of the API is a JSON, containing much more information than needed. Thus, a large part of the response was pruned before being saved in the database, in order to save disk space. Pruned information is, for example, written instructions.

The Direction's API returns multiple levels of detail. The broadest level is a route, describing the total distance and duration of the journey. This route can be split into legs, representing connections between origin, intermediate stops, and the destination. Legs consequently equal a connection in a time-expanded model. Further, each leg holds information about its distance and duration, as well as the geographic location of the start and end points. Legs, in turn, can be separated into steps. Steps represent a segment between two points on the road network, where a driver has to take an action. The vertices of a step are therefore usually located at junctions, crossings and so on. The distance and duration of a step, as well as the coordinates of its vertices, are given. The most detailed level are paths. Paths are the actual route on a road network and thus contain of hundreds of thousands of vertices. Segments of paths do not provide information about the duration or distance.

⁷ <https://developers.google.com/maps/web-services/> - Google Maps APIs > Web Services (Accessed: 06.03.2017)

⁸ <https://geocode.arcgis.com/arcgis/index.html> - ArcGIS Online Geocoding Service (Accessed: 06.03.2017)

⁹ <http://www.geonames.org/export/ws-overview.html> - Geonames is a free online gazetteer offering free public APIs. (Accessed: 06.03.2017)

Result: The resulting data set contains information about the geographic location of stops. Furthermore, the potential route of the driver has been calculated in multiple levels of detail. All the additional data from the enriching was stored in the MySQL database. Therefore, the database contains four tables: *Offers*, *Routes*, *Legs*, and *Steps*. The path of a step is contained in the tuple of a step.

By reconsidering the structure of a time-expanded model, it becomes apparent that the minimum requirements are connections defined by a trip and its departure and arrival times at stops. Due to the enrichment process, missing time information at intermediate stops and the destination can be calculated using the duration information of the legs. Additionally, the proposed merging & linking technique requires a spatial component on stop locations, which could be also supplemented during the enrichment process. Consequently, the enriched carpooling data fulfills all the requirements of a time-expanded model and the proposed merging & linking technique.

5.1.3 Road Network Dataset

The linking approach explained in the chapter 4.3 Model Merging & Linking requires drive time areas around public transportation stops. These areas are pre-calculated, as they are considered stable. Drive time areas are calculated using a road network, containing extensive information. Either the road network must contain the duration for a road segment or the distance plus the speed limit.

For this step, Esri's StreetMap Premium¹⁰ was used. This set contains the 2016 European road network by HERE¹¹ in the form a file geodatabase for the use with ArcGIS for Desktop and the Network Analyst extension. The road network is updated twice a year and is therefore ideally suited to calculate drive time areas. In addition, this data set integrates historic traffic information, in order to improve the correctness of drive time areas for specific times during a day.

¹⁰ <http://www.esri.com/data/streetmap> - StreetMap Premium is a road network data set for the ArcGIS Platform. (Accessed: 20.03.2017)

¹¹ <https://here.com/> - HERE is company providing mapping data. (Accessed: 20.03.2017)

5.2 Data Structures & Preprocessing Stable Components

After collecting and enriching all the necessary data sets, prerequisites were defined and stable components were calculated. At the moment, carpooling offers are not in a format, which allows the generation of a time-expanded graph. Furthermore, arrival and departure times at intermediate stops and destinations have not been calculated yet. Thus, the following prerequisites can be defined:

- **Data structure:** All transportation data used must be structured in a sufficient way, so that building a time-expanded graph is straightforward. In addition, the data structure shall allow interoperability based on current standards.
- **Data enhancing:** Missing discrete departure and arrival times of carpooling offers need to be calculated.

Considering preprocessing, the proposed merging technique explained in chapter 4.3 Model Merging & Linking requires drive time areas of public transportation stops in order to combine different networks by representing fuzziness and preserving flexibility of carpooling. Since road networks and train stations do not change often, drive time areas are defined as a stable component. The fare scheme of the Swiss railway network changes only once a year and is therefore considered stable as well. As previously mentioned, the financial cost cannot be integrated directly into the network graph. Hence, an additional system is set up. This system must further allow updating prices once a year.

This section presents solutions for the defined prerequisites and explains the implementation and calculation of stable components. The following section will build up upon this preprocessing.

5.2.1 General Transit Feed Specification (GTFS)

The first prerequisite defined is an adequate data structure for the railway network and the carpooling data. Considering the components of a time-expanded model, at least the information about trips and their arrival and departure times at stops are required. Additionally, transfers shall be enabled by interlinking time nodes at a stop. Consequently, the data should be structured into at least trips, stops, times, and transfer regulations. Additionally, the railway network and carpooling both provide information about the operator of a trip. Carpooling offers also contain information about the rating of a driver, as well as the type of car. Although this information is conceptually not needed for a time-expanded model, it might be of interest for a user. The large amount of data, as well as the difference in contents of the railway network and carpooling require a well-engineered data structure, preferably similarly structured as the time-expanded model. Such a structure can be achieved by following the *General Transit Feed Specification (GTFS)*.

The *General Transit Feed Specification*, is a common way to model static public transportation schedules (Antrim & Barbeau, 2013). GTFS was originally developed by Google and aims to deliver an interoperable way for transport agencies to publish and share their schedules (Google, 2016). In addition, GTFS feeds are also widely used in other domains. For example, Esri offers a tool for their Network Analyst to import GTFS feeds. Even OpenTripPlanner¹², an open source routing engine, is based on GTFS. In current research, GTFS feeds are well-known and often used for case studies (Antrim & Barbeau, 2013; Bast & Storandt, 2014; Bit-Monnot et al., 2013; Bucher et al., 2017; Hillsman & Barbeau, 2011). Despite the fact that GTFS feeds are not graphs, they are a good way of structuring data. Especially when planning on creating time-expanded graphs, GTFS feeds provide a similar structure. Hence, generating graphs from GTFS feeds is straightforward.

A GTFS feed consists of several text files describing the schedule. A typical GTFS structure can be seen in Figure 26. Contained files and their relations to each other are represented. Basically, an agency operates routes, which are used by trips. Since trips are the time dependent component of a route, they incorporate stop times defining the departure and arrival times of a trip at a stop. Stop times of the same trip are related to each other, representing a train driving from stop to stop. Further, in a GTFS feed, stops are split up into parent stops, representing a train station, and child stops located at a parent, representing platforms.

A GTFS feed may contain much more information, but since they are not relevant for this thesis, they are disregarded; only the used components are elucidated.

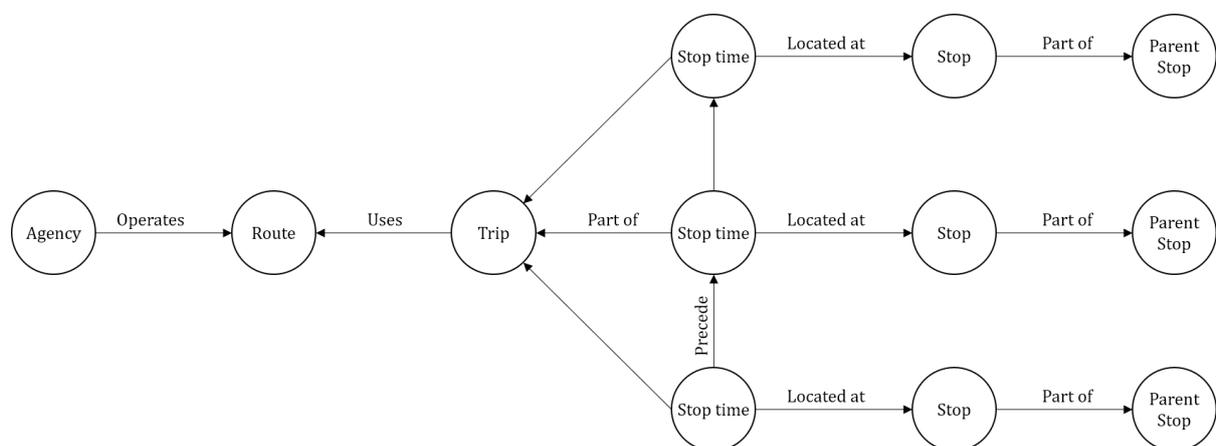


Figure 26 Schematic representation of a GTFS feed and its relations, according to Van Bruggen (2015).

¹² <http://www.opentripplanner.org/> - OpenTripPlanner (OTP) is an open source platform for multi-modal and multi-agency journey planning. (Accessed: 02.04.2017)

Agency: An operator of a public transportation service is called agency. Multiple operators can be stored in a single file and be associated with the routes they operate via a unique ID. Typical attributes are a name, a URL and the time zone where the agency is located. Table 1 shows all attributes of the agency file. The orange highlighted attributes are required.

According to a time-expanded model (cf. Pajor (2009); Schulz (2005)), agencies are conceptually not a part of this model. Nevertheless, the information about agencies can be appended to any route without a negative impact to the overall model.

Table 1 Structure of the agency file within a GTFS feed. Required parameters are highlighted in orange.

agency_id	agency_name	agency_url	agency_timezone	agency_lang	agency_phone	agency_fare_url	agency_email
Unique identifier	Name	URL	Time zone where to agency is located	Primary language	Phone number of	URL to the ticket	Email address

Routes: Often, public transportation offers fixed routes. For example, trains, trams or buses are divided into lines, always serving the same destinations. In GTFS these lines are represented as routes.

A route is operated by an agency and has a name and a type. The name, for example, is the number of a line and/or the lines destination. The type specifies the mode of transportation and is encoded using integers. Unfortunately, no transportation type is defined for carpooling (Google, 2016). Table 2 shows all attributes. The orange highlighted attributes are required. Either the route short name or the route long name is required. A short name, for example, is the line number, whereas the long name might be a combination of the line number plus the head sign. A route itself is time-independent since it describes a line and not the trip. Thus, a route is made up of trips.

A route of a GTFS feed is the same as a route in the time-expanded model. Conceptually, a route is not needed for routing purposes, but can be integrated to represent the structure of a transportation network.

Table 2 Structure of the routes file within a GTFS feed. Required parameters are highlighted in orange.

route_id	agency_id	route_short_name	route_long_name	route_type
Unique identifier	ID of the Operator	Short name	Long name	Mode of transportation

Trips: An actual journey is represented as a Trip, hence figuratively a train or bus. A Trip is therefore time-dependent and uses a route. Specific departure and arrival times are defined by a sequence of stop times which are stored separately. The availability of a trip during a week is also defined and stored in the Calendar file. Furthermore, because a trip can be considered a train or bus, additional information such as “bikes allowed” or “wheelchair accessible” can be specified. This information is, however, optional.

As mentioned above, trips are the time-dependent component of a route and represent an actual vehicle on a network. Trips in a GTFS feed have the same meaning as a trip in a time-expanded model. A time-expanded model is event driven, where an event is an elementary connection $c := (\text{train}, \text{start}, \text{departure}, \text{destination}, \text{arrival})$ (cf. 2.1.4.4 Modeling Public Transportation). Consequently, trips are mandatory.

Table 3 Structure of the trips file within a GTFS feed. Required parameters are highlighted in orange.

trip_id	route_id	service_id	trip_headsign
Unique identifier	ID of the route	Id of the calendar	Trips destination

Stop times: Exact departure and arrival times as well as the time a vehicle stays at a stop are represented by stop times. Stop times are part of a trip and are located at a stop. Since a trip has at least two stop times (starting point and destination), it is of high importance to order the stop times by defining a sequence.

Stop times of the same trip are related to each other and define the direction of a trip. As show in Figure 26, a sequence of stop times builds a path connecting different stops. Figuratively, a vehicle follows the path along stop times.

Stop times equal times in a time-expanded model. Both are defined by a tuple of departure and arrival times. Table 4 shows the structure of a stop time in a GTFS feed. In addition to the time information, the IDs of the related trip and the stop there are located is required. Further, the stop sequence indicates its position along a trip.

Table 4 Structure of the stop times file within a GTFS feed. Required parameters are highlighted in orange.

trip_id	arrival_time	departure_time	stop_id	stop_sequence
ID of the trip	Time of arrival at a specific stop	Time of departure from a specific stop	ID of the stop the stop time is located at	Order of the stop times along a trip

Stops: Stops in a GTFS feed equal stops of a time-expanded model. Furthermore, in chapter 4.1 Public Transportation (Railway) Model, the concept of a meta-stop was. A similar concept is also given in a GTFS feed, whereas platforms are modeled separately and are related to a parent train station.

A parent station is used to define the affiliation of platforms to a train station. For example, “Zurich HB” would be a parent station and “Bahnhof Löwenstrasse”, “Gleis 1”, etc., its children. Stop times are always located at child stops since trains arrive at platforms. Exceptions exist when a train station only has one platform. Then, stop times are located at the parent.

Table 5 Structure of the stops file within a GTFS feed. Required parameters are highlighted in orange.

stop_id	stop_code	stop_name	stop_lat	stop_lon	parent_station
Unique identifier	Short text or number. (e.g. abbr. of Stop or platform code)	Name of the Stop	Latitude	Longitude	ID of parent station

Calendar: By default, a GTFS feed assumes a certain frequency of trips. For that reason, the Calendar file declares the availability of a trip during the week. Moreover, the duration of validity of a service/trip must be defined.

Table 6 Structure of the calendar file within a GTFS feed. Required parameters are highlighted in orange.

service_id	Monday	Tuesday	start_date	end_date
Unique identifier, referenced in trips	Binary (0,1 true, false)	Binary (0,1 true, false)			Start date of the service	End date of the service

Calendar Dates: While the Calendar file in a GTFS feed only declares the validity of a trip on weekdays, the Calendar Dates file is used to define exceptions. Exceptions are defined by dates, describing when a certain connection is not available. As mentioned above, this concept requires frequent connections in order to be useful. Nevertheless, in case of a non-frequent MOT, it is allowed to omit the Calendar and only use calendar dates. In this case, rather than defining exceptions, only dates of availability are stored in the Calendar Dates file.

Table 7 Structure of the calendar dates file within a GTFS feed. Required parameters are highlighted in orange.

service_id	date	exception_type
Unique identifier, referenced in trips	Date	Int (1,2 valid, invalid)

Transfers: A GTFS feed can contain an additional file defining regulations for transfers. In more detail, an agency can specify the minimum transfer time between two stops. Consequently, it is possible to define variable transfer times between platforms. In the present experiment, this information can be used to specify the lower bounds of the conditional used to find possible transfers (cf. 5.3.1).

The transfer file in a GTFS feed is optional, but if it exists, at least the origin stop and the destination stop of the transfer, as well as the minimum transfer time must be defined.

Table 8 Structure of the transfers file within a GTFS feed. Required parameters are highlighted in orange.

from_stop_id	to_stop_id	min_transfer_time
ID of stop	ID of stop	Duration in seconds

The explanation of the GTFS structure mentioned above shows that a transportation schedule stored in this format already holds all the necessary information to generate a time-expanded graph. Furthermore, the relations between the different files equal the relations in a time-expanded graph. Consequently, converting carpooling offers into a GTFS feed can be considered as a first step towards a time-expanded carpooling graph. In addition, converted carpooling offers allow the use of other systems (cf. OpenTripPlanner, Esri's Network Analyst) and enables other researchers or companies to work with this data. At this moment, carpooling as a type of transportation in the GTFS is not defined, but can be unofficially defined by the present author.

5.2.2 Converting Carpooling Offers into a GTFS Feed

As mentioned in the previous section, GTFS feeds can be used to create a time-expanded graph in a straightforward way. Furthermore, during the conversion to GTFS, missing arrival times at carpooling stops can be calculated and stored in the according file. Hence, a complete carpooling schedule results after the conversion, which will be used for the further graph generation.

In this section, the process of converting the crawled carpooling offers into a GTFS feed is described. Therefore, the data structure of the gathered data is recapped based on an example. Additionally, the generation of each GTFS file is described individually.

The crawler plus the data enrichment process resulted in tuples of data. As mentioned before, multiple levels of abstraction exist. The broadest scale returned by the Directions API is a route, whereas only the start and end location, as well as total distance and duration are contained. A route, however, consists of legs that represent the connections from the start via the waypoints to the destination. Each leg holds also the information about the distance and duration. A leg can be separated into steps. Steps are basically segments at which the driver has to take an action at the end (e.g. turn right). Steps hold the same information as legs: start and end location, as well as duration and distance of a step. The most fine-grained scale is a path. A path is the actual route on the road network. Hence, the polyline of a path consists of hundreds of thousands of vertices. Paths do not hold information about the distance or duration from one vertex to another.

The proposed merging technique aims to find intersections between a drive time area of a public transportation stop and a carpooling journey. The appropriate scale derived from the API must thus be determined. Routes and legs cannot be used, as they are too generalized and an intersection with a drive time area would most likely be wrong. Steps, however, seem to be ideal, because they have an adequate number of vertices. Furthermore, the vertices lie at points at which a driver needs to take an action. Thus, these points are defined as *Points of Action (POA)*. Steps are very detailed in urban areas and loose in rural areas. Paths, by contrast, are far too detailed (>100k vertices per route) and do not have duration information for all segments. According to Pajor

(2009), merging networks is directed. It makes no sense to connect every junction of a road network to a public transportation stop. Consequently, a generalization of the steps will be made.

A typical journey divided into steps looks as follows:

Table 9 Structure of a carpooling offer divided into steps. The offer ID as well as the route ID are unique. The step ID is unique as well and further defines the order of the steps in a route. Duration is in seconds. The start and end locations hold an array of latitude and longitude coordinates.

Offer ID	Driver	Time	Date	Car	Rating	Price	Route ID	Start	End	Step ID	Start		Duration
											Loc.	End Loc.	
123	Joe	12:00	18.3.16	VW	4.7	8€	456	Bern	Olten	1	[x, y]	[x, y]	180
123	Joe		18.3.16	VW	4.7	8€	456	Bern	Olten	2	[x, y]	[x, y]	240
123	Joe		18.3.16	VW	4.7	8€	456	Olten	Basel	3	[x, y]	[x, y]	500
123	Joe		18.3.16	VW	4.7	8€	456	Olten	Basel	4	[x, y]	[x, y]	380
123	Joe		18.3.16	VW	4.7	8€	456	Olten	Basel	5	[x, y]	[x, y]	670

↓ Next Offer ↓

Table 9 illustrates that multiple steps have the same start and end location. That is because the Directions API does not return place names for vertices of steps. Hence, the place name information from the legs was chosen. The correct place names can later be inherited from public transportation stops during merging. At this moment, arrival and departure times of each step are unknown and need to be calculated based on the duration.

With the information quoted above, offers can be converted into a GTFS feed. Therefore, a converter in JavaScript was implemented. The converter is a command line tool using Node.js¹³ and takes a JSON file containing all steps of all offers as input. The JSON file has the same structure as the above-described Table 9. The following offers a step-by-step break-down, file by file, of how the data was converted:

Agency.txt: The minimum requirements of an agency are a name, a URL, and a time zone. However, in order to link a driver to a route, an ID is also needed. The agency name can simply be defined by the driver's name, whereas the time zone is, because of the crawling for Switzerland, always CET. Since a driver has no URL, a placeholder "n/a" is used. The driver's ID is a conversion from his name to a numeric (e.g. J(10) + O(15) + E(5) = 10155).

The crawled carpooling offers also contain a driver rating value. The rating can, as mentioned before, be an indicator for convenience. Therefore, this property was added to the agency.txt since it is driver specific and does not change for different offers.

¹³ <https://nodejs.org/en/> - Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine. (Accessed: 21.03.2017)

An entry in the agency.txt therefore looks as follows:

ID	name	URL	time_zone	rating
10155	Joe	n/a	CET	4.7

Routes.txt: The minimum requirements of a route are an ID, a short name and/or a long name and a type. The ID of a route can be defined by the route ID of an offer. The short name as well as the long name can be defined using the start and end location (e.g. B->B, Bern-Basel). The type defines the MOT of a route. As carpooling has not been defined by Google, "carpooling" is used as type. An entry in the routes.txt therefore looks as follows:

ID	short_name	long_name	type
456	B->B	Bern-Basel	"carpooling"

Trips.txt: In GTFS, a trip must have at least an ID, the ID of the route it uses and a service ID linking to the calendar_dates.txt where the date of the offer is defined. The route ID is present in the input offer, whereas the trip ID is not known. Since a route ID is already unique, a trip's ID is defined as route ID + 1/10 (e.g. trip_id = 456.1). The service ID is defined as the trip ID + ".s" (e.g. 456.1.s). Since the gathered carpooling offers have the information about the car type, a property for this is added as well. It is added to a trip because a driver might have multiple cars, so the car type is trip specific. This information is not needed for routing purposes, but might be of interest for a passenger.

An entry in the trips.txt therefore looks as follows:

ID	route_id	service_id	car_type
456.1	456	456.1.s	VW

Calendar_dates.txt: Calendar dates define the validity of a trip. They must at least have a service ID, a date and an exception typ. The service ID can be retrieved from the referenced trip, whereas the date can be specified by the offers date. The exception type defines, whether an offer is valid (1) or not (2) on this date. An entry in the calendar_dates.txt therefore looks as follows:

service_id	date	exception_type
456.1.s	2016-03-18	1

Stops.txt: Creating stop entries from carpooling offers requires some caution. The principle, however, is simple. The most important stop locations are the start, via and end points, which can easily be extracted. The first entry of an offer, thus the one with the lowest step ID, is the start point, whereas the one with the highest step ID and the same offer ID is the destination. As mentioned above, we have many stop locations (vertices of steps) along a route. In urban areas, the density is especially high. In light of the future merging of carpooling and public transportation, these detailed steps do not carry additional information. Hence, for stop locations along a route, a helper function is applied, which removes vertices within a self-defined distance. The idea is straightforward:

Reduce Steps

```

1   reference_bb <- BoundingBox(p_x)
2   if (p_{x+1} inside reference_bb) then
3       omit p_{x+1}
4   else
5       keep p_{x+1}
6       reference_bb <- BoundingBox(p_{x+1})

```

Consequently, the closest a point can lie to its predecessor is the distance predefined. For this thesis, a distance of 1km has been chosen.

Vertices which lie along a route have no place name. Hence, a placeholder name is defined, which will be later replaced by the merging technique. To nevertheless add some meaning, the definition was chosen to be the place name of the predecessor stop plus “Around” before the name, in order to indicate where the vertex is approximately located. Also, stops which are predefined origins, vias or destinations and lie along a route are marked with an additional property, to be able to later identify the type of stop (cf. 5.4). Major stops are defined by 1 and *POA* stops by 2.

The stops.txt file represents a set of distinct stop locations containing a unique ID, a place name, and a geographic location in latitude and longitude. Hence, a concatenation of the coordinate pair is used as an ID for a stop. An entry in the stops.txt therefore looks as follows:

ID	name	latitude	longitude	stop_type
xy	Bern	y	x	1

Stop_times.txt: Stop times represent arrival and departure times of a trip at a stop. The minimum properties are a trip ID, an arrival time, a departure time, a stop ID and a stop sequence. The trip ID can be retrieved from the referenced trip, while the departure time and arrival time need to be calculated using the duration of a step. Stop times are created together with stops. Whenever a stop is created, a corresponding stop time is implemented as well. Thus, the stop time receives the ID of the created stop. Arrival and departure times can be calculated using the summarized durations of all before steps added to the original departure time of the offer. Durations for omitted steps are part of the summary as well. As the waiting time at a stop is unknown, arrival and departure time are set the same. The stop sequence indicates the position of a stop time in a trip. An entry in the stops.txt therefore looks as follows:

trip_id	stop_id	arrival_time	departure_time	stop_sequence
456.1	xy	12:00	12:00	1
456.1	xy	12.03 (12:00 + 180s)	12.03 (12:00 + 180s)	2

The process of the converter can be found in the pseudo code block below.

Converter

```

input file: JSON File containing offers divided into steps. S
parameter: Generalization threshold t
1  foreach step  $s \in S$  do
2      if (inside t || new offer) then
3          write to agency.txt <- { id: s.driver as numeric, name: s.driver }
4          write to routes.txt <- { agency: aid, id: s.rid, type: "carpooling" }
5          write to trips.txt <- { route: s.rid, id: tid, service: cdid, sn: "", head: "" }
6          write to calendar_dates.txt <- { id: cdid, date: s.date, exception_type: 1/2 };
7          time <- s.time + sum(previous s.duration)
8          if (! last step of offer) then
9              write to stops.txt <- { id: sid, name: start name, lat: start lat, lng: start lng}
10             write to stoptimes_.txt <- { trip_id: tid, stop_id: sid, arrival: s.time,
                                         departure: time, stop_sequence: sid }
11         elseif (last step of offer) then
12             write to stops.txt <- { id: sid, name: end name, lat: end lat, lng: end lng}
13             write to stoptimes_.txt <- { trip_id: tid, stop_id: sid, arrival: time,
                                         departure: s.time, stop_sequence: sid }

```

Result: The conversion of a subset of approx. 2000 carpooling offers resulted in a GTFS feed which can be directly used to generate a time-expanded graph. A total of 1238 distinct drivers could be found, offering 2212 different trips. The reduction of vertices along a route resulted in 7312 distinct stop locations, containing 49630 stop times. In chapter 5.3.2 the generation of a time-expanded graph using this GTFS feed is further elucidated.

5.2.3 Calculating Drive Time Areas

A drive time area (aka. service area) is a region which encompasses all streets that can be reached within a certain amount of time from a predefined point (Esri, n.d.-b). Hence, service areas can be used to evaluate the accessibility of a point. Drive time areas can be calculated in two directions, either accumulating the duration away or towards a point of interest. Due to one-way streets, the direction may have a large impact. Further, drive time areas can be calculated using traffic models. These areas are likely smaller during rush hour.

For the experiment of this thesis, drive time areas have been calculated using the ArcGIS Network Analyst¹⁴ extension on a StreetMap Premium road network (cf. 5.1.3 Road Network Dataset). No traffic model was used, thus only one area was calculated per public transportation stop. Using multiple drive time areas at a stop at different times during the day may improve the proposed merging technique even more. This consideration could be part of a further study.

As drive time polygons are also used to exploit new stops along a carpooling route, a driver passes the area in both directions; first towards the stop and then away from the stop. Thus, independent of the direction of the drive time area, the extent in one direction is always not completely correct, but an approximation. For this thesis, the direction away from a point has been chosen.

Drivers of carpooling offers usually tend to accept a detour time of somewhere between 0 and 30 minutes. In this thesis, a static detour time of 15 minutes is assumed. Consequently, for a 15-minute detour, the drive time area must not be larger than 7.5 minutes. Therefore, drive time areas have been calculated for a 5-minutes radius, which leaves enough time in case of traffic or imperfection of one direction. Also, in a further study, drive time areas of different sizes could be used, in order to precisely match a driver's preferences.

A 5-minute drive time area also means that in a worst-case scenario, a driver has to take a 10-minutes detour, plus consider potential traffic delays. Consequently, with the size of the used drive time areas, a driver can maximally take one detour per trip.

¹⁴ <http://www.esri.com/software/arcgis/extensions/networkanalyst> - The ArcGIS Network Analyst is an extension for ArcGIS for Desktop and provides network analysis tools for complex routing problems. (Accessed: 24.03.2017)

Drive time areas are only calculated for public transportation stops and not for carpooling offers. Since in real-life carpooling offers are only valid once and the (pre-calculated) route may be uncertain, drive time areas for carpooling offers cannot be considered stable. For every single new offer inserted into the network graph, extensive preprocessing is needed before it can be added to the graph. In exchange, calculating drive time areas for public transportation stops can be considered stable, because variations of these stops are extremely rare.

Results: A total of 1897 drive time areas at public transportation stops have been calculated. Figure 27 shows a map of all resulting drive time areas. It is important to mention that a small number of drive time areas lie outside the Swiss borders, since the SBB serves individual train stations in the neighboring countries. All the calculated areas can be used to link carpooling stops to accessible public transportation stops at start, via, and end locations, as well as to exploit new stops along carpooling routes. In order to later assign drive time areas to public transportation stops, each polygon contains the stop ID as an attribute.



Figure 27 Resulting drive time areas (grey) of train stations (purple) served by the SBB. Drive time areas have been calculated in the direction away from train stations for a 5-minute drive.

5.3 Network Graph Generation

In chapter 4.3 Model Merging & Linking, a modeling approach for a public transportation and a carpooling network were proposed. Both cases were based on a time-expanded model. A common approach for creating multimodal networks is to first model each MOT separately and then merge and link it afterwards (Bast et al., 2015). As already mentioned in chapter 2 Related Work, networks are modeled according to graph theory. The use of *multidigraphs* is sufficient for this purpose (cf. (Foulds, 1992)). However, in the light of multimodality, where routing is based on label-constraints (cf. 2.2 Multimodal Routing), labeling edges and nodes is mandatory. Hence, the use of a labeled graph is necessary. Furthermore, as mentioned in 4.3 Model Merging & Linking, a slight adaption of the transfer modeling in a time-expanded graph is used. Rather than implementing an additional transfer node (cf. Pajor (2009)), labeled edges (or edge properties) were implemented. Although the modeling of transfers was adjusted slightly, the concept remains the same. A transfer is indicated by a label rather than by a node.

The implementation of this experiment was conducted on a Lenovo ThinkPad T430s with an Intel i7-3520 CPU @ 2.9 GHz and 16 GB memory. The graph was built using Neo4j¹⁵ 3.0.8, a widely used graph database. Neo4j was chosen for this experiment as it comes with a small set of spatial procedures needed for the proposed merging technique. Furthermore, simple routing algorithms (Dijkstra's algorithm, A*) are already implemented in the APOC¹⁶ plugin. Neo4j also allows adding properties to both nodes and edges (also called relationships), as well as defining labels and relationship types. This principle is a so-called labeled property graph¹⁷.

Syntax: In the following sections, we will present pseudo code describing import and graph operations:

create:	Creating a new node in the database
match:	Getting a node from the database
call:	Calling a specific external procedure.
set:	Setting a property or label of a node/ edge
:Label:	Label of a node/ relationship type
{key: value}:	Property of a node/ relationship
(n:Label {key: value}):	A node n with a label and properties
()<-[:Label]->():	Labeled edge between two nodes. <> direction of an edge

¹⁵ <https://neo4j.com/> - Neo4j is a highly scalable native graph database. (Accessed: 20.03.2017)

¹⁶ <https://github.com/neo4j-contrib/neo4j-apoc-procedures> - APOC: Awesome Procedures for Neo4j. (Accessed: 20.03.2017)

¹⁷ <https://neo4j.com/developer/graph-database/> - The Property Graph Model. (Accessed: 01.04.2017)

Remembering the structure of GTFS feeds, the set up consisted of several files which make up a schedule and if related lead to a time-expanded model. Each entry in a file can be considered a node, which is related to others via unique IDs. Further, GTFS feeds represent a certain hierarchy, which helps in loading data into the graph. Consequently, each file is loaded individually starting at the top and the created nodes are related afterwards. The naming of relationship types follows the in Figure 26 used syntax based on Van Bruggen (2015). Figure 28 highlights the structure and relation of a GTFS feed.

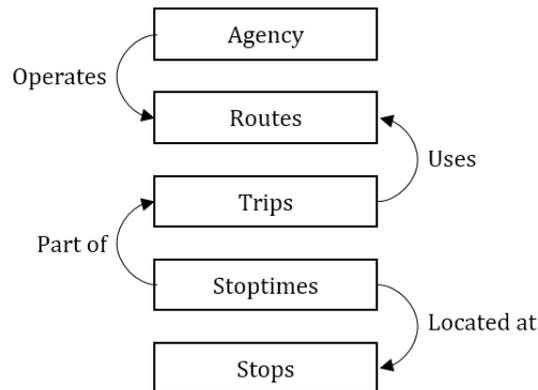


Figure 28 Structure and relations of a GTFS feed. This structure can be followed (top to bottom) during graph generations.

As both network datasets, public transportation and carpooling are present as GTFS feeds, constraints that apply for both can be defined:

- Agency.id is unique
- Route.id is unique
- Trip.id is unique
- Stop.id is unique

Further, indexes for properties with a high number of accesses can be created. Theoretically, indexes on the following properties are not needed, but lead to a performance improvement, which will not be discussed further:

- Index on Stop.name
- Index on Stoptime.stop_sequence

In the following sections, the graph generation of both the public transportation and carpooling networks are elucidated. The following chapter will then explain the merging and linking process.

5.3.1 Public Transportation Network Graph

The public transportation schedule used in this example has been retrieved in the form of a GTFS feed containing more information than conceptually needed. As already outlined in 5.1.1 Railway Data, additional information about the agency or the type of a train is given. This information is not concerned during routing, but may be of interest to a user. Thus, this information is integrated as well.

The creation process of a time-expanded public transportation graph follows the structure of the available GTFS feed (cf. 5.1.1 Railway Data). The first step is to load entries as nodes into the graph and relate them as quickly as possible.

In Neo4J, nodes can have multiple labels and properties. While labels are denoted as `:Label` and represent the type of a node or relation, properties are denoted as an object `{key: value}` describing the characteristics. In light of the later merging with carpooling, every node is assigned a `:Train` label. This allows to distinguish between public transportation a carpooling in a later step.

Agencies: Loading agencies to the temporarily empty graph G is straightforward. Each line of the `agency.txt` file represents an agency, hence a node $a \in A$ in G . All existing properties of an agency are assigned to a . The procedure looks as follows, resulting in $G = (A)$.

Load agency.txt	
1	<code>foreach line do</code>
2	<code>create (a:Agency:Train {id: line.id, name: line.name, url: line.url, timezone: line.timezone})</code>



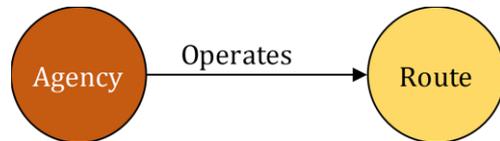
Routes: Routes are operated by an agency. Hence, each route $r \in R$ can be related to an agency $a \in A$, represented by an edge (a, r) . In order to improve performance during the import of routes, the related agency of the route is matched to the route ID of the entry. Consequently, a route node r and the according edge (a, r) are created in one go, where (a, r) is directed $a \rightarrow r$.

Load routes.txt

```

1   foreach line do
2       match (a:Agency:Train {id: line.agency_id}) as agency
2       create (r:Route:Train {id: line.id, short_name: line.short_name, long_name: line.long_name,
type: line.type) <- [:Operates] - (agency)

```



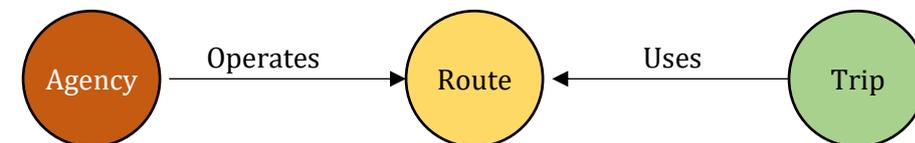
Trips: Trips are the time-dependent part of a route, hence representing the actual trains using a route. Consequently, a trip $t \in T$ is related to a route $r \in R$. The same procedure as for routes above can be applied for trips. For each entry in the trips.txt, the matching route node r is matched and related to a newly created trip node t by an edge (t, r) , resulting in $G = (A, R, T)$.

Load trips.txt

```

1   foreach line do
2       match (r:Route:Train {id: line.route_id}) as route
2       create (t:Trip:Train {id: line.trip_id, service_id: line.service_id, headsign: line.trip_headsign,
short_name: line.trip_short_name}) - [:Uses] ->(route)

```



Stops: In 4.1 Public Transportation (Railway) Model it was decided to use meta-stops. This circumstance is also existent in GTFS feeds, where platforms are modeled as child stops of a parent station. Consequently, during import, additional label *:Metastop* can be assigned to every stop node $s \in S$ where the *parent_station* property is null. All other entries are thus child stops representing a platform and are labeled with *:Track*. Although stops can be differentiated, they all belong to the subgraph of stops and are additionally labeled with *:Stop*.

Stops further contain a geographical location (latitude, longitude). As for loading data from a text file, latitudes and longitudes are parsed to Floats to enable calculations.

In chapter 5.2.3 Calculating Drive Time Areas, the stable component of drive time areas for each train station were preprocessed. These drive time areas are later used to link public transportation and carpooling. In order to allow future carpooling offers to be added to the network on-

demand without any preprocessing, the drive time areas are imported into the graph. Neo4J supports spatial components through the Neo4j Spatial¹⁸ plugin. Geometries can be added as a property of a node in the form of a *Well-known text (WKT)* string. Consequently, a new property *wkt* is implemented on *s*, holding the geometry of its drive time polygon.

Neo4J pursues the principle of spatial layers. In order to work with the drive time polygons stored in *s.wkt*, a new spatial layer must be created and every *s.wkt* added to this layer. Neo4J automatically creates a spatial index on every geometry, enabling spatial queries such as *closest*, *intersects* or *withinDistance*.

Load stops.txt

```

1   foreach line do
2       create (s:Stop:Train {id: line.id, name: line.name, stop_code: line.stop_code latitude:
          toFloat(line.latitude), longitude: toFloat(line.longitude), parent_station: line.parent_station})
2       if s.parent_station is null
2           set s:Metastop
3       else
4           set s:Track

```

Load drive time areas a

```

1   foreach a do
2       match (s:Stop:Metastop {id: a.stop_id})
3       set s.wkt = a

```

Create spatial layer 'geom'

```

1   call spatial.addWKTLayer('geom', 'wkt')

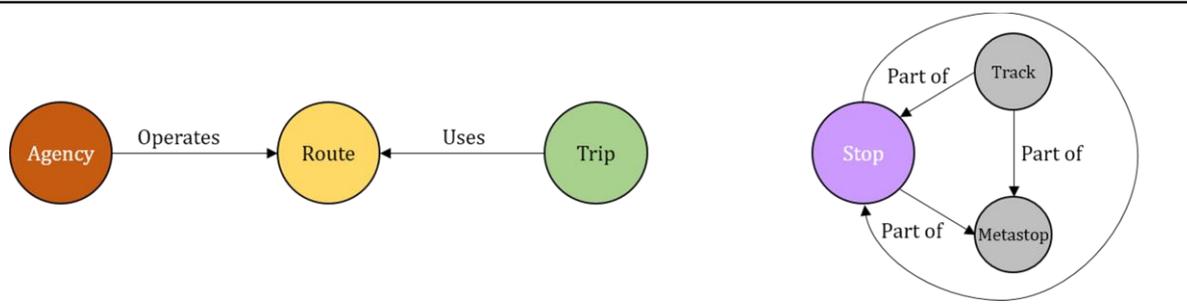
```

Adding Geometries to 'geom'

```

1   match (s:Stop:Metastop)
2   call spatial.addNodes('geom', s)

```



¹⁸ <https://github.com/neo4j-contrib/spatial> - Neo4j Spatial is a library of utilities for Neo4j that facilitates the enabling of spatial operations on data. (Accessed: 20.03.2017)

Stop times: Stop times $st \in ST$ define when a trip $t \in T$ stops at a specific stop $s \in S$. Hence, stop times are a part of t , located at s . Therefore, stop times st share an edge (st, t) with t and an edge (st, s) with s . Further, stop times are ordered by a *sequence* along a trip t . Consequently, stop times st_x and st_{x+1} belonging to t share an edge $e = (st_x, st_{x+1})$, representing a connection from stop s_x to stop s_{x+1} . Edge e therefore needs to be weighted to indicate the cost of travel from s_x to stop s_{x+1} along t . Hence, a property $e.duration = d(st_x, st_{x+1})$ is defined.

Load stoptimes.txt

```
1   foreach line do
2       create (st:Stoptime:Train {trip_id: line.trip_id, stop_id: line.stop_id, arrival_time: line.arrival_time, departure_time: line.departure_time, stop_sequence: toInt(line.stop_sequence)})
```

Relate stoptimes st with a trip t

```
1   match (t:Trip:Train)
2   match (st:Stoptime:Train {trip_id: t.id})
3   create (st)-[:PART_OF_TRIP]->(t)
```

Relate stoptimes st with a stop s

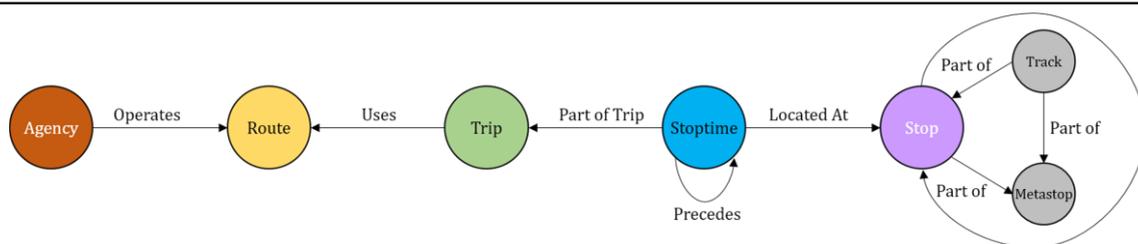
```
1   match (s:Stop:Train)
2   match (st:Stoptime:Train {stop_id: s.id})
3   create (st)-[:LOCATED_AT]->(s)
```

Relate stoptimes st_x with st_{x+1}

```
1   match (st_x:Stoptime:Train)
2   match (st_{x+1}:Stoptime:Train {trip_id: st_x.trip_id, stop_sequence: st_x.stop_sequence + 1})
3   create (st_x)-[:PRECEDES]->(st_{x+1})
```

Calculating edge weights of $e = (st_x, st_{x+1})$

```
1   match (st_x:Stoptime:Train)-[e:PRECEDES]->(st_{x+1}:Stoptime:Train)
2   set e.duration = st_{x+1}.arrival_time - st_x.departure_time
```



Provisional result: After loading every entry contained in the GTFS feed, the result is a graph $G = (A, R, T, S, ST)$. This graph represents the schedule of the SBB. Further, the considerably stable drive time areas around train stations have been integrated and indexed, allowing spatial queries. The graph is, at the moment, only routable for direct journeys using basic routing techniques. Time-expanded models, however, require modeling transfers at stops to route for indirect journeys (Pajor, 2009). Consequently, transfer edges between stop times need to be created (cf. 4.1 Public Transportation (Railway) Model).

Furthermore, the use of goal directed routing algorithms such as A* is currently not possible. Regarding the explanation in chapter 2.1.2.2 Goal-Directed Techniques, A* uses a heuristic to only route towards the destination. Usually, the geographic location, i.e. the distance to the destination is used. Since in a time-expanded graph one usually routes via stop times, A* is not able to use the geographic location without any augmentation, as stop times do not have coordinates. In order to allow the use of A* anyhow, the latitude/longitude information of the related stop is copied to the stop time.

Copy Coordinates

```
1    match (st:Stoptime:Train)-[:LOCATED_AT]->(s:Stop:Train)
2    set st.latitude = s.latitude, st.longitude = s.longitude
```

Transfers: According to Pajor (2009), transfers in a time-expanded graph are modeled using additional transfer nodes. Since a labeled property graph is used in this experiment, the use of transit nodes is relinquished, and instead labeled relationships are used to differentiate between transfer and connection edges. Based on the presented modeling approach in chapter 4.1 Public Transportation (Railway) Model, conditionals are used to define possible transfers between trips at a stop.

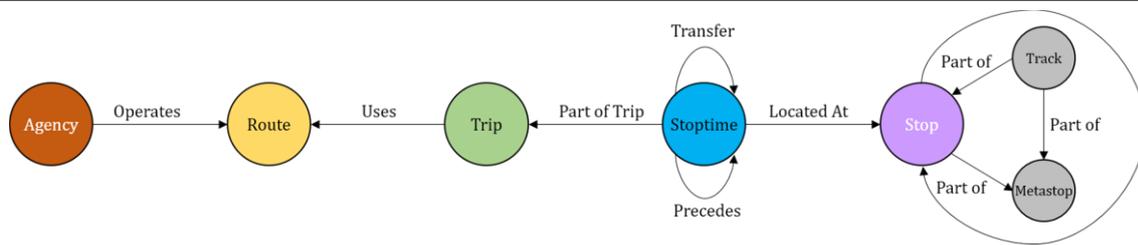
A GTFS feed can contain a file defining minimum transfer times at a stop or between child stops. The feed used in this example encloses such a file, which defines minimum transfer times between meta-stops and at a stop. Unfortunately, the transit time between child stops is not given. Nevertheless, this file is used to define the conditional for viable transfers. Since the transfers.txt file does not specify a maximum transfer time, twice the minimum transfer time is used as upper limit. Consequently, a transfer edge $e = (st_A, st_B)$ where st_A belongs to t_A and st_B belongs to t_B is created, if the difference between the arrival time of train A and the departure time of train B is larger than the minimum transfer time and smaller than twice the minimum transfer time. Further, transfer edges shall not be self-relating, as staying in the same train is not a transfer. The process can be described as follows:

Load transfers.txt

```

1   foreach line do
2       match (s1:Stop {id: line.from_stop_id})--(st1:Stoptime)
3       match (s2:Stop {id: line.to_stop_id})--(st2:Stoptime)
           where not ID(st1) = ID(st2)
           and st2.departure_time - st1.arrival_time < (2 * line.min_transfer_time)
           and st2.departure_time - st1.arrival_time > line.min_transfer_time
4       create (st1)-[:TRANSFER {duration: st2.departure_time - st1.arrival_time }]->(st2)

```



Result: The resulting time-expanded public transportation (railway) graph finally allows querying for indirect routes. Additionally, the use of goal directed techniques (A*) has been enabled due to adding a spatial component to stop times. The use of additional labels for meta-stops (*:Metastop*) and platforms (*:Track*) helps to distinguish between a train station and the actual platform.

Using labeled transfer edges (*:TRANSFER*) further aid in interpreting a query result and thus evaluates if transfers occur or not. Also, this label further enables to only search for direct routes by omitting every transfer edge during a shortest path query.

Due to consistency reasons, the use of date information is disregarded (cf. 5.3.2 Carpooling Network Graph).

5.3.2 Carpooling Network Graph

In chapter 4.2 Carpooling Model use a time-expanded modeling approach for a carpooling network was argued for. A time-expanded model allows the representation of a schedule as a graph and further allows the proposed merging technique. Chapter 5.1.2 describes how carpooling offers can indeed be modeled time-expanded, but require further data enrichment in order to retrieve approximate time information along a route. Chapter 5.2 showed that GTFS feeds serve as an adequate data structure for the future time-expanded graph building. Consequently, in 5.2.2 Converting Carpooling Offers into a GTFS Feed, a set of roughly 2k carpooling offers were converted into a GTFS feed. A subset was used to improve performance during graph operations. However, in order to have sufficient offers in the network, the date component will be neglected. A carpooling offer is contemplated to be available on every date.

As a GTFS feed contains the carpooling offers gathered from BlaBlaCar.de, the same procedure as for the public transportation schedule elucidated in 5.3.1 can be applied. However, rather than using a `:Train` label on every node created, a `:CP` label is used instead to indicate their affiliation. Additionally, carpooling stops do not hold a drive time area. As elucidated in chapter 5.2.3 Calculating Drive Time Areas, drive time areas around carpooling offers can be considered instable, hence leading to an increased computational cost without any further benefits.

As the procedure of creating a time-expanded graph from a GTFS feed has been extensively explained in the previous section, the focus is now directed only at deviations.

Agency: In chapter 5.2.2 Converting Carpooling Offers into a GTFS Feed, the `Agency.txt` is extended by a driver's rating, indicating his reliability and thus the overall convenience. Consequently, an additional property `rating` was assigned to a driver's node: `(a:Agency:CP {..., rating: -.})`.

Routes: A GTFS feed specifies the MOT of a route. As already mentioned before, carpooling has not been adopted as an official route type. Hence, in 5.2.2 Converting Carpooling Offers into a GTFS Feed, the route type is defined to be "carpooling". Consequently, rather than using an integer for the type, a string is used: `(r:Route:CP {..., route_type: "carpooling"})`.

Trips: A trip is, in this case, a driver using a specific route. Since the car type and thus also the number of available seats may be important, an additional property `car_type` on a trip was needed. As previously mentioned, this property needs to be set on a trip rather than a route, because a driver may have multiple cars and thus be driving the same route with different ones. Consequently, a `car_type` property was added to a trip node: `(t:Trip:CP {..., car_type: ""})`.

Stops: Carpooling stops do not yet have parent stations. Parent stations are used to define the affiliation of platforms (child stops) to a train station. Carpooling does not follow the concept of platforms and thus does not require any child-parent relations. Nevertheless, according to the merging/ linking approach proposed in chapter 4.3 Data Modeling and Graph Merging & Linking, carpooling stops will later be appended to train stations. With that said, at the moment, no definitions of meta- and child stops have to be made yet. Stops can be simply loaded into the graph.

Stop times: Due to the enrichment process and the conversion into a GTFS feed, the in chapter 4.2.2 Carpooling as a Timetable mentioned fictional time nodes t_f could be augmented with discrete time information. Hence, they are no longer fictional and can be used to build a real time-expanded graph. The structure of a stop time node is the same as for public transportation anyhow.

Calendar Dates: As mentioned at the beginning of this section, the date information of carpooling offers was disregarded in order to virtually increase the set of valid offers. Therefore, no calendar dates have been added to the graph.

Transfers: Transfers between carpooling offers have basically been implemented the same as public transportation transits. The main difference is the minimum transfer time. While the SBB defines a minimum transfer time for a valid transit, carpooling inherently does not have specified transfer times since it is unusual to change a carpooler during a journey. Consequently, it was not possible to implement meaningful variable transit times. Thus, static measures have been used. In this experiment, a maximum transfer time of 10 minutes and a minimum of 3 minutes were defined.

Enabling A*: In the previous section, the process of copying the geographic location of a stop to all related stop times were elucidated, in order to enable the use of a goal directed algorithm such as A*. The same procedure was naturally applied to the carpooling graph as well.

Result: The resulting time-expanded carpooling graph (cf. Figure 29) contains every offer of the input subset, modeled as nodes and relation, without a true geographical route. The use of basic and goal directed algorithms is enabled by a slight modification and allows for indirect journeys.

Date information is not contained in the graph, as, on behalf of the small set of offers, it is assumed that each offer is valid on every day.

As previously mentioned in chapter 4.2.2 Carpooling as a Timetable, a time-expanded graph is not able to adequately represent the fuzziness of carpooling offers at stop locations. Consequently, the stop locations loaded into the graph are still imprecise. Nevertheless, fuzziness will be implemented with the help of the proposed merging/linking technique, by relating carpooling and public transportation stops based on drive time. Therefore, the next section explains the implementation of the proposed merging/linking technique.

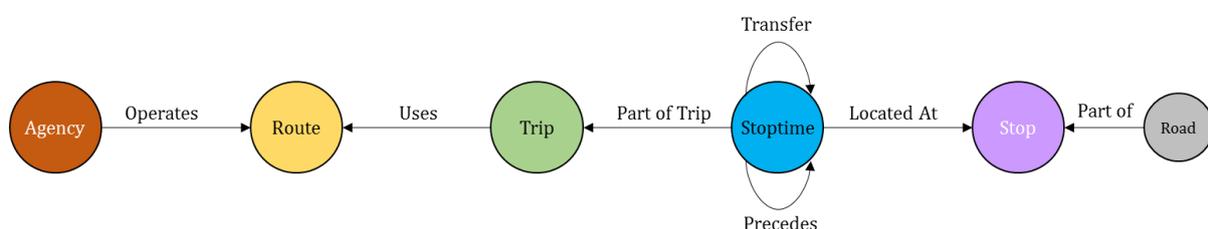


Figure 29 Schematic representation of the meta graph of the carpooling network.

5.4 Network Merging & Linking

According to the current state of the art, multimodal networks are created by merging individual networks and linking stop location by its geographic location (Bast et al., 2015). A common approach is to solve the *Nearest Neighbor Problem (NNP)*, resulting in linking the closest stop locations of both networks (Pajor, 2009). Thereby, linking is an asynchronous process where, for example, all stops of a public transportation network are linked to the road network, but not all nodes of a road network are linked to the public transportation network (Pajor, 2009). Although this approach can be computed quite easily, it also has some drawbacks. Solving the *NNP*, i.e. connecting closest nodes, is not feasible in the present case. Closest distance, even if constrained (cf. Delling, Pajor, et al. (2009a); Dibbelt et al. (2015)), does not account for real-life situations. A carpooler, for example, is bound to the road network and defines a maximum detour time. Consequently, the threshold varies for every stop depending on the road network. Furthermore, it may even differ for different carpoolers, as the maximum detour time can vary. A closest point linking approach is therefore not suitable. Also, connecting only two stops with each other may not be sufficient as a carpooler may be able to approach multiple stops within his/her detour time, thus revealing some sort of fuzziness. In addition, chapter 5.1.2 Carpooling Data illustrates that carpooling stops are very imprecise, making it impossible to use an accurate, closest point's link. Thus, a carpooling stop should be connected to multiple public transportation stops. The same principle can be applied to stops along a carpooling route where the detour time is decisive.

The merging/linking approach proposed in 4.3 Model Merging addresses the drawbacks of the *NNP*, by implementing a fuzzy technique based on driver's standards. The technique can be considered fuzzy as it uses drive time areas around public transportation stops, allowing the definition of multiple possible stops. Furthermore, new (public transportation) stops along a carpooling route can be exploited without forcing the driver to take unnecessary detours. Hence, the merging/linking approach proposed also accounts for the flexibility of carpooling.

Since this fuzziness and flexibility could not be implemented into the carpooling network graph itself, the merging/linking approach shall re-construct the lost features. This section therefore applies this technique to the in 5.3.1 and 5.3.2 created individual networks.

Merging: Conceptually, the individually created graphs $G_{CP} = (A, R, T, S, ST)$ and $G_{PT} = (A, R, T, S, ST)$ need to be merged into one single large graph $G_{Multi} = (A, R, T, S, ST)$, where $A = (A_{CP}, A_{PT}), R = (R_{CP}, R_{PT})$ and so on. But because a graph database (Neo4J) is used, both network graphs have already been created in the same database. This is also a reason for labeling each node with either $:Train$ or $:CP$ (cf. 5.3.1, 5.3.2). Consequently, no merging has to be performed, as a graph $G_{Multi} = (G_{CP}, G_{Train})$ already exists.

Linking: The linking process, i.e. implementing link edges between both networks, finally re-constructs carpooling's flexibility and deals with imprecise stop locations. The proposed linking technique deals with these two problems. First, carpooling stop locations are imprecise in terms of their exact spatial location, hence they are fuzzy. This first problem is addressed by relating the geocoded carpooling stop to every public transportation stop reachable within a 5-minute drive. Secondly, carpooling has the flexibility to deviate from the original route to pick-up or drop-off a passenger at a "new" stop location. This second problem is addressed by analyzing whether or not a carpooling route intersects a drive time area of a public transportation stop. As not every intersection automatically means a stop is possible, e.g. a driver on a highway may not have the possibility to approach the desired stop, chapter 5.2.2 Converting Carpooling Offers into a GTFS Feed elucidates the use of an abstracted, action oriented path. A stop location and time node for every place the user needs to take an action (e.g. turn right at a junction) was implemented. Furthermore, this set of stop locations was generalized by a 1 km distance in order to omit close-to-each-other points which are irrelevant. Thus, rather than intersecting the true path with drive time areas, the areas can be intersected with *Points of Action (POA)*. Consequently, problem two can be solved with the same approach as problem one.

In section 5.3.1, drive time areas of meta-stops were integrated into the network graph. Due to a spatial index, spatial operations are enabled. Therefore, the linking process can be started by intersecting all carpooling (: *CP*) stop nodes and *POA* nodes with the drive time polygons. Whenever an intersection is found, the investigated node is related to the meta-stop of the drive time area. Furthermore, during the conversion of carpooling offers to a GTFS feed (cf. 5.2.2), *POA* stops do not have a name. Therefore, a placeholder "Around ..." was set. By appending a carpooling stop to a meta-stop of public transportation, the placeholder can now be replaced with the actual name of the meta-stop.

As *POA* nodes will be linked to a stop a certain distance away, stop times may change. This phenomenon has been already discussed in chapter 4.3.3. The simple link to a reachable stop, however, does not automatically adjust the stop time. In order to bypass this problem, a penalty for transfers at *POA* stops is added, which will be discussed later in this section.

The process of linking can be described as follows:

Linking Carpooling and Public Transportation Graphs G_{CP} & G_{PT}

```

1      match (s:Stop:CP)
2      call spatial.intersect(drive time areas, s) yield node as metastop
3      if intersection
4          set s.meta_names = metastop.name
4          create (s)-[:LOCATED_AT]->(metastop)

```

First, a carpooling stop is matched and intersected with the spatial layer holding all meta-stops and their drive time polygons. Whenever an intersection is found, it yields the meta-node of the intersected drive time polygon and creates an edge between the carpooling stop and the meta-stop. Consequently, the carpooling stop is appended to the meta-stop as a child. This process is performed for every carpooling stop contained in G_{CP} .

Mode change (Transfers): The above merging and linking processes created a single multimodal graph $G_{Multi} = (G_{CP}, G_{PT})$. However, transfers between carpooling and public transportation are not enabled yet, as transfer edges between carpooling and public transportation connections were not created. According to Bast et al. (2015), one might add penalties for mode changes. In the present example, these penalties can be used not only for mode changes, the modification of stop times at *POA* stops. Because a *POA* stop was appended as a child stop to a meta-stop of public transportation, the real arrival time at the meta-stop may be at a maximum of half the drive time area size later. Consequently, the upper limit of the conditional used to find possible transfers (cf. 4.3.2 or 5.3.1 & 5.3.2) can be increased by half the size of a drive time area. If a driver actually drives a detour, all future stop times would consequently also change, but as the size of the drive time areas only allow one detour per trip, the future stop times do not need to be adjusted.

Major stop locations such as start, via, and end locations are approached directly by a driver. Thus, a drive time area based penalty on the transfer conditional is not needed. It would be possible to add a penalty as well because of the mode change. In this experiment, however, a static transfer time with the same bounds as for carpooling is used (cf. 5.3.2).

The process of creating mode changes can be separated into two operations. First mode changes at major stop locations are created. Later, the penalty extended transfers are implemented at *POA* stops.

Transfers at major stop locations

```

1   match (m:Metastop)<-[:LOCATED_AT]-(s1:Stop:CP {stop_type: 1})--(st1:Stoptime:CP)
2   match (m)—(st2:Stoptime:Train)
      where not ID(st1) = ID(st2)
      and st2.departure_time - st1.arrival_time_s < 600
      and st2.departure_time - st1.arrival_time_s > 180
3   create (st1)-[:TRANSFER {duration: st2.departure_time - st1.arrival_time }]->(st2)

```

Transfers at *POA* stop locations

```

1   match (m:Metastop)<-[:LOCATED_AT]-(s1:Stop:CP {stop_type: 2})--(st1:Stoptime:CP)
2   match (m)—(st2:Stoptime:Train)
      where not ID(st1) = ID(st2)
      and st2.departure_time - st1.arrival_time_s < 600 + 300
      and st2.departure_time - st1.arrival_time_s > 180
3   create (st1)-[:TRANSFER {duration: st2.departure_time - st1.arrival_time }]->(st2)

```

Result: After merging graphs G_{CP} and G_{PT} to G_{Multi} , creating links, and adding transfer edges, the result is a full multimodal time-expanded graph. The usage of drive time areas allowed linking a carpooling stop to multiple public transportation stops, hence representing the fuzziness of carpooling stops in the graph itself. With the use of the drive time polygons, new stops along a route could be exploited by linking *POA* stops to public transportation stops. Also, the definition of different carpooling stops (major stops, *POA*) enable the modeling of diverse transits.

The final multimodal time-expanded graph is still routable using basic techniques. Further, the adjustments of stop times allow the use of the goal directed A* algorithm. Not only shortest routes can be retrieved from this graph, but also, for example, all shortest paths in terms of the number of hops. Therefore, the next section describes basic queries on the in this chapter generated multimodal graph.

5.5 Querying

The previous sections demonstrated the building of a multimodal graph based on the explanations of chapter 4 Data Modeling and Graph Merging & Linking. The result is a time-expanded graph, which entails fuzziness and flexibility of carpooling. All nodes are labeled according to their affiliation to an MOT. Instead of creating additional transit nodes, labeled edges were used. Even though this is a difference to e.g. Pajor (2009), it should not change the principle of a time-expanded model.

A time-expanded model, hence also the multimodal graph, can be queried with basic techniques. Other techniques can also be applied on the graph, in order to for example only query for direct routes, or query for all shortest routes in terms of the number of hops. Therefore, this section aims to provide an overview of possible query techniques on the graph.

Basic Principles: Usually, a user wants to query routes from a start location to an end location. However, the platform, thus the child stop of the start and end location, is not known beforehand. Thus, the routing algorithm shall start at a meta-stop and find the shortest way to the desired destination.

In a time-expanded graph, stops are connected via stop times (cf. Figure 30). Hence, a routing algorithm may follow the edges between stop times. Consequently, routing happens on the sub-graph of stop times. However, having a meta-stop as the origin for the routing algorithm does not allow choosing either a specific departure time or a time range, as the time information is represented by the stop times.

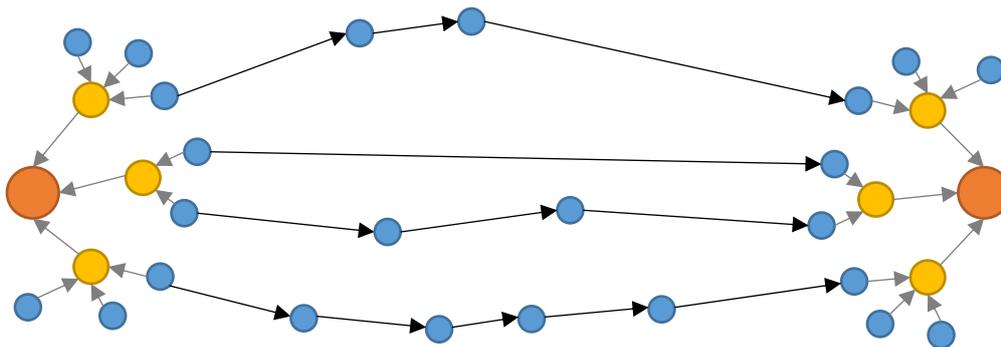


Figure 30 Schematic representation of two interconnected stops. Meta-stops (orange), child stops (yellow), stop times (blue).

Number of hops: In the presented graph, it is possible to query all directions between meta-stops. Rather than searching for the shortest path, a search is performed for any path from an origin to a destination. It is crucial that without any restriction, a huge number of possible connections will be found. Therefore, it makes sense to restrict the connections to a maximum number of hops. Figure 31 shows an example where the number of hops has been limited to 3 along stop times or 7 by considering the path from meta-stop to meta-stop. In this case, paths between the origin and destination are not ordered by the cost, but by the number of hops.

Searching for paths with a small number of hops can be used to query for multiple connections between two stops. Furthermore, routes with a small number of stops tend to be faster as they approach their destination more directly and do not lose time at intermediate stops. It is also important to mention that this must not be true in every case. Nevertheless, a user may prefer a journey with less intermediate stops or transfers, which can be calculated using this approach.

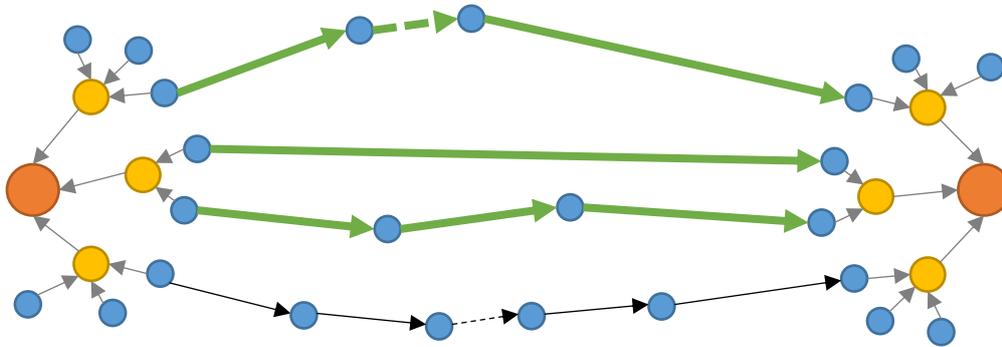


Figure 31 Schematic representation of valid routes in terms of the number of hops. Valid routes are highlighted in green, meta-stops in orange, child stops in yellow and stop times in blue.

Direct routes: In the former sections, the use of labeled transfer edges was elucidated. Standard connection edges are labeled with *:PERCEEDES*, while transfer edges are labeled *:TRANSFER*. Consequently, by pruning all paths which contain a *:TRANSFER* edge, the result allows querying only for direct routes. Figure 32 shows a case where only routes without transfers from the origin to the destination are found.

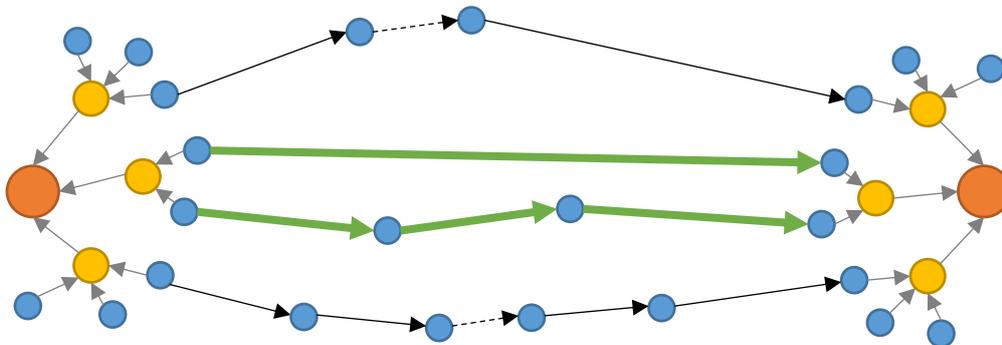


Figure 32 Schematic representation of direct routes (no transfers). Valid routes are highlighted in green, meta-stops in orange, child stops in yellow and stop times in blue.

Range Problem: The *Range Problem* describes the circumstance that a user may not specify a discrete time of departure, but a time range (cf. 2.1.4.2 Range Problem (RP)). Consequently, not all stop times are valid. The routing algorithm shall not consider, hence not follow, stop times which lie outside of the defined time range. In the presented graph, the *RP* can be solved by not starting at the meta-stop of the start location, but rather providing the algorithm a set of stop times.

As an example: A user wants to travel from A to B and leave between 08:00 and 10:00 o'clock. First, all stop times located at the child stops from A are matched where the departure time is later than 8:00 but earlier than 10:00. These stop times can now be used as start locations for the routing algorithm. Consequently, the routing algorithm can only return the shortest path leaving at one of these stop times.

Figure 33 shows a schematic example of this process. Stop times with departure times outside the time range are hollow, while all valid stop times are blue. Even though there are other possible connections between A and B (light grey), they are not queried, as their stop times lie outside the time range.

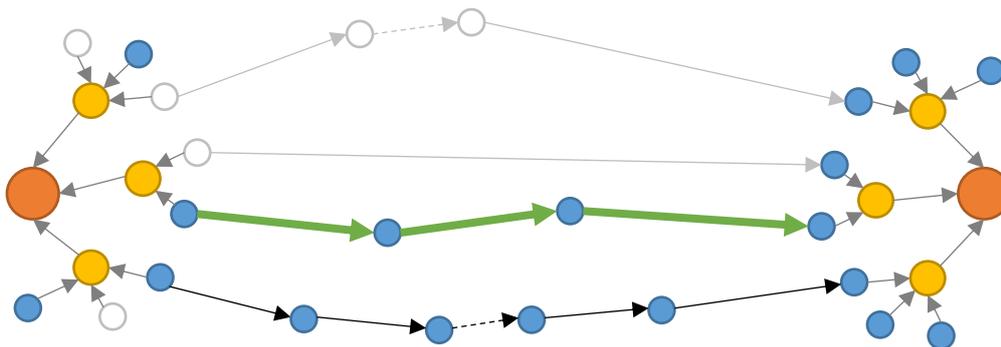


Figure 33 Schematic representation of the Range Problem. Greyed out stop times are not used as origins of the routing algorithm. Valid routes are highlighted in green, meta-stops in orange, child stops in yellow and stop times in blue.

In the present graph, the *RP* is solved by first matching all stop times at the origin's meta-stop which lie in the time range. In a second step, all stop times at the destination's meta-stop are matched. With these stop times, a Dijkstra's algorithm can be run on *:PRECEDES* and *:TRANSFER* edges of the stop times, which returns the quickest path departing between 08:00 and 10:00 o'clock.

Range Problem

```

1   match (m:Metastop)—(st:Stoptime)
      where st.arrival_time < 08:00
      and st.arrival_time > 10:00

2   with st as origin

3   match (m2:Metastop)—(st2:Stoptime)

4   with st2 as destination

5   call apoc.algo.Dijkstra(origin, destination, "PERCEEDES>|TRANSFER>", duration)

```

Inverse Range Problem: Similar to the *RP*, a user might want to arrive at the destination at a certain time, but does not know when he/she should leave at the origin. The same procedure as for the *RP* can be applied. However, instead of reducing the set of stop times at the origin, the set of stop times at the destination is reduced. Hence, a path can only be found if it ends at one of these stop times (cf. Figure 34).

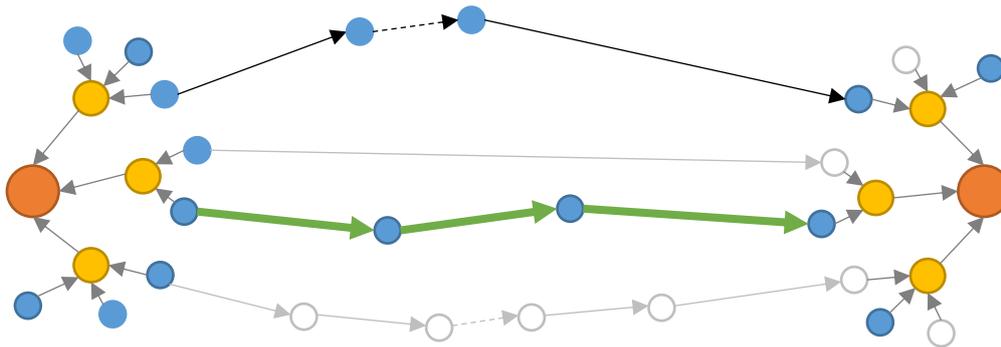


Figure 34 Schematic representation of the inverse *RP*. Greyed out stop times are not used as destinations/origins of the routing algorithm. Valid routes are highlighted in green, meta-stops in orange, child stops in yellow and stop times in blue.

This problem can be solved in two different ways. Either a forward search or a backwards search is performed. In the forwards approach, the first step entails matching all stop times at the origin's meta-stop which lie within the given arrival time range. In a second step, a Dijkstra's algorithm is run starting from any stop time at the origin to the selected stop times at the destination. Consequently, the routing algorithm returns a path which must end within the given time range.

In the backwards approach, routing is performed in the opposite direction of the edge direction. Thus, a Dijkstra's algorithm is run starting from the selected stop times at the destination to any stop time at the origin.

Inverse Range Problem

```
1      match (m:Metastop)—(st:Stoptime)
          where st.arrival_time < 08:00
          and st.arrival_time > 10:00
2      with st as origin
3      match (m2:Metastop)—(st2:Stoptime)
4      with st2 as destination
5      call apoc.algo.Dijkstra(destination, origin, "<PERCEEDES|<TRANSFER", duration)
```

6 Evaluation & Results

Current research in multimodal routing focuses on the *Nearest Neighbor Problem (NNP)* as solution to network linking (Bast et al., 2015; Dibbelt et al., 2015; Pajor, 2009). Chapter 5.4 Network Merging & Linking illustrates that this approach is not adequate for carpooling as it cannot handle the imprecise stop locations and the flexibility of carpooling offers along the route.

Focusing particularly on carpooling as a part of a multimodal network, in all conscience, no studies exist which investigate a suitable modeling approach for carpooling. In general, only two studies exist which consider carpooling and multimodal routing. Their idea is not to integrate carpooling as a full-fledged part into a network, but to substitute parts of an already existing multimodal journey.

Whereas they consider the flexibility of carpooling, imprecision of carpooling stops is not discussed. The reason for this might be the different nature of their carpooling offers. In contrast to the data set of this thesis, their offers have not been frequent, and were located on a small scale.

Depending on the characteristics of carpooling offers, it must be evaluated which modeling approach is best suited in terms of adequate network links, and thus offers improvements for all containing networks, and would thus lead to meaningful routing results.

Therefore, this section analyzes the characteristics of the crawled carpooling offers in order to assess the benefits of carpooling in a multimodal system. Furthermore, the proposed merging/linking technique is evaluated in terms of representing fuzziness and retaining flexibility. Additionally, the improvement of the overall network is analyzed. Lastly, different shortest path queries are presented with a focus on multimodal trips containing carpooling.

6.1 RO 1.1: Characteristics of Real-Life Carpooling Offers

As previously mentioned, the carpooling data set used in comparable studies (cf. Aissat & Varone (2015a)) differs drastically from the carpooling offers in this thesis. Whereas their carpooling offers are frequent and on a regional scale, carpooling offers from real-life platforms such as BlaBlaCar.de are volatile. Also, the crawled offers are on a much larger (national to international) scale. Unfortunately, the authors do not sufficiently justify their approach based on the carpooling data set. Thus, in this section, the crawled data set is analyzed and evaluated in order to give a general overview of real-life, volatile carpooling offers

Especially for a small country like Switzerland, carpooling offers cannot be restricted to a national scale, as most offers either have the start or destination in a foreign country. Hence, in this section, the crawled carpooling offers are described. In a first step, general characteristics are discussed. Later, the spatial component will be presented. Finally, the technical feasibility of real-life, volatile

carpooling offers is investigated based on the before ascertained characteristics and the quality of the crawled data. With all this steps, the time-expanded modeling approach is justified or disproved.

6.1.1 General Characteristics of Carpooling Offers

As mentioned in the introduction, carpooling tends to be a MOT usually used for long-haul journeys. Thus, it can be assumed that these offers do not reflect daily commuting. Figure 35 shows the amount of carpooling offers per weekday, which seems to conduce to the before assumption. Most journeys start around the weekend, especially on Thursday (15.6%), Friday (19.9%) and Sunday (16.3%). This implies travels of weekly residents, meaning people travelling back home for the weekends.

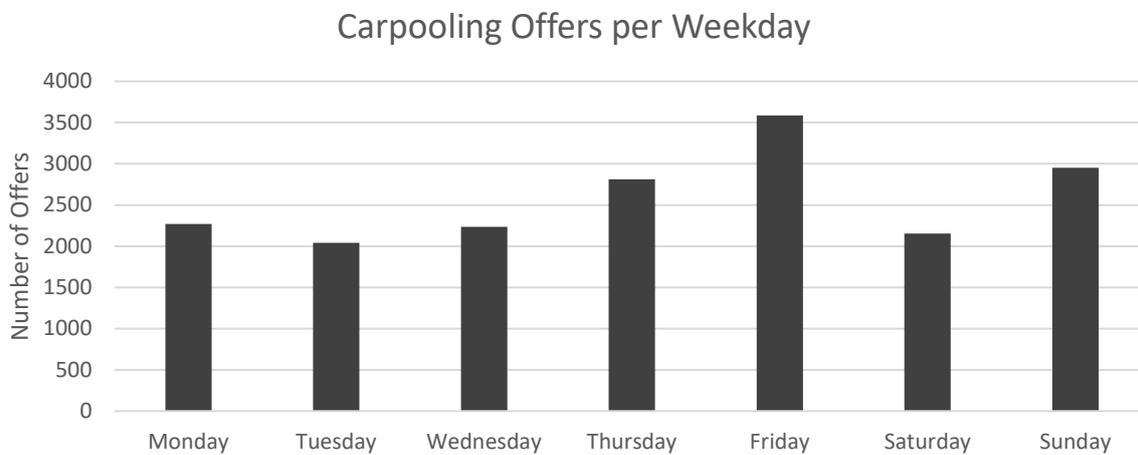


Figure 35 Number of offers on each weekday. With 20%, Friday has the most offers, followed by Sunday with 16.5%. Lowest days are Tuesday and Saturday with approximately 11%. The data set contains of approximately 18k carpooling offers and has been crawled within an 8-month period.

Most of the offers, roughly 50%, leave at around noon, indicating longer distance travel not for commuting. However, a slight increase during the evening rush hour (16:00- 18:00) can be seen (cf. Figure 36). By considering the weekdays, it is visible that these are weekend travels, as Thursday, Friday and Sunday are above the average and all other days around or below the average number. Although data before 11:00 o'clock was not available, it can be assumed that the number of offers may be relatively low, since carpooling from BlaBlaCar.de tends to be for leisure travel rather than commuting. It is important to note that offers before 11:00 o'clock were not fetched by the crawler. The reason for this is that many drivers post offers in the early morning which depart later that day. In order to not stress the servers of the crawled platform, the crawler was run only once a day for offers of that day. In order to not miss offers posted in the early morning, the crawler was scheduled at 11:00 o'clock (cf. chapter 5.1.2.1).

Considering the participation in carpooling, 7674 different drivers can be identified. Figure 37 shows the distribution of offers per driver. Most offers are served by a small number of drivers, whereas many drivers (approx. 25%) only offered a single ride. The top 5% drivers offered 30% off all trips, and the top 10% offered 43% of all trips (cf. Table 10). The participation in carpooling thus follows the long tail theory.

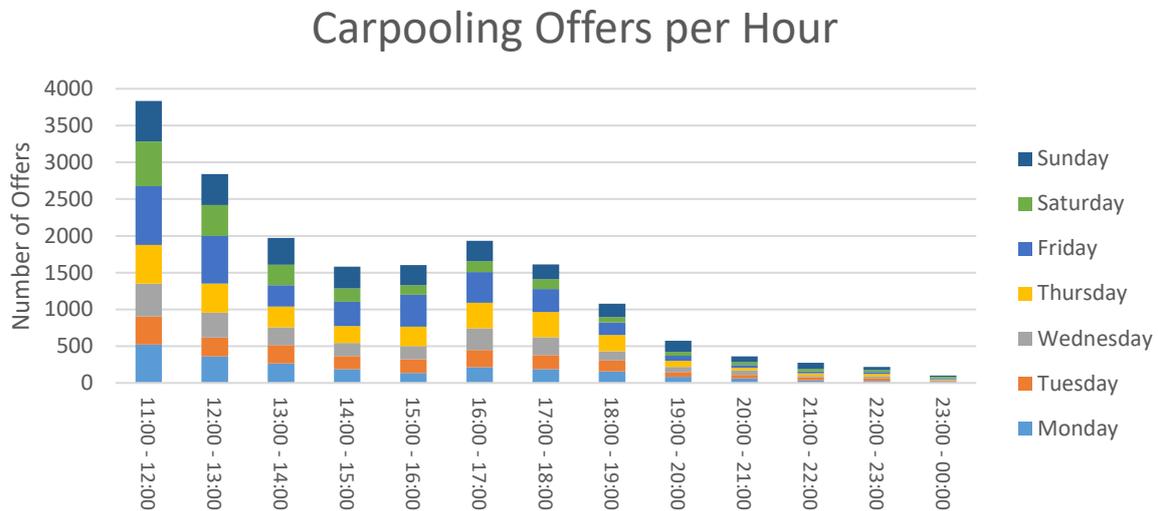


Figure 36 Number of offers in a one hour interval of each weekday.

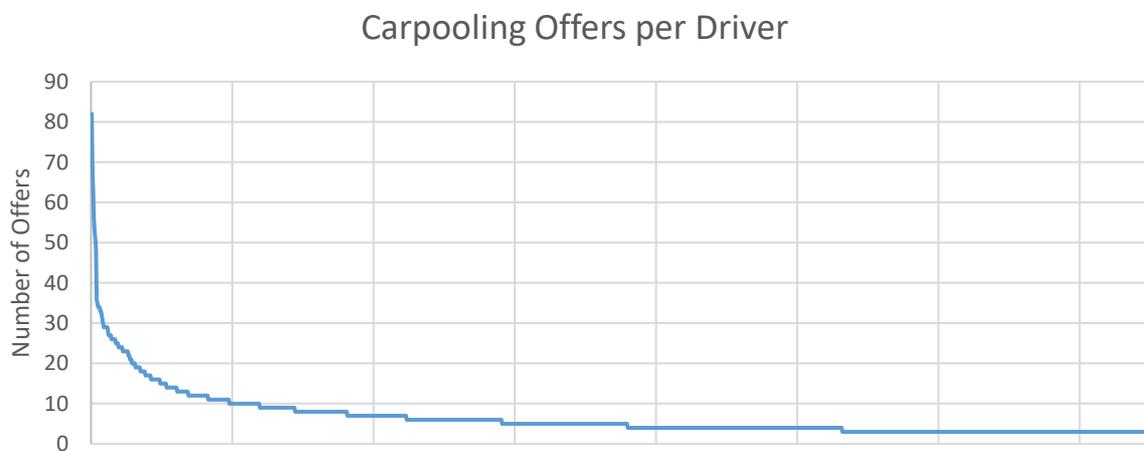


Figure 37 Distribution of offers per driver. Most offers are offered by a small number of drivers, whereas many drivers (approx. 25%) only offer a single ride. The distribution follows the Long Tail Theory.

Table 10 Number of drivers and offers and the percentage of the total amount.

	5%	10%	25%	50%	100%
Driver	384	767	1919	3837	7675
Offers	5538	7710	11189	14208	18043
% of total	30.69	42.72	61.99	78.73	100%

6.1.2 Spatial Characteristics of Carpooling Offers

This data set contains only offers that have at least one stop location within the borders of Switzerland. However, the crawled carpooling offers cannot be restricted to a country scale. Thus, this section shows the spatial distribution of carpooling offers across Europe. In a first step, the stop locations are described, followed by the routes.

It is important to mention that the same subset of 2k carpooling offers as used in the implementation is analyzed.

Stop locations: As previously illustrated, carpooling offers tend to be for longer journeys. The average length is 480 km, indicating that most of the offers extend across the borders of Switzerland. Figure 38 shows a choropleth map of European countries, classified by the number of stop locations. It becomes visible that especially neighboring states have a high number of stop locations.

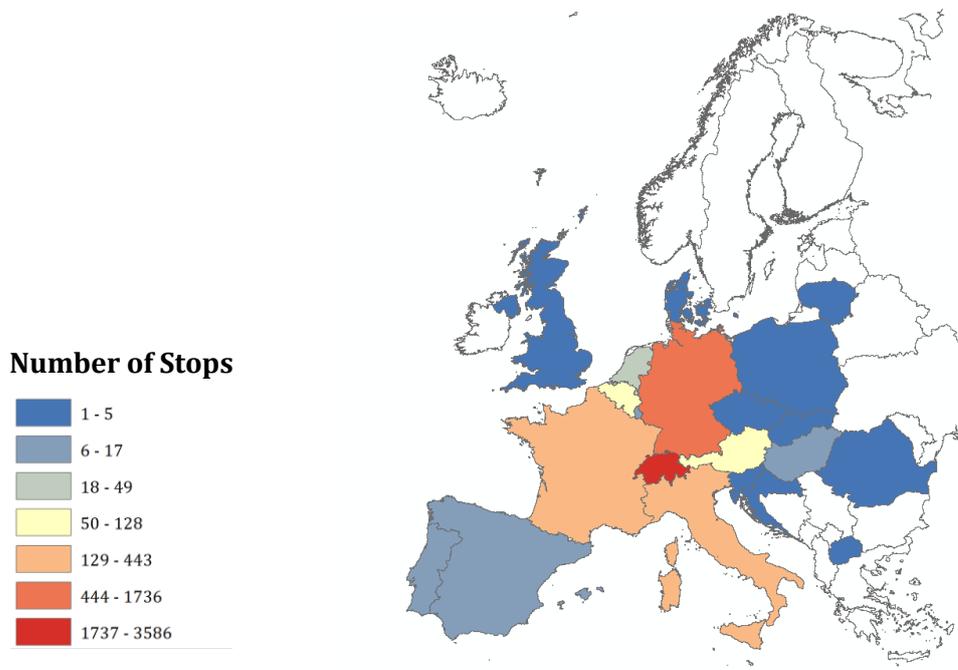


Figure 38 Number of stop locations per country across Europe. Neighboring states of Switzerland exhibit a particularly high number of stop locations.

A closer look at the different countries in Table 11 shows that twice as many offers start in Switzerland than they end. In return, the neighboring states have far higher numbers of end locations than start locations. In addition, Switzerland shows a high number of via locations, indicating Switzerland to be a transit country. Due to the high difference in start and end location, offers tend to be single service.

Table 11 Top 10 most important countries according to stop locations.

	Total	%	Start	Via	End
Switzerland	3586	100	1426	1451	709
Germany	1736	48.4	462	347	927
France	443	12.4	108	114	221
Italy	383	10.7	105	58	220
Austria	128	3.6	26	69	33
Belgium	115	3.2	51	1	63
Netherlands	49	1.3	22	2	25
Hungary	17	0.5	9	0	8
Lichtenstein	17	0.5	2	15	0
Portugal	17	0.5	16	0	1

14 countries with 11 (2x), 5 (2x), 3 (1x), 2 (5x) and 1 (3x) stop locations are excluded from this table.

28 countries have 0 stop locations.

Analyzing the numbers concerning Switzerland shows that the railway network has a much higher density of stop locations than carpooling. Especially in the “Mittelland”, the railway network as well as the carpooling network have a high density. Even though the railway network has far more stop locations, carpooling serves additional areas that are not accessible by train. The southern region “Wallis” is a good example. The Wallis is surrounded by mountains and thus not well suited for the railway network. However, it must be kept in mind that these areas can also be reached by bus. Nevertheless, this fact is another argument for using carpooling as a complement to the railway network. Figure 39 illustrates the prevalence of carpooling stops and train stations.

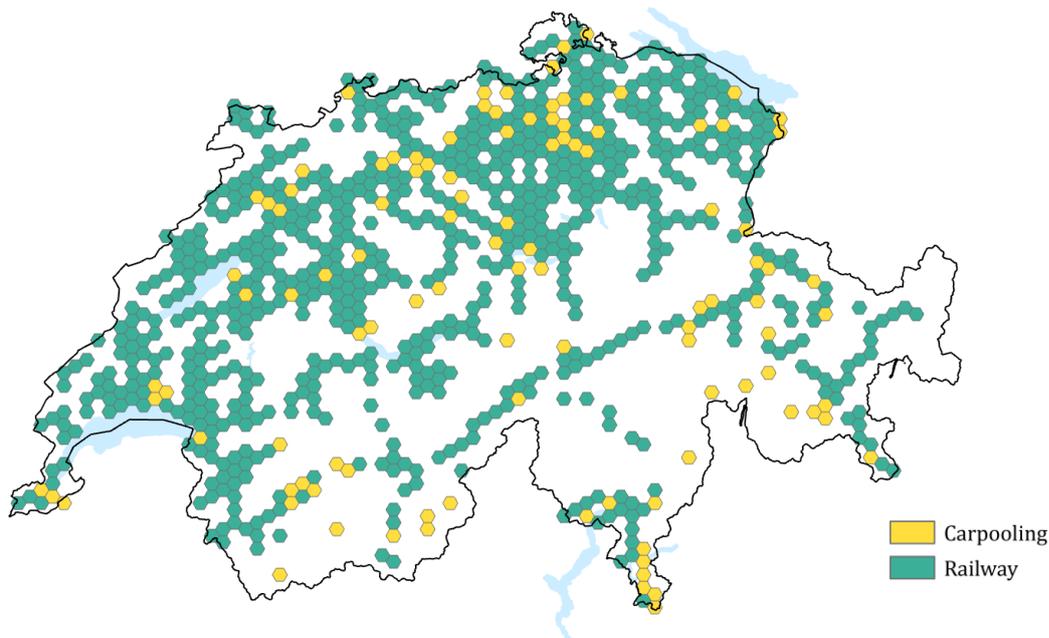


Figure 39 Stop locations of the 2k carpooling offers and all train stations of the Swiss railway network within the borders of Switzerland for a 5 km tessellation. Hexagons indicate the occurrence of stop locations. Yellow and green colors indicate which MOT has more stops within a hexagon.

Routes: Figure 40 shows a subset of around 2000 carpooling routes across Europe. The actual routes have been generalized using the step segments returned by the Directions API (cf. 5.1.2.2 Data Enrichment). It becomes visible that especially central Europe has a high coverage of carpooling routes. In addition, a trend in north-running offers can be seen. Furthermore, east-west and north-south connections seem to develop. Carpooling routes also serve long-haul travels, indicated by routes to Portugal or Turkey.

On a national scale, Switzerland shows a high density of carpooling routes along the east-west and north-south axis. This correlates with the Swiss highway network, as two major highways cross Switzerland in these directions. Also, the figure shows that a high number of offers enter or leave Switzerland, which indicates the larger scale of carpooling.

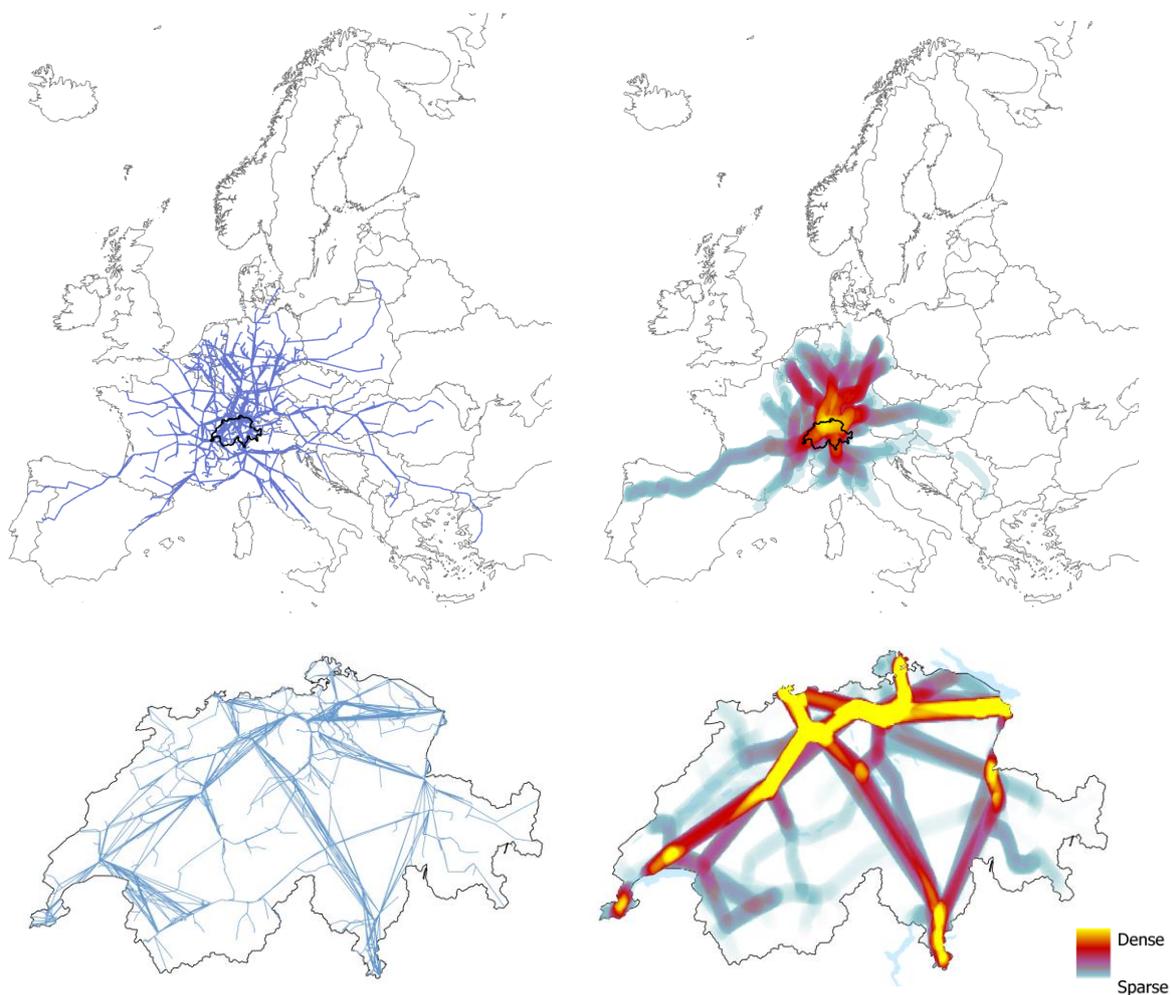


Figure 40 Subset of 2k carpooling routes (generalized) across Europe/Switzerland (left) and a line density calculation (right). A clear tendency of north-running offers can be seen. Switzerland shows a high density of carpooling routes along the east-west and north-south axis (bottom).

Connections: In this thesis, the goal is to combine the Swiss railway network and carpooling. Thus, it is of interest whether or not carpooling can provide connections not already serviced by the railway network. Based on the subset of 2000 carpooling offers, connections from a train station, to a train station, and completely independent connection have been analyzed (cf. Table 12). Carpooling exploits 71 new connections starting from a train station and heading to a stop unknown to the railway network, 47 from a stop unknown to the railway network to a train station, and 878 independent connections. Independent means that no train station is involved and thus neither the origin nor the destination can be reached with public transportation. These connections do not include *POA* stops, but are calculated based on the predefined origin, via and destination stops. In order to identify which carpooling stop and public transportation stop are approximately at the same location, the approach for spatially allocating carpooling stops from chapter 4.3.1 has been used.

Table 12 The number of connections not covered by the railway network of Switzerland. PT -> CP indicates the possibility of reaching the origin by train, but not the destination. CP -> PT indicates the possibility of reaching the destination by train, but not the origin. CP -> CP indicates that neither the origin nor the destination can be reached by train.

PT->CP	CP->PT	CP->CP
71	47	878

6.2 RO 1.3: Merging & Linking

The in chapter 4.3 Model Merging & Linking proposed merging/linking technique aims to retain or restore the fuzziness of carpooling stops and the flexibility of driving detours. A striking benefit of this approach is the possibility to link imprecise carpooling stops to multiple public transportation stops. Moreover, it can exploit new stop locations along a carpooling route by analyzing an intersection with a drive time area of a public transportation stop. The drive time areas are calculated using a road network and an approximation of maximum allowed detour times. Thus, an intersection with such an area indicates the possibility of picking-up or dropping-off a passenger. It can be assumed that the more links between the carpooling and the railway network are established, the more transfers are possible and thus the multimodal network will exhibit larger coverage and connectivity.

In chapter 5.4 Network Merging & Linking the proposed linking approach with real-life data was implemented. This section therefore analyzes and evaluates the resulting multimodal network based on:

- Stop allocation
- Transit possibilities
- Improvements of the overall network
- Quality of queries

6.2.1 Multimodal Graph

The resulting multimodal graph consist of approximately 955k nodes, whereas most of them (890k) belong to the railway network. Additional 2.55M relationships have been created. Table 13 shows the number of different types of nodes for carpooling, railway, and the total amount. It can be stated that the number of nodes is fairly low. A set of 2k carpooling offers produces only 64k nodes. Crucially, the most nodes are created for stop times. Depending on the frequency of an MOT, the number of trips can be high as well. In the present experiment, the railway network offers 71k trips, which is roughly 30x more than carpooling. Nevertheless, the impact of carpooling on the multimodal network is high in some areas. This will be shown later in this section.

Considering relationships/edges as shown in Table 14, for each node, approximately 2.5 relationships are created. It must be mentioned that transfer edges were excluded from this table. Transfer edges will be discussed later in this section. Again, stop time related edges cover the largest part of all relationships. This is crucial and also explains the 2.5 times higher number, as each stop time is related to a trip (:PART_OF_TRIP), a stop (:LOCATED_AT), and, excepting origin and destination, to a successor (:PRECEDES). Transfer edges will further massively increase the number of edges at a stop time.

Table 13 Number of nodes of different types for all contained MOT.

Node Type	Carpooling	Railway	Total
Agency	1237	62	1299
Route	2211	28454	30665
Trip	2211	69150	71361
Stoptime	51198	789330	840528
Metastop	-	1912	1912
Track	-	2771	2771
Road	7613	-	7613
Stop	7613	4683	12296
Total	64'470	891'679	956'149

Table 14 Number of relationships of different types for all contained MOT.

Edge Type	Carpooling	Railway	Total
:OPERATES	2211	28454	30665
:USES	2211	69150	71361
:PART_OF_TRIP	51198	789330	840528
:LOCATED_AT	51198	789330	840528
:PART_OF	6652	2771	9423
:PRECEDES	48987	720180	769167
Total	162'457	2'389'215	2'551'672

6.2.2 Stop Allocation & Exploiting

Stop Allocation: A major part of the linking approach is to connect the imprecise carpooling stops to train stations. Rather than linking only one stop to another, the benefits shall be in connecting a carpooling stop to multiple stops, hence retaining fuzziness. Thus, Table 15 shows the number of links between carpooling and the railway network. In the subset of 2k carpooling offers, 1061 main stops (origin, defined via, destination) and 6552 *Points of Action (POA)* could be identified. The Swiss railway network has 4683 train stations.

Table 15 illustrates that with the use of drive time areas, 296 carpooling main stops can be linked to train stations (CP->Train) with a total of 969 links (#Links). Subsequently, a main carpooling stop connects on average to 3.27 different train stations (\emptyset Links). Concerning *POA* stops, 1619 out of 6552 could be linked to train stations, with an average of 3.51 links per *POA*. Consequently, 1619 new stops along carpooling routes could be exploited. It is important to note that a large number of Main stops (765) and *POA* stops (4933) could not have been linked to train stations (No Link). This can be explained by the fact that large parts of carpooling routes lie outside of the borders of Switzerland where no railway network was available.

In order to draw a comparison to the state of the art approach *NNP*, a small calculation has been performed. The *NNP* has been solved for the 1061 main carpooling stops out of the set of 7613 stops (Main & *POA*) and the 4683 train stations. Thereby, a common approach is to define a threshold distance (cf. Dibbelt et al. (2015)). The *NNP* has been solved for maximum Euclidean distances of 1 km, 2 km, and 5 km. The author argues that a drive time area may span somewhere between these 1 to 5 km. The *NNP* for a threshold of 1 km connects to 199 main carpooling stops, to 254 main stops for 2 km, and to 306 stops with a 5 km distance (*NNP* 1-5km). Crucially, only one link per carpooling stop is initiated. The number of connected carpooling stops to train stations may be lower for the *NNP* of 1 km and 2 km, as a drive time area of 15 minutes can span larger than 2 km. Furthermore, the generalization approach presented in chapter 5.2.2 has potentially removed stops close to a train station. The threshold of 5 km equals the presented approach the most. It can therefore be assumed that a driver can drive up to 5 km within 15 minutes. The state of the art in linking stop locations using the *NNP* only links known stops. In the present case, these are the main stop locations (origin, via, destination), derived from the carpooling offers. Hence, the *NNP* are not solved for *POA* stops, as these stops are part of the proposed linking technique (cf. 4.3 Model Merging & Linking).

Table 15 Number of connected carpooling stops (Main, POA) and the number of established links in comparison to the NNP. "No Link" represents the number of carpooling stops not linked to train stations, "CP->PT" the number of stops connected to train stations, "#Link" the number of established links between carpooling stops and train stations and " \emptyset Links/CP" indicates the average number of train stations linked to a carpooling stop. NNP 1-5 km indicate the number of linked carpooling stops and train stations for different thresholds.

Stop type	CP	Train	No Link	CP -> Train	#Links	\emptyset Links	NNP 1km	NNP 2km	NNP 5km
Main Stops	1061	4683	765	296	969	3.27	199	254	306
POA	6552	-	4933	1619	5683	3.51	-	-	-
Total	7613	4683	5698	1915	6652	3.47	199	254	306

Exploiting: As previously mentioned, the proposed merging technique shall also exploit new stops along a carpooling route and thus allow transfers to either the railway network or to other carpooling offers connected to train stations. Consequently, carpooling trips have been analyzed in order to evaluate the average number of possible transfers on a trip.

On average, as shown in Table 16, a carpooling trip consist of 2.914 main stops and 20.25 POA stops. Median values are slightly lower due to a few ultra-long journeys. During an average trip, a passenger may transfer to 6.23 different train stations at main stops or to 34.23 different stations from POA stops.

Table 16 Number of main and POA stops and the number of relations to train stations. Median values are slightly lower due to ultra-long trips exhibiting many stops.

	Stops		Relations	
	Average	Median	Average	Median
Main Stop	2.914	3	6.23	5
POA	20.25	19	34.23	27
Total	23.16	22	40.44	34

A more detailed analysis shows how many different train stations could be reached per 1, 10, 25, and 50km. Trips were classified based on their length. As this thesis manly focuses on a national scale, trip lengths of up to 500 km are of interest (Switzerland: E-W 350km, N-S 250km). Table 17 shows that in general, the shorter the trip, the more train stations can be reached per distance. Especially on ultra-short trips up to 25km, a passenger could transfer to train stations every kilometer. This seems to be confusing as one would assume that the longer the trip, the more train stations are passed. However, by reconsidering the input for the linking technique, this fact is crucial. In order to link carpooling and train stations along a route, POA are implemented. On longer trips, one usually drives on highways and thus train stations cannot always be reached, even though the route intersects a drive time polygon. Long-haul trips outside the borders of Switzerland could also not be connected to train stations, as only the Swiss railway network were used. In order to evaluate long-haul trips, railway networks of all affected countries would need to be considered.

Nevertheless, on average, a passenger may change every 10 km to 3, every 25 km to 6.42, and every 50 km to 10.25 different train stations. Again, the median differs strongly.

Table 17 Number of links to train stations per 1, 10, 25 and 50km of trips of different lengths. Short trips show an especially high number of links.

Length of trip	AVG Dist.	1km	10km	25km	50km
0-10	4	1.09	-	-	-
10-25	18	0.60	6.00	-	-
25-50	38	0.62	6.19	15.49	-
50-75	65	0.45	4.51	11.26	22.53
75-100	88.70	0.40	3.99	9.97	19.94
100-250	172.71	0.25	2.45	6.13	12.27
250-500	357.98	0.13	1.28	3.20	6.41
500-750	627.06	0.06	0.63	1.59	3.17
750-1000	847.82	0.05	0.54	1.36	2.72
>1000	1371.41	0.09	0.94	2.34	4.69
Average	426	0.37	2.95	6.42	10.25
Median	349	0.12	1.15	2.72	4.69

6.2.3 Transfers

An important part of a transportation network is the possibility of transfers. Therefore, the process of creating variable transfers for the railway network and static transfers for carpooling has been elucidated in chapters 4.1 & 4.3.2. Rather than creating an additional transfer node between time nodes, labeled edges (aka. relationship types) were used. Thus, the number of nodes as well as edges can be conceptually reduced.

The process of creating transfers is demanding and time-consuming. The calculation and creation of all possible transfers took up to half a day on a Lenovo ThinkPad T430s with an Intel i7-3520 CPU @ 2.9 GHz and 16 GB memory. The more connections exist, the longer the processing. It is essential to mention, however, that in this thesis, a new multimodal network was built and thus all possible transfers had to be calculated. Adding single or just a few offers does not demand recalculating every transfer, but only those from and to the new connection.

By evaluating the multimodal network built in this thesis, a total of 5.57M transfer edges were created, representing all possible transfers. It is crucial that most of the transits arise from the railway network, due to the high number and constant frequency of trips (cf. Table 18). Thus, 4.8M edges belong to this network. In comparison to carpooling, this amount is tremendous, where only 123k transfers are possible.

Considering mode change, the number of transfers from carpooling to train and vice versa is similar. Approximately 75,000 transfers exist at main stops and 240,000 at POA stops. Consequently, mode changes are more than twice as usual than transferring to another carpooler.

Table 18 Number of transfer edges created between carpooling and train trips.

	Train -> Train	CP -> CP	CP -> Train	Train -> CP
Main Stops	-	16,795	75,067	76,987
POA	-	106,394	244,081	240,343
Total	4,812,926	123,189	319,148	317,330
Total Number of Transfer in the Network			5,572,593	

A closer look at possible transfers along a route reveals that carpooling and the railway network behave similarly. Figure 41 shows the normalized number of possible transfers for different lengths of routes, indicated by the number of stops, and for mode changes or same mode transfers. Absolute numbers of transfers have been normalized by scaling them to a range of 0 to 1.

Being on a carpooling route, the number of transfers remains equal over an increasing number of stops. This accounts for changes to another carpooler as well as for a mode change to the railway network. Fluctuations occur on routes with a very high number of stops (>45). This fluctuation may have two reasons: either the route is long, but only a small part lies within Switzerland (low transfer value), or the route has a long distance within Switzerland (high transfer value).

Considering train routes, especially routes with a length between 15 and 25 stops, show a high possibility for either changing to carpooling or staying on trains. It can be assumed that such routes represent regional trains and thus have a high number of stop locations allowing transits. Consequently, carpooling was also linked not only to major train stations but also to smaller regional stations.

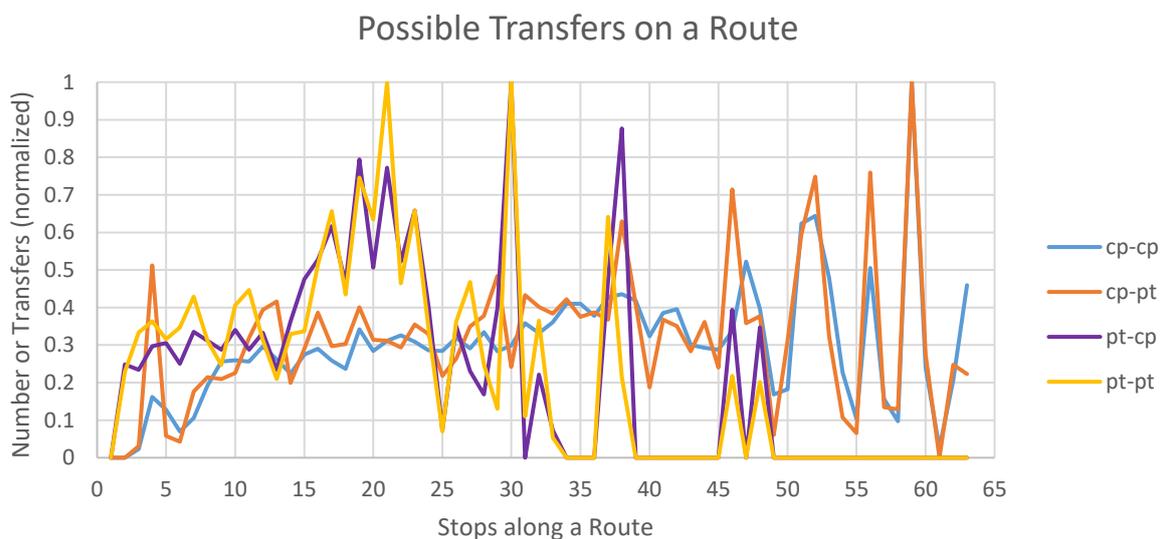


Figure 41 Normalized number of possible transfers along a route of different length indicated by the number of stops.

6.2.4 Network Improvement

This thesis focuses on integrating carpooling as a full-fledged part into a multimodal routing system. Thus, this shall also lead to a general improvement of the multimodal network. As elucidated above, carpooling has the ability to serve areas where no trains operate. The flexibility of carpooling allows to drive detours, thus exploiting new connections and consequently enables more transfer possibilities.

In this section, the impact of the subset of 2k carpooling offers on the multimodal network is evaluated by analyzing the importance of stop locations in the graph.

Common approaches to evaluate the importance of nodes in a graph are Centrality measures. As many different Centrality measures exist, some algorithms, which could be used to evaluate important nodes in a transportation network are listed below:

- **Degree Centrality:** Degree Centrality calculates a score for a set of nodes based on their associated relations. In a *multidigraph* graph, two measures, in-degree and out-degree, can be calculated. Thus, in Degree Centrality, a node is important if it links to many other nodes or many other nodes link to it (Franceschet, 2014).

In a transportation network, Degree Centrality can be used to evaluate if one network improves the other based on the additional number of connections at a stop.

- **Betweenness Centrality:** Betweenness Centrality (cf. Freeman (1977)) is a measure to quantify how many times a node lies on paths between other nodes. A node with a high Betweenness value is thus important for the overall network, as a lot of information passes through it. Hence, removing such a node may lead to a way less interconnected network (Franceschet, 2014). The requirement of calculating many shortest paths results in a high complexity of $O(n * m)$.

In a transportation network, Betweenness Centrality can be used to detect system critical stops. As an example, in Switzerland, the train station of Olten lies between 4 major cities (Basel, Bern, Lucerne, and Zurich). Connections between these cities pass through Olten. Consequently, if the train station of Olten would be impassable, all connections are disrupted.

- **Closeness Centrality:** Closeness Centrality calculates the mean distance between nodes. Thus, a node receives a high score if it is close to many other nodes (Franceschet, 2014). In a transportation network, closeness centrality can be used to detect central train stations. As an example, a large city may have a main station and multiple smaller stations (subways, regional trains, etc.) close to it. Thus, the main station will receive a high closeness score. Consequently, Closeness Centrality can expose potential hubs based on closeness.

Similar to Betweenness Centrality, Closeness Centrality is also bound to the complexity of $O(n * m)$.

- **PageRank Centrality:** PageRank was developed by Sergej Brin and Larry Page to be used in Google Search. The PageRank algorithm calculates a score of a node based on three different factors: The number of incoming links, the link propensity of the linkers, and the centrality of the linkers. Thus, in PageRank centrality, a node can be considered important if it is linked by important nodes or if it is highly linked in general.

Originally, the PageRank algorithm was developed to rate the importance of websites and thus improve Google Search. Nevertheless, this measure can be also used in Transportation networks to expose hubs (cf. Chan & Teknomo (2016)). Other than Closeness Centrality, the hub is defined by the importance of other nodes linking to this node.

The different centrality algorithms can all be used to analyze a transportation network in this thesis, as the aim is to detect improvements of connections at stops. Furthermore, important stop locations shall be exposed. Subsequently, Degree Centrality and PageRank Centrality were selected for use. It is important to mention that especially Betweenness Centrality could be used to detect if important stop locations in a multimodal network changed, compared to the individual networks. Unfortunately, Betweenness Centrality is a global graph algorithm using shortest paths. The complexity is therefore $O(n * m)$. Within the scope of this thesis, it was not possible to calculate this measure with the given computational power.

Degree Centrality: As mentioned above, Degree Centrality rates nodes according to the number of edges coming either in or out. Thus, this measure can be used to evaluate if carpooling linked to train stations can significantly improve the number of connections at a stop.

Figure 42 illustrates the number of train stations and the percentage of improvement. Only train stations which are linked to at least one carpooling stop and exhibit an improvement $>0\%$ are considered. Most of the train stations that contain carpooling offers have not been improved higher than 1% for either incoming or outgoing connections. “Both” describes improvements where the direction is neglected.

Although most stops (55%) could not be improved more than 1 %, some stops show higher numbers. In general, 25% of the stops could be improved by $>2\%$ and 7.5% of the stops could be improved by more than 5%.

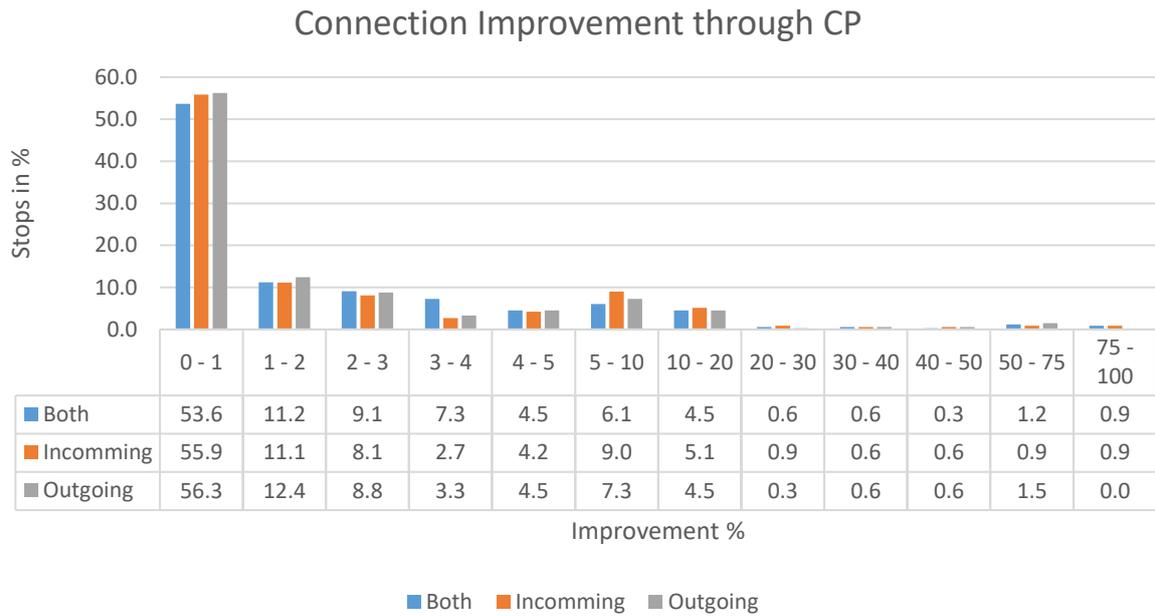


Figure 42 Percentile improvement of connections based on Degree Centrality. Around 55% of all stops holding at least one carpooling stop show an improvement of less than 1%.

Considering the spatial location of the improved stops, Figure 43 shows high improvements for train stations close to the Swiss border (south, south-west) and within the Alps. This fact seems to be crucial. On the one hand, the railway network is not as well developed in the mountains as the road network and, on the other hand, carpooling offers cover a much larger area than only Switzerland.

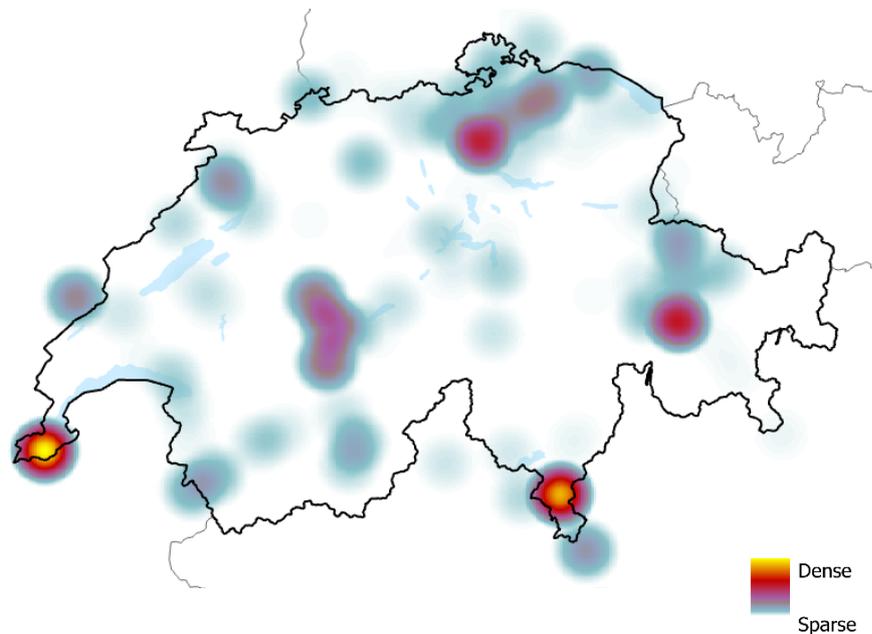


Figure 43 Heat map of the top 25% of stop locations improved more than 5%. These stops are mostly concentrated in border regions and in alpine areas.

PageRank: It was previously elaborated on that the PageRank Centrality can be used to detect hubs in a transport network based on the importance of a node linked to other nodes. Thus, the PageRank is used to find differences between the individual networks carpooling and railway, and the multimodal network. Figure 44 shows three maps for each network. Stops with the highest PageRank score (> 1.5 Standard Deviation) are indicated by a yellow dot. The underlying heat map shows the density of stop locations, weighted by their PageRank score. Thus, it can be assumed that dense areas are more important for the network.

The top-left map shows the situation for the carpooling network. The most important areas of the network appear to be either large cities (Zurich, Berne, Lucerne, etc.) or border points (e.g. top-right, south). This further indicates that carpooling operates on a large scale.

The top-right map shows the situation of the railway network. Compared to the carpooling network, less dense areas are visible. Exceptions are Zurich and Lausanne. The network is spread more equally, and is thus more continuous. In contrast to the carpooling network, regions close to the border are less important. Alpine regions (in the center and East of Switzerland) seem to be less important.

Considering the multimodal network, it becomes apparent that both networks have been merged. Border regions are more important compared to the railway network a finding that seems to correlate with the fact that public transportation stops in these areas have been improved the most. In addition, alpine regions have developed as well. This entails that more stop locations gain a high PageRank score. Consequently, the multimodal network retained the importance of border regions as well as the continuity of the public transportation network. Additionally, larger parts of Switzerland are important to the network, indicating better coverage. Furthermore, it can be stated that, with an increasing number of important nodes, the network is more stable, meaning that it is less affected by a failure of one or more stops.

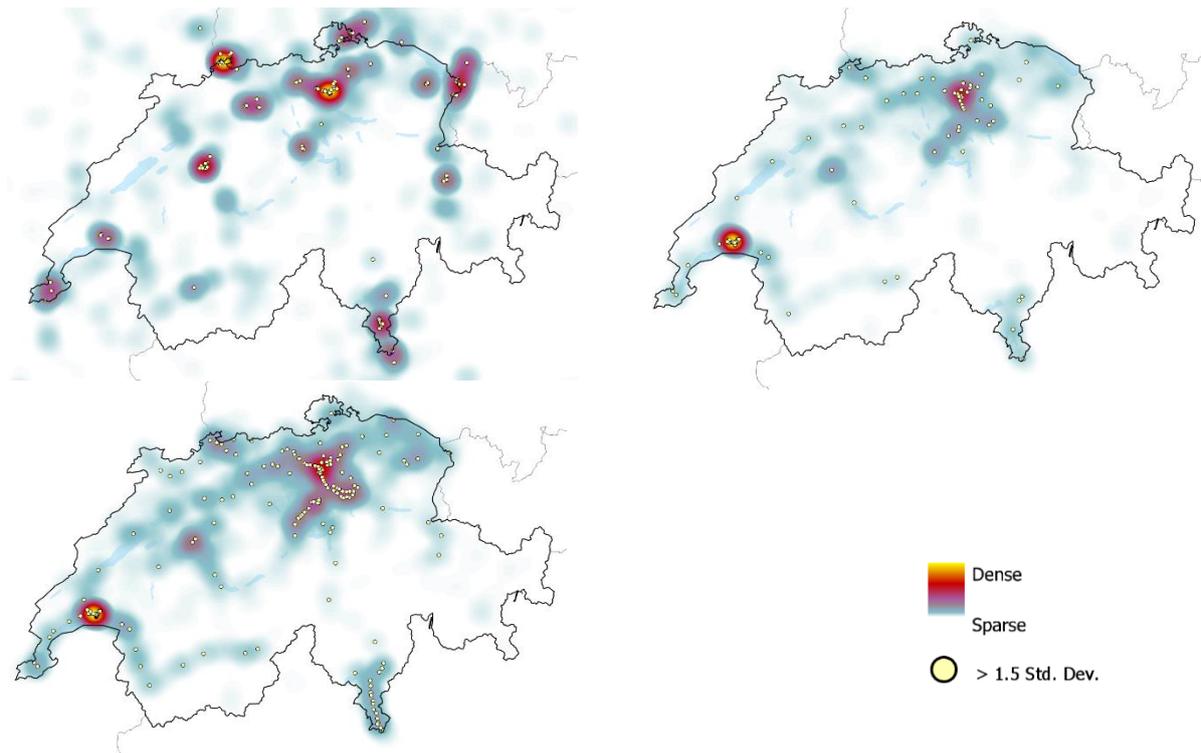


Figure 44 Heat maps of stop locations weighted by their PageRank score. Top-left shows the individual carpooling network, top-right the individual railway network and bottom-left the multimodal network.

6.2.5 Queries

In this section, query results are presented based on an example of a trip from Olten to Basel (cf. Figure 47) and a fictional scenario for a trip from Olten to the Expo in Milano. A further focus is directed to two additional examples for multimodal trips containing carpooling. The aim is to show that with the system implemented in this thesis, meaningful shortest paths for either the railway network, carpooling or a combination of both can be retrieved. As a full multimodal network was created in this thesis, the focus lies especially on the latter.

However, it is to mention that routing algorithms are not a major part of this thesis. Thus, focus is not put on either speed-up techniques or multicriteria optimization. Nevertheless, it will be shown that, indeed, meaningful shortest paths can be derived from the implemented system, which yield correctness within the given system.

Chapter 5.5 Querying elucidated how all direct connections between two stops can be retrieved. Solving the *Range Problem (RP)* was illustrated. Thus, in the following, results for both problems will be demonstrated. As the graph consists of different MOT, it will present results for carpooling and public transportation only, as well as a multimodal trip.

All Direct Routes: When querying for all direct routes, the time component is subsidiary. Thus, one can simply find all relations between two stop locations via stop times. However, paths containing transfer edges shall be pruned. Thus, no multimodal trips can be found.

As the number of direct routes between two stops can be very large, only a subset of two routes is shown in the results. Nevertheless, a full query may provide many more connections.

Figure 45 shows an example of a query for direct carpooling routes between Olten and Basel. Blue nodes represent stop locations along the journey. Green nodes are the stop times (arrival time, departure time) at a stop, related to a yellow trip node. Pink nodes are route and agency (driver) nodes. In this example, both carpooling routes follow the same sequence of stop locations from Olten to Basel.

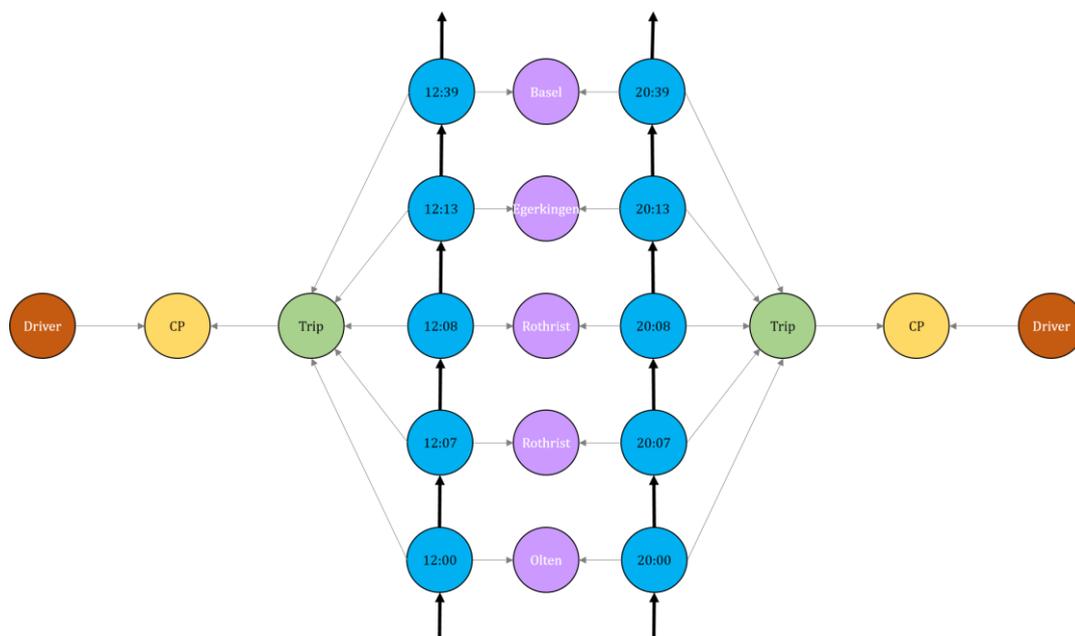


Figure 45 Direct routes from Olten to Basel using carpooling. Blue nodes, belonging to a trip (green) define arrival and departure times at a stop (purple). Trips belong to a route (yellow) offered by a driver (red).

Table 19 shows these two routes in a tabular form. Rothrist is listed twice. This does not indicate a transfer, but rather the fact that a user may hop-on or hop-off at two different points in Rothrist. These two stops are due to the generalization approach (cf. 5.2.2) at least 1 km apart. The price listed for both routes is the price a driver requests for the whole advertised trip.

Table 19 Schedule of the carpooling routes from Olten to Basel, visualized in Figure 45.

Time Route 1	Time Route 2	Stop
12:00:00	20:00:00	Olten
12:07:19	20:07:19	Rothrist
12:08:33	20:08:33	Rothrist
12:12:58	20:12:58	Egerkingen
12:39:38	20:39:38	Basel

Figure 46 shows a query result for the same route between Olten and Basel, but this time by train. In this example, two different trains were found. The first one on the left has no stops in between, whereas the second route on the right stops in Liestal. Table 20 shows the exact schedule of these routes from Olten to Basel.

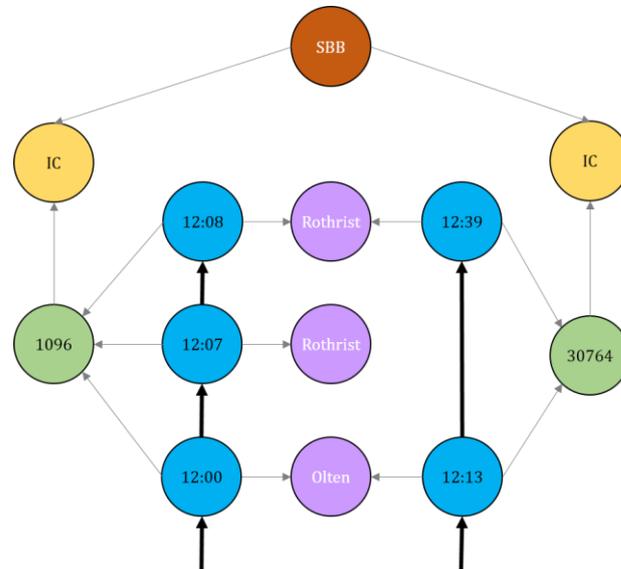


Figure 46 Direct routes from Olten to Basel by train. Blue nodes, belonging to a trip (green) define arrival and departure times at a stop (purple). Trips belong to a route (yellow) offered by a driver (red).

Table 20 Schedule of the train routes from Olten to Basel, visualized in Figure 46. Route 1 has an additional stop in Liestal, whereas route 2 drives directly to Basel.

Time Route 1	Time Route 2	Stop
23:33:00	19:06:00	Olten
23:48:00		Liestal
23:59:00	19:33:00	Basel



Figure 47 Carpooling and railway routes from Olten to Basel

Range Problem: In chapter 5.5 Querying, it was illustrated how the *RP* can be solved within the presented network. By simply specifying a set of stop times at the origin, the shortest path starting from one of these points can be search for. Thus, the path must start within the given time range.

In the following, a query result based on an example is presented. A user living in Olten wants to visit the Expo in Milano (cf. Figure 48). Thus, he/she must travel from Olten to Rho Fiera Milano, the train station of the Expo. As he/she will stay in Milano during the entire weekend, the plan is to arrive in the early evening of Friday. Thus, the trip shall start in the early afternoon.

Querying for a public transportation trip starting at around 14:30 o'clock (14:15 – 14:45) results in the route presented in Table 21. In total, the user has to transfer 3 times: Arth-Goldau, Monza, Milano Porta Garibaldi. According to the SBB, this trip costs CHF 102 without any reduction.

It is to mention that this trip information has been retrieved from the SBB online schedule¹⁹, as the developed system does not deliver pricing information for the railway network.

Table 21 A train route from Olten to Rho Fiera Milano. Three transfers are needed at Art-Goldau, Monza and Milano Porta Garibaldi.

Departure	Arrival	Stop	Transfer	Price
14:30:00		Olten		
15:46:00	15:50:00	Arth-Goldau	Yes	CHF 102
18:21:00	18:32:00	Monza	Yes	
18:51:00	19:02:00	Milano Porta Garibaldi	Yes	
	19:11:00	Rho Fiera Milano		

An alternative to the public transportation journey above can be retrieved from the implemented system. By additionally defining that trips should start and end with a carpooler, queries can be made for carpooling trips. Unfortunately, no carpooling only trip was available at around 14:30. Consequently, the time range was extended to 15:00, resulting in a direct trip starting at 14:50 in Olten and arriving at 18:10 in Rho Fiera Milano. The journey is offered by Justus-Florian S in a Mercedes Benz C230 for a price of CHF 16 (cf. Table 22). The exact query can be found in the Appendix.

¹⁹ <http://www.sbb.ch/> - Online schedule of the Schweizerischen Bundesbahnen

Table 22 Schedule of a carpooling trip from Olten to Rho Fiera Milano. This is a direct connection. Hence no transfers are needed. Additionally, the driver, his car and the requested price are presented.

Departure	Arrival	Stop	Driver	Car	Price
14:50:00		Olten			
14:57:19	14:57:19	Rothrist			
15:27:04	15:27:04	Luzern Allmend			
15:40:55	15:40:55	Kriens Mattenhof	Justus-	MERCEDES-	CHF 16
17:42:45	17:42:45	S. Antonino	Florian S	BENZ C 230	
17:50:37	17:50:37	Chiasso			
18:05:57	18:05:57	Around Como, IT			
	18:09:37	Rho Fiera Milano			

Although no carpooling only trip was available at around 14:30, the implemented system can be used to search a multimodal path. Table 23 presents the trip information returned by the system. The trip starts with the same train as in the public transportation trip, but the user has to leave the train in Lucerne and change to a carpooler (Mahite O) which brings him to Chiasso where the user again must transfer to another carpooler (Justus-Florian S), taking him directly to Rho Fiera Milano. In Chiasso, the user has an extended stop (arrival 17:23, departure 17:50). Even though the returned route is correct, the system should suppress such long waiting times. An explanation of this problem can be found at the end of this section.

This multimodal trip starts within the desired range and requires two transfers. Consequently, less transits are required than in the public transportation trip. The user will arrive at the destination earlier. Considering the price, the first train segment costs, according to the SBB, CHF 22. Additionally, the first carpooling segment prices at CHF 9.60 and the second at CHF 16. The total cost of the trip is CHF 47.6. It is important to note that the price listed for carpooling segments is the price a driver requests for his whole advertised trip and not only for this segment. Nevertheless, the provided multimodal trip is cheaper than the public transportation trip.

Table 23 Schedule of a multimodal journey from Olten to Rho Fiera Milano. The trip requires two transits (Train -> Carpooling -> Carpooling).

Departure	Arrival	Stop	Transfer	Mode	Driver	Car	Price
14:30:00		Olten		Train			CHF 22
	15:05:00	Luzern	Yes	Train			
15:13:32		Luzern		CP	Mahite O	CITROEN	CHF 9.60
17:15:22	17:15:22	S. Antonino		CP		XSARA	
	17:23:14	Chiasso	Yes	CP			
		Chiasso		CP			
17:50:37		Chiasso		CP	Justus-	MERCEDES-	CHF 16
18:05:57	18:05:57	Around Como, IT		CP	Florian S	BENZ C 230	
18:09:37	18:09:37	Rho Fiera Milano		CP			

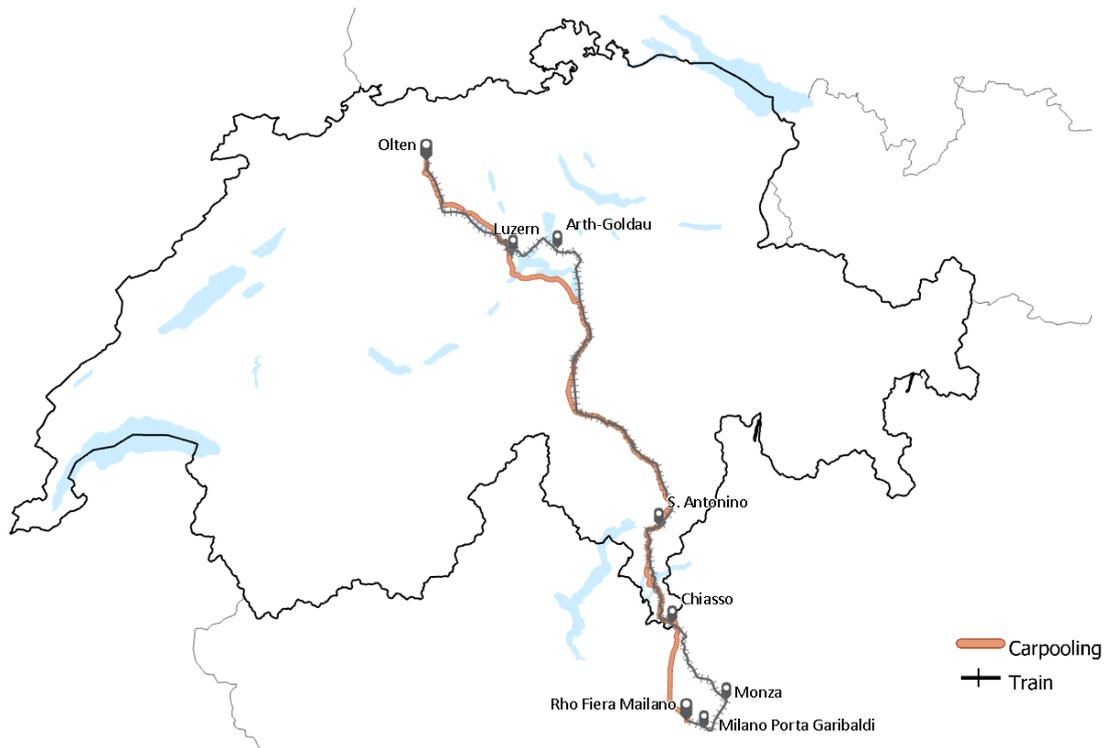


Figure 48 Carpooling and train route from Olten to Rho Fiera Milano.

To deliver proof of concept for multimodal routes, two additional examples are presented. Soccer fan Adriana went to watch a game of her favorite team, FC Bern, at their Stadium in Bern Wankdorf. The game finishes at 16:00. Therefore, she is looking for a trip back to Olten at around 16:30. Querying the system, a multimodal trip from Bern Wankdorf to Egerkingen with carpooler Jordan L in a BMW 318 can be found. In Egerkingen, Adriana has to take a connecting train to Olten (cf. Figure 49). The exact query can be found in the Appendix.

Compared to the multimodal trip above, the carpooling segment in this trip for CHF 63 is quite expensive. This is the price for the whole advertised trip and not only for this segment.

Table 24 Schedule of a multimodal journey from Bern Wankdorf to Olten. The trip requires one transit (Carpooling -> Train).

Departure	Arrival	Stop	Transfer	Mode	Driver	Car	Price
16:38:27		Bern Wankdorf		CP	Jordan L	BMW	CHF
17:07:32	17:07:32	Schönbühl Shoppyl land		CP		318	63
	17:08:20	Egerkingen		CP			
17:11:00	17:11:00	Egerkingen	Yes	Train			CHF
17:14:00	17:14:00	Hägendorf		Train			3.30
17:17:00	17:17:00	Wangen bei Olten		Train			
17:19:00	17:19:00	Olten Hammer		Train			
17:24:00	17:24:00	Olten		Train			



Figure 49 Multimodal route from Bern Wankdorf via Egerkingen to Olten.

Timo is a passionate snowboarder. Thus, after a day of snowboarding above Chur, he is looking to take a trip back to Olten at around 16:00 (cf. Figure 50). With the use of the implemented system, the following multimodal trip presented in Table 25 can be found. The trip starts at 15:51 in Chur with carpooler Frank S and passes Untervaz-Trimmis on its way to Sargans. In Sargans, Timo has to transfer to a connecting train to Zürich HB, where he has to transfer a second time to another train. The duration of the overall trip is 2h and 9 minutes. At CHF 5.30, the carpooling part is quite cheap.

Table 25 Schedule of a multimodal journey from Chur to Olten. The trip requires two transits (Carpooling -> Train -> Train).

Departure	Arrival	Stop	Transfer	Mode	Driver	Car	Price
15:51:00		Chur		CP	Frank S	n/a	CHF 5.30
16:06:00	16:06:00	Untervaz-Trimmis		CP			
	16:18:08	Sargans		CP			
16:28:00		Sargans	Yes	Train			
	17:23:00	Zürich HB		Train			
17:30:00		Zürich HB	Yes	Train			CHF 54
	18:00:00	Olten		Train			



Figure 50 Multimodal route from Chur via Sargans and Zurich to Olten.

Problems: Although the system is able to calculate shortest paths between two stops with un-augmented versions of Dijkstra's and A* routing algorithms, some problems occurred. Querying for indirect routes requires allowing the routing algorithm to traverse over transfer edges. In chapter 4.1 & 4.3.2, the implementation of transfer edges is elucidated by relating every stop time at a stop location if a certain condition is fulfilled. Subsequently, a stop time may be related to many different other stop times.

Using a standard routing algorithm may lead to inadequate results, because multiple transfer edges can follow each other. Figure 51 shows an example where the routing algorithm follows two consecutive transfer edges (highlighted with bold arrows). Subsequently, a route from A via B to C with a transfer at B may lead to the route A -> B -transfer-> B -transfer-> B -> C.

This problem can also be seen in the multimodal trip above where Chiasso is listed three times. Thus, the routing algorithm followed two transfer edges after each other. Theoretically, the result is not wrong and simply results in a longer waiting time in Chiasso, but the overall route may change. To solve this problem, one can either adjust the query algorithm to prohibit multiple, following transfer edges or split time nodes in departure and arrival nodes. This issue will further be discussed in the following chapter.

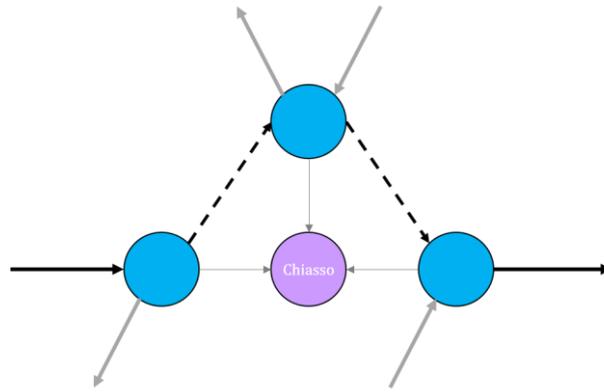


Figure 51 Schematic representation of routing two consecutive transfer edges (dashed). This should be not possible, as it bypasses the conditional of maximum transfer times.

7 Discussion

Today's systems supporting multimodality usually contain MOT which either have a fixed timetable or allow access on demand such as the road network, where a journey is not bound to a discrete time. The former timetable-based MOT further entail a certain frequency which helps a user to travel more conveniently, as connections mostly start at the same times during the day. Moreover, for example, public transportation offers the same routes in the opposite direction. Hence, it is ideally suited not only for leisure travel, where the goal is the travel itself or the activity at the destination, but also commuting.

Not only does the user profit from frequent MOT. Modeling timetable-based networks entailing a high frequency are well studied. Therefore, most of the existing modeling approaches are tailored for timetable-based networks with frequent connections. Additionally, the General Transit Specification Feed (GTFS) assumes frequent trips by default, as only exceptions need to be defined and not the available dates of a trip.

Newer transportation modes, however, might show a different behavior. Carpooling, for example, does indeed exhibit a timetable, but lacks frequency. Nevertheless, carpooling can be seen as a way of making private cars part of a transportation network (Bit-Monnot et al., 2013). As private cars travel on the road network, they further allow some flexibility of driving detours. Also, they might cover areas which are not serviced by public transportation. Carpooling therefore entails parts of different networks. On the one hand, it is bound to discrete departure times, but, on the other hand, it runs on the road network and thus allows highly flexible journeys. Currently, as already mentioned in the section on the research gap, no modeling approach for these undetermined networks exists.

When it comes to multimodal routing, flexibility is even less considered. The state of the art in combining networks is to simply solve the *Nearest Neighbor Problem (NNP)*, hence linking the closest stops of networks. While this is adequate for traditional transportation networks such as trains, buses or trams, where the vehicles must stop at predefined stations, it is inappropriate for a flexible mode like carpooling, which has the possibility to approach stations that are not originally part of a route.

Research in routing and multimodal routing has currently no solution to this problem. Furthermore, in all conscience, only two studies exist which attempt to combine the flexible mode of carpooling with public transportation. Although they provide solutions considering carpooling's flexibility of driving detours, they do not elucidate a specific modeling approach, but use the road network to find substitutions for already existing multi- or unimodal trips. In addition, the dataset

of carpooling offers used showed a high frequency on a small scale. As carpooling is only used as a substitution of an existing trip, it seems to not be possible to bypass areas not already covered.

Therefore, in this thesis, the aim was to propose a technique to integrate carpooling as a full-fledged part into a multimodal routing system without losing any of the given flexibility. Crucially, an adequate modeling approach for carpooling, as well as a sophisticated merging technique, had to be elaborated. Since the proposed technique shall be useful for real-life carpooling offers, it is necessary to understand the characteristics of these offers.

In chapter 3, research objectives were defined accounting for the research gap in carpooling and multimodal routing. As mentioned above, finding an adequate way of modeling carpooling for the future integration into a multimodal routing system requires understanding its characteristics. The evaluation of whether carpooling is potentially a useful MOT in a multimodal routing system had to be elaborated. This is focused by the first sub-objective:

RO 1.1: The characteristics of real-life carpooling offers shall be investigated in terms of quantity, temporal scale, and spatial scale to assess the benefits of carpooling in a multimodal routing system.

Regardless of the benefits of carpooling offers, the second sub-objective focuses on modeling carpooling offers:

RO 1.2: In the light of a future integration into a multimodal network, a model which allows retaining flexibility and fuzziness has to be elaborated for carpooling.

The third sub-objective considers a conceptual model and a technical implementation of a multimodal routing system containing carpooling. As mentioned above, today's technique in linking MOT is not adequate for carpooling:

RO 1.3: Network merging & linking techniques have to be developed, providing a multimodal network which:

- a. Retains flexibility and fuzziness of the contained MOT networks*
- b. Improves the overall network*
- c. Is able to calculate useful multimodal journeys using current shortest path algorithms*

In this chapter the proposed merging technique, as well as the experiment and its outcomes are discussed. The used dataset of real-life, volatile carpooling offers are interpreted based on the results presented in section 6.1.

In the first part of the discussion, the current situation of carpooling offers in Switzerland, derived from a real-life carpooling platform, is elucidated. Further, the technical feasibility of these offers is brought into the context of routing and multimodal routing. In particular, the minimum requirements for modeling purposes are discussed.

In the second part, research objective 1.3 is discussed based on the outcomes of the experiment.

7.1 RO 1.1: Characteristics of Real-Life Carpooling Offers

Carpooling in Switzerland is not as prominent as it is in other countries. A reason for this might be the extremely good coverage and the high frequency of public transportation. Nevertheless, even in Switzerland, cars only average 1.6 passengers per trip (Tages-Anzeiger, 2013). However, it is not specified if the number of passenger differs between commuting and leisure travel, and it can be assumed that this number might be lower for commuting. Carpooling for commuting is therefore a good way of reducing traffic on the street network and improving the ecological footprint of an individual (Ghoseiri et al., 2011).

In the next section, the characteristics of the retrieved carpooling offers will be elucidated in order to qualitatively assess the benefits of carpooling in combination with the Swiss railway network.

7.1.1 Carpooling in Switzerland

First at all, it is important to mention that in the scope of this thesis, only the current situation in Switzerland can be described. Future insights would be vague. Nevertheless, it is important to understand what kind of carpooling offers have been used in this thesis, as the understanding of carpooling can differ. One form of carpooling is ridesharing for daily commuting, as it is often performed in the US, where special lanes on highways only for carpools exist. Another form is leisure travel (cf. Heinze (2000)), where either the journey itself or the purpose at the destination (e.g. vacation) is the goal. The latter is consequently more sporadic and does not entail frequent journeys. In this section, the acquired carpooling data set is discussed based on:

- Number of offers
- Temporal component of trips
- Spatial component of trips

Using these measures, the purpose of carpooling in Switzerland can attempt to be defined.

Number of Offers: Considering carpooling offers from a real-life platform it becomes apparent that not many offers in Switzerland exist. Crawling offers for the 17 biggest cities in Switzerland plus Olten as a main part of the railway network only resulted in 18k offers within an 8-month period. Additionally, carpooling offers for Switzerland are very volatile, meaning that the number of offers differs drastically from day to day. This is a first indication that carpooling in Switzerland is not frequent and thus not used for daily commuting. Nevertheless, a slight overall increase is

visible. Furthermore, it should be considered that the amount of data could have been increased by the use of an official API. In this case, carpooling offers could have been also crawled for rural regions and not only for large cities and their suburbs.

In addition, as shown in 6.1.1, most of the carpooling trips are offered by only a few individual drivers. This phenomenon follows the *Long Tail Theory*, which in Social Media and Online Communities is also known as the *90-9-1 Rule in Participation Inequality*. It states that approximately 90% of the users never contribute, 9 % contribute only a little and 1 % are responsible for most content (Nielson, 2006). A reason for this phenomenon in carpooling cannot be easily identified. However, several possibilities are conceivable, e.g. these few drivers are commuting or that there is no need for carpooling because of the dense public transportation network, or even that carpooling is still unknown in Switzerland. An exact reason, however, is not part of this thesis. One strongly influential factor is, however, also named in literature. A routing system, which an online rideshare platform can be assumed to be, must deliver understandable, suitable, and convenient results. The effort to find a journey must be as small as possible (Furuhata et al., 2013; Geisberger et al., 2009; Sierpiński et al., 2014). For carpooling in Switzerland, this is often not the case, as offers do not start at the desired destination and time. Thus, a user would have to consider additional routing systems if he wanted to take up a carpooling offer.

Generally speaking, based on the pure number of offers, carpooling currently has a very limited use in a multimodal routing system, as too few offers exist and thus often no carpooling route can be found. However, to promote carpooling, integration is indeed suggestive, as it simplifies and improves the access to carpooling. There might be a good chance that more people will take part in ridesharing.

Temporal component of trips: Carpooling offers for Switzerland mostly start around the weekends (Thursday, Friday, Sunday). In addition, this thesis succeeded in showing that these offers start at around noon. Consequently, the assumption of commuting can be discarded. It is important to keep in mind, however, that due to the crawling, carpooling offers before 11:00 were not available. It can only be assumed that the morning peak might look similar to the evening peak, as commuters should be returning then. Even in this case, there is no indication for commuting. In order to have proof, additional crawling in the morning would have to be performed. Nevertheless, a reason for less to no commute offers might also be the way BlaBlaCar.de works. Each offer represents a trip valid only once and not a frequent schedule. Thus, the effort needed to post a commute offer for every day is high. Hence, a system which aims to simplify the access to carpooling might also consider a driver's side allowing posting offers in a convenient way.

Based on the temporal signature of carpooling offers, it can be stated that the impact on a multimodal network is small. Even if the number of offers rises, the offered trips are not well spread

during the day or the week. Consequently, in the current situation, carpooling is not well suited for spontaneous travel or commuting during the week. Nevertheless, it is well suited for weekend leisure travels.

Another temporal component is detours. As carpooling uses private cars and the road network, routes driven can be changed on demand. Thus, a driver is able to approach stops which were not originally planned. Usually, a driver specifies a maximum detour time he/she is willing to undertake. Therefore, carpooling entails a flexibility which is not given in traditional MOT such as trains, buses or airplanes. Nevertheless, as shown, the proposed linking technique is able to retain the flexibility of carpooling offers and combine it with traditional non-flexible MOT.

Spatial component of trips: Considering the spatial characteristics of carpooling offers for Switzerland, it was demonstrated that it could not be restricted to a country scale. The average length of a carpooling trip is roughly 480 km, which is more than three times longer than an average train connection (130 km). Crucially, this is another indicator against commuting. Nevertheless, considering stop locations within Switzerland, most can be found in areas where also many train stations exist. Additionally, even with a relatively small subset of 2000 carpooling routes, connections not served by the Swiss railway network could be found. Especially regions within the Alps or border regions are serviced better by carpooling.

In addition, in the context of the spatial extent of carpooling, the pricing is an important factor. As demonstrated earlier in this thesis, carpooling serves for a very low price. On average, it is 10 times cheaper than the Swiss railway network. Consequently, this is a stimulus for people to take part in carpooling as riders, which in reverse is also good for drivers, as they can fill up empty seats and thus earn more. The proposed system should therefore also consider pricing optimization to propose cheap trips to riders.

Based on the spatial component of carpooling trips, it can be stated that carpooling shall indeed be integrated into a multimodal network as it covers and serves areas not offered by the railway network. Even if this thesis focuses on Switzerland, carpooling is well suited for long-haul journeys to foreign countries at low prices. Consequently, carpooling is a good complement to public transportation and should be investigated on a larger scale.

7.2 RO 1.2: Modelling Carpooling

In chapter 4.2, two different modeling approaches (road network, time-expanded model) for carpooling were discussed. By comparing the minimal requirements for the road network approach and the time-expanded model (cf. Table 26), it becomes apparent that the time-expanded model requires more information. Whereas the road network only demands a geographic location of stops, the time-expanded model further requires exact departure and arrival times at stops, as well as trip information (cf. Pajor (2009)), describing a car driving from A to B.

Considering the crawled carpooling offers, it can be seen that the most important information, namely an exact geographic stop location is missing. An offer also usually only contains an approximate departure time, but no arrival and departure time at intermediate stops and the destination. Whereas this is not problematic for a road network, where time information can be automatically derived from a shortest path calculation, it is a problem in a time-expanded model, as it can be only used to find optimal paths if the times are given at every stop. Furthermore, these times are needed to model transfers at stop locations.

Table 26 Requirements of the discussed modeling approaches for carpooling and the information provided by the offers.

Requirement	Road Network	Time-Expanded	Carpooling
Geographic Location	X	X	City Level (no coordinates)
Exact Time Information		X	Departure Only
Trip/ Driver		X	Yes, Driver and Trip signature

Carpooling offers crawled from a real-life platform do not fulfill the requirements of either the road network approach or the time-expanded model. Consequently, data enrichment processes are needed in any case. Regarding the technical feasibility of the used carpooling offers, it can be stated that feasibility is only given after data enriching.

With the use of a road network approach, the flexibility of carpooling offers could be enabled in the individual model, which, as shown, is not possible with a time-expanded model. This might be a reason why both carpooling in multimodal routing studies relied on the road network (Aissat & Varone, 2015a; Bit-Monnot et al., 2013).

Although the road network seems to be more feasible than the time-expanded model, one reason to not use it is the imprecision of carpooling stops. Since in carpooling the driver usually specifies a city and not an exact address, it is not clear where to start a shortest path calculation in a road network. The time-expanded model, however, does not represent a physical network, but a schedule of an MOT. Thus, an offer can have multiple potential start locations. This allows representing the fuzziness of imprecise carpooling stops. The missing information can be derived through a

previous routing. It is assumed that this process is even more efficient than the road network approach. In the road network approach, the route on the network has to be calculated for every query, where in the time-expanded model, the physical route itself is not important anymore, but only the time information at stop locations.

7.2.1 Summary of RO 1.1 & RO 1.2

Carpooling in Switzerland is still a niche MOT. However, a high potential for long-haul journeys can be seen. It is therefore to be recommended that a multimodal system which integrates carpooling should be implemented for a larger scale than only Switzerland. This however does not mean that public transportation schedules of every country a carpooling offer passes must be integrated, but that stops in foreign countries shall not be omitted. Considering the frequency, carpooling has just a few frequent travels. Furthermore, carpooling offers are only valid at a specific date. Thus, a modeling approach must be efficiently updatable. According to Müller-Hannemann & Schnee (2009), this is given in the time-expanded model. In addition, they showed that even life information can be integrated into the time-expanded graph. Thus, this model suites the characteristics of carpooling well, as past offer can be removed in an efficient, straightforward way and even live information such as traffic delays could be integrated.

Considering the technical feasibility, it was shown that real-life, volatile carpooling offers, derived from an online platform, are per se not qualified to use in a time-expanded model. Nevertheless, as demonstrated, with simple data enriching processes the minimum requirements can be fulfilled. Rather than routing the road network for every query, the route was preprocessed and only the information needed extracted. Based on these facts, this thesis argues to have chosen the correct model (time-expanded) for carpooling.

With the current situation in Switzerland, carpooling is based on the number of offers and the temporal scale not well qualified for the integration into a multimodal network. During a week and especially during rush hour, not many offers exist. Considering the spatial scale, however, carpooling indeed exhibits a high potential, since it covers areas not serviced by public transportation and offers long-haul journeys for a good value. The fact that carpooling is on average ten times cheaper than the train, is a high inducement for choosing carpooling. Integrating carpooling into a multimodal routing system simplifies and improves the access to carpooling offers. It is assumed that this simplification boosts the acceptance of carpooling and will animate more people to participate. Consequently, even if carpooling, at the moment, does not fulfill the minimum requirements and shows low participation in Switzerland, research on its integration into a multimodal routing system is promising.

7.3 RO 1.3: Merging & Linking

7.3.1 Fuzziness and Flexibility

Stop Allocation: As previously described, carpooling entails imprecise stop locations. Contrary to a traditional public transportation network, stop locations are not at a specific location, but are defined on a city scale. Thus, the *NMP* is not able to find the correct adjacent public transportation stop. The proposed linking technique therefore uses drive time areas around public transportation stops in order to evaluate if a carpooler is able to reach a specific stop. Currently, there is no similar approach in literature.

With the use of drive time areas in the size of half the maximum allowed detour time of a driver, multiple reachable train stations can be found. On average, a predefined stop of a carpooling offers could be linked to 3.2 different public transportation stops, in comparison to the *NMP*, where only one link would be initiated. This is also user friendly. Rather than travelling to one specific location, the user can be picked up at the most desirable linked stop. Crucially, this not only represents the fuzziness of carpooling stops, but also leads to an improvement of the accessibility. Nevertheless, it is to mention that a user still has to arrange a pick-up location with a driver, but with the use of this approach, a set of possible locations is given so that a driver does not have to take inconveniently long detours. Thus, this approach helps both the user and the driver.

Drive time areas were calculated around train stations and not around a carpooling offer. The reason for this is clear. Because train stations very rarely change, they can be considered stable. This also has a large impact on the future integration of new carpooling offers. Rather than calculating drive time areas for every offer, the already existing drive time areas of public transportation stops can be used. Hence, the cost of preprocessing can be reduced drastically.

Flexibility, Exploiting Stops: Carpooling travels on the road network in private cars. Thus, it is possible to drive detours. Subsequently, a driver can depart from his/her original route to pick-up or drop-off a passenger. This flexibility is also considered by Aissat & Varone (2015) and Bit-Monnot, Artigues, Huguet, & Killijian (2013). However, in their approaches, potential stop locations along a carpooling route are not represented in the model. Consequently, they use carpooling as a substitution to an already existing multimodal or unimodal trip. The stop locations along the existing trip can be used to analyze whether or not a carpooler drives a similar route and could potentially pick-up a user at one of these stops and drop him off at a later stop. Thus, they do not exploit all potential stops along a carpooling route.

The proposed technique, however, exploits all potential stops along a carpooling route during the insertion into the multimodal graph. Similar to the stop allocation, drive time areas are used to evaluate whether a train station is reachable within the given detour time. It is further explained that an intersection of a route with a drive time area of a train station inconclusively indicates a

potential link. Thus, *Points of Action (POA)* were introduced. A *POA* is a point along a route where a driver has to perform an action (leave highway, turn right/left, etc.). Consequently, *POA* are located at crossings, junctions, ramps, etc. It is assumed that if such a *POA* lies within a drive time area, its train station must be reachable. Hereto, it must be mentioned that this still might be not true in every case. For example, a highway ramp might be situated inside the drive time area, but lead the driver outside of it. Nevertheless, *POA* are assumed to be an adequate option.

A *POA* located inside a drive time area of a train station is linked to the meta-stop of the train station, making the *POA* a carpooling child stop of the train station. The author is aware of the fact that the arrival time at a *POA* stop differs from the actual arrival time at the train station it is linked to. In a worst-case scenario, the arrival time is the size of the drive time area later (plus eventual traffic delays). Consequently, penalties for transfers at these stops were implemented, which will be discussed later in this section.

In case a driver approaches a reachable train station along a route, every later departure and arrival time changes by the amount of time needed to drive the detour. Subsequently, every later time needs to inherit the additional time taken. In this thesis, it is argued, however, that a driver shall not drive multiple detours along a route. On the one hand, this might lead to very inconvenient trips for a driver (driving many short connections) and on the other hand, this would change the overall nature of carpooling, where driver and passenger usually have the same origin and destination. Consequently, this thesis argues that drive time areas shall be at maximum half the size of the allowed detour time. For the purpose of this study a static detour time of 15 minutes and hence 5-minute drive time areas to account for potential delays were determined. Multiple drive time areas could be calculated considering different detour times (e.g. 5, 10, 15, and 30 minutes) and traffic models for different times of the day. It is assumed that the correctness of linked stops could be increased drastically, as for example during morning rush hour, drive time areas are much smaller in size than in the afternoon. The evaluation of this statement could be examined in a further study.

Using *POA* along a route has some drawbacks. In urban areas, the density of *POA* is high, as many crossings and junctions exist, whereas in rural areas a driver rarely has to take an action. Subsequently, without any generalization approach, many *POA* of nearly the same location would be connected to the same train station. This is similar to linking a road network with public transportation stops, where the station is linked to the nearest location of the road network and not vice versa. Linking locations along the road network to the nearest station would create a huge amount of link edges without any further information (Pajor, 2009). Thus, in this thesis, a straightforward generalization approach of *POA* was applied. Every *POA* which lies within a 1 km range

of its accepted predecessor is removed. Consequently, the amount of *POA* in a city is reduced drastically and “useless” link edges are avoided. In rural areas, this generalization approach has less impact. This simple generalization approach can be considered naive, meaning that *POA* stops which would have created a link can be removed and thus not all connections to the railway network are caught. Therefore, it is argued that in a further study, a more sophisticated generalization approach should be used. A viable solution would be to not remove *POA* which are very close to a public transportation stop and lie within its drive time area.

Nevertheless, with the used approach, 1600 *POA* in total could be linked to train stations. Each linked *POA* shares links with 3.5 different stations, indicating a high interconnectivity between carpooling and the railway network. Considering an average carpooling trip with 23 stops (3 Main, 20 *POA*), the main stops are connected to 6 different train stations and the *POA* to 34 stations. Especially shorter carpooling trips show many links to public transportation per kilometer. This is crucial as longer trips are usually driven on highways, where the number of *POA* is low.

The proposed technique thus works well to exploit potential stop location along a carpooling route. Furthermore, this flexibility can be stored in the time-expanded graph. Hence, routing shall be very efficient, compared to the before mentioned studies, where potential detours are determined during querying.

Transfers: Linking a carpooling stop to multiple public transportation stops has a crucial influence on possible transits. Furthermore, the exploited stops enable additional transits. In the approach of e.g. Aissat & Varone (2015), no new stops are exploited, as only a part of an existing trip is bypassed with carpooling. Furthermore, they project the existing trip onto the road network, leading to a not true multimodal network. Hence, carpooling and other MOT are not on the same graph. Thus, there is no need to calculate possible transfers at every exploited carpooling stop.

The proposed technique, however, creates a true multimodal time-expanded graph. Consequently, transfers between every trip at a stop must be modeled. Even though this requires exhaustive calculations, the flexibility of carpooling is able to be stored in the developed network. Furthermore, as carpooling stops are linked to train stations, the minimum transfer time at a stop can be used, provided by the SBB, to model variable transfer times. However, as stated above, linking a *POA* to a train station requires adding penalties for mode changes, as the arrival time at a *POA* does not equal the actual arrival time at a train station. Thus, the information of the minimum transfer time was extended with a penalty of the size of the drive time area. In case a driver arrives earlier at the train station as the arrival time at the *POA* plus the size of the drive time area, transfers are not modeled. Thus, this approach lacks correctness. Nevertheless, all modeled transfers are passable. A solution to resolve the issue could be to apply and additional routing on a road network to derive the time needed to travel from the *POA* to the actual train station. Thus, the

exact arrival time is known. Nevertheless, this approach requires much more exhaustive preprocessing, as routing has to be performed for every *POA* contained by a drive time area.

Even though calculating transfer edges in a time-expanded model is exhaustive, profit can be drawn from the given complexity, because the stable transfers between trains need to be calculated only approximately once a year in case of a change of schedule. Inserting a new carpooling offer therefore only requires the new calculation of transfers at the stops along its route.

In the experiment with 2000 carpooling offers and the Swiss railway network, a large amount of possible transfer could be identified. The 296 main carpooling stops each linked to 3.2 different train stations, as well as the 1619 exploited *POA* stops each linked to 3.5 different train stations exhibit a total of roughly 640'000 possible transfer. Approximately 50% are from a carpooling stop to a train station and vice versa. Considering carpooling and train routes, it was illustrated that, especially on train routes between 15 and 25 stops, a high number of transfers to carpooling is possible. As these routes are usually regional trains, the proposed approach also connected carpooling to smaller suburban train stations. This is essential as it further increases the interconnectivity of carpooling and public transportation. Also, the access to carpooling is improved. A user, for example, therefore does not have to travel far to be picked-up by a carpooler. Additionally, carpooling would in this case perfectly suit as transportation mode for commuting, as a user might share the ride with a driver, rather than first driving to a hub and then to the destination.

The proposed technique further showed that it is possible to highly interconnect carpooling and public transportation. Further, the access to carpooling offers could be improved by linking carpooling to smaller train stations. This circumstance is important as routing systems are only used if they return adequate, understandable journeys (Sierpiński et al., 2014).

In the undertaken experiment, the combination of carpooling with the railway network was put in focus. Extending these with smaller scale MOT like buses and trams has the potential to further increase the interconnectivity, as the number of total stops increases. Carpooling could be reached from more stations, even closer to a user and thus simplify the access. Not only a user may profit from this, but also a driver, as he/she does not have to drive far in order to pick up a rider. A further study could therefore evaluate the proposed linking approach in consideration of smaller scale MOT.

7.3.2 Network Improvement

A large benefit of multimodal routing systems is the fact that journeys can consist of different MOT. With the use of multimodality, areas can be covered which are not reachable with an individual mode. In addition, in case of an impassable stop in one network, it might be possible to bypass this stop with another mode. Thus, research objective 1.3 also posed the question if carpooling, with the use of the proposed merging & linking technique, is able to improve the overall network.

Improvement of Connections: In section 6.2.4 the individual networks and the multimodal network were evaluated using Centrality measures. First, the Degree Centrality was used to find public transportation stops, which could be improved in terms of new connections. More than half (55%) improved by a maximum of 1%, which can be considered as no improvement. Approximately 5% could be increased by more than 2% and 7.5% by more than 5%. Considering the stops and the geographic location of the improved public transportations stops, it was found that the highest improvements were identified for smaller train stations in border regions and in the Alps. Considering the characteristics of the crawled carpooling offers, this is crucial as most of the offers use Switzerland as a transit country. Consequently, stop locations are often found at the border. In addition, carpooling offers heading to alpine areas have a higher impact, as the railway network is not as dense in the Alps as in the “Mitteland. The railway network could be even more improved with a higher number of carpooling offers.

The impact of the small subset of 2k carpooling offers on the Swiss railway network is, based on the improvement of connections, rather small. However, smaller train stations (especially in border regions and the Alps) could be improved the most. This furthermore correlates with the statement about transfers, where it was illustrated that carpooling has been connected to multiple smaller train stations.

PageRank: Not only the improvement of connections is of importance, but also the distribution of important hubs in a network. Thus, PageRank Centrality has been used to identify important stop locations in both individual networks and the resulting multimodal network.

In the individual carpooling network, a city centric network was identified, meaning that the overall network is not well spread over Switzerland. Most of the offers start/stop in big cities. Even though carpooling shows a high coverage over Switzerland, smaller stops are not approached. This statement is however biased, as carpooling offers have been only crawled for big cities. Nevertheless, another concentration of important stop locations can be found in border regions. This can be further explained by the characteristics of the crawled carpooling offers, where most have either the origin or destination in a foreign country.

The railway network shows opposite behavior. Public transportation is much more equally spread over Switzerland. A reason for this might be the high number of stations. Except for Zurich and Lausanne, the identification of critical hubs is difficult. The Swiss railway network can therefore be denoted as highly interconnected and thus less vulnerable to a failure.

The combination of carpooling and the railway network on the basis of the proposed merging & linking technique results in a multimodal network showing characteristics of both contained MOT. It has been shown that important stop locations are even more spread over Switzerland. Furthermore, the merging approach retained the importance of stop locations in border regions.

In general, even smaller train stations gained importance as they can be approached by many carpooling offers. In general, the more stops of high importance, the better the network. Thus, it can be assumed that a multimodal system containing carpooling and the railway network might be less vulnerable to disturbances of an individual MOT. In addition, integrating small scale MOT such as buses and trams could potentially improve the overall network further and make it even less vulnerable to failures.

Consequently, real-life carpooling offers integrated into a multimodal system using the proposed linking technique has improved the overall network. Thus, the technique can be considered successful in terms of improvements.

7.3.3 Carpooling in Multimodal Journeys

Proposing a new merging and linking approach for multimodal routing requires proof of concept by being able to query for adequate shortest paths. Without the possibility of routing the final network, the merging technique can be considered as unsuccessful.

In general, multimodal networks are routed by solving the label-constrained shortest paths problem. It is an augmentation of the standard shortest path problem with the principle of computing journeys that obey constraints on the MOT (Bast et al., 2015; Pajor, 2009). Basically, a label-constrained path is a shortest path based on a language given to the query. Consequently, the computed journey results in a multimodal trip following the language. Basic and goal directed techniques have been augmented to adapt this concept (Barrett et al., 2008; Bast et al., 2015).

In the scope of this thesis, sophisticated routing algorithms were not considered, as the aim was to present meaningful shortest paths with un-augmented routing techniques. In section (6.2.5) several multimodal trips containing carpooling were presented.

With standard Dijkstra's and A* algorithms, no true label-constrained shortest paths can be calculated. Nevertheless, it was demonstrated that train, carpooling, and multimodal trips can indeed be queried. By defining the start and end MOT, the mode a journey shall start and end can be forced. This can be considered a primitive form of a language. Nevertheless, a sequence of MOT

could not be specified. The implemented system can be further queried without specifying any language. In this case, the system will return a shortest path without considering MOT at origin and destination.

Nevertheless, the calculated shortest paths in section 6.2.5 demonstrate that adequate journey, addressing the *Range Problem (RP)*, can be derived. This thesis illustrates that multimodal trips containing carpooling are faster than train trips in general. Additionally, the number of transits (max. 2x) can be considered adequate for a journey crossing half to the whole of Switzerland. Even though the price of the whole advertised carpooling journey is listed, they serve for a good value. Thus, in terms of multimodal journeys, carpooling is well suited for being integrated into a multimodal system. In addition, even with a small set of 2k carpooling offers, journeys could be found. This fact is also assigned to the proposed linking technique, which, as illustrated, created a high interconnectivity between carpooling and the railway network. However, the date component of trips has been neglected in this thesis. Consequently, in a real-life scenario, the number of valid trips is even lower. Promoting carpooling in Switzerland is thus an inevitable measure that must be taken.

The price of carpooling offers in a multimodal system has not been considered in this thesis. However, the application of such a system requires solving the problem of pricing single segments. The multimodal path presented in 6.2.5 from Bern Wankdorf to Egerkingen is a relatively short segment of approximately 54 km. Not decomposing the overall price leads to an amount of CHF 63, which is very high. Thus, the problem of calculating an appropriate price must be considered in a real-life system. Estimating a price based on the length or duration of a carpooling segment may be critical. In general, it was shown that carpooling offers are very cheap even on long-haul travels. It is questionable if a driver would accept a fraction of the advertised fare. Consequently, a solution for a payment system must be found. Allowing a user to pay every segment individually could be inconvenient on trips with multiple transits. A solution might be to either restrict the number of transits or introducing an overall payment system. If no solution is found, the system entails a barrier which hinders users to use such a system (cf. Sierpiński et al. (2014)).

Although adequate multimodal journeys can be derived from the implemented system, a problem regarding transit modeling could be identified. In chapter 4 arrival and departure times were modeled as tuples $t = (arrival, departure)$. Further, instead of using transit nodes. Labeled edges between times t of different trips at a same stop have been implemented. Using standard routing algorithms, paths can contain journeys with two or more consecutive transfer edges. Thus, a user would transfer multiple times at the same stop, arriving, transferring to another train/carpooler, but not taking it, transferring to another train/carpooler, and so on. The resulting journey would lead a user to the destination anyway, but waiting times at a stop could be longer

than defined during transfer edge creation. This problem can be solved by either augmenting a routing algorithms to not allow more than one transit edge at a stop, or by splitting the time tuple into an arrival and a departure node. Consequently, a transit can only lead from an arrival to a departure node and not vice versa.

However, the problem of two consecutive transfer edges arose, it can be argued, that using one time node instead of two helps to decrease the graph size, which is a known problem in time-expanded models (Pajor, 2009). As time nodes make up the largest part of a graph, they can be reduced by the factor of two. Further, this shifts parts of the model complexity to the query. It is supposed that the needed augmentation of the routing algorithm will be straightforward. This would need to be confirmed in a further study.

In conclusion, this problem does not have an influence on the correctness of the proposed merging and linking technique, as it only has an impact on shortest path calculations and not on the main goal of representing fuzziness and retaining flexibility of carpooling in a multimodal network.

7.3.4 Summary of RO 1.3

The proposed merging and linking technique in this thesis can successfully be used to integrate MOT with fuzzy stop information. In addition, the entailed flexibility can be stored into the multimodal graph, which theoretically allows fast queries. A key benefit of this stored flexibility is the allowance of modeling potential transfers. Thus, carpooling offers can be used to bypass areas where no other trip is available.

Considering transfers, the principle of linking carpooling to multiple train stations increases the amount of possible transfers along a route drastically. This crucially leads to a more interconnected multimodal network. Using PageRank Centrality it was illustrated that the combination of carpooling and the Swiss railway network leads to a better connected, less vulnerable overall network.

Unfortunately, no current research has considered a similar or different linking technique which retains fuzziness and flexibility of a MOT like carpooling. Thus, it was not possible to draw a meaningful comparison. Consequently, linking technique developed in this thesis can successfully be used to represent fuzziness and retain flexibility of an MOT, improve an overall network, and still allows routing for shortest uni- and multimodal paths.

In addition, since the overall approach is based on a time-expanded model, speed-up techniques, and more sophisticated routing algorithms can be used. A further study could therefore investigate the feasibility of the proposed approach in combination with state of the art routing algorithms.

8 Conclusion

8.1 Achievements & Implications

Traditional MOT in multimodal networks, such as trains, buses, airplanes etc., all entail fixed stations and exhibit frequent schedules. They all travel on predefined routes. A common approach is therefore to solve the *Nearest Neighbor Problem (NNP)* in order to link these MOT to a multimodal graph. However, newer MOT like carpooling do not specify an exact stop location. Additionally, even though carpooling offers are scheduled, they are not bound to a fixed route and thus allow making detours to non-planned destinations. Current state of the art approaches have been identified as inadequate, as they are not able to link imprecise stop locations and cannot exploit potential stops along routes.

Consequently, the aim of this thesis has been to develop a graph merging and linking technique for multimodal routing which is able to represent fuzziness and retain flexibility of a given MOT. Therefore, a conceptual model has been proposed which uses drive time areas to allocate imprecise stop locations to multiple possible stations of another MOT. The concept of *Points of Action (POA)* in combination with drive time areas has been introduced to exploit potential, reachable stop locations along a route.

Specifically, the conceptual model was developed for the newer MOT carpooling in combination with public transportation. Carpooling, in general, shows fuzzy stop locations and further entails the flexibility of driving detours. This circumstance is a major difference to traditional MOT such as trains, buses or trams.

Based on an experiment with real-life carpooling offers derived from an online platform and the Swiss railway network, the conceptual model was implemented and evaluated. The major achievements of the proposed merging and linking technique can be describe as follows:

- **Stop Allocation:** With the use of the proposed merging technique, it is possible to allocate stops without an exact position to multiple possible stations of another transportation network.
- **Exploiting potential Stops:** The concept of *POA* along a route on the road network in combination with drive time areas defining the maximum allowed detour can be used to exploit previously unknown stops.

This thesis shows that not only MOT with fuzzy stop locations and flexible routes can be integrated into a multimodal routing system, but that the proposed linking technique has a considerable impact on the interconnectivity of different MOT. The possibility of linking stop locations to multiple other reachable stations enhances the number of potential transfers.

In addition, it has been demonstrated that the system implemented using the proposed linking technique **retains the possibility of calculating unimodal and multimodal paths**. Furthermore, multimodal trips containing carpooling have shown the tendency to be faster and cheaper. It can therefore be reasoned that carpooling is indeed qualified as an MOT in multimodal systems.

The proposed linking technique has initially shown that fuzzy and flexible MOT can be fully integrated into a multimodal routing system using a well-known modeling approach, the time-expanded model. This research can furthermore be implemented in real-life systems of online rideshare platforms or public transportation agencies to provide multimodal trips.

8.2 Future Work

The conceptual model presented in this thesis has shown the possibility of integrating fuzzy and flexible MOT such as carpooling into a multimodal system. Even though the proper functioning could have been demonstrated, improvements and additional evaluation steps can be defined for further research.

The drive time areas employed in this thesis used to identify reachable stations were calculated using static detour times. It is therefore of great interest to implement variable drive time areas in order to potentially improve correctness of links. Variable drive time areas could be calculated using traffic models and different detour times.

Additionally, the proposed linking technique is limited to the number of detours per driver. Although this was identified as a desired aspect, evaluating the linking technique in combination with time-dependent models is encouraged.

Considering routing for optimal paths, the investigation of the applicability of the proposed approach in combination with current speed-up techniques further seems promising. Carpooling entails additional properties such as a drivers rating which could be used in multicriteria optimization. Fast routing results and multicriteria optimization can further improve the proposed system.

Another important factor to investigate is the scalability of the proposed merging technique in larger or productive multimodal systems. Considering the fact that this thesis has experimented with a small subset of 2000 carpooling offers, a future aim could be to evaluate how linking behaves with larger amounts of data or even with real-time information.

Bibliography

- Aissat, K., & Varone, S. (2015a). Carpooling as Complement to Multi-modal Transportation. In *Enterprise Information Systems - 17th International Conference, {ICEIS} 2015, Barcelona, Spain, April 27-30, 2015, Revised Selected Papers* (pp. 236–255).
- Aissat, K., & Varone, S. (2015b). Real-Time Ride-Sharing Substitution Service in Multi-modal Public Transport Using Buckets. In *Modelling, Computation and Optimization in Information Systems and Management Sciences - Proceedings of the 3rd International Conference on Modelling, Computation and Optimization in Information Systems and Management Sciences - {MCO} 2015, Metz, France*, (pp. 425–436).
- Amit, P. (n.d.). Introduction to A* Title. Retrieved February 22, 2017, from <http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html>
- Antrim, A., & Barbeau, S. J. (2013). The many uses of GTFS data--opening the door to transit and multimodal applications. *Location-Aware Information Systems Laboratory at the University of South Florida*, 4.
- Arz, J., Luxen, D., & Sanders, P. (2013). Transit node routing reconsidered. In *International Symposium on Experimental Algorithms* (pp. 55–66).
- Banister, D. (2008). The sustainable mobility paradigm. *Transport Policy*, 15(2), 73–80.
- Bannister, M. J., & Eppstein, D. (2012). Randomized Speedup of the Bellman-Ford Algorithm. In *Proceedings of the Meeting on Analytic Algorithmics and Combinatorics* (pp. 41–47). Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.
- Barrett, C., Bisset, K., Holzer, M., Konjevod, G., Marathe, M., & Wagner, D. (2008). Engineering label-constrained shortest-path algorithms. In *International Conference on Algorithmic Applications in Management* (pp. 27–37).
- Bast, H., Brodesser, M., & Storandt, S. (2013). Result diversity for multi-modal route planning. In *ATMOS-13th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems-2013* (Vol. 33, pp. 123–136).
- Bast, H., Delling, D., Goldberg, A. V, Pajor, T., Müller-hannemann, M., Wagner, D., & Werneck, R. F. (2015). Route Planning in Transportation Networks, 1–65.
- Bast, H., & Storandt, S. (2014). Flow-based guidebook routing. In *Proceedings of the Meeting on Algorithm Engineering & Experiments* (pp. 155–165).
- Bellman, R. (1958). On a routing problem. *Quarterly of Applied Mathematics*, 16(1), 87–90.

- Berrettini, E., D'Angelo, G., & Delling, D. (2009). Arc-flags in dynamic graphs. In *OASIScs-OpenAccess Series in Informatics* (Vol. 12).
- Bit-Monnot, A., Artigues, C., Huguet, M.-J., & Killijian, M.-O. (2013). Carpooling: the 2 synchronization points shortest paths problem. In *13th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS)* (p. 12--p).
- BlaBlaCar. (n.d.). Über uns. Retrieved February 3, 2017, from <https://www.blablacar.de/ueber-uns>
- Brodal, G. S., & Jacob, R. (2004). Time-dependent networks as models to achieve fast exact timetable queries. *Electronic Notes in Theoretical Computer Science*, 92, 3–15.
- Bucher, D., Jonietz, D., & Raubal, M. (2017). A Heuristic for Multi-modal Route Planning. In *Progress in Location-Based Services 2016* (pp. 211–229). Springer.
- Calvo, R. W., de Luigi, F., Haastrup, P., & Maniezzo, V. (2004). A distributed geographic information system for the daily car pooling problem. *Computers & Operations Research*, 31(13), 2263–2278.
- Chan, J., & Teknomo, K. (2016). Hub Identification of the Metro Manila Road Network Using PageRank. In *Proceedings of the 8th ATRANS Symposium Young Researcher's Forum*.
- Commission of the European Communities. (2001). White paper-European transport policy for 2010: time to decide. Office for Official Publications of the European Communities.
- Communities Commission of the European. (2011). White Paper on Transport: Roadmap to a Single European Transport Area: Towards a Competitive and Resource-efficient Transport System. Publications Office of the European Union.
- Delling, D., Dibbelt, J., Pajor, T., Wagner, D., & Werneck, R. F. (2013). Computing multimodal journeys in practice. In *International Symposium on Experimental Algorithms* (pp. 260–271).
- Delling, D., Giannakopoulou, K., Wagner, D., & Zaroliagis, C. (2008). Timetable information updating in case of delays: Modeling issues. *Arrival Technical Report*.
- Delling, D., Pajor, T., & Wagner, D. (2009a). Accelerating multi-modal route planning by access-nodes. In *European Symposium on Algorithms* (pp. 587–598).
- Delling, D., Pajor, T., & Wagner, D. (2009b). Engineering time-expanded graphs for faster timetable information. In *Robust and Online Large-Scale Optimization* (pp. 182–206). Springer.

- Delling, D., Pajor, T., & Werneck, R. F. (2014). Round-based public transit routing. *Transportation Science*, 49(3), 591–604.
- Delling, D., Sanders, P., Schultes, D., & Wagner, D. (2009). Engineering route planning algorithms. In *Algorithmics of large and complex networks* (pp. 117–139). Springer.
- Dibbelt, J., Pajor, T., & Wagner, D. (2015). User-constrained multimodal route planning. *Journal of Experimental Algorithmics (JEA)*, 19, 2–3.
- Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1), 269–271.
- Esri. (n.d.-a). Algorithms used by the ArcGIS Network Analyst extension. Retrieved February 22, 2017, from <https://desktop.arcgis.com/en/arcmap/latest/extensions/network-analyst/algorithms-used-by-network-analyst.htm>
- Esri. (n.d.-b). Service area analysis. Retrieved March 24, 2017, from <http://desktop.arcgis.com/en/arcmap/latest/extensions/network-analyst/service-area.htm>
- Euler, L. (1741). Solutio problematis ad geometriam situs pertinentis. *Commentarii Academiae Scientiarum Petropolitanae*, 8, 127–140.
- Firmani, D., Italiano, G. F., Laura, L., & Santaroni, F. (2013). Is timetabling routing always reliable for public transport. In *ATMOS-13th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems-2013* (Vol. 33, pp. 15–26).
- Foulds, L. R. (1992). *Graph Theory Applications*. Springer New York.
- Franceschet, M. (2014). Network Science. Retrieved April 3, 2017, from <https://www.sci.unich.it/~francesc/teaching/network/>
- Freeman, L. C. (1977). A set of measures of centrality based on betweenness. *Sociometry*, 35–41.
- Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.-E., Wang, X., & Koenig, S. (2013). Ridesharing: The state-of-the-art and future directions. *Transportation Research Part B: Methodological*, 57, 28–46.
- Geisberger, R., Luxen, D., Neubauer, S., Sanders, P., & Volker, L. (2009). Fast detour computation for ride sharing. *arXiv Preprint arXiv:0907.5269*.
- Geisberger, R., Sanders, P., Schultes, D., & Vetter, C. (2012). Exact Routing in Large Road Networks Using Contraction Hierarchies. *Transportation Science*, 46(3), 388–404.

- Ghoseiri, K., Haghani, A. E., & Hamed, M. (2011). *Real-time rideshare matching problem*. Mid-Atlantic Universities Transportation Center Berkeley.
- Goczyła, K., & Cielątkowski, J. (1995). Optimal routing in a transportation network. *European Journal of Operational Research*, 87(2), 214–222.
- Goldberg, A. V., & Harrelson, C. (2005). Computing the Shortest Path: A Search Meets Graph Theory. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms* (pp. 156–165). Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.
- Golledge, R. G. (1995). Path selection and route preference in human navigation: A progress report. In *International Conference on Spatial Information Theory* (pp. 207–222).
- Google. (2016). GTFS Static Overview. Retrieved December 19, 2016, from <https://developers.google.com/transit/gtfs/>
- Grotenhuis, J.-W., Wiegman, B. W., & Rietveld, P. (2007). The desired quality of integrated multimodal travel information in public transport: Customer needs for time and effort savings. *Transport Policy*, 14(1), 27–38.
- Gunkel, T., Schnee, M., & Müller-Hannemann, M. (2011). How to find good night train connections. *Networks*, 57(1), 19–27.
- Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2), 100–107.
- Heinze, G. W. (2000). Transport and leisure: growth as opportunity. *ECMT Round Table*, 111, 5–51.
- Hilger, M., Köhler, E., Möhring, R. H., & Schilling, H. (2009). Fast point-to-point shortest path computations with arc-flags. *The Shortest Path Problem: Ninth DIMACS Implementation Challenge*, 74, 41–72.
- Hillsman, E. L., & Barbeau, S. J. (2011). *Enabling Cost-Effective Multimodal Trip Planners through Open Transit Data*.
- Holzer, M. (2008). *Engineering planar-separator and shortest-path algorithms*. Karlsruhe Institute of Technology.
- Huang, H., Klettner, S., Schmidt, M., Gartner, G., Leitinger, S., Wagner, A., & Steinmann, R. (2014). AffectRoute – considering people’s affective responses to environments for enhancing route-planning services. *International Journal of Geographical Information Science*, 28(12), 2456–2473.

- Kenyon, S., & Lyons, G. (2003). The value of integrated multimodal traveller information and its potential contribution to modal change. *Transportation Research Part F: Traffic Psychology and Behaviour*, 6(1), 1–21.
- Loo, B. P., & Chow, S. Y. (2006). Sustainable urban transportation: Concepts, policies, and methodologies. *Journal of Urban Planning and Development*, 132(2), 76–79.
- Martins, E. Q. V. (1984). On a multicriteria shortest path problem. *European Journal of Operational Research*, 16(2), 236–245.
- Müller-Hannemann, M., & Schnee, M. (2006). Paying less for train connections with MOTIS. In *OASIS-OpenAccess Series in Informatics* (Vol. 2).
- Müller-Hannemann, M., & Schnee, M. (2007). Finding all attractive train connections by multi-criteria pareto search. In *Algorithmic Methods for Railway Optimization* (pp. 246–263). Springer.
- Müller-Hannemann, M., & Schnee, M. (2009). Efficient Timetable Information in the Presence of Delays. In R. K. Ahuja, R. H. Möhring, & C. D. Zaroliagis (Eds.), *Robust and Online Large-Scale Optimization: Models and Techniques for Transportation Systems* (pp. 249–272). Berlin, Heidelberg: Springer Berlin Heidelberg.
- Müller-Hannemann, M., Schulz, F., Wagner, D., & Zaroliagis, C. (2007). Timetable Information: Models and Algorithms. In F. Geraets, L. Kroon, A. Schoebel, D. Wagner, & C. D. Zaroliagis (Eds.), *Algorithmic Methods for Railway Optimization: International Dagstuhl Workshop, Dagstuhl Castle, Germany, June 20-25, 2004, 4th International Workshop, ATMOS 2004, Bergen, Norway, September 16-17, 2004, Revised Selected Papers* (pp. 67–90). Berlin, Heidelberg: Springer Berlin Heidelberg.
- Nielson, J. (2006). The 90-9-1 Rule for Participation Inequality in Social Media and Online Communities. Retrieved April 13, 2017, from <https://www.nngroup.com/articles/participation-inequality/>
- Pajor, T. (2009). *Multi-Modal Route Planning*. Universität Karlsruhe.
- Prandtstetter, M., Straub, M., & Puchinger, J. (2013). On the way to a multi-modal energy-efficient route. In *Industrial Electronics Society, IECON 2013-39th Annual Conference of the IEEE* (pp. 4779–4784).
- Pyrga, E., Schulz, F., Wagner, D., & Zaroliagis, C. (2004). Towards realistic modeling of time-table information through the time-dependent approach. *Electronic Notes in Theoretical Computer Science*, 92, 85–103.

- Pyrga, E., Schulz, F., Wagner, D., & Zaroliagis, C. (2008). Efficient models for timetable information in public transportation systems. *Journal of Experimental Algorithmics (JEA)*, 12, 2–4.
- Rehrl, K., Bruntsch, S., & Mentz, H. J. (2007). Assisting Multimodal Travelers: Design and Prototypical Implementation of a Personal Travel Companion. *IEEE Transactions on Intelligent Transportation Systems*, 8(1), 31–42.
- Sanders, P., & Schultes, D. (2006). Robust, Almost Constant Time Shortest-Path Queries in Road Networks. In *The Shortest Path Problem* (pp. 193–218).
- Sanders, P., & Schultes, D. (2007). Engineering fast route planning algorithms. In *International Workshop on Experimental and Efficient Algorithms* (pp. 23–36).
- Schlich, R., Schönfelder, S., Hanson, S., & Axhausen, K. W. (2004). Structures of leisure travel: temporal and spatial variability. *Transport Reviews*, 24(2), 219–237.
- Schulz, F. (2005). *Timetable information and shortest paths*.
- Sierpiński, G. (2013). Changes of the modal split of traffic in Europe. *Archives of Transport System Telematics*, 6.
- Sierpiński, G., Celiński, I., & Staniek, M. (2014). Using Trip Planners in Developing Proper Transportation Behavior. *International Science Index*, 8(11), 482–490.
- Sniedovich, M. (2006). Dijkstra’s algorithm revisited: the dynamic programming connexion. *Control and Cybernetics*, 35(3), 599.
- Tages-Anzeiger. (2013, October 14). Pendler wollen allein im Auto sitzen. Zürich. Retrieved from <http://www.tagesanzeiger.ch/schweiz/standard/Pendler-wollen-allein-im-Auto-sitzen/story/15232287>
- Van Bruggen, R. (2015). Loading General Transport Feed Spec (GTFS) files into Neo4j - part 1/2. Retrieved March 20, 2017, from <http://blog.bruggen.com/2015/11/loading-general-transport-feed-spec.html>
- Varone, S., & Aissat, K. (2015). Multi-modal Transportation with Public Transport and Ride-sharing - Multi-modal Transportation using a Path-based Method. In *Proceedings of the 17th International Conference on Enterprise Information Systems* (pp. 479–486).
- Wagner, D., Willhalm, T., & Zaroliagis, C. (2005). Geometric containers for efficient shortest-path computation. *Journal of Experimental Algorithmics (JEA)*, 10, 1–3.

Appendix

Neo4j Query: Olten -> Rho Fiera Milano (carpooling only)

```

match (tu:Stop:CP { name: "Olten" })--(st:Stoptime)
  where st.departure_time_s > 52200
  and st.departure_time_s < 54000
with st
match (ant:Stop:CP { name: "Rho Fiera Milano" })--(st2:Stoptime)
with st2, st
call apoc.algo.aStarConfig(st, st2, 'PRECEDES>', { weight: 'duration', default: 10, x:
  'longitude', y: 'latitude' })
  yield path, weight
with nodes(path) as n, weight as w
unwind n as nodes
match (nodes)-[:LOCATED_AT]->(s:Stop),
  (nodes)-[:PART_OF_TRIP]->(t:Trip)-[:USES]->(ro:Route)-[:OPERATES]-
  (a:Agency)
return s.platform_code, nodes.departure_time, nodes.arrival_time, s.name, t.car,
  ro.short_name, a.name, a.rating

```

Neo4j Query: Bern Wankdorf -> Olten (multimodal)

```

match (tu:Stop:CP { name: "Bern Wankdorf" })--(st:Stoptime)
  where st.departure_time_s > 59816
  and st.departure_time_s < 60000
with st
match (ant:Stop:Track { name: "Olten" })--(st2:Stoptime)
with st2, st
call apoc.algo.aStarConfig(st, st2, 'PRECEDES>|TRANSFER>', { weight: 'duration', de-
  fault: 10, x: 'longitude', y: 'latitude' })
  yield path, weight
with nodes(path) as n, weight as w
unwind n as nodes
match (nodes)-[:LOCATED_AT]->(s:Stop),
  (nodes)-[:PART_OF_TRIP]->(t:Trip)-[:USES]->(ro:Route)-[:OPERATES]-
  (a:Agency)
return s.platform_code, nodes.departure_time, nodes.arrival_time, s.name, t.car,
  ro.short_name, a.name, a.rating

```


Personal Declaration

I hereby declare that the submitted thesis is the result of my own, independent work. All external sources are explicitly acknowledged in the thesis.

April 19, 2017

Julian Kissling