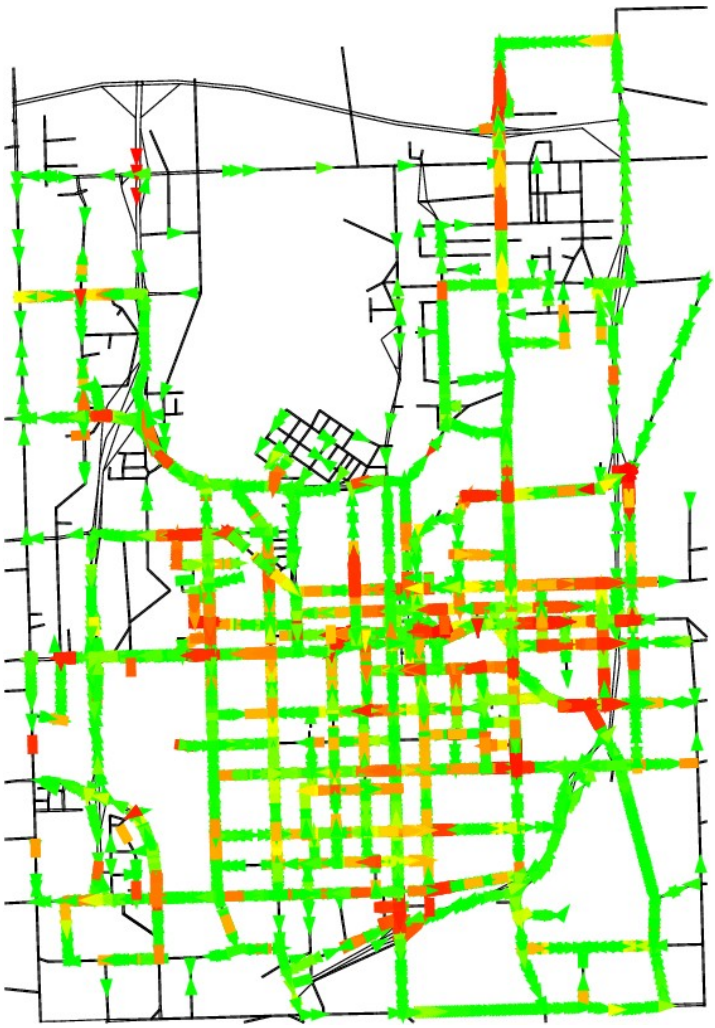

Implementation of an autonomous taxi service in a multi-modal traffic simulation using MATSim

Master thesis in Complex Adaptive Systems

Sebastian Hörl
Göteborg, 2016

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Energy and Environment

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

IVT Institut für Verkehrsplanung und Transportsysteme
Institute for Transport Planning and Systems

Implementation of an autonomous taxi service in a multi-modal traffic scenario using MATSim

SEBASTIAN HÖRL

Claes Andersson, Associate Professor (Docent)
Chalmers, Examiner

Dr. Alexander Erath
ETHZ, Supervisor

Pieter J. Fourie
ETHZ, Supervisor

Department of Energy and Environment
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2016

Implementation of an autonomous taxi service in a multi-modal traffic scenario using MATSim
SEBASTIAN HÖRL

Department of Energy and Environment
Chalmers University of Technology
SE-41296 Göteborg
Sweden
Telephone +42 (0)31-772 1000

Cover:
Snapshot of a traffic simulation using MATSim. Graphic created using SENOZON VIA.

Department of Energy and Environment
Göteborg, Sweden 2016

Abstract

In the foreseeable future it will be possible to have self-driving cars in the city environment. A number of simulations have been performed to assess the impact and possibilities of autonomous vehicles, mainly seeing them as a means of public transportation. While previous studies have focused on the replacement of existing private car trips, the interplay between a newly introduced autonomous vehicle taxi service with the existing means of transport in a traffic scenario will be investigated here.

The study is based on the agent-based traffic simulation framework MATSim and covers the whole process from the implementation of the transport mode, several extensions of existing MATSim assets and the simulation results.

It has been found that the introduced AVs are mainly competing with established public transport services and that the net driven distance by cars in the network is increased. It is claimed that varying pricing schemes alone cannot lead to a more balanced and sustainable mode share, but that they need to be supported by pinpointed incentive and/or taxation policy.

Keywords

autonomous, taxi, agent-based model, multi-modal, traffic simulation

Acknowledgements

The computational work for this thesis was done on resources of the National Supercomputing Computer, Singapore (https://www.nscc.sg).

Contents

1 Introduction

Autonomous vehicles are expected to have a great impact on how the traffic situation in our cities will look like in the not so far future. During the last years, autonomous vehicle technology took great leaps forward, examples being Google's self-driving cars (Google, 2016) or Tesla's latest software updates on autonomous parking functionality (Tesla, 2016). Furthermore, renowned car manufacturers such as Audi recently joined the competition. Ambitious tests in real-world conditions are fostered all over the world, examples being Volvo's DriveMe project with 100 autonomously driving cars on the highways of Gothenburg (City of Gothenburg, 2016) and the LUTZ pathfinder project, establishing the use of three autonomous transport pods in the city center of Milton Keynes (Transport Systems Catapult, 2016).

Especially big cities like Singapore could gain a lot from autonomous car technology, be it from privately owned cars to publicly owned services. A floating fleet of autonomous vehicles (AVs) could have a drastic impact on land usage, reducing the area of parking space needed in the urban environment. At the same time, it has the potential improve road safety, access to transportation, congestion, and emissions (Tan and Tham, 2014). In fact, it is predicted that an AV fleet size of one-third the size of the number of private cars could serve the associated demand that is generated today (Spieser et al., 2014). From a user perspective, shared autonomous taxis could solve the classic "last mile problem" where AVs could bridge the transport gap from the closest public transport facility to the homes of the people (Litman, 2015). As soon as a particular supply of AVs is reached, prices are predicted to become highly competitive to if not even cheaper than private car ownership (Chen and Kockelman, 2016).

On the other side, with the adaptation of autonomous transport, fundamental moral problems (Hevelke and Nida-Rümelin, 2015) need to be discussed, and related questions in liability and ownership need to be answered (Anderson et al., 2016). In general, predicting how and when autonomous vehicles will be on the roads is a highly complex problem, and most of the proclaimed benefits can easily be defeated due to a lack of reliable data.

To give an example, one of these benefits is having less congestion (International Transport Forum, 2014). However, it is claimed that to make a passenger comfortable in a self-driving car, significantly smaller accelerations and decelerations are allowed, and thus the overall movements on the roads will be slowed down (Le Vine et al., 2015). Therefore, having only assumptions on how people would experience shared AV services, replacing a

huge percentage of contemporary cars with autonomous ones could in fact increase overall congestion.

In consequence, while the technological development in autonomous driving already came a long way, there is still much demand for research on an upper level, looking at the overall economic and societal picture (Silberg et al., 2013; Schoettle and Sivak, 2014). A valuable tool for doing this is the use of traffic simulation (Fagnant and Kockelman, 2014; International Transport Forum, 2014). Numerous efforts have been made in predicting the usefulness of AVs for a city regarding AV supply, i.e. answering the question how many shared AVs would be needed to serve the existing demand (Boesch et al., 2016). On the contrary, fewer results are available on how the new travel mode would interact with existing travel options and how customer preferences influence the mode choice.

The agent- and activity-based traffic simulation framework MATSim (Horni et al., 2015) allows for such research (Boesch and Ciari, 2015) by taking into account assumed customer preferences and letting people interact with a newly added means of transportation. The framework has been successfully used in a range of studies from autonomous taxi services in Berlin and Barcelona (Bischoff and Maciejewski, 2016) up to the simulation of the whole transport network of Singapore (Erath et al., 2012). While the framework allows for a quite precise prediction of traffic flows for scenarios involving car traffic and public transport, it yet does not allow the simulation of dynamically acting autonomous vehicles embedded in the overall traffic situation.

Therefore, the purpose of this thesis will be to implement means of simulating autonomous taxis within the MATSim framework. Subsequently, measurements on how people would switch to the new transport mode given certain supply levels, acceptance levels, and pricing schemes will be made. A major challenge will be to account for an acceptable simulation speed while keeping the dynamic detail, which is needed in order to simulate intelligently acting AV taxi fleets. At the same time the functionality should be kept as versatile and extendable as possible in order to make it possible to gain research results on a multitude of factors, which influence the adaptation of autonomous vehicles.

In the scope of this thesis, a basic model of autonomous vehicles, which are transporting one passenger, is developed. Many interesting aspects such as shared AVs, AVs for the purpose of feeding public transport facilities or simulating a refuel/recharging infrastructure can be subject of future research. Since the model relies on data on how likely people are to use autonomous technology, the model at the moment will be mainly pointed towards finding qualitative results on the interplay between factors that define the traffic situation. However, more and more data on customer preferences, actual experiences of

AV technology and pricing information will become available over the next years, leading the model to give a more accurate and reliable look into the future.

Given those data sets, the simulation developed in the thesis has the potential to act as a valuable tool in transport planning and urban development. It will give great aid in the implementation of the needed infrastructure, the development of pinpointed transport solutions and a restructuring of the present traffic network to account for the adaptation of autonomous vehicles.

The thesis at hand is structured in four main parts: Section 2 will introduce how traffic simulation is performed in the MATSim framework and highlight the aspects that are important to know for the implementation of autonomous vehicles. Section 3, as another prerequisite for setting up a meaningful simulation, covers the adaptation of a readily available MATSim traffic scenario to the increased network resolution, which is needed here. In sections 4 and 5, the technical implementation of the AV simulation will be explained in detail on two abstraction layers, first introducing a new framework for simulating dynamically acting agents in MATSim and then creating a model of autonomous taxis based on that. Then, in section 6 the model will be tested with a variety of parameter configurations, giving insights on the general working of the model and predictions of AV usage in the artificial Sioux Falls test scenario. Finally, section 7 will give an outlook on a multitude of possible extensions of the model and further research questions that can be answered using the model at hand.

2 Simulation Framework

The investigations of autonomous vehicles in this thesis will be based on the agent-based transport simulation MATSim (Horni et al., 2015). The following sections will outline how the framework works and where it can be extended to shape it towards an autonomous taxi simulation. An overview will be given on which components need to be modified and how the final transport situation will result from all the different parts that are playing together in MATSim.

2.1 Agent-based transport simulation

The approach that is used in MATSim is to simulate a virtual population of a city on a per-agent timestep-based level. At the beginning of a simulation run, each person in the synthetic population has an initial plan of what it is supposed to do during the simulated day. Those those plans consist of two elements:

Activities have a start and an end time, as well as, depending on the respective scenario, certain constraints on when the earliest start or latest end time could be, i.e. how much the current values could be alternated to still give a realistic time frame. Furthermore, minimum durations can be defined for which and agent has to stay at a certain activity location. Those locations are given as facilities, which have specific coordinates on the scenario map. Usual activities are "home" (which usually is the first plan element of a day) and "work". More elaborate simulations can additionally use an arbitrary number of secondary activities.

Legs are the second type of plan elements. These describe connections between two activity locations and contain information like which mode of transport the agent will use (e.g. "car", "public transport", or "walk") and, depending on the selected mode, further data like the route that should be taken through the street network.

A typical day plan of a MATSim agent can be seen in fig. 1. The agent starts at home, then walks to his job, stays there for a certain time and then goes back home. Each agent in a popuation (which can range from several hundred to hundeds of thousands) has its individual plan that is executed when one day is simulated.

The network, on which the simulation is taking place, is described through nodes and connecting links. These are defined by a specific flow capacity which tells the simulation

Figure 1: A typical agent plan in MATSim

how many vehicles are able to pass the link within a certain time frame while the length and an average link speed determine how fast vehicles will travel to the next node.

The whole MATSim simulation is performed time-step by time-step. In a single simulation step (usually 1 second) the agents that are currently in an activity do not need to be taken into account unless their scheduled activity end is reached. The other agents, which are currently on a leg, are simulated in a dedicated traffic network simulation. This simulation moves the agents according to the capacity and current congestion from the start node to the end node of the respective links. Traffic is not simulated on a micro-level (i.e. the vehicles do not have distinct positions along the link) to increase the simulation speed. Consequently, congestion on the traffic network is emerging from the single travel plans of all the agents. If there are too many agents who try to use one route at the same time, the overall congestion will increase.

The actual routes that are being taken by the agents are generated based on the fastest path through the network for a certain leg, with a certain amount stochasticity added into the pathfinding process in order to avoid artificially created bottlenecks. This approach allows the optimization of the plans to be separated from the actual simulation of the plans, but is not suitable for the simulation of an AV taxi service, where routes through the network must be generated dynamically, based on the current demand. Therefore, to simulate dynamic agents for AVs, which are not statically residing in a fixed-timed activity or a predefined leg route, significant changes need to be made, and some of the computational advantages of the simulation approach need to be partly circumvented. This, however, mainly addresses implementational details and will be discussed later throughout the thesis in section 4.

All steps described above, i.e. the simulation of activities and legs during a whole simulation day, are summarized as the "Mobility Simulation" or, in short, Mobsim of the MATSim framework.

2.2 Utility-based scoring

As pointed out in the last section, individual travel decisions might lead to waiting times on the network, which then can also lead to late arrivals at the designated activity locations, which usually would be disadvantageous in the real world. Therefore, it might be beneficial for an agent to reconsider the time and route choices that had been made for the current day, just as a real person would do.

In order to "know" whether a plan worked out well or was disadvantageous, the single elements need to be weighed and quantified. This is done using the Charypar-Nagel scoring function (Horni et al., 2015):

$$S_{plan} = \sum_{q=0}^{N-1} S_{act,q} + \sum_{q=0}^{N-1} S_{trav,mode(q)} \tag{1}$$

It combines the marginal utilities of all activities ($S_{act,q}$) ranging over the N activities in a plan and the marginal utilities for all the legs in between.

The marginal utility for the activities, among other factors, depends on how long the activity has been performed, whether the agent needed to wait to start it (due to an early arrival) or whether it was forced to leave early. For more details the complete computation is shown in (Horni et al., 2015).

For the scope of this thesis the travel utility is more interesting. A basic version for a single leg q can look as follows:

$$S_{trav,mode(q)} = C_{mode(q)} + \beta_{trav,mode(q)} \cdot t_{trav,q} + \gamma_{d,mode(q)} \cdot \beta_m \cdot d_{trav,q} \tag{2}$$

Mode Choice The first term, $C_{mode(q)}$, describes a constant (dis)utility for the choosing a certain mode for the leg. It can be interpreted as how "favorable" going on a trip in the specific mode is and is negative. An classic use of the parameter is to model constant costs per trip in a mode.

Travel Time The parameter $\beta_{trav,mode(q)}$ is the marginal utility of traveling, which is multiplied by the time spent on the leg $t_{trav,q}$. It signifies how favorable it is to spend time on such a leg, i.e. the longer one needs to stay in a car or public transport, the larger the disutility gets (and therefore the parameter is usually negative). Values which are smaller, therefore, stand for travel modes where time is spent less useful or comfortably. The unit is "utility per time".

Travel Cost The third element involves the (positive) marginal utility of money β_m, which is a universal simulation parameter and describes how the utility of money can be weighed against e.g. time., the unit being "utility per monetary unit", e.g. EUR. It is multiplied by the (negative) monetary distance rate $\gamma_{d,mode(q)}$, which states, to how much disutility per spanned distance the leg will lead. For the given example it would be stated as "EUR per meter". The parameter is useful for imposing distance-based fares in a transport mode and thus making it monetarily attractive or unattractive compared to other ways of traveling.

Beyond these parameters, which are usually used for all travel modes, there are a number of additions, e.g. for public transport, or yet unused options, such as a direct marginal utility of distance traveled.

For the purpose of simulation autonomous taxi services, one addition is made:

$$S_{av} = C + \beta_{trav,av} \cdot t_{trav} + \gamma_{d,av} \cdot \beta_m \cdot d_{trav} + \beta_{wait,av} \cdot t_{wait} \tag{3}$$

Here, β_{wait} is the marginal utility of waiting time, quantifying how disadvantageous it is to wait for an AV to arrive.

The utility computations presented here are used in the "scoring phase", which is taking place in MATSim after one simulation day (in fact, usually 30h are used) has been performed. Then all experienced legs and activities are scored, and each agent is assigned a final score, depending on which further replanning is done, as described in the next part.

2.3 Evolutionary replanning

The last step is to make all the agents replan their day in order to "learn" more optimal plans which make sure that they arrive on time at the activity locations and avoid congestion. This is done using an evolutionary algorithm in the following way:

Usually, an agent will start out with one quite random plan, go through it during one whole day and then get a score for it. Afterward, in some iterations, this plan is copied and modified slightly. Those modifications can happen with respect to start and end times of activities, mode choices for certain legs, etc. So after one iteration the agent might already have two plans to select.

Before the next day starts, one of the available plans is selected according to a certain strategy. The standard approach is to do a multinomial selection with respect to the

previously obtained plan scores. So one after another plans, will be created, scored, modified, rescored and so on. Because of the selection process, which favors high scores, better and better choices will be made.

However, this is done for each and every agent, so while improving the performance of one agent's plans, this might effect the performance of other agents negatively, which is especially true if one thinks about the example of highly congested roads due to too many agents choosing the same route. Finally, though, the algorithm reach at a quasi-equilibrium, which in MATSim is usually referred to as the "relaxed state", in which the average score of the used plans stabilizes within a reasonable variance.

Each "day" that is simulated in this manner is usually called an "iteration" of the simulation. It is common to divide these iterations into two parts. The first one is the "innovation phase", where plans have a certain probability to be modified while innovation is turned off in the second phase. This means that the agent will only choose among the present plans in his repertoire (usually around 5) and rescore them again and again, until the most favorable is selected.

All of the above is known as the "replanning" in MATSim. Putting everything together, a whole cycle in a MATSim simulation can be seen in fig. 2. Since the final traffic situation evolves from the evolutionary choice in the replanning and selection, as well as from the emergent congestion in the Mobsim, this whole cycle is usually referred to as a "co-evolutionary" algorithm.

Figure 3 shows a typical progression of the population-wide score in a MATSim simulation. What one can see there is an average of the worst and best plans of each agent. Additionally the mean of the per-agent average scores in their list of plans is displayed. Finally, one can see the average of all the plans that have been executed by the agents in a particular iteration.

The first phase until iteration 100 is the innovation phase, where time and mode choices can be made, while it is turned off at iteration 100. There, because now the best plans must adapt to the overall situation, the best plans are loosing in utility. Finally, a quite stable population-wide relaxed state is reached.

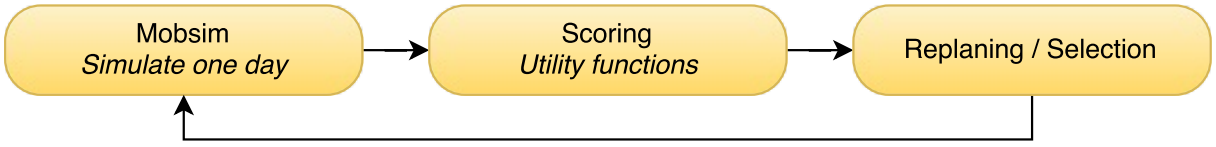

Figure 2: The basic co-evolutionary algorithm of MATSim, showing the three main stages: Mobsim, Scoring and Replanning/Selection

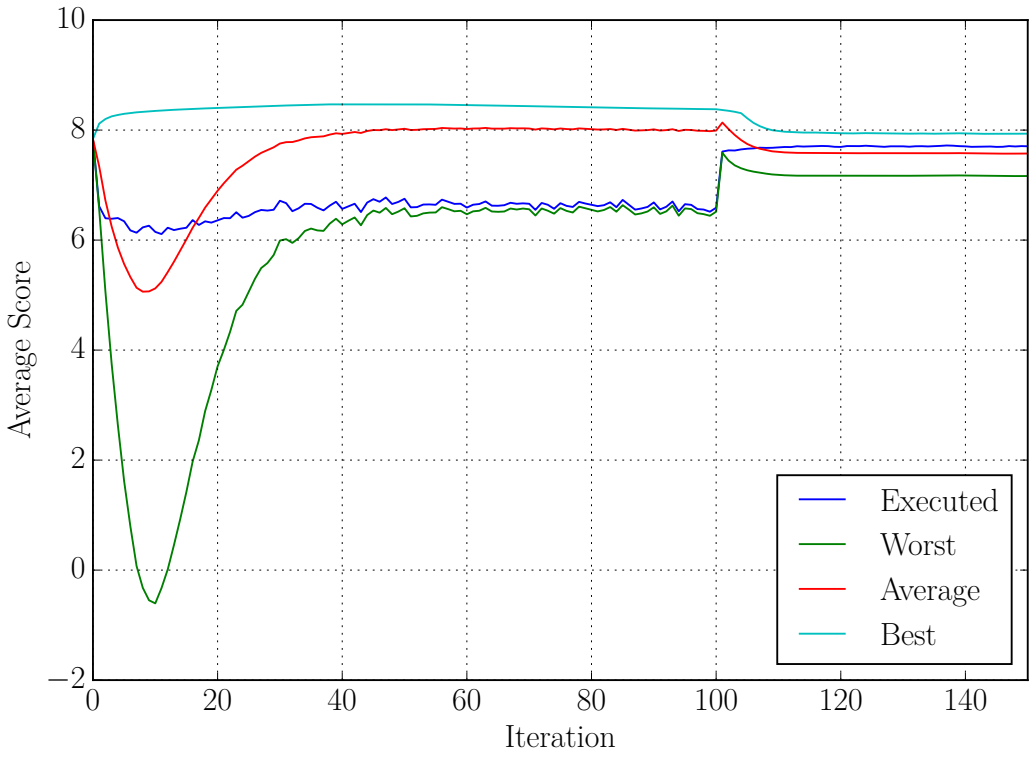

Figure 3: Typical progression of the agent scores throughout a MATSim simulation. "Worst" means that the plans with the worst scores from all the agents are taken and averaged, the same applies for the other graphs.

2.4 Queue-based simulation

The heart of the simulation in MATSim is the queue simulation, or more briefly, the QSim. This part of the framework is iterating through all the agents that need to take action in the current simulation step.

Itself, the QSim is split into the handling of agents which are currently performing an activity (i.e. are in an idle state in terms of the traffic simulation) and another part, the Netsim, which is simulating the traffic network.

When running the Netsim, the agents are moved on the network according to their current route and dependent on congestion. After moving an agent, the Netsim checks whether the agent should end its trip at the current link to which he has been moved. If this is the case, the agent is removed from the Netsim and the next agent state is computed.

The result of this computation is either that the agent wants to start a leg, in which case he is reinserted into the Netsim, or that he wants to start an activity, which will make the agent be added to the activity queue.

This activity queue is the other main component of the QSim. In fact, it is a priority queue, where all agents are sorted according to the time at which they want to end the next activity. So when adding an agent to the activity queue, it first is checked when the activity should be ended and then the agent is inserted at the corresponding position in the queue.

The processing of this queue in the QSim for each simulation step then works as follows: The first element of the queue is looked at and it is checked, whether the agent should already end the activity. If this is not the case, the simulation step is already finished. On the other hand, if the activity should be ended now (or previously if the time resolution of the simulation is quite high), the agent is removed from the top of the queue and the next state is computed as described above. Then the new top element of the queue is examined. The whole QSim simulation is schematically rendered in fig. 4.

The big advantage of this simulation architecture is as follows: When an agent is in an activity, no computation needs to be performed. So instead of polling all the agents in every simulation step to check if they want to end an activity, the computational demand is much lower when using the queue, since many agents can be skipped. The sequential processing of the priority queue is very fast in terms of computational complexity ($\mathcal{O}(1)$ for checking the top element and $\mathcal{O}(\log n)$ for fetching it) (Java API, 2016).

Furthermore, the same concept is used within the Netsim to speed up the computation of the traffic situation. In both cases, for the QSim and Netsim, those savings in computation time naturally decrease the versatility of the simulation environment.

One major drawback is that if an agent, which is already queued, should abort its current activity, it needs to be removed from the queue and added at a new position. Both operations are quite costly for the priority queue (Java API, 2016), so if more and more reschedulings are needed, the computational overhead can become quite large.

However, for truly dynamic agents, like autonomous vehicles, it is necessary to adopt their plans frequently and thus some thought needs to be put into how to achieve this freedom while still keeping as many advantages from the existing simulation architecture. Section 4 will explain how this problem has been overcome in this thesis.

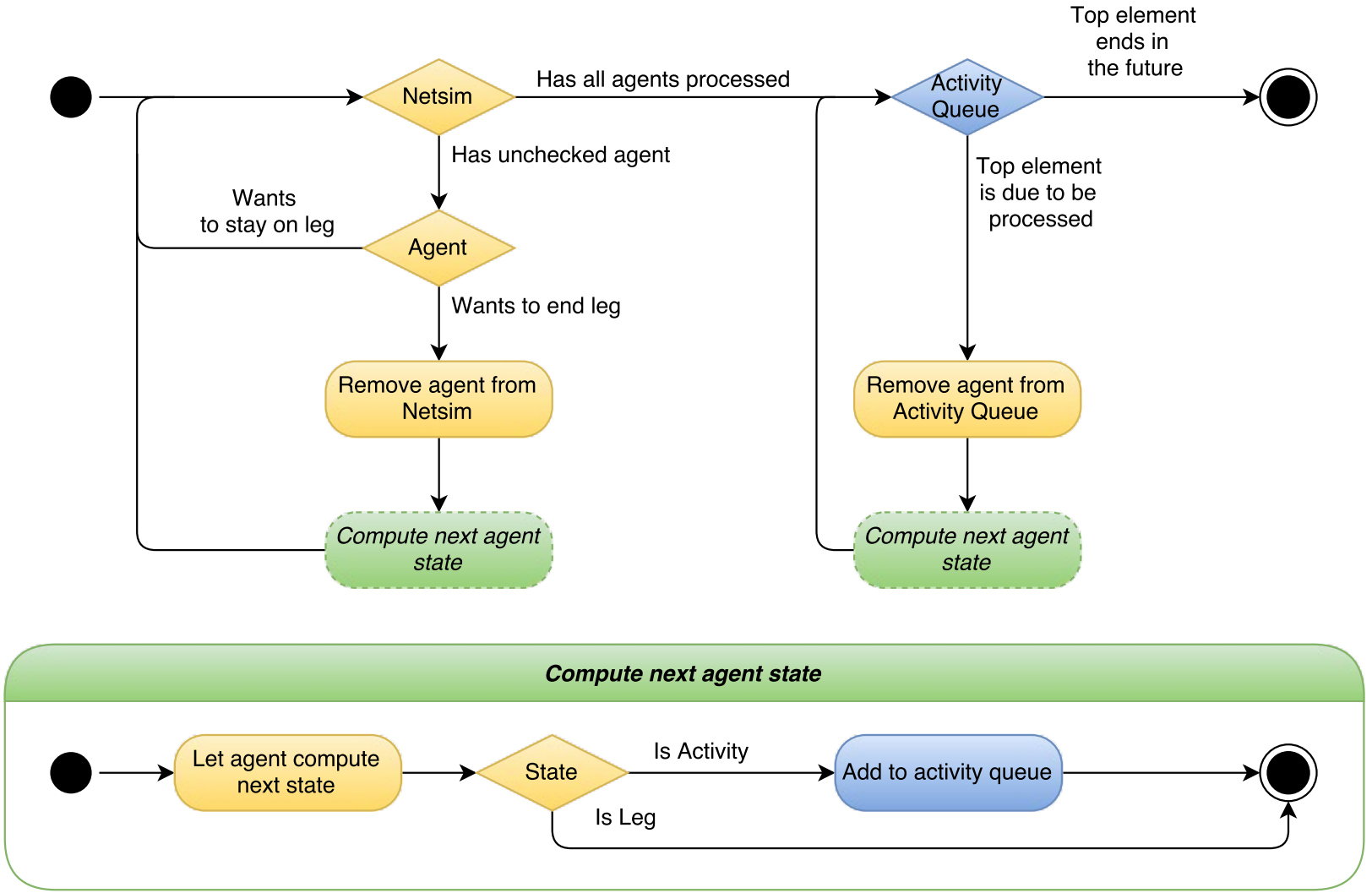

Figure 4: Simplified structure of the QSim in MATSim.

3 The Sioux Falls Scenario

The City of Sioux Falls in South Dakota (fig. 5) has been a classic test case in transport research for more than four decades, being first mentioned in this context in Morlok et al. (1973). While none of the numerous implementations of the network are intended to be accurate and realistic with respect to the actual City of Sioux Falls, they are merely aiming towards providing downscaled, computationally tractable test cases for transport planning and simulation problems.

In Chakirov and Fourie (2014), the scenario ("Sioux-14", fig. 6) has been adapted to the MATSim framework. A sparse street network, which is computationally easy to handle by the simulation, was introduced. It consists of 27 nodes and 76 links, representing the main arterial roads of the city, split further down into 282 nodes and 334 links in order to arrive at partial link sizes of less than $500m$. This is necessary because MATSim agents start their travels at the start node of a link and thus a high resolution is needed to avoid unrealistic clustering.

On the supply side, there is furthermore a public transport network, which consists of 5 lines with bus stops along the arterial roads in a distance of $600m$. The stops are placed at a distance of $5m$ perpendicular to the road and departures from the lines' respective start links take place every 5 minutes.

Much of effort has been put into the modeling of the demand side, covering the realistic generation of home locations, workplaces, secondary activity locations, as well as the distribution of socio-economic factors such as age, car ownership and gender across a synthetic population, as it is needed for the agent-based simulation.

In this regard, the scenario provides a lightweight example of a complete MATSim simulation, where it is easy to test different parameters and extensions to the framework on a near-realistic baseline scenario. However, while working with the original Sioux-14 network during the course of this thesis, it became evident that the simplified structure of the underlying traffic network does not provide enough resolution for the simulation of autonomous vehicles.

The main reason is that agents, who choose to take a car in the Sioux-14 network, probably living in the middle of the rectangular regions of the network would be teleported to the nearest link to start their travels. This would also be true for autonomous vehicles, which is not a realistic assumption in both cases, especially when comparing it with public transport, where agents in MATSim are explicitly penalized for covering the distance between home and bus stop by foot. Furthermore, the effect of people being

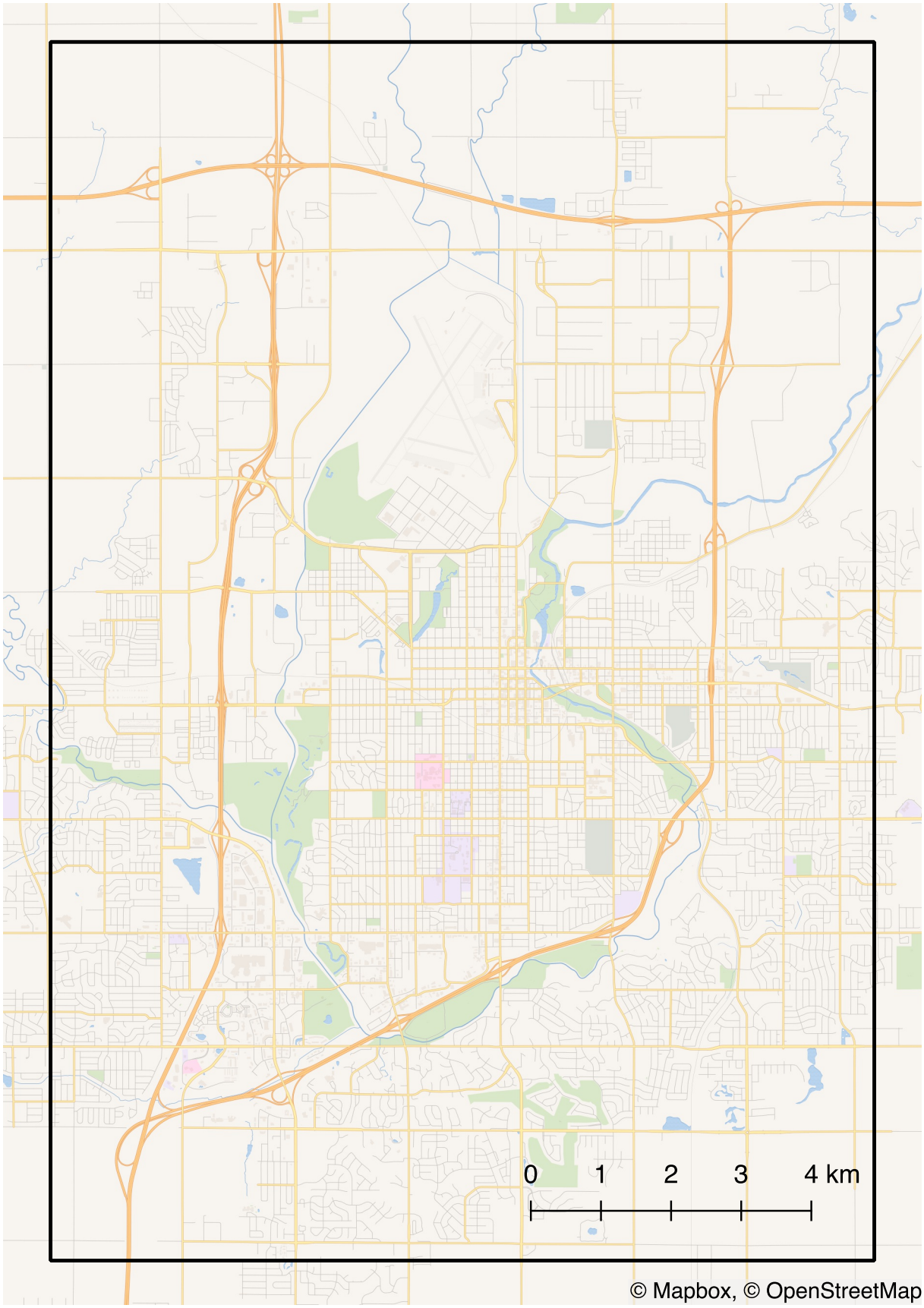

Figure 5: Map of Sioux Falls. The bounding box encapsulates the processed region with longitude from $-96.8105°$ to $-96.6653°$ and latitude from $43.4729°$ to $43.6286°$.

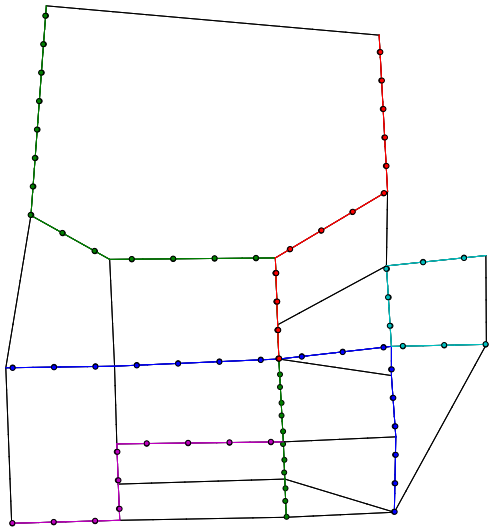

Figure 6: Sioux-14 road and public transport network

more inclined to opt for an autonomous taxi when living far from public transport can only be convincingly simulated on a finer network.

The following sections will describe how, starting from the initial Sioux-14 scenario, a new more fine-grained versatile test scenario for MATSim has been developed, which in turn has been used as the basis of the following investigations in this thesis.

The new Sioux-16 network, which has been developed in this thesis, is based on the demand model of Sioux-14, which means that all locations for homes, workplaces and secondary activities are kept equal, while it differs on the supply side, aiming to resemble the original scenario as closely as possible. The following sections will describe, how the Sioux Falls network from OpenStreetMap (with the state as of 18 Apr 2016) has been converted and adjusted to be compatible with MATSim and closely match the prior version of the test scenario. Furthermore, it will be explained, how the public transport network has been adapted to the fine-grained Sioux-16 scenario.

3.1 Network generation and adjustment

For the creation of the new scenario, an area covering Sioux Falls has been captured from OpenStreetMap, which can be seen in fig. 5. Using the MATSim exporter in the JOSM tool [1], a simplified, MATSim-compatible network of the selected region has been created. In this process, all primary, secondary and tertiary streets have been selected, while road types further down the hierarchy (for instance residential streets) have been omitted.

[1] https://josm.openstreetmap.de/

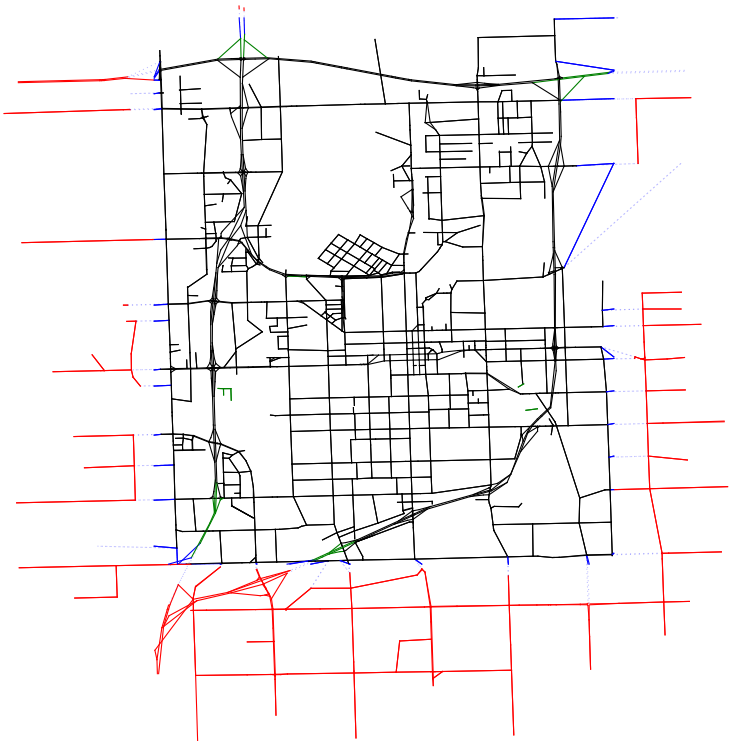

Figure 7: Automatic network modifications: Red links have been removed while blue links have been shrunk to the closest border points of a bounding box enclosing the facilities of the scenario. Green lines have been removed either for being detached from the main network being sinks/sources.

However, some adjustment was needed for the network to play nicely with the given facilities from the Sioux-14 scenario. Most importantly, the network from OSM was defined in the EPSG:3857 coordinate system, while Sioux-14 uses EPSG:26914. Therefore, in a first step, the coordinates of all the generated nodes needed to be converted to the old system.

In a second step, the positions of all facilities in Sioux-14 have been obtained and a bounding box with a margin of $500m$ around them has been computed. Subsequently, all links, which were located out of the bounding box, were removed from the network, while those who were crossing the borders have been cut to fit into the area. The initial network with all removed (red) and cut (blue) roads can be seen in fig. 7. In total 352 outside links have been removed, and 83 links have been adjusted.

After that, a cleanup up the network needed to be done, partly because of artifacts due to the filtering of road types, partly because of the removal of the external sections. This removal was done in a couple of steps:

1. The lengths of all the links have been updated to the L^2 distance of their respective start and end nodes in the new coordinate system.

2. The network has been searched for sources (nodes that only have outgoing links) and sinks (nodes that only have incoming links). Those have been removed, because they make no practical sense. This proecdure lead to 53 removed links in total.

3. One seed node has been defined, which definitely belongs to to the street network and then by traversing the all paths from that node, it has been determined, which streets belong to the main network. All remaining nodes and links, which have become detached from this main network have been removed (11 nodes and 14 links).

4. Finally, the network has been searched for duplicate links with the exact same start and end node. Only the first of those links have been kept, which lead to the removal of 3 duplicates.

5. In Sioux-14, the links have been split such that there are no connections longer than $500m$. This procedure has also been applied to the network at hand, leading to 389 split links, which have been cut into a number of equal parts with less than $500m$ length depending on their overall length.

Regarding nodes, these procedures in total lead to an increase of nodes from 1392 to 1806 and in an increase of links from 2957 to 3335. The links that have been removed during the cleanup are colored in green in fig. 7.

3.2 Public Transport Adaptation

The adaptation of the public transport network to the new ("Sioux-16") scenario posed some challenges that needed to be solved:

- The links of the original network did not exist anymore, obviously the routes for the different public transport needed to be mapped to the new network.

- The stops from the original network could not be mapped easily to the new roads, partly because some streets in Sioux-14 were "invented", only approximating connections in the finer network on a very coarse level, but also because roads that consisted of two overimposed links for both directions were now split up into two spatially distinct lanes (as can be seen in the upper left part of fig. 8).

In order to get a rough routing for the bus lines, the main nodes of the Sioux-14 network have manually been mapped by hand to nodes in the Sioux-16 network. This made it possible to obtain new public transport roads in terms of those guide points: For each of the lines it has been defined, which guide points should be traversed and in which order. Then, the Dijkstra algorithm (Dijkstra, 1959) with travel time as the objective has been used to find the shortest path from waypoint to waypoint. After fitting those partial paths together, the whole routes in terms of the links of the Sioux-16 network have been obtained. As can be seen in the colored routes in fig. 8 this made it easy to obtain routes that take into account the specific map structure (for instance the highway on the upper left or the one-way streets, which are traversed by the blue line in the center of the map).

The generation of bus stop facilities was a more challenging problem. Given the location from Sioux-14, one could roughly match the stops to links along the new routes, which worked in principle. However, using this approach, bus stops were cluttered all along the lines. With the intention of having a rather realistic network some conditions needed to be taken into account:

- The bus stops of one line should be on opposite sides of a street and not have a large longitudial distance. While satisfiying results could obtained, for instance in the center with the blue line, the same approaches did not work well for the highway connectors on the left for the green line, and vice versa.

- The bus stops of parallel lines should be at the same locations, i.e. in the center where the green line uses the same roads as the red one, the same locations should be chosen. This constraint usually interfered with the approaches that took into account the first constraint (especially in the center for the blue line).

- Finally, some of the Sioux-14 locations where completely off the network in Sioux-16, for instance for the red line, where one can see a bend in the diagonal connection, which was modelled as a straight line in Sioux-14.

Given all those constraints the best approach seemed to put in manual work with some automated help. The final approach made use of a small program, written to manually choose the stop locations along all roads and then subsequently choose which stop locations should belong to which line. In this step one did not take into account the cases of two lanes, as just the general positions needed to be known (e.g. for the blue line in the center an approximate position between the two lines would be chosen).

In another step, the locations that had been assigned to each line have been assigned to the respective links along the line. So for the blue one, the average points in between would have been assigned to a link underneath for one direction, and another stop would be created for the link above. This resulted in a final set of stop locations.

Furthermore, some stop locations were located on one single link. This can happen if there is a link of roughly $500m$ and the stops are for instance located at $1m$ and $499m$ along the link. In those cases, the network needed to be broken up at those positions. In the current scenario, this leads to the splitting of only four links.

Finally, according to the setup of Sioux-14 the stop locations have been moved to a position normal to the link direction, with a distance of $5m$. Additionally a schedule with buses departing in $5min$ intervals has been created.

The final public transport network can be seen in fig. 8. It features 150 stops with an average distance of $520m$ and a median of $566m$. This is a result of not being able to put the stops as accurately in intervals of $600m$ as it is possible in Sioux-14. Moreover, the L^2 distance between two stops along the routes have been used instead of a measure along the actual path. The minimum and maximum distance between stops are $218m$ and $847m$, respectively.

All steps that have been described above have been implemented in a reusable and parametrized way. For instance, one could choose to allow a maximum link length of $5km$ and would still obtain a valid network at the end, probably just with a larger number of split links during the processing of the public transport.

3.3 Scenario Calibration

The Sioux-14 network features artificially chosen link capacities, which are based on a minimum of two and a maximum of three lanes per link, where three have only be chosen for a fraction of highway links (Chakirov and Fourie, 2014). While the added minor streets in Sioux-16 should not make too much of an impact on overall capacity, the highways and main arterials of the real network usually feature greater capacity (for instance by providing *four* lanes on the western highway).

Looking at fig. 9, it can be seen that the travel time (of cars) in the new network is lower (red), compared to Sioux-16 (black). This effect can partly be explained by the increased capacity, but also by the multitude of new options to choose the most effective route for a trip through the fine-grained network. In fact, the route choice might be the major

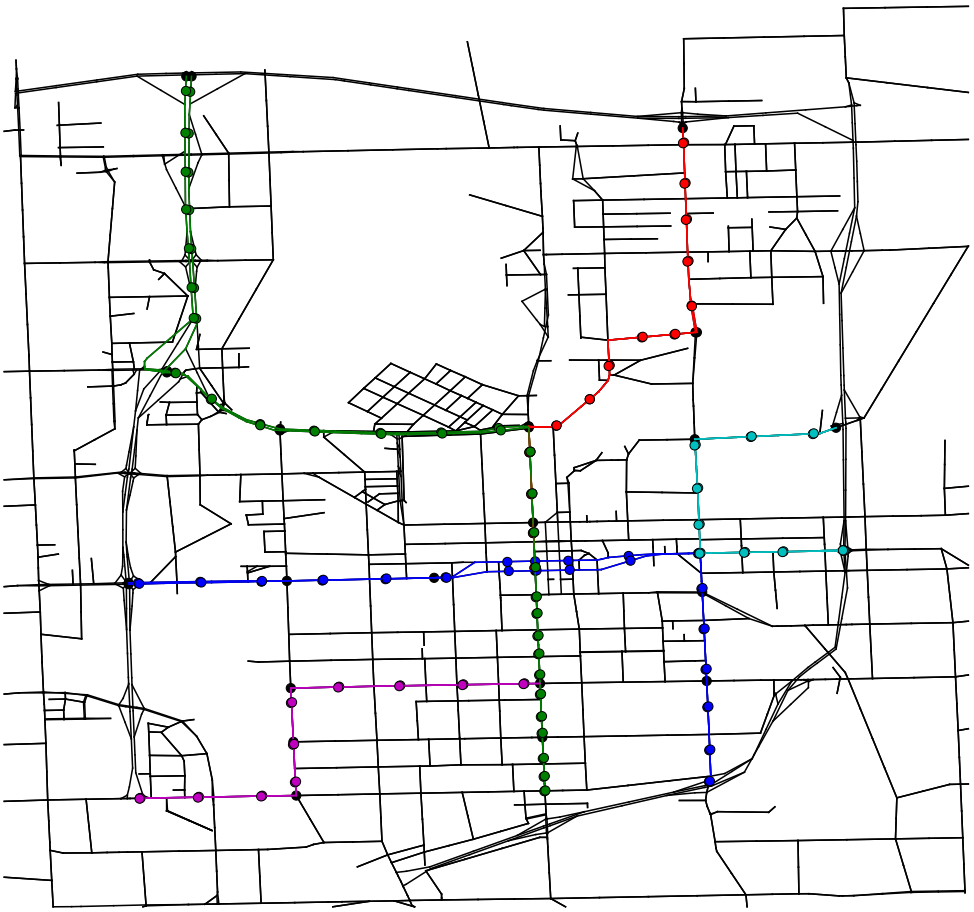

Figure 8: Sioux-16 network with five public transport lines and respective stop facilities

influence when comparing with the data from fig. 10. There the decrease in link speeds (relative to the freespeed) can be seen, which is quite similar, indicating a comparable amount of congestion in the network.

Depending on how important the comparability to Sioux-14 in a specific scenario is and what time of the day should be compared, it might be beneficial to adjust the flow capacity of the network[2]: In terms of travel times a scaling of 50% would resemble the afternoon peak way better than the 100% version, while the morning peak would best be recovered by a value in between (fig. 9). A similar situation arises for the link speeds in fig. 10, where a value of 50% is better suited for comparing the morning peak while the 100% scaling creates more comparable results in the evening.

Furthermore, in terms of link speeds, it can be seen that the Sioux-16 scenario features less congestion during the off-peak hours due to the possibility of distributing trips all over the network.

[2]In MATSim, this can be easily done in the Mobsim configuration, e.g. `qsim.flowCapacityFactor` for QSim

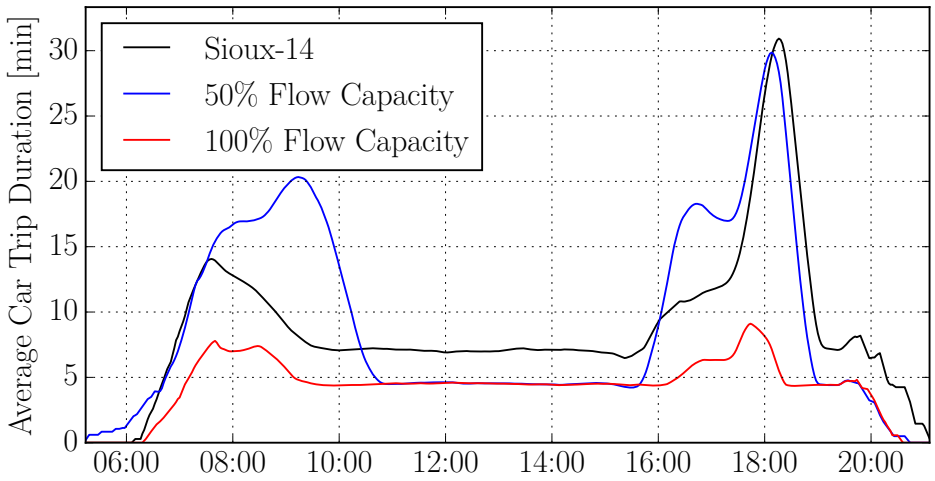

Figure 9: Comparison of the average duration of car trips by day time in Sioux-14 and Sioux-16 with 50% and 100% flow capacity.

In any case it has to be kept in mind that comparing the speed decreases is only a rough measure of network congestion. Most importantly, the values displayed are average values, which means that they biased towards outliers, i.e. capturing the changes in main arterials. That is a good comparison to Sioux-14, but on the other hand, the average is also taken over an increased number of links, of which some might rarely be used and therefore dragging the average down.

A comparison of the scenarios in terms of average travel times, distances and more shares can be found in table 1. What can be seen is that averaged over the whole day, values stay roughly the same, with the biggest differences being present in car traffic. There, especially the decrease in travel distance is noticeable but expected, since more direct routes can be taken. For public transport, the result of Sioux-16 is similar to Sioux-14, which is an indicator that the network is strongly resembling the former version.

Looking at the travel time and distance for the transit walk to and from public transport facilities, one can see that changes are quite small, which allows the conclusion that the fine-grained network does not incline agents to switch to the car mode on the new network. This is verified by a comparison of the mode shares, which stay roughly constant for the scenarios. A major difference can be seen in the mode shares of walking and public transport legs, where roughly two to three percent of the agents in the network switch from the walking mode to public transport.

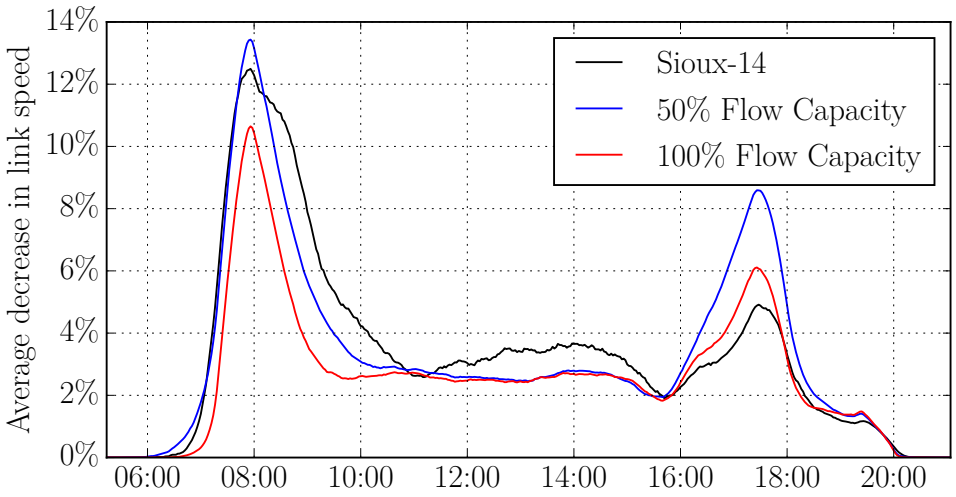

Figure 10: Comparison of the decrease of link speeds by day time in Sioux-14 and Sioux-16 with 50% and 100% flow capacity.

Table 1: Comparison of Sioux-14 and Sioux-16 in terms of travel measures. The respective distances and travel times of the walking legs to and from public transport facilities are included in the measures for public transport

Scenario Flow Capacity	Sioux-14	Sioux-16 50%	Sioux-16 70%	Sioux-16 100%
Travel Distances [km]				
Car	5.30	3.79	3.74	3.70
Walking	1.31	1.29	1.27	1.26
Public Transport	3.73	3.82	3.86	3.89
(Transit Walk)	1.31	0.53	1.28	1.28
Travel Times [mm:ss]				
Car	11:56	08:39	06:59	05:08
Walking	26:16	25:48	25:19	25:08
Public Transport	32:37	29:05	28:32	28:14
(Transit Walk)	24:05	21:50	21:55	21:59
Mode Shares				
Car	63.57%	63.23%	64.84%	65.71%
Walking	9.29%	7.50%	6.80%	6.56%
Public Transport	27.14%	29.27%	28.36%	27.72%

4 Dynamic Agents in MATSim

Several approaches have been proposed to allow for dynamic behavior in MATSim. The decision on which option is best depends mainly on what a level of complexity in the behavior is needed.

A simple version of dynamic behavior is implemented in the public transport (PT) extension, where passenger agents are saved in a list as soon as they reach the link in the network, where they should be picked up by a public transport agent. As soon as the PT agent (e.g. a bus) arrives at that link, persons from the link are deregistered from the list and "teleported" to the destination stop as soon as the vehicle arrives there. This setup fits very well into the queue based structure of MATSim.

An even more dynamic approach is used in the DVRP framework, which will be discussed in the first part of this section. At the time of this thesis and for the specific use case of autonomous vehicles, some drawbacks will be shown and finally a new abstraction layer for dynamically acting agents will be presented, which has been part of the thesis work.

4.1 DVRP

The DVRP extension (Maciejewski and Nagel, 2012; Horni et al., 2015) is designed to provide a level of abstraction to the simulation of dynamic transport services, such as taxis. Its general structure is quite flexible so that it is easy to implement for instance electric vehicles (Bischoff and Maciejewski, 2014) (which need to recharge at some point during the simulation) or taxis, which are roaming randomly through the city and serving requests when they are made by passengers (Maciejewski and Bischoff, 2015).

However, this flexibility comes at a cost: The architecture of DVRP circumvents the efficient queue simulation of MATSim for activities. While "normal" agents are simulated as described before, dynamic agents (DynAgents) have two modifications:

Legs are sent to the conventional Netsim. However, agents have the ability to change their paths dynamically, i.e. one can modify the route of the agent during the simulation steps and the next time the Netsim wants to move the agent or checks whether the agent should arrive at the current link, its response is calculated from the updated path.

Activities are simulated separately from the non-dynamic agents. As soon as a DynAgent starts an activity it is added to a list of active agents. In each simulation step a

specific callback for each of those agents is called and then it is checked whether the agent wants to end this activity or not. This polling approach, as depicted in fig. 11, makes sure that it is not necessary to know when an activity (for instance a taxi waiting for any requests) should end or how long it should take. On the other hand, this approach is much more computationally expensive than the efficient queue simulation. Given that the simulation is done on a second-by-second basis, an activity that would take one hour, would cost only one insert and one lookup on the acitivty queue. In the polling approach it costs 3600 calls to the simulation step callback (even if it does not actually compute anything, this adds a computational overhead) and 3600 checks whether the activity should be ended.

Using DVRP, Bischoff and Maciejewski (2016) ran a study on autonomous vehicles, where the demand in Berlin has been obtained using an ordinary MATSim simulation. Then the link speeds of the underlying network have been modified, to resemble the traffic situation in a congested situation and then *only* autonomous vehicles have been simulated. This means that the simulation could be run once, and the results were obtained because only the QSim was used and not the evolutionary learning of MATSim.

However, in the project of this thesis, autonomous vehicles should be tested in an existing multi-modal scenario, where the evolutionary learning is necessary for the agents to arrive at their quasi-optimal mode choices. So if Bischoff and Maciejewski (2016) mentions computational times of 20h for a large number of agents, it is a measure for one iteration of the evolutionary algorithm. For the scenarios that will be tested here, iterations still do not reach a satisfactory equilibrium, although having a theoretical computation time of 2000h already.

Measurements have been made to determine, how much the simulation of an idle agent, which always stays in one activity, would cost regarding execution time on a test machine[3]. The final results gave an average time of 25ns. So simulating for instance 8000 agents for a common simulation time of 30h, would lead to an execution time of:

$$T_{30h} = \underbrace{25}_{\text{Nomial}} \cdot \underbrace{3600 \cdot 30}_{\text{One day in seconds}} \cdot \underbrace{8000}_{\text{Agents}} = 21.6s \tag{4}$$

For the whole MATSim simulation, this needs to be multiplied by at least 100 iterations, so that the overhead of a simulation of 8000 agents doing nothing would already reach a (highly optimistic) computational overhead of

[3]Intel(R) Core(TM) i7-3612QM CPU @ 2.10GHz

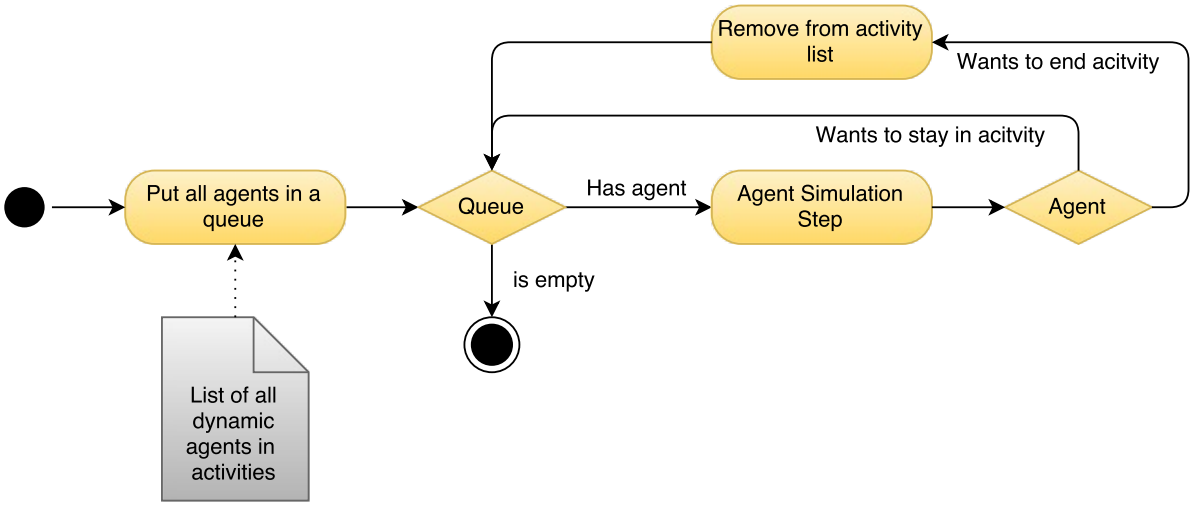

Figure 11: Polling approach of the DVRP framework

$$T_{sim} = 21.6s \cdot 100 = 36min \tag{5}$$

Furthermore, while working with DVRP, it has been found that one needs to put increased efforts into making the framework compatible with parallelized computation in the Netsim, which would have lead to a neglect of a good way to improve the computational performance of the simulation if ignored.

As a summary, one can say, that DVRP allows for a great freedom in simulating dynamic behavior and its structure and signaling flows are very straight-forward and easy to work with. However, for large numbers of iterations, it would be beneficial to find more efficient ways for the simulation.

4.2 The AgentLock framework

As an alternative to the simulation of dynamic agents using DVRP, the AgentLock framework has been developed as part of this thesis. It tries to combine the advantages of a queue simulation while providing as much flexibility as possible to create rich and dynamic agent behaviors in MATSim.

The basic idea is as follows: Usually agents in MATSim do not need much processing power, i.e. the per-agent simulation step of DVRP is hardly ever used and can usually replaced by some callbacks and event handlers outside of the actual agent simulation.

Furthermore, this means that an activity just means that an agent is residing at a certain position in the network and not taking part of the network. So an activity for an agent

is just keeping the agent back from joining the traffic network again after a certain time or event.

With this idea in mind, three different types of ways to "lock" an agent into an activity have been proposed:

Blocking activities will just let the agent reside in this activity until it is released through a call from outside.

Time-based activites are the ones from the basic MATSim simulation: They have a certain duration and therefore a fixed end time.

Event-based activities let the agent reside in the activity until a certain event happens.

The heart of the AgentLock extension is the LockEngine. Whenever it encounters an agent that wants to start a blocking or event-based activity, it is removed from the simulation and only added back manually or as soon as the event occurs. Then it either is passed to the ordinary Netsim or the next activity is started, just as requested by the agent logic.

For time-based activities, a similar approach to the activity queue for ordinary agents is taken. The first idea that would come to one's mind is to order the agents by the end time of their activity and as soon as there is a change in plans, remove the element from the priority queue and add it back at a certain position.

Compared to DVRP, where a change of plans would cost nothing, here this would lead to an overhead of $\mathcal{O}(\log n)$ and $\mathcal{O}(n)$ for these operations (Java API, 2016). So it is necessary to weigh the overhead of the polling in DVRP against the overhead of the rescheduling, which mainly depends on how often such reschedulings appear. In the concrete example of autonomous taxis, this itself depends directly on the travel demand.

If one sacrifices a slightly increased memory consumption for the sake of having a faster computation time, this setup can be improved further, as done for the AgentLock framework. Here, every time a time-based activity is started, a handle to the corresponding agent is saved into the priority queue, ordered by the end time of that activity. Furthermore, an indicator is saved, whether the handle is still valid. So if in the meantime the plan is changed (i.e. before the end of the activity has been processed), this handle will be invalidated, and it will simply be ignored when processing the priority queue.

The whole process (fig. 12) works as follows: In every simulation step, the top element of the priority queue is checked. If the top lock handle is scheduled for the current or a past time step, the associated agent is saved for further processing. If the handle

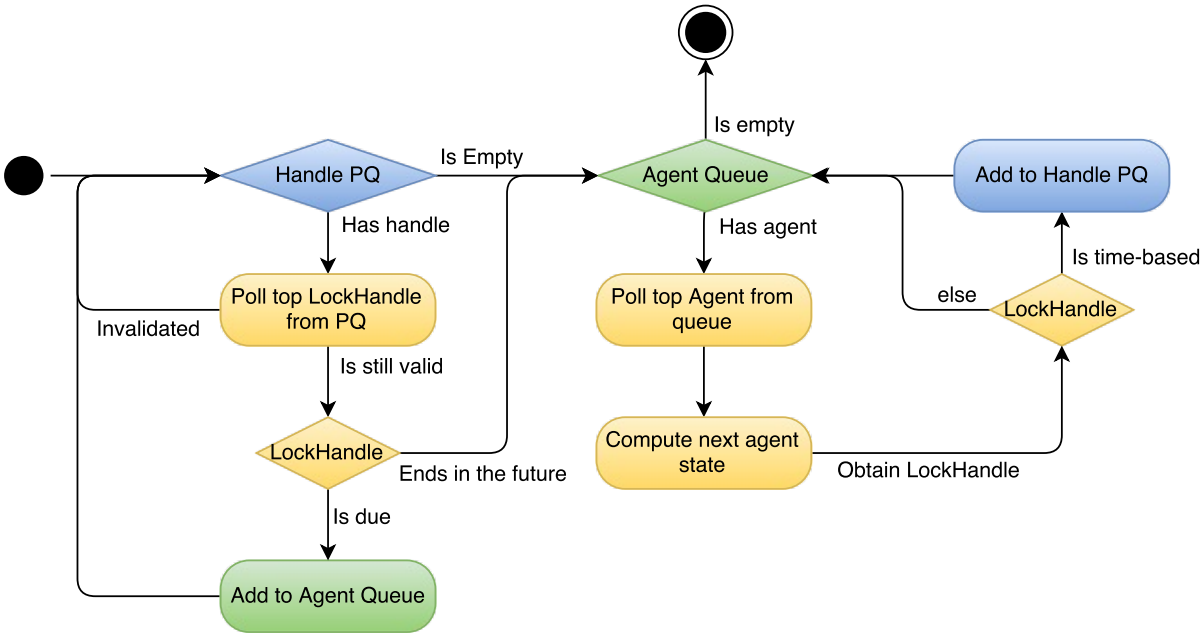

Figure 12: Per-Iteration procedure of the LockEngine in the AgentLock framework

already had been invalidated, the processing continues with the updated top element of the priority queue until a handle is found that has an expiry time which lies in the future. Subsequently, the new state (activity or leg) is computed for each saved agent, and a new lock handle is added to the handle queue if it is time-based.

The main advantage of this setup is that elements are only removed at the top of the queue, which is significantly less expensive than removing elements at arbitrary positions within the queue.

Furthermore, the AgentLock framework provides methods for dynamically ending legs, and it has been made sure that all the functionality has a high degree of compatibility with the existing parts of MATSim, such as the multi-threaded Netsim.

4.3 AgentFSM

While the AgentLock extension mainly provides an abstraction layer for the dynamic rescheduling of activities and legs, another layer has been developed, which is loosely resembling a finite state machine, tailored towards agents in MATSim.

In this framework all the activities and legs are predefined states in a finite state machine and the transitions from one step to another can either be triggered manually (which is the *blocking* lock from above), by time (*time-base lock*) or by an event (*event-based lock*).

The main task of the AgentFSM framework is to encapsulate common programming steps when designing dynamic behavior in MATSim. This means that one usually just has to define which states the behavior is built of and how the transition from one to another works. All of this functionality is provided with a simple programming interface.

In more detail, each state is either a LegState or an ActivityState. For each state, an *enter* callback exists, which the programmer can use to execute arbitrary code. It needs to either return how this state is locked from the three options above or issue a direct switch to another state.

As soon as the state is ended due to the above conditions, the *leave* callback of a state is called, which must return to which next state the simulation should switch.

The concrete implementation for the autonomous vehicles will be discussed in section 5.

4.4 Comparison

An implementation of autonomous vehicles in DVRP has been compared to an implementation using the AgentLock framework. A specific number n of autonomous vehicles are created and put into a queue. As soon as an agent wants to start a leg using an AV, the top element of the queue will drive to that position, pick up the passenger, bring him to the final destination and drop him off. Then the AV will be added back to the end of the queue. This dispatching algorithm is quite inefficient but sufficient to do investigations in terms of computation time for the two implementations since the behavior is equal and easily understandable.

Simulations with $n = 2000, 4000, 8000$ have been done on the Sioux-14 scenario. What can be seen in fig. 13 is the number AV legs in the simulation and the associated computation time. For a large range, the implementation using AgentLock/AgentFSM (solid) is much faster than the corresponding DVRP implementation (dashed). In fact, for $n = 8000$ only at a (quite unrealistically big) AV share of around 60%, DVRP starts to be more efficient.

Important to notice is that the performance of DVRP stays roughly the same over the whole range of shares. This is because all 8000 AVs are simulated in every single time-step, no matter if they are active or not. In the AgentLock implementation, however, agents are only simulated if they are really in use. At $110k$ legs, though, when the reschedulings of used AVs are getting too frequent, the repeated queue operations get more expensive than the polling and therefore DVRP becomes more efficient.

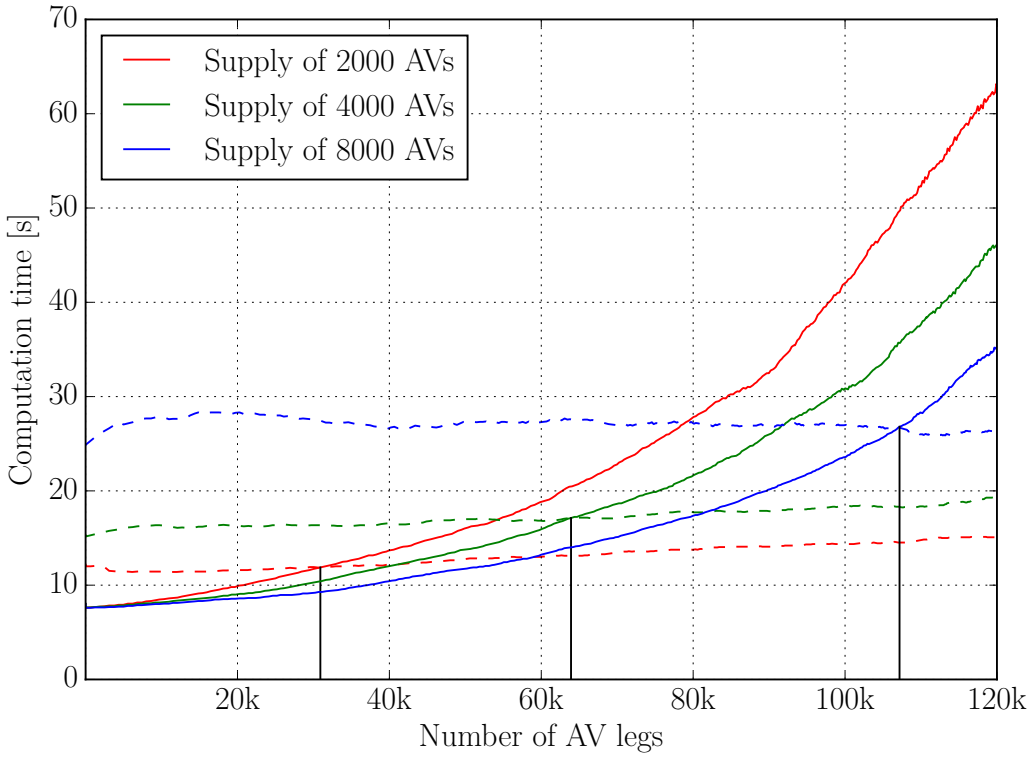

Figure 13: Comparison of Mobsim runtimes between DVRP (dashed) and AgentLock/AgentFSM (solid) implementation, dependent on the number of AV legs. The scenario is Sioux-14 with a total number of legs of around 180k.

As a result, one can say that the AgentLock implementation is usually more efficient than the DVRP version if there is a limited number of reschedulings. If they get too frequent, i.e. the behavior is characterized by many alterations of some initial plan, the polling structure of DVRP gets more efficient. For the purpose of simulating autonomous taxis in this thesis, the number of such reschedulings should be minimal, as it will be described in section 5. Therefore, the development of the AgentLock framework indeed bears a computational advantage for this thesis.

Regarding a multithreaded Mobsim, only tests with the new AgentLock implementation could be done, as shown in fig. 14. Obviously, though the improvement is small, the simulation with two threads performed best. This shows that it is an advantage to be able to use the multithreaded Mobsim. Similar results have been found for other scenarios. For instance, the optimal number of threads for the Singapore scenario has been found to be four (Erath et al., 2012).

In conclusion, while the AgentLock framework is suited for the investigations in this thesis, a hybrid framework, offering both approaches, would be highly beneficial. Even more, it

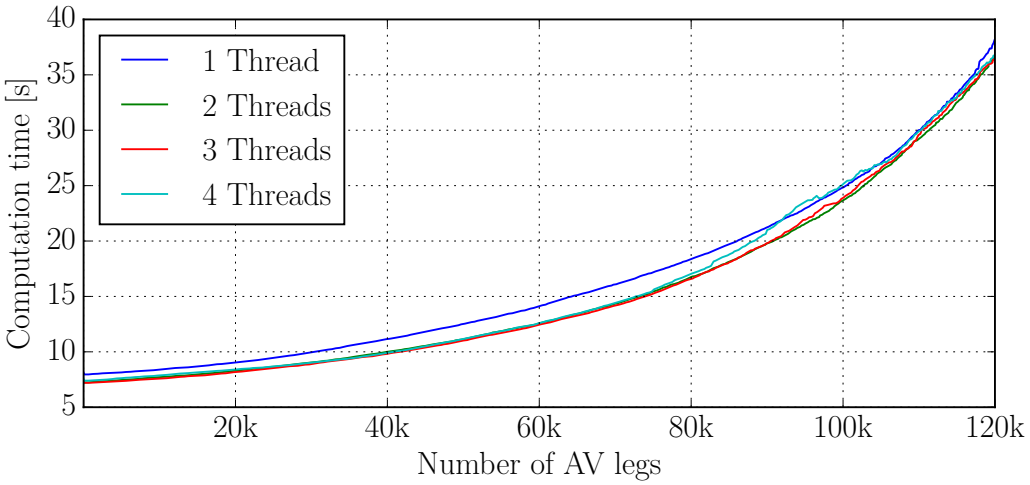

Figure 14: Comparison of different numbers of threads for the Mobsim with the AgentLock implementation

would be interesting to investigate adaptive algorithms, which could switch intelligently between the processing modes, dependent on the actual computation time and number of reschedulings in a specific simulation.

5 Autonomous Taxi Fleet Model

Based on the previously presented layers for the simulation of dynamic agents in MATSim, a model for the specific case of autonomous vehicle taxi agents is developed in the following sections. First, the general behavior is defined, based on a logical chain of actions, which an AV agent needs to follow to serve a customer. Furthermore, the questions of how to distribute AVs at the beginning of the simulation and how to assign AVs to pending requests in an efficient manner are answered. Finally, algorithms for routing AVs through the street network are presented and evaluated.

5.1 Agent behavior

The behavior of an autonomous taxi is not a lot different from an ordinary one, except that there is no need of random roaming to find new customers. Therefore, the AV behavior presented here is mainly inspired by the model in Maciejewski and Bischoff (2015), where ordinary taxi services have been simulated.

The behavior of an autonomous taxi agent can be described in terms of a task life cycle. When the agent is not on a task, he is in idle mode, basically (for the basic model) meaning that he will just stay at the current position and wait for further instructions. The following steps in the life cycle are depicted in fig. 15.

1. **Pickup Drive** is the phase where the AV has got a task and is moving to the requested pick up location. If the AV is already at the right location, this point can be skipped. Otherwise, it represents a leg driving from the current position to the pickup location.

2. **Waiting** is the phase in which the AV has arrived at the pickup location, but the passenger is not there yet. This can only happen if passengers request cars in advance, prior to the time when they want to be picked up. If the passenger is already present at the time of the arrival at the pickup location, this state can be skipped.

3. **Pickup** is the state in which the passenger is picked up. It is modeled as a fixed time, e.g. one minute and starts as soon as both the AV and the passengers are present at the pickup location.

4. **Dropoff Drive** represents the leg going from the pickup location to the dropoff point.

5. **Dropoff** is the point where the passenger leaves the vehicle. Again, this is modeled as a fixed time interaction.

6. **Idle** is the final (and initial) state of the AV life cycle. At this point, the AV will just wait until it receives a new task to pick up another passenger.

The states described above fit very well to the distinction of Activities and Legs in MATSim as well as to the structure of the AgentFSM framework, which has been developed for this purpose. Resulting from this behavioral model, a couple of parameters end implementational questions arise, which must be configured accordingly:

The Idle behavior for the basic model just means that the AV will stay at its last position. Future extensions could make use of parking facilities or do an intelligent repositioning to improve the overall performance of the service.

Pickup and Dropoff duration need to be specified. In accordance with Bischoff and Maciejewski (2016), $t_{pickup} = 120s$ and $t_{dropoff} = 60s$ have been chosen. The time used for the pickup action will not be counted as waiting time (which, in turn, would be penalized through the marginal utility of waiting as described in section 2.2).

5.2 Distribution Algorithm

At the beginning of a day simulation in MATSim, all the n available AVs must be placed somewhere in the network. Two simple distribution algorithms have been implemented for the purpose of testing in this thesis.

The first one is **Random Distribution**. Here n links are chosen randomly from all available network links. While this is an easy distribution strategy for testing, it creates a quite unrealistic scenario. Obviously, in a real AV service, vehicles would be relocated overnight, such that they can serve as many passengers as possible during the morning peek.

Therefore, a more elaborate distribution strategy has been implemented, which should lead to more realistic results. One can define a density over the network, such that every link has a certain probability to get an AV assigned. Then, in n steps, the n available AVs are added to a link dependent on the assignment probability. This probability density can be based on many factors. The ones that are implemented for this thesis are:

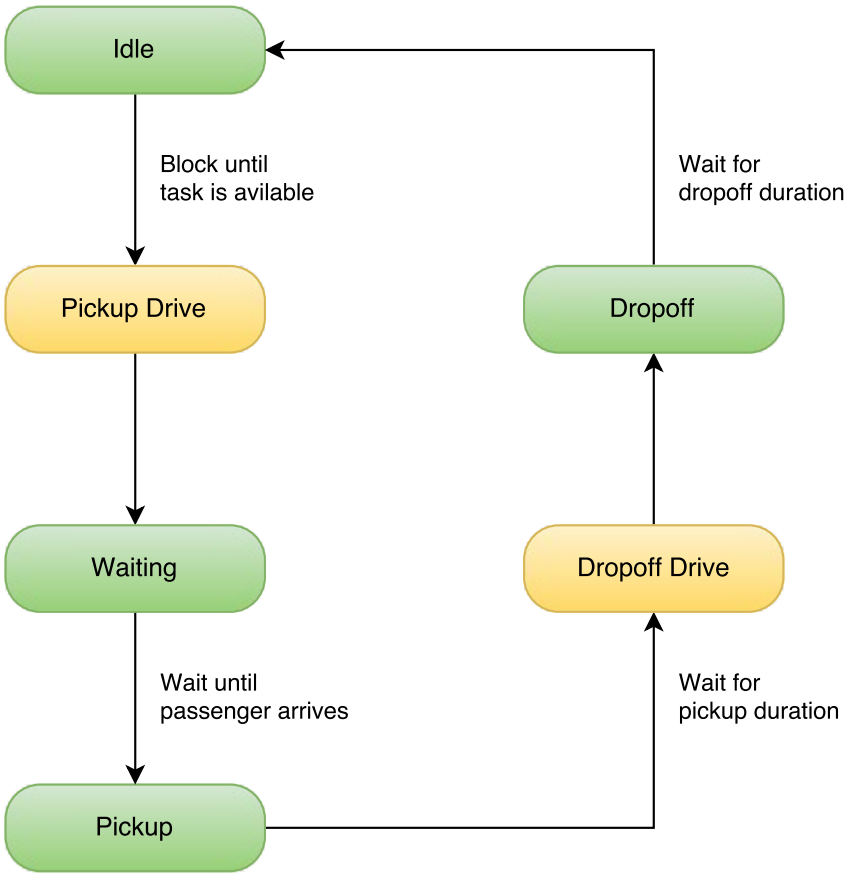

Figure 15: State chart of an AV agent. Activity states are displayed in green while Leg states are yellow.

Population Density The population density is measured in terms of how many people are having their "home" activity on a certain link.

AV Density For each link it is counted how many agents have chosen the AV mode for their first leg (the one that is starting from home).

While the prior one is static, the latter on is dynamic in the sense that for every iteration in the evolutionary learning, the density will change. So while the population-density based distribution is a fixed constraint, the latter one captures the factor of availability. For instance, if there are many agents using AVs in a certain area, the availability is very high, and thus, more people might be inclined to use it. On the other hand, if availability is low in an area, it might lead to longer waiting times and people are less likely to use AVs.

While it might be interesting to observe different emerging patterns of availability, the population density is better suited for the scope of this thesis. A detailed investigation would need to be done if any results from the AV leg density come from the mode choice of the agents or of feedback behavior within the algorithm itself.

5.3 Dispatcher Algorithm

The dynamic dispatching of taxi vehicles is quite a complex scheduling problem, which is quite hard to solve or needs numerous assumptions to be feasible. In general, the problem will be NP-hard, and heuristical solution strategies need to be applied if fast solutions are needed. An overview, as well as a proposed heuristic algorithm, can be found in Maciejewski and Bischoff (2015); Bischoff and Maciejewski (2016).

The algorithm there can be described in two different states:

Oversupply occurs when there are more available autonomous vehicles than requests. This means that each request can be assigned without delay. In that case, as soon as a request arrives at the dispatcher, the closest vehicle to this request is searched and assigned to serve the customer.

Undersupply is the case when all autonomous vehicles are occupied. In that case requests will stack up, which cannot be handled immediately. In this case the algorithm works the other way round: As soon as an autonomous vehicle gets available, it is dispatched to the closest request.

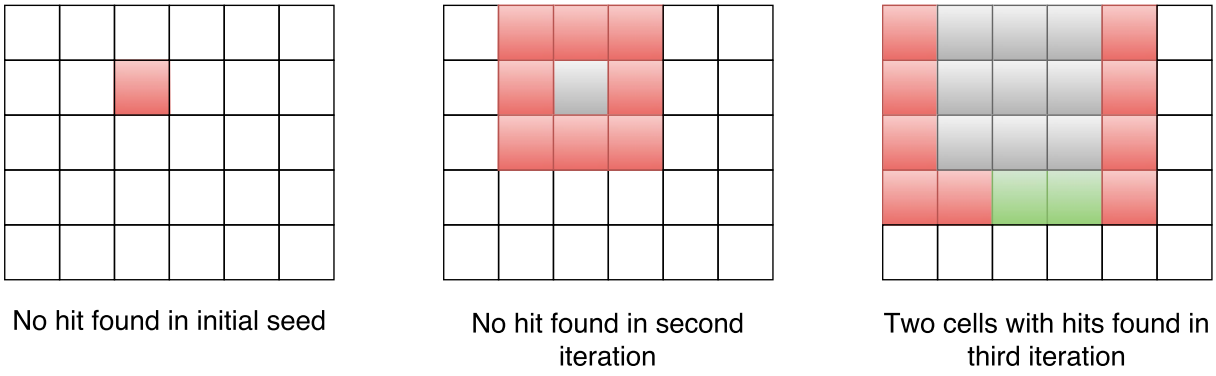

Figure 16: Schematic grid search algorithm for the dispatcher

According to the beforementioned papers this strategy gives a near-optimal solution, although providing fast computational times.

The implementation for this thesis is based on a spatial relaxation of the traffic network. This means that a grid with a specific resolution in x and y is fitted over the links. One of these grids saves the locations of all the available AVs while another grid saves the locations of all open requests. Because the grids have a fixed structure, finding the closest AV or request is quite simple, since each position in x and y belongs to one specific cell of the grid. If no option is found in a certain cell, the search continues with all cells in its Moore neighborhood (all eight surrounding cells). If this still gives no result, the radius is increased and so forth. As soon as one or more candidates are found in a cell, a random one is selected as the result of the search algorithm. This means that no further weighing of the candidates is performed, which would lead to increased computation cost. The procedure is depicted in fig. 16.

This search algorithm imposes new parameters to the implementation, which are the cell counts in horizontal and vertical direction. Depending on the topology of the network, different parameters might be more efficient. If the grid is chosen to be too dense, many iterations are needed in the algorithm, while a too sparse one in the extreme case can lead to finding most of the items in only one single cell, effectively transforming the to a simple random selection.

In fact, for some topologies, it might probably be more efficient to use different data structures like a binary tree or quadtree to improve the search procedure. Also, more elaborate network search algorithms could be used, which make direct use of the topology.

For the Sioux-14 scenario, it has been tested which impact the choice of the grid size has on the results. In fig. 17 one can see, that very high-resolution grids ($n = 200$) lead to an increase in computational time, due to the necessary “expansion” of unoccupied cells,

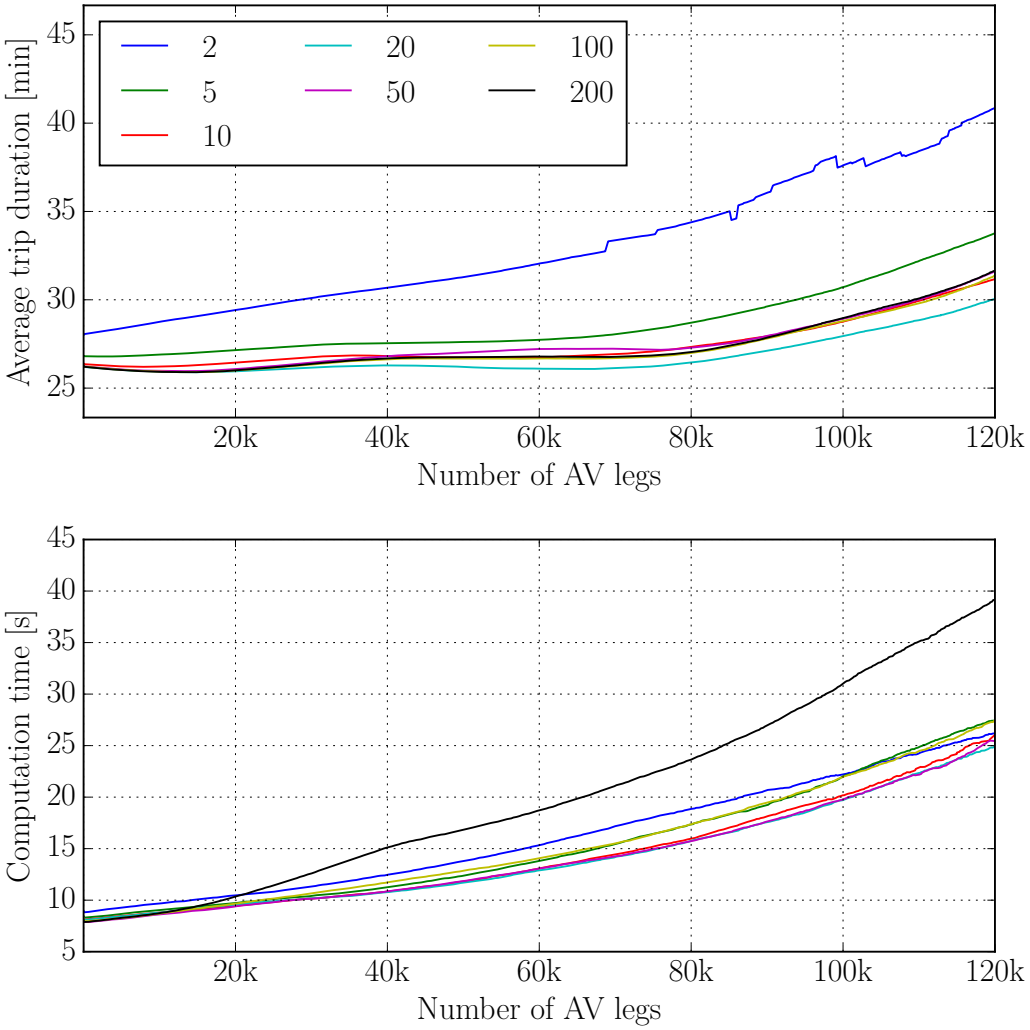

Figure 17: Performance of different grid sizes in the dispatching algorithm for Sioux Falls

as described above. On the contrary, very low grid sizes lead to poor results in terms of the overall average travel time of the agents, because selecting a random agent from one of a few cells is just a random assignment. A good value for the Sioux Falls network has been found to be $n = 20$, which is computed fast over the whole range of AV legs and furthermore does a good job in decreasing the overall travel time.

Once one open AV request is assigned to one idle AV using these two grids, the trip is computed, which mainly consists of finding a route for the AV to move to the pickup location and finding a route from there to the dropoff point. The actual implementation of this routing process will be described later (section 5.4).

While in the first version of the dispatcher the routing was performed sequentially after all trips had been assigned, the dispatching process could be improved by parallelizing

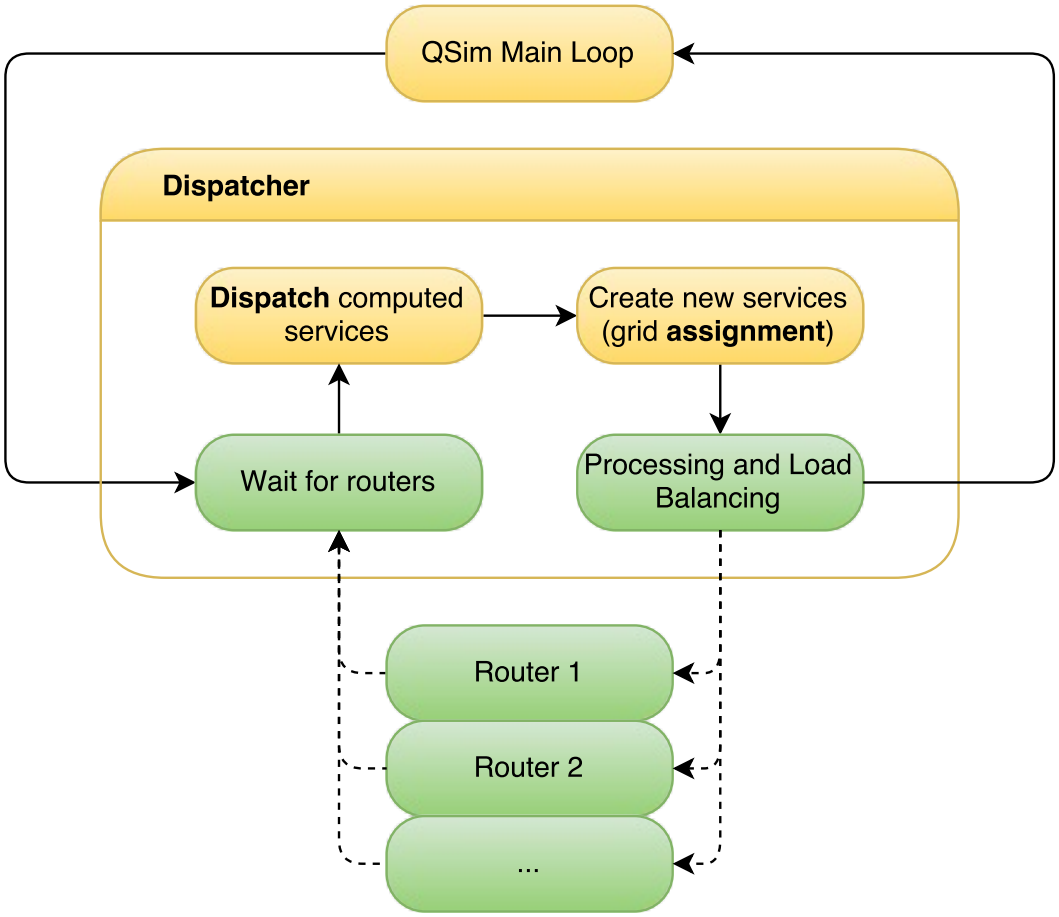

Figure 18: Dispatcher implementation with routing parallel to the MATSim loop

the routing stage with the whole QSim cycle as shown in fig. 18. At one point in the dispatcher, the assignments between requests and AVs are made, just as before. Then, though, the actual task of finding the routes is delegated to an arbitrary number of parallel routers. Each of those routers has a queue which can be used to spread the routing tasks over all routers (i.e. if there are two routers and six tasks, each one will process three tasks, which are added to the respective queues). After all tasks have been sent to the routers, the MATSim simulation continues as usual; the "work-in-progress" services are saved temporarily in the dispatcher. After one whole QSim loop (i.e. simulating the traffic network, static activities, ...) the dispatcher is called again. Then it waits for all the routers to finally dispatch the requests that have been assigned in the previous simulation step. Subsequently, the assignment step for the current simulation step is performed.

Figure 19 shows the impact of adding this feature: Clearly, adding the parallel functionality accounts for a significant decrease in computation time. However, using more than one parallel router does not further increase the performance. This is a strong indicator

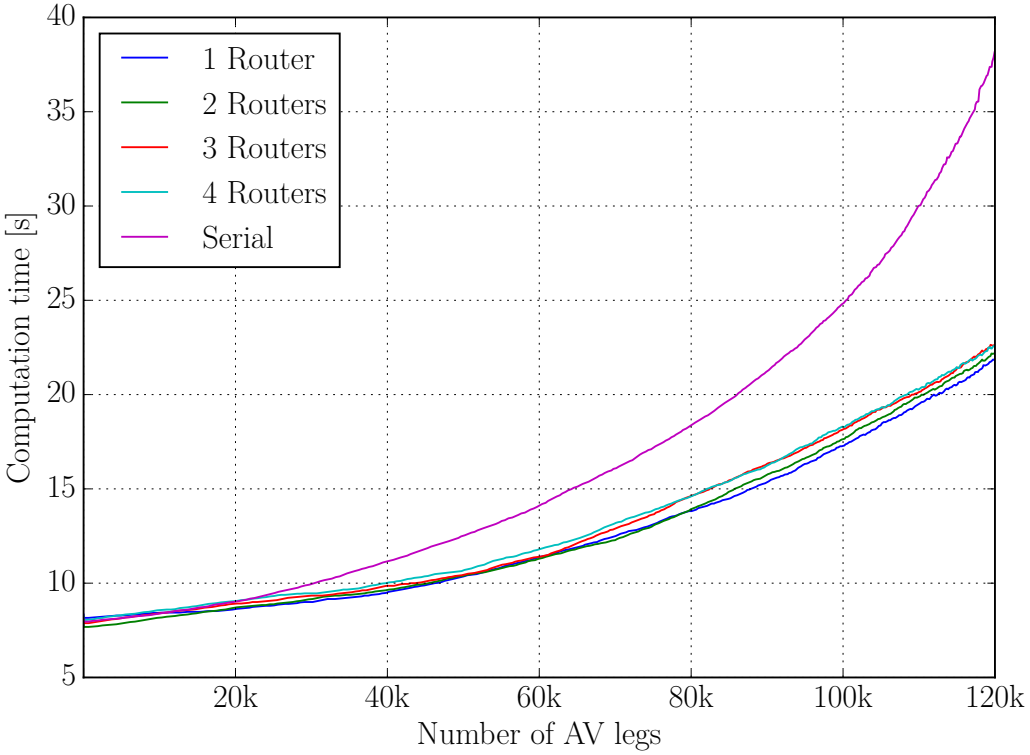

Figure 19: Computational performance of the dispatcher with sequential and parallel routing

for the assignment step (together with the QSim) for being the restricting time factor here. The routing of all tasks takes less time than the other processes, even when only one router is present, and thus no further gain in computation time can be made. Looking at the end of the graph, one can see though that the offset in computation time that is accounted to the routing is increasing faster than the computation time of the rest of the simulation. Therefore, with a higher count of AV legs, there should appear the point where two routers are more efficient than one.

In theory, the parallelization of the dispatcher could be pushed even further. However, the assignment step, which is the other expensive component in the algorithm, would probably be hard to parallelize because concurrent workers would need to access the same grid structures. The overhead of managing which worker has access is likely to add more overhead than improvement to the simulation quickly. One could, however, put the whole assignment process as a serial procedure in parallel to the QSim, as done with the routers. This could be a promising approach to decreasing the computation time even further.

5.4 Routing Algorithm

The routers in the AV simulation have the task to find a route through the street network. For each AV service, there are two of such routes: The pickup route from the AV's current position to the customer and the drop off route on which the customer is transported to the predefined drop off location. To find the routes, the Dijkstra algorithm is used (Dijkstra, 1959), with travel time as the objective to be minimized. This travel time is computed using the length of the selected links and the speeds on those links.

The most simple idea for those speeds, however, would be to use the predefined free-flow speed values in the scenario network. While this approach works for small shares of AVs, it leads to heavy overcongestion with high numbers of AVs on the streets. The reason for this is that for similar trips, similar routes are chosen, leading AVs to make extensive use of the main arterials, but also of specific low-capacity links, which are optimal for topological reasons. By not taking into account any information on how congested the links are, more and more vehicles attempt to enter them, creating huge traffic jams.

In the top-level MATSim loop, such congestion is mainly avoided by introducing small stochastic and congestion-based alterations to the car-using agents. This way, iteration by iteration, such routes are eliminated from their plans because they lead to a bad scoring of the respective plans.

In the following sections, two more advanced approaches for defining the link speeds are presented. First, continuous online tracking of link speeds during the day is discussed in section 5.4.1, followed by an approach based on the introduction of stochasticity into the process (section 5.4.2).

The problem of routing *all* vehicles in a network (which is in principal the case for very high shares of the AV mode), is a complex problem in itself. In this perspective, the presented routing algorithms are versions of a local optimization of the route for each agent and trip. However, this does in no way guarantee that a global optimum is reached (depending on what the objective is). In that regard global routing algorithms and heuristics might be worth to study in the future.

5.4.1 Online Speed Calculation

One approach that has been tested for the routing is to base the link speeds on actual online measurements in the network. The traffic situation on a link can be described by the three interconnected measures flow, density and speed. Flow describes the throughput

of the link, i.e. at the start or the end of the link it is measured how many vehicles cross that link in a certain time interval while the density describes how many vehicles are on the link at a certain time.

Each link in MATSim has a certain flow capacity q_c in $\left[\frac{veh}{h}\right]$ defined, stating the maximum number of cars per hour, that are allowed to pass the link. Furthermore, for each link it can be counted how many cars are on the street by counting the number of cars going in n_{in} and out n_{out}, whenever either event happens. This allows one to computed the density on the link:

$$d = \frac{\Delta n}{l} = \frac{n_{in} - n_{out}}{l} \tag{6}$$

with l being the length $[km]$ of the respective link. These values can be combined to arrive at the current link speed by computing:

$$v = \frac{q_c}{d} = \frac{q_c \cdot l}{\Delta n} \tag{7}$$

The result is a velocity in $\left[\frac{km}{h}\right]$. It states the maximum possible speed when a certain density d appears on a link with the maximum flow capacity of q_c.

Of course, computing this value might lead to speeds, which exceed the predefined maximum free-flow speed of the link, so the final value is bounded from above. Also, for the proper working of the Dijkstra algorithm, a minimum speed is defined to $1\frac{km}{h}$:

$$v' = \begin{cases} 1\frac{km}{h} & \text{if } v < 1\frac{km}{h} \\ v & \text{else} \\ v_f & \text{if } v > v_f \end{cases} \tag{8}$$

In order to avoid the same computation whenever a link should be traversed, the network is computed collectively after a fixed time span, e.g. every $30s$ in simulation time.

5.4.2 Stochastic Variation Model

The second idea is to add stochasticity to the Dijkstra process, thus creating slightly perturbed routes. What needs to be answered for using this approach is, which magnitude those perturbations should have. To solve this problem, the Sioux-16 baseline scenario

has been examined. The 30h day has been divided into $N = 720$ bins ($2min$ each) and for each the relative decrease in speed for all link passages starting has been computed:

$$r_{i,j} = \frac{1}{n_i} \sum_{j=1}^{n_i} \frac{v_{i,j,free-flow} - v_{i,j,simulation}}{v_{i,j,free-flow}} \tag{9}$$

Here n_i is the number of link passages (all agents and all links) that occurred in bin number i and $r_{i,j}$ is the relative decrease of speed in each bin for each passage.

As can be seen in fig. 20, the distribution of decreases in travel speeds of the network links can roughly be approximated using a Gamma distribution. The outliers, which can be seen on the right, are those cases where there is a much congestion during peak hours. Because those are exactly the cases that one wants to avoid here, they are omitted in the following steps. The assumption of a Gamma distribution is wholly motivated by the shape of the histogram, performing an educated derivation of the distribution shape would be an interesting investigation, which is out of the scope of this thesis.

A Gamma distribution is defined by a shape parameter k and a scale parameter θ. For all bins in the baseline scenario, the respective parameters have been obtained using the MLE estimator for θ and a Newton approximation for k as described in Minka (2002). Figure 21 shows the parameter values for each bin on top, after they have been smoothed using a moving average filter of length 20. Also, for the times before roughly 5 am and after 10 pm, average parameter values over the off-peak hours in the middle of the day have been inserted due to a lack of sufficient data for fitting the speed decreases in these outer areas.

Also shown in fig. 21 (on the bottom) is the mean of the speed decrease at each time of the day, framed by the 10% and 90% quantiles, now instead of simulation results based on the statistical model. It can be seen that due to the Gamma model, the range of probable decreases is high during peak hours, up to 70%, while the 10% stays quite constant during the day. The graph of the mean suggests that the presented model is a reasonable approximation, which qualitatively makes sense.

The final travel speed for the router can now be described as a random variable:

$$V_{i,j,routing} = v_{i,j,free-flow} \cdot (1 - \min\{R_i, 1\}) \tag{10}$$

with $R_i \sim \Gamma(k_i, \theta_i)$ being Gamma-distributed for each bin.

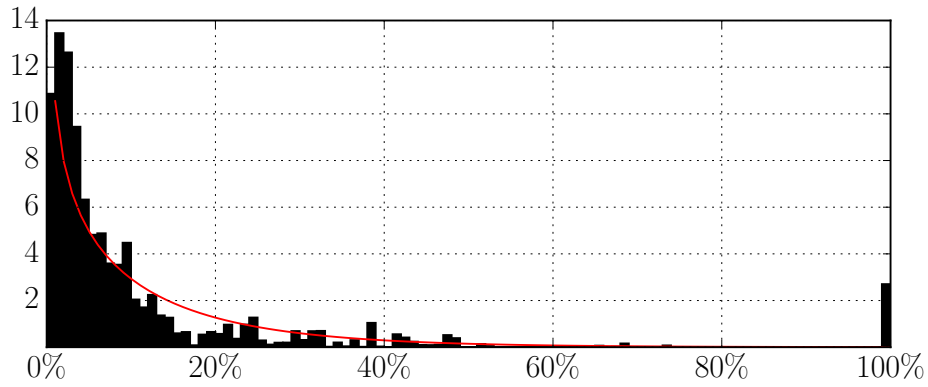

Figure 20: Example distribution (pdf) of relative decreases in link speed

The overall effect of this model is then as follows: There is a certain increase in travel time expected for each link, which depends on its free-flow speed and the time of the day. However, especially during peak hours, when the bottlenecks would be a critical problem, more variation is added to the expected increases in travel time and thus a better variation in the routing is reached. In each case, the expected increases are founded on the actual network characteristics of the Sioux-16 network.

5.4.3 Performance Measures

Basic tests of the AV taxi fleet model without the evolutionary replanning have been performed to test the performance of the routing algorithms. For that purpose, the scenario has been simulated for one iteration, without any further refining of the agent plans. First, the plans of 30% of the agents using a car in the population have been changed to use AVs. Figure 22 shows how the average travel time in the network changes with an increasing number of AVs. If only a few AVs are available, the waiting time for an AV is very high, because there are not enough vehicles to cover the demand. With an increasing number of available AVs the congestion in the network increases and therefore the average travel time (of cars) in the network increases. Clearly, the online speed tracking performs better in preventing congestion while the Gamma distribution approach shows a similar performance to a simple routing based on the free-flow speed.

Another case is presented in fig. 23. There the number of available AVs has been fixed to 2000, and different shares of AV trips in the population have been examined. In that dimension that waiting time increases with a higher share of the population, since more demand is generated, which the taxi service is less and less likely to cover the more agents are using it. Interesting is the decrease in overall travel time, which occurs because at

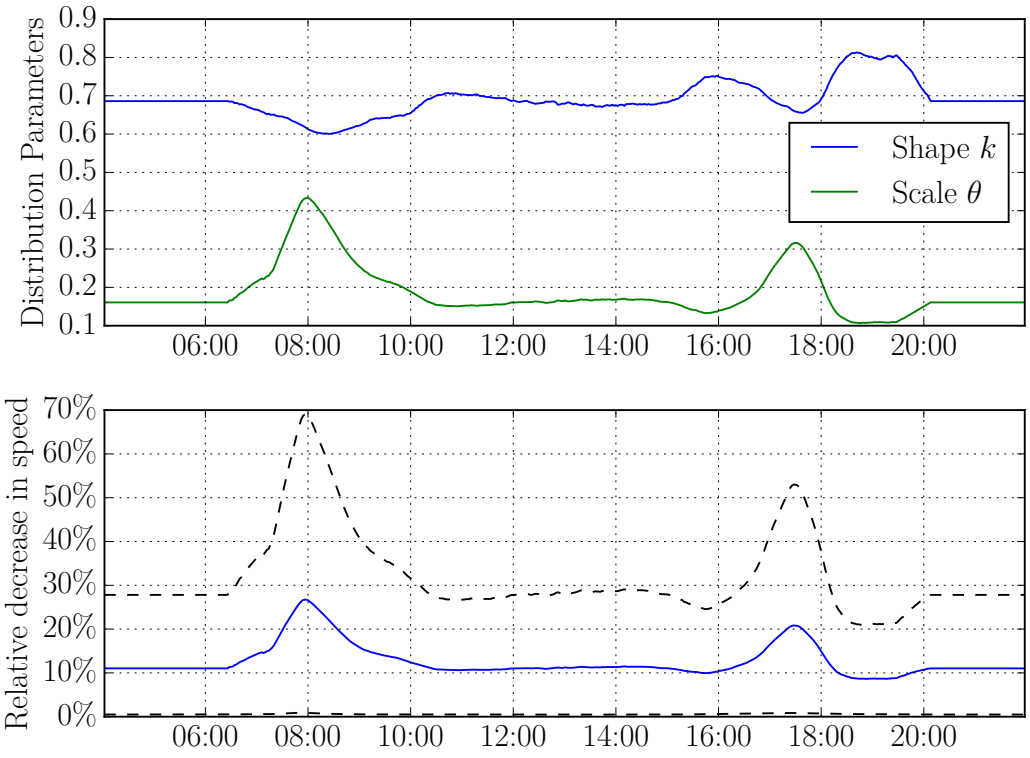

Figure 21: Statistic model for the relative decrease in link speed. Top: Parameters for the Gamma distribution model by day time. Bottom: Mean value of the respective Gamma distribution and the 10% and 90% quantiles.

some point the number of waiting people exceeds the number of people on the road, which is freeing the network from congestion, while letting people wait for a much longer time.

Finally, while the former cases were "relaxed" in the way that although leading to high waiting times, the demand could be covered in the 30h simulation day, fig. 24 shows the case were this is not the case anymore. In that figure, 70% of all car trips have been replaced by AV trips. The solid lines show the average travel time of cars in the network for cases in which all agents could finish their plans; the dotted lines indicate cases where agents were not able to do so. In the beginning of the graph, for a very low number of AVs, all algorithms break down in this regard. This is due to the disability to serve the demand with that number of vehicles.

For the online calculation approach, one can see, that it is performing badly throughout the whole range of values. One explanation for that is that AVs react to any slowdown in the network, effectively trying to find alternative routes to prevent them. This leads the AVs to find unnecessarily long trips, which take irrational routes through the network. What is missing in this routing approach is a sense of expected congestion. In a network, where the capacity of the roads is not able to allow for a free flow at all times, the routing algorithm should be able to bargain between joining a traffic jam and expecting it to dissolve soon or choosing a completely different route through low-capacity links. In this approach, the latter will lead to even more severe congestion because those links are used more frequently than would be necessary.

Comparing the free-flow speed router and the stochastic one, the performance is similar. There is a gap in the middle of the graph, showing the area in which too much congestion is produced by the AVs, such that agents get stuck. Before the gap, one has the regime where AVs serve multiple requests since there are not enough AVs available to have one AV for each request. Then, if the number of AVs gets bigger, it happens that because more and more users get their "unique" AV, more cars go on the street, and too much congestion occurs. Finally, if there are even more AVs, the situation relaxes again, because now almost every request has it's own car. There is no need to perform additional pickup trips to the customers since there are enough AVs distributed over the network.

The gaps presented in this scenario are likely to be resolved when the evolutionary re-planning is introduced into the model. Because people can reschedule their departures, the situation can be relaxed. Also in the replanning, agents would choose other means of transport if the use of AVs gets to disadvantageous. However, it has been found that also with the replanning, for some scenarios it occurs that not all agents arrive at the end of the day. As can be seen in fig. 24, the stochastic approach can cover a larger range of

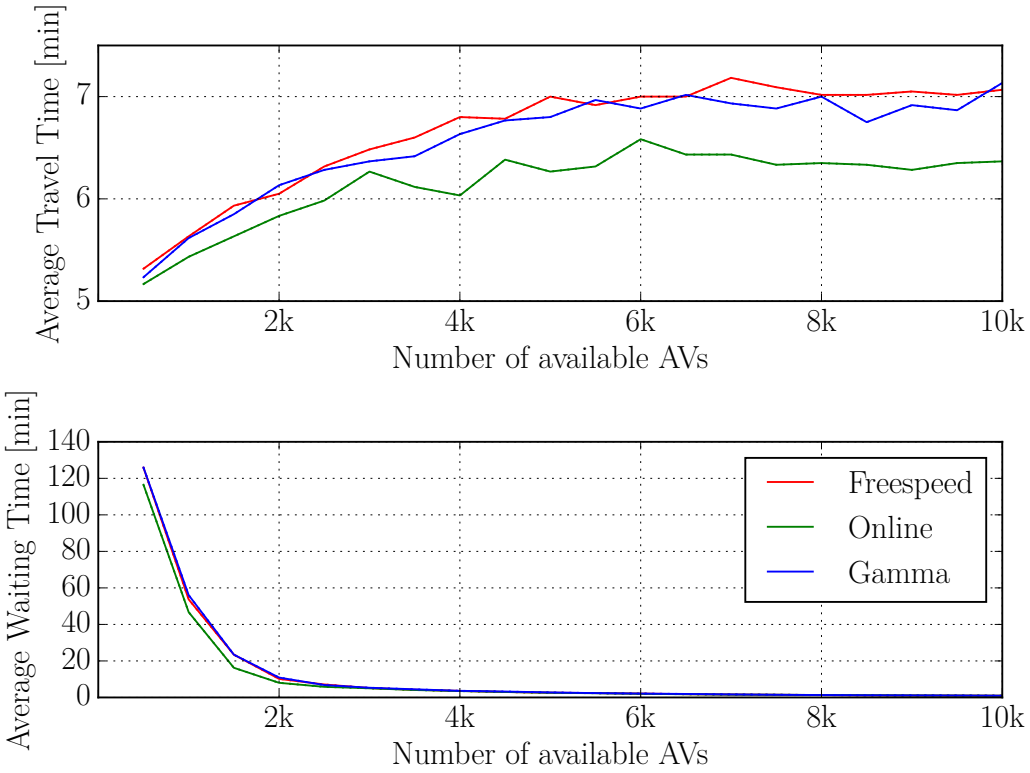

Figure 22: Comparison of different routing algorithms in terms of average travel time and average waiting time with a fixed share of 30% of car users in the population using AVs.

cases, and it also has been found that in combination with the replanning, this approach yields the strongest reliability. Therefore, it is chosen in the following simulations of the whole model.

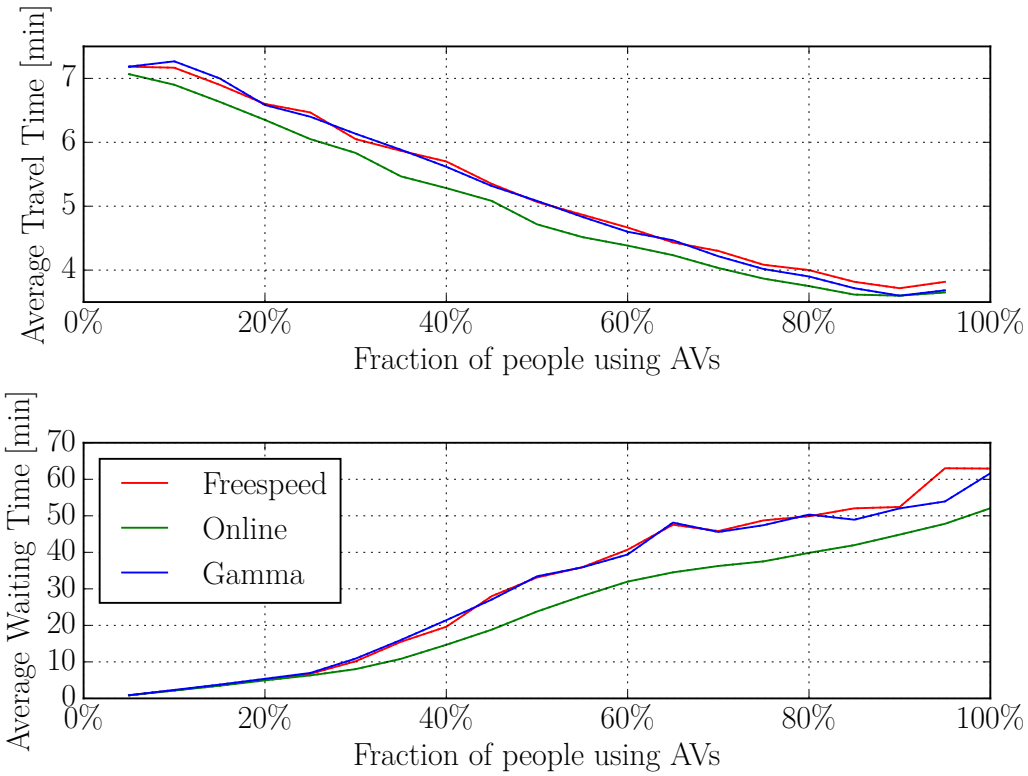

Figure 23: Comparison of different routing algorithms in terms of average travel time and average waiting time with a fixed number of 2000 available AVs.

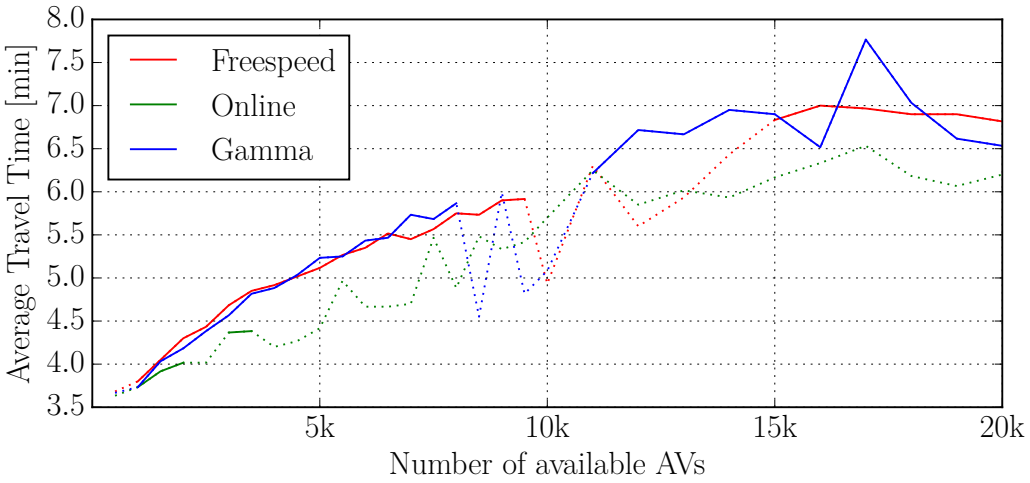

Figure 24: Comparison of different routing algorithms in terms of average travel time with a fixed share of 70% of car users in the population using AVs. Dotted lines show areas where agents were not able to finish their plans during the 30h simulation period.

5.4.4 Static Trip Replacement

From the data for the stochastic routing, it was possible to find a structured dependency of the number of available AVs and the number of replaced cars in terms of the waiting time for passengers without performing any additional measurements. The result is presented in fig. 25. There, the existing data points are depicted as crosses, while all combinations with an average passenger waiting time of less than 10min have been highlighted. A pareto front over the existing measurements has been constructed. So in the case of a bare replacement of car trips by AV trips, one can see that in order to replace 40% of the car fleet, 10,000 AVs are necessary to reach the desired waiting time, which is only 15% of the cars in the baseline scenario. So in total one would be left with a total car and av fleet size of 75%. In the case of a replacement of 70% of private cars with AVs, one would need to convert 30% of vehicles to AVs, which would yield a total fleet size of 60%.

Furthermore it should be mentioned, that taking the average as a waiting time threshold might yield misleading results, i.e. if the distribution is heavily biased towards low waiting times. Experiments with using quantile functions instead of the average showed that in fact the opposite case is observed here. The shown average coincides quite accurately with the 80% quantile, thus yielding an answer to the more solid question: How many AVs are needed to replace a certain percentage of the initial car fleet if 80% of the waiting times should be less than 10 minutes.

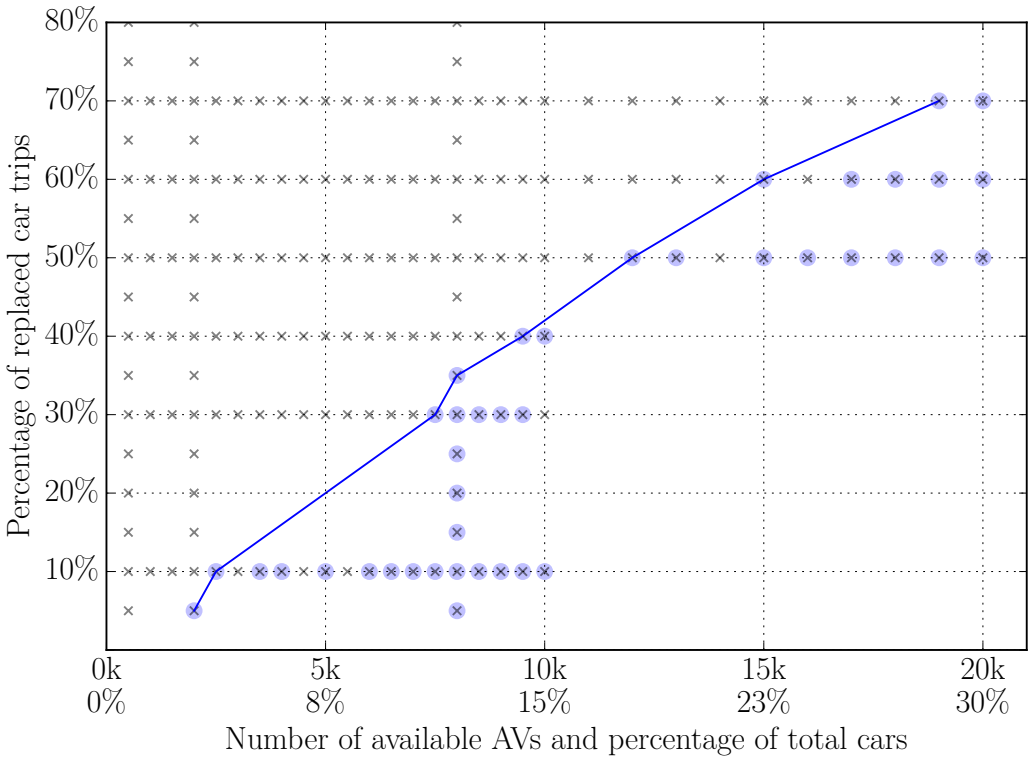

Figure 25: Dependency of the number of available AVs and the number of replaced cars. The crosses indicate distinct measurements; the highlighted ones yield an average passenger waiting time of less than 10 minutes.

These results are a first indicator of the performance of the model, but of course lack the flexibility of the evolutionary replanning. By letting agents alter their departure times the demand situation should become more relaxed and therefore waiting times should decrease. This, in turn would mean that the displayed front would decrease in slope. The next section will cover the results gained from the combined simulation framework, where all components are added together.

Table 2: Baseline Scenario Parameters

Parameter	Computation	Baseline Value
Constant Utility per Trip	$C_{av} = -\beta_m \cdot \$3.60$	-0.2232
Marginal Utility of Travel Time	$\beta_{trav,av} = \beta_{trav,car}$	0.0
Monetary Distance Rate	$\gamma_{av} = 2.19\$/km$	0.00219
Marginal Utility of Wait Time	$\beta_{wait,av} = \beta_{wait,pt}$	-0.18

6 Simulation Results

6.1 Baseline Scenario

The model with the described implementation has been tested on the Sioux-16 network with a flow capacity scaling of 70% to allow for a reasonable amount of congestion.

The travel disutility parameter has been chosen to be zero, as is the travel disutility for taking a car in Sioux-16. The reason behind this is that there are numerous studies indicating positive and negative effects of autonomous vehicles, so it is hard to decide whether the perception of AVs will tend towards one side or the other compared to cars.

The constant disutility for cars in the Sioux-14 scenario has been computed by combining the travel disutility for 10min walking (as to account for getting to and from a parking lot) and the monetary disutility for paying \$6 for parking. For the AVs in this simulation it has been assumed that there is no such additional cost, but a monetary fee per AV trip. This assumes that a fictitious operator already included the costs of parking into the pricing scheme (which is reasonable taking the values from actual taxi services).

Since the values in Sioux-14 are based on measurements in Sydney, the current maximum charges for taxi trips there have been used as a reference (Transport for NSW, 2016). According to this source, an initial charge of \$3.60 has been set as the constant disutility per trip, while the monetary distance factor for AVs has been set to \$2.19 per km.

Finally, the disutility for waiting for an AV has been assumed to be the same as the waiting disutility for public transport from Sioux-14, which itself is just a vague assumption (Chakirov and Fourie, 2014), but at least allows for a systematic comparison.

Using these parameters, which are summarized in table 2, the scenario has been simulated until relaxation. The following paragraphs will show the respective results in terms of the traffic situation. A sufficiently high number of available AVs ($N = 8000$) has been chosen to show how agents make a choice for taking an AV based on their utility evaluation.

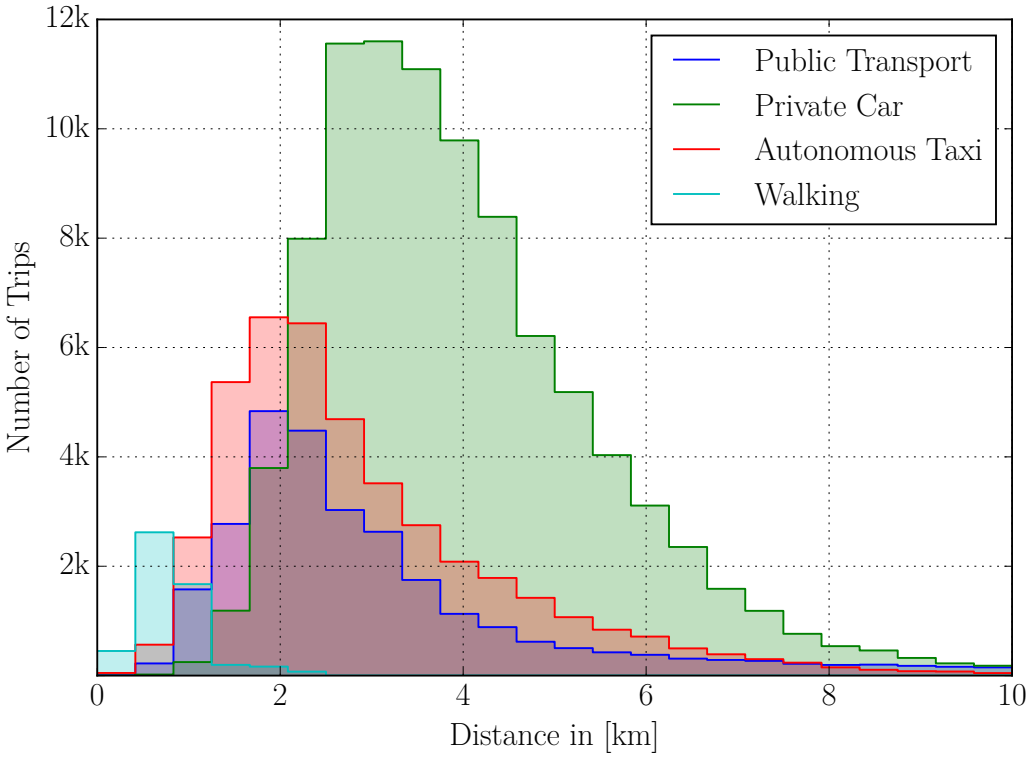

Figure 26: Number of trips for each node by traveled distance

6.1.1 Trip Statistics

Table 3 shows the basic trip statistics of the case where AVs have been introduced to the baseline scenario. It can be seen that with the given utility parameters, the AVs reach a share of around 25% averaged over the day, mainly decreasing the share of public transport and walking while also attracting some of the former private car users.

Interesting to see is that for public transport and walking agents the travel distances decreases because relatively long trips in those modes will be replaced by AVs, thus drawing the average down to shorter trips, while it increases for cars. Here, mainly the shorter car trips are replaced by AVs.

These results can also be in fig. 26, where the distribution of travel distances by mode is presented. Clearly, AVs act as a competitor towards public transport there, serving mainly the same range of trips with the assumed utility parameters.

In terms of travel times, it can be seen that there is a slight increase for the car mode, which stems from the same argument as before. The decrease in travel time for public transport and walking agents is quite significant, though. For the walking agents, the change is obvious as described before. The decrease for public transport can be explained

Table 3: Traffic measures for the relaxed AV baseline scenario

	Baseline	With AV
Travel Distances [km]		
Car	3.73	4.04
Walking	1.25	0.82
Public Transport	3.88	3.25
Autonomous Taxi		2.94
Travel Times [mm:ss]		
Car	07:20	08:07
Walking	25:01	16:29
Public Transport	28:48	20:41
Autonomous Taxi		10:41
Mode Shares		
Car	65.04%	55.08%
Walking	6.79%	3.08%
Public Transport	28.17%	16.65%
Autonomous Taxi		25.19%

by the switch of agents, who needed to have long walking distances to the closest bus stop, which is included in the calculation. By only keeping those agents at using public transport, who live nearby a bus stop, the overall travel time decreases quite substantially.

The waiting times for AVs in the baseline scenario are on average around 04:40 min in the morning peak and 02:55 min in the afternoon, while the daily average lies at 01:40 min. A more detailed analysis of waiting times is given in section 6.2.

In terms of travel distances the total amount of kilometers driven increased from around 424,000 km in the baseline to 553,000 km in the AV scenario. The amount of kilometers driven by AVs is 162,500 km from which around 37,800 are for the purpose of picking up passengers, i.e. they are unoccupied while covering this distance, which is roughly 23%. This is around half compared to ordinary taxis with 52% as stated in recent statistics for Oslo (Geir Martin Pilskog, 2015) or around 50% in Barcelona (Amat et al., 2014).

6.1.2 Mode Choice

Table 4 shows how the mode choice that takes place after AVs have are introduced. The rows show the original modes while the percentages indicate how many of the initial users switch to the mode in the column after the introduction of AVs. What can be seen is that 44% of all initial public transport users and 56% of all walking people opted for taking an AV while only 14% of car users switch modes. This again shows that with the baseline

Table 4: Migration matrix showing which agents switched from one mode to another: The rows resemble the initial choices of the agents while the columns resemble the mode choice after the introduction of AVs. The percentages denoted how many users of the original mode switched to another option.

	AV	Car	PT	Walking
Car	13.69%	84.41%	1.68%	0.23%
Public Transport	44.29%	0.60%	54.91%	0.21%
Walking	56.21%	0.18%	1.38%	42.24%

parameters, AVs rather work as a competitor against public transport while additionally drawing new adopters from the walking people. Therefore, this scenario represents the rather unwanted case where AVs lead to a less optimal situation on the road, leading to more congestion and less use of collective transportation.

A further impression on the choice behavior of the agents can be obtained through fig. 27. In the upper plot one can see the public transport trips in the baseline without AVs (blue), which have not changes during the introduction of the new mode while the red dots show those combinations of trip duration and distance in the original scenario, which have changed to AV. Comparing the blue and red areas it becomes evident that users with rather long trips in terms of travel time switch to AVs. The green dots show the combinations after the change has been taken place, i.e. the travel duration and distance in the converted AV trips. It can be seen that after the introduction of AVs the travel durations get much less, so for the public transport users, the AV mode is mainly attractive because it provides shorter net travel durations.

The lower plot in fig. 27 shows the same arrangement for private car users. One can see that the switching users (red) are clustered for short trips in distance and duration. Their travel times, contrary to the public transport users, increase when using the AVs, indicating that the monetary benefit of using the AV instead of going on a private car trip (and paying for parking) is stronger than the desire to have a minimal travel time.

6.1.3 AV Operation and decreased supply

Finally, fig. 28 shows the states of the AVs during the day. While the lines show how many AVs are currently performing either a pickup or dropoff task, i.e. being “en tour”, the shaded areas show how many passengers have been picked up or dropped off at a certain time of the day. In this baseline scenario, only around 3000 cars are actually

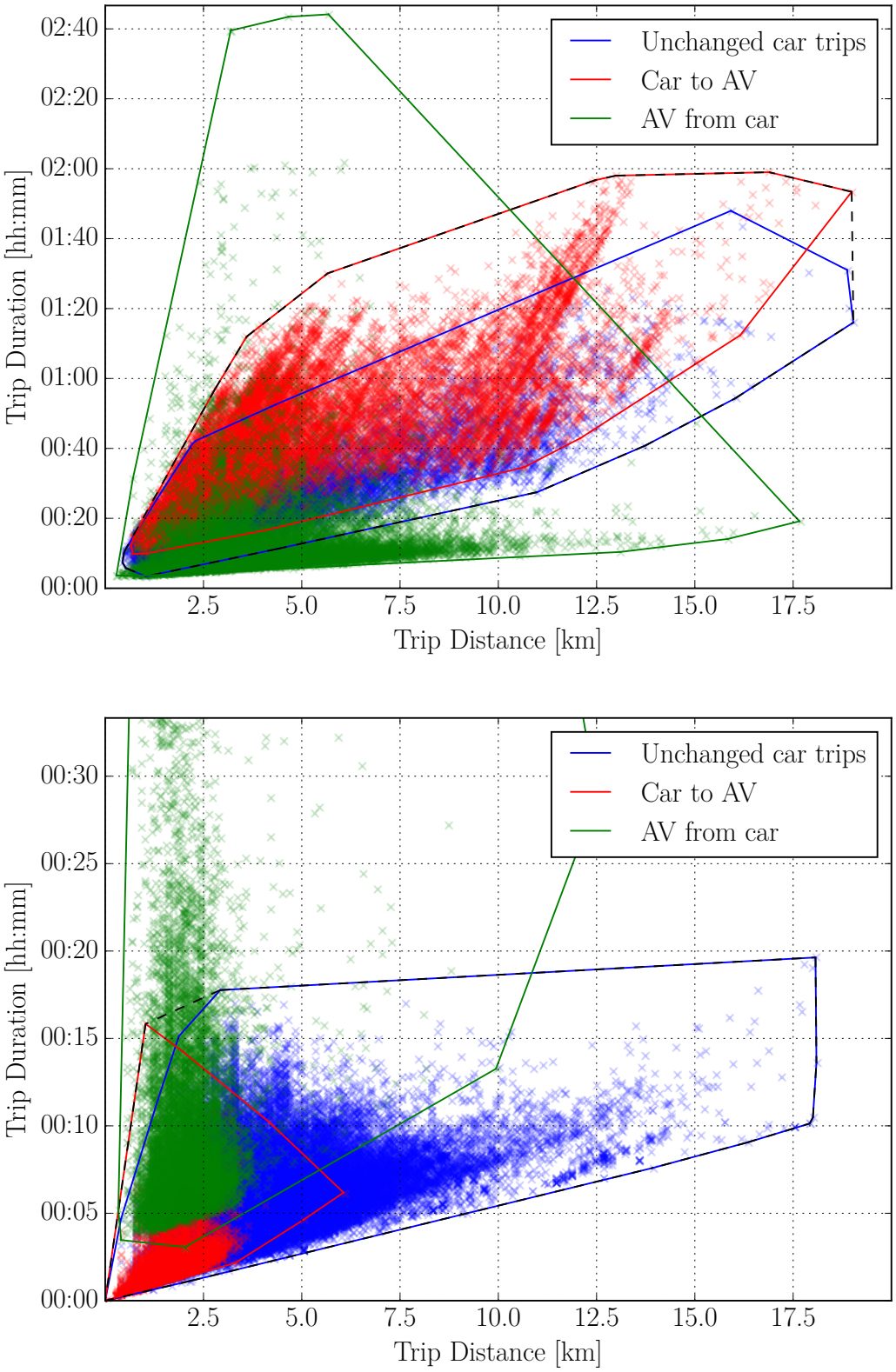

Figure 27: Analysis of the distribution of public transport (top) and car (bottom) trips in terms of travel distance and duration before and after the introduction of autonomous vehicles.

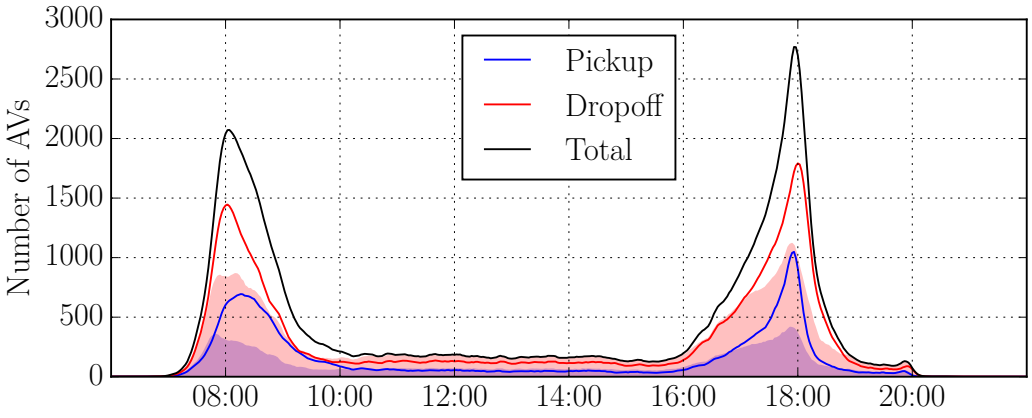

Figure 28: Activities of AVs during the day in the relaxed (8000 AVs) AV baseline scenario. The solid graphs show the number of vehicles, which are "on tour", while the shaded area denotes the number of pickup and dropoff interactions with the passenger.

active of the available 8000, so from the perspective of an AV operator the scenario would not be an ideal case, because the usage of the AVs is not nearly close to saturation.

Such a case is depicted in fig. 29. It shows the states of the AVs during the day if only 1000 of them are available. Shaded areas indicate those times of the day where the dispatchment mode changes from oversupply to undersupply, where there are more requests than available AVs. At those peak times, one can see that the number of active AVs goes into saturation. The number does not go to 1000 exactly since only driving AVs are measured, whereas some might be in the 120s pickup or 60s dropoff activities.

Compared to the high supply case, the share of AV trips drops from 25.19% to 18.92%. While the travel time stays roughly the same for the AV mode, the average distance increases slightly from 2.94km to 3.18km, indicating a shortfall of short trips. Around half the amount of private car users switch to AVs (6.65%, before 13.69%); for public transport users, the decrease is less significant from 44.29% to 40.29%. From that one can conclude that the attractiveness of AVs for private car users is decreasing substantially with a constrained supply while the induced longer waiting times seem to be tolerable by former public transport users.

From an environmental perspective, this scenario is worse than the former one. While car users continue using their private vehicles, public transport users switch to additional cars on the road. In terms of distance (fig. 30), the total amount of kilometers driven increases further because of an increase in excess travel distance of the AVs. Since fewer agents are using the service, the vehicles have to cover longer distances to get to the next

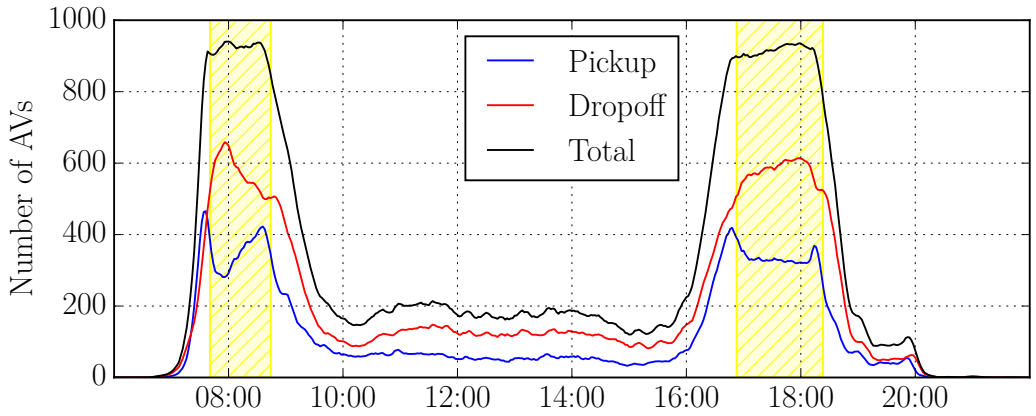

Figure 29: Activities of AVs during the day in the constrained (1000 AVs) AV baseline scenario. The show the number of vehicles, which are "on tour", while the shaded area denotes times where the undersupply dispatchment mode is active.

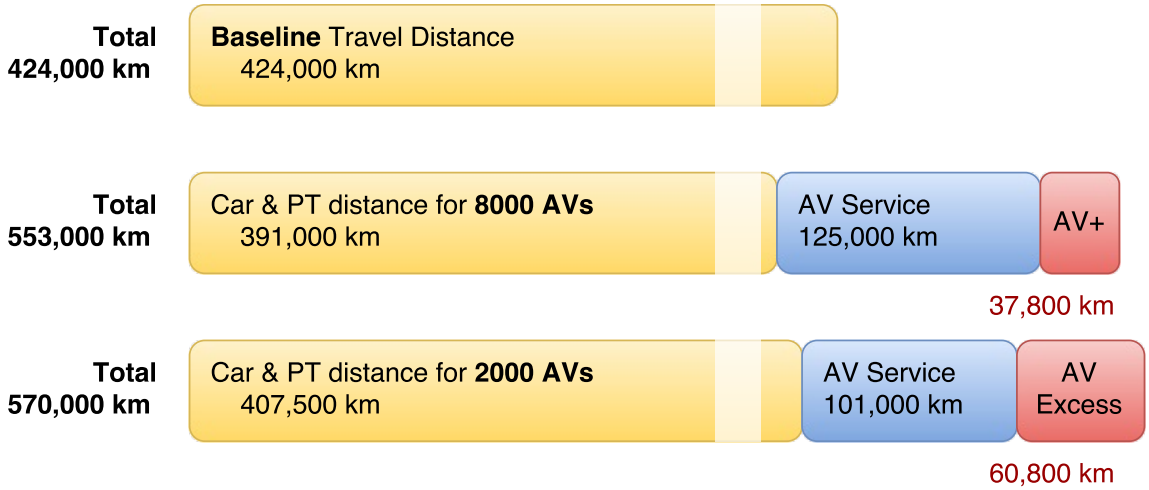

Figure 30: Comparison of total travel distances in the low and high supply scenarios.

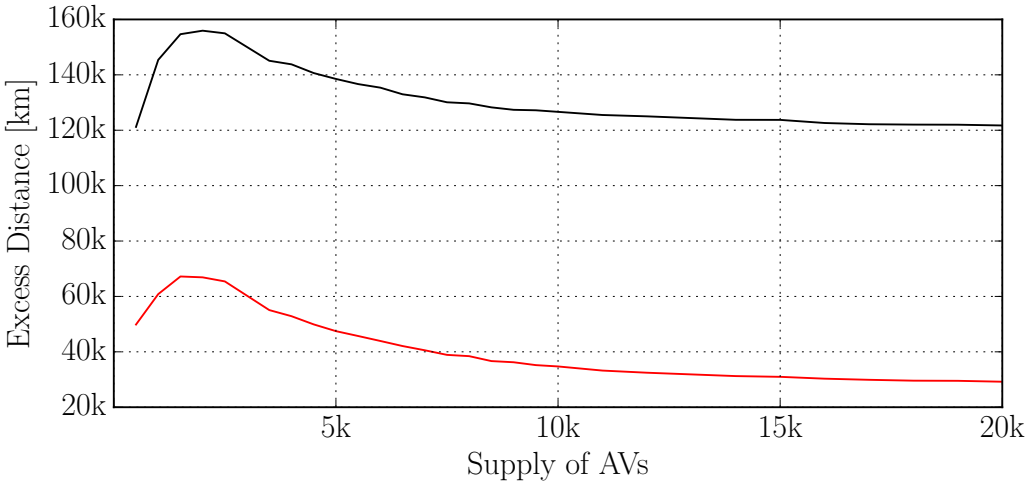

Figure 31: Evaluation of the total added travel distance of all vehicles compared to the base scenario without AVs (black) and excess driving distance for AVs for the respective supply (red).

customer. Such a case is disadvantageous for the service operator, so it will be interesting to see how the additional costs of excess mileage affect the overall economic evaluation of the provider (section 6.4). The next chapter will give a more detailed relation of the supply level on the total traveled distance.

6.2 Supply Analysis

Figure 31 shows the relation of travel distance and supply in a more detailed way. At around 1000 vehicles, there is a peak of the net driven distance in the network (black), which is relaxed if the supply is increased. The stable added number of kilometers is then around 120,000km. However, the peak is only 40,000km bigger than this value, which itself is a quarter of the initial 400,000km in the base scenario. Looking at the red graph, which shows the added miles of empty drives in the AV services compared as an offset to the total number of AV miles, one can see that it shows the same peak, i.e. the excess mileage is responsible for the increase in total travel distance. In this regard, the service operator and public administration would have the same priority to avoid this peak (in terms of profit one one side and regarding environmental policy and congestion on the other).

In terms of waiting time it has been found that per 1000 AVs around 4% of initial car trips in the scenario could be replaced with static demand in section 5.4.4. Looking at the waiting times on top in fig. 32 one can see that the mean value as well as the 90% quantile

of the waiting time t_W is under the threshold of 10 minutes, which has been examined before. Furthermore, the middle of fig. 32 shows $P(t_W \leq 10min)$, i.e. the probability of having a waiting time of less than 10 min in the simulated supply scenarios. While this probability clearly decreases with small fleet sizes, it still stays rather high at 90%. That is the case, because due to high waiting times, fewer trips are being made. For higher supplies, the quantile finds an equilibrium-like state at around 97%, which can be interpreted as a measure of how tolerable increased waiting times are in a certain scenario.

Additionally, the bottom plot in fig. 32 shows the replaced percentage of trips dependent on the amount of available AVs. Because of the preferences that are induced through the utility-based learning, considerably fewer trips are converted to AVs although staying in the waiting time limits. While in the static analysis 5000 AVs can replace 20% of private car trips, in the dynamic one it is only 15%. For the static case 60% are simulated at 15,000 AVs, but here the replacement fraction remains at 15%. This shows that the usefulness of the mode, which is quantified by the utility, is the restricting factor, despite a large available margin in waiting time efficiency.

6.3 Cost Dependencies

Intuitively, the behavior of the utility parameters should be quite clear: If the utility is increased, the AV mode gets more favorable, if it is decreased, less people will use it. However, in such a complex traffic system there are secondary effects, which influence the adaptation of AVs.

Figure 33 shows the share of the AV mode in the baseline scenario (top) with different pricing schemes, given through a price per kilometer and a price per trip. For very cheap services, the share reaches 90%, while for a combination of \$7 per trip and \$3 per kilometer the share drops down to under 10%. It can be seen that for a very low travel utility (lower left) the threshold in the shares gets steeper while it dilutes for very high travel utility (i.e. acceptance) of the AV mode (lower right). This hints at the fact that the more accepted AV technology is in the population, the more people will use it while the pricing scheme can have a huge impact on adaptation if there is a considerable amount of skepticism towards the technology.

The shaded areas in fig. 33 show those parameter combinations, where $P(t_w \leq 10min) \geq 0.9$, i.e. where the probability to wait less than 10 minutes for an AV per trip is higher than 90%. Taking that as another criterion to assess the performance of an AV service, one can see that it puts a restriction on how low the prices can drop to allow for a smooth

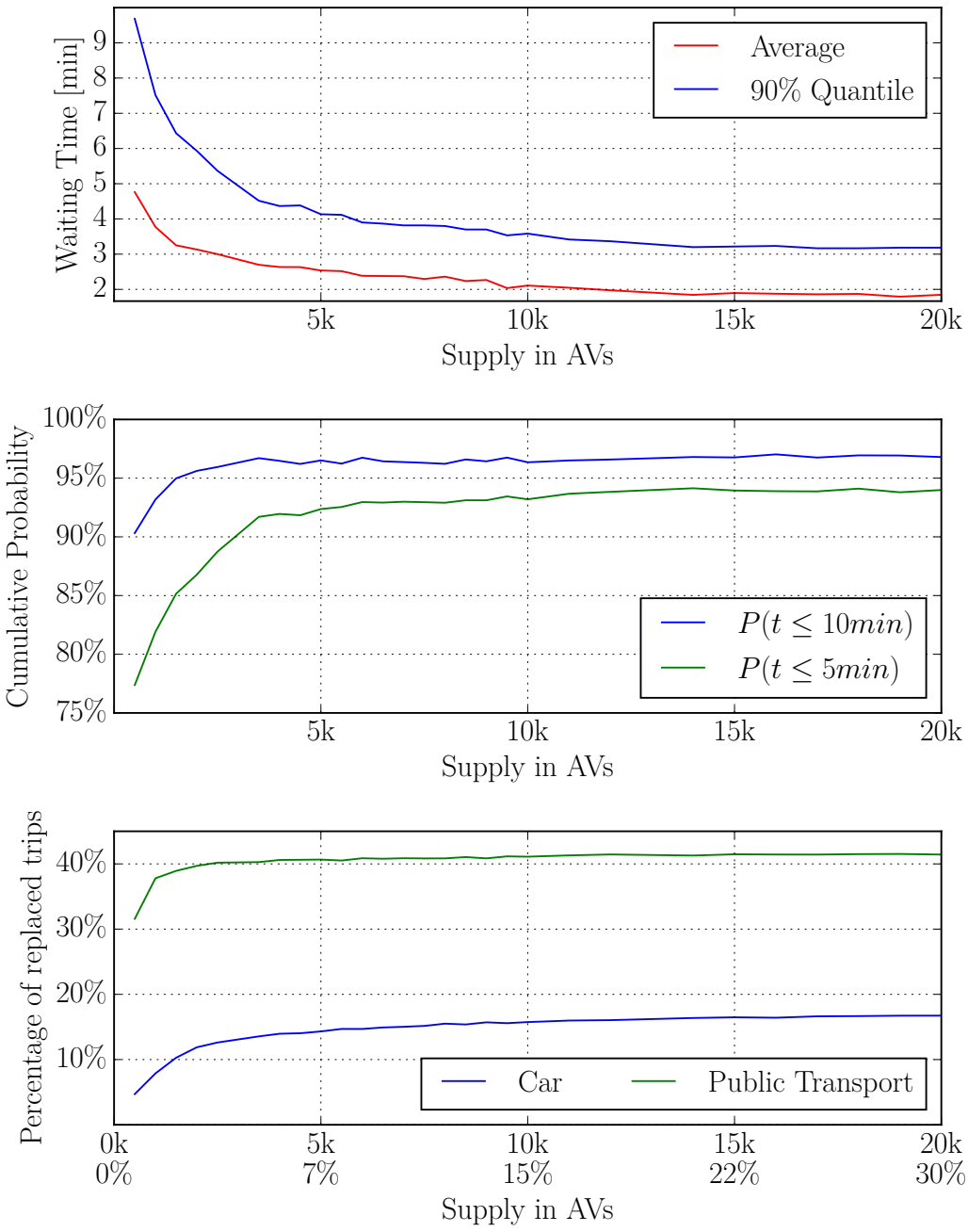

Figure 32: Top: Mean value and 90% quantile of waiting time for different supply levels. Bottom: Cumulative probability of observing a waiting time less than 10 minutes.

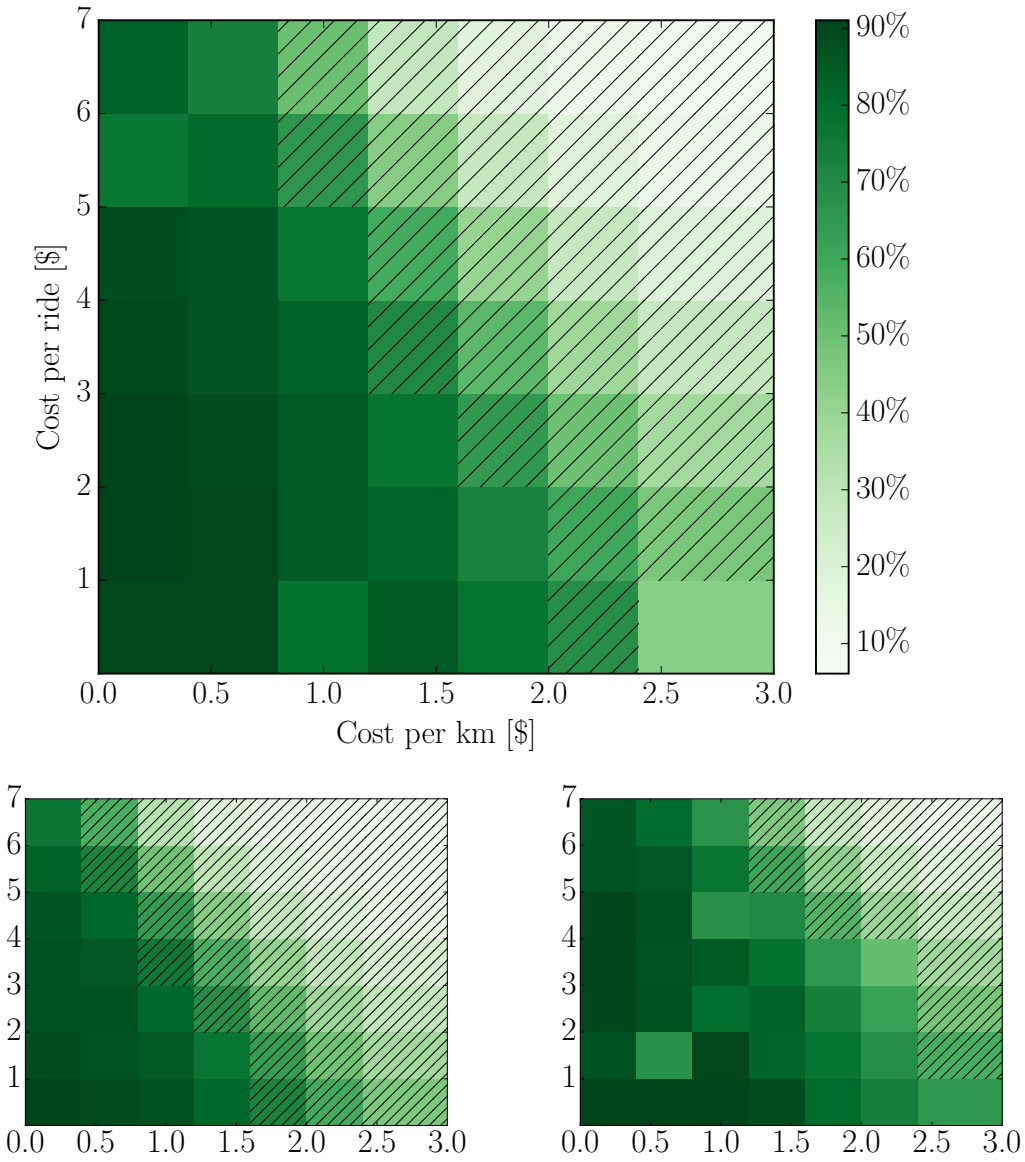

Figure 33: Dependency of the AV mode share on the pricing scheme. Top: Baseline scenario. Left: Low utility of traveling ($\beta_{trav,av} = -0.5$). Right: High utility of traveling ($\beta_{trav,av} = 0.5$). Shaded areas show parameter combinations where the waiting time for an AV is shorter than 10 min in 90% of the cases.

operation of the service. In fact, if waiting time is a constraint, only moderate shares of AVs can be reached on any level of acceptance.

What needs to be kept in mind here is that no further investigations on the disutility of waiting time have been performed here, but it is rather based on an assumption taken from Chakirov and Fourie (2014). Nonetheless, the result is surprising since, in extreme cases, agents accept a waiting time of 30 minutes or more in 90% of trips, as can be seen in fig. 34. Mainly, this depends on all the utility parameters in the scenario, also the utility of performing an activity, the disutility of using other means of transport and so forth. Accepting such a high waiting time might be an indicator that the utilities in the Sioux scenario should be further improved to lead to better results. However, the interpretation is tricky for very low prices, since they also resemble quite unrealistic situations, where only times are weighed against each other: If there are no monetary costs, a trip in terms of utility costs as much as not performing an activity for the travel time.

Furthermore, the results of the simulation are surprising when looking at the share of public transport in fig. 35. The general tendency makes sense: Lower prices lead to lower shares of public transport because using an AV gets more advantageous. Also, having very high prices, the public transport share stays at its initial baseline level. Nevertheless, one would expect people to react more abrupt to the pricing scheme on the per-trip side than on the per-km side. So far each trip in the Sioux Falls scenario costs \$2. Imposing no per-trip fee for AVs, but different per-km fares should show a smooth transition as can be seen in fig. 35: The shares should change depending on the price and the trip distance distribution. On the other hand, if no per-km fare is imposed, but only per-trip payments, the transition should be more abrupt. This effect, however, might be smoothed out by AVs taking less travel time.

This is only true though if people can make rational decisions about the total costs of travel. When looking at the beforementioned plots, one can see that there is are quite linear nivau lines, meaning that if a per-km cost is given, one can easily obtain the per-trip cost in order to stay at a certain level of service. For instance, this means that for the agents, paying \$5 per trip and \$1.60 per km is equal to paying \$3 per trip and \$1.90 per distance. In reality, the perception of the high per-trip fare might be different to the lower per-km fare, especially compared to the initial \$2 per trip.

Combining the results of this section, it becomes apprent that in the given scenario, also the share of public transport has a lower bound if a specific service level in terms of waiting time should be maintained. On the other hand, the introduction of AV services will diminish the share of public transport in any case. As one conclusion it can be

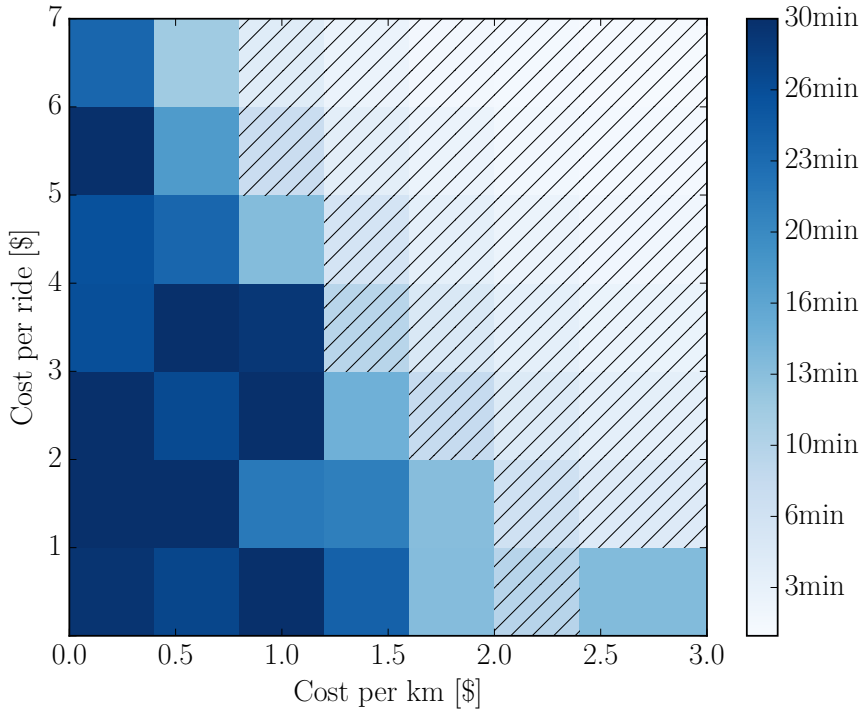

Figure 34: 90% quantile of the waiting time in the baseline scenario with different pricing schemes. The scale is truncated at 30 minutes.

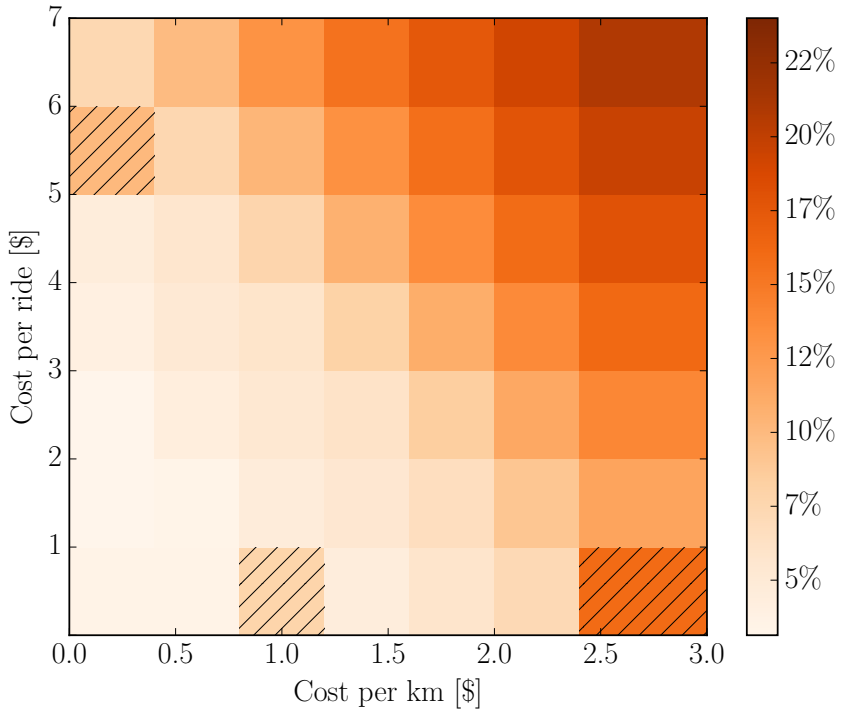

Figure 35: Share of public transport trips. Shaded areas indicate not completely relaxed simulation runs with stuck agents.

therefore stated that without any policy-based incentives, it is not possible to maintain the level of public transport while motivating private car owners to switch to AVs.

Another point that has to be taken into account regarding these considerations is the profitability of the service for the operator, which will be the subject of the next section.

6.4 Economic Analysis

The operator model for the net income z proposed in this thesis can be stated as follows:

$$z = \underbrace{p_{km} \cdot d_{dropoff} + p_{trip} \cdot n_{trips}}_{GrossIncome} - \left(\underbrace{\$6 \cdot n_{veh} + c_{pd} \cdot n_{veh} + \gamma_{d,car} \cdot d_{total}}_{Expenses} \right) \tag{11}$$

On the income side of the operator, there is the total distance of dropoff (i.e. occupied) trips $d_{dropoff}$, multiplied by the price per km and the number of AV trips n_{trips}, multiplied by the price per trip. The expense side has been modeled to be comparable with the car mode in the Sioux-16 scenario. It involves a cost for parking ($6) as well as running costs per km for private cars multiplied by the combined total distance for pickup and dropoff trips. Of course, this choice bears a lot of uncertainty, it might be a reasonable guess though, since increased costs for insurance and decreased costs for (electric) operational costs might weigh out each other (Chen et al., 2015). Finally, a cost per day c_{pd} is introduced for each supplied AV taxi.

That cost has been modeled as follows: Chen et al. (2015) states predictions of (electric) AV taxi prices of around $c_{veh} = \$62,000$ (converted to AUD) and states lifetimes of around $d_{max} = 370,000km$. From the simulation the average driven distance of one day is known as $d_{avg} = d_{total}/n_{veh}$ per vehicle. Those values can be used to obtain a vehicle lifetime, assuming that the amount of kilometers driven stays constant over the lifetime τ:

$$\tau = \frac{d_{avg}}{d_{total}/1d} = \left[\frac{km}{km/d} \right] = [d] \tag{12}$$

Then the costs per vehicle per day can be stated as:

$$
\begin{aligned}
c_{pd} &= \frac{c_{veh}}{\tau} = d_{avg} \cdot \frac{c_{veh}}{d_{max}} \\
&= d_{avg} \cdot \nu \\
&= \nu \cdot \frac{d_{total}}{n_{veh}}
\end{aligned} \tag{13}
$$

with

$$
\nu = 0.17 \frac{\$}{km}. \tag{14}
$$

Inserting this equation into eq. (15) effectively cancels out the number of available AVs from the investment costs and integrates them into the per distance costs:

$$
\begin{aligned}
z = & \underbrace{p_{km} \cdot d_{dropoff} + p_{trip} \cdot n_{trips}}_{GrossIncome} \\
& - \left(\underbrace{\$6 \cdot n_{veh} + (\nu + \gamma_{d,car}) \cdot d_{total}}_{Expenses} \right)
\end{aligned} \tag{15}
$$

Therefore the net income is characterized by a complex relation of the total distance driven, the occupied distance, the number of cars and the number of AV trips. Applying this model to the previously introduced pricing scheme map gives the result in fig. 36. For very low prices the service clearly is not profitable in the proposed model, while it is possible to maintain the service in a moderate price range. In fact, the area with most profit is covered by the formerly introduced condition on waiting times (hatched area). However, if a share of 15% of public transport share should be maintained, the operator would need to offer the service in the crossed area. There the profit is decreasing because of a smaller number of users.

For the baseline scenario, the operator scenario has been tested with different supply levels. The results in fig. 37 show that over the whole range of AVs there the service is profitable, especially at 4000 AVs. Over the whole depicted range the constraints on waiting time and public transport are fulfilled. In that sense those scenarios are quite optimal cases, where the share of public transport stays above 15%, the waiting times are usually less than 10 minutes and the operator has a large margin. Incorporating the results from section 6.1, the right-most cases are the best because additionally the overall excess mileage is smallest.

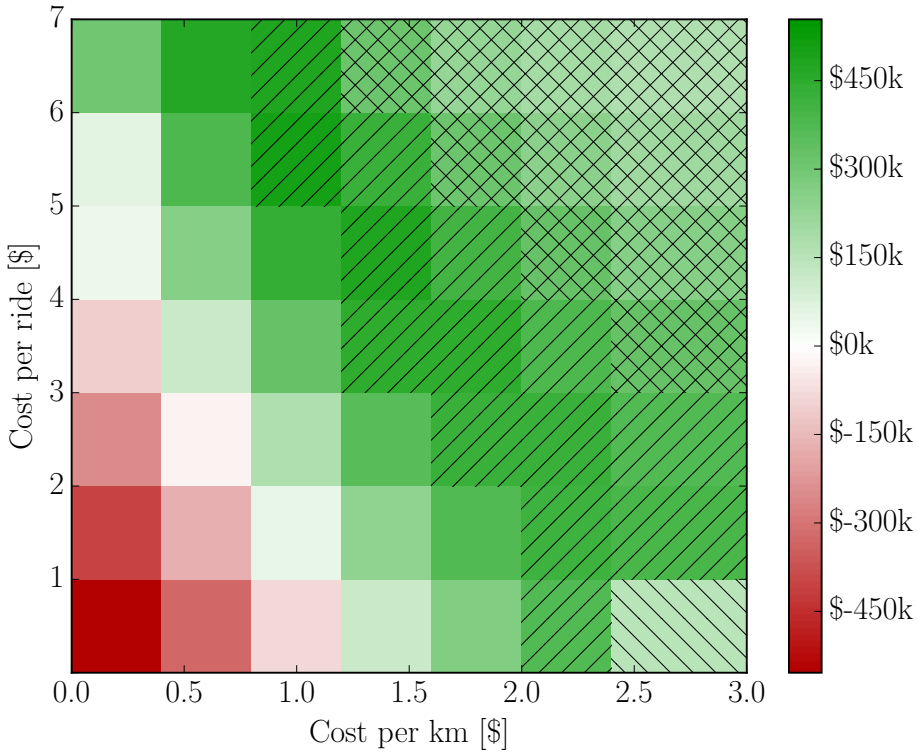

Figure 36: Net income of the AV operator. The shaded areas represent acceptable waiting times (bigger area) and public transport mode shares of more than 15% (smaller area).

The large margin is an indicator that the baseline scenario is a setup that could "work" in a city similar to the Sioux Falls network. Contrary to the financial analysis in Chen et al. (2015), here infrastructure costs have not been included in the analysis, which could be covered by that profit of the operator.

6.5 Summary

In the previous chapters the quantitative results of the AV simulation have been demonstrated in detail. However, the simulation at hand is by nature more suited for giving qualitative insights into the traffic system rather then quantitative since it is based on a virtual test scenario. The following paragraphs will summarize the beforementioned results and give a qualitative overview of the findings, putting a focus on the main subject of the thesis: The interaction of the new travel mode with the established means of transport.

One of the first results, which could be obtained, is that AVs in the test scenario will lead to an increased overall milage, which is an adverse effect looking from an environ-

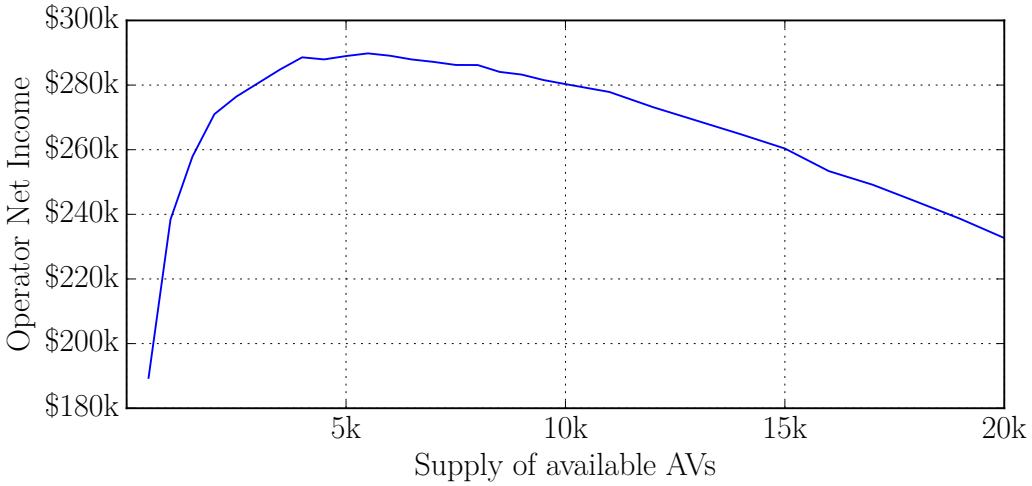

Figure 37: Dependency of the operator net income in the baseline scenario on the number of supplied AVs.

mentally perspective, but is also disadvantageous in terms of congestion in the network. Implementation-wise it has been found that considerable work needs to be put into intelligent ways of making AVs facilitate the existing infrastructure in order to avoid artificially made traffic jams, which is a problem that gets crucial with an increasing number of AV trips and therefore increasing mileage.

In that regard the introduction of AVs plays against one positive effect of public transport: to have fewer vehicles one the road. As could be shown in the results, former public transport users are the main adapters of autonomous vehicles in the given scenario. While private car users have a financial motivation to switch to the AV mode and accept longer travel times, public transport users mainly do the switch to reduce their accumulated travel time consisting of the walking to the stop facilities and the ride itself, possibly with several line switches.

This choice behavior goes along with a high tolerance for waiting times for the AV mode. Because public transport users are used to having a long travel time, the performance that could be reached in the given network was clearly enough to serve the demand. Nevertheless, the public transport users are not the audience that a policy maker would want to attract with AVs. From the findings in the simulations one can state (at least for the Sioux network), that it is very hard to impossible to maintain a certain level of public transport while getting private car owners to using the AV service. In fact, applying pinpointed incentive or taxation schemes to the network might be necessary to reach at the desired results.

In terms of pricing it has been found that the more inclined people are to spend time in an AV, the less constrained the financial structure of the service needs to be in order to reach certain AV shares. For the baseline scenario it has been found that it is possible to operate an AV service in a profitable way while still maintaining a share of %15 public transport and providing waiting times of less than ten minutes. While doing this, there would be a financial margin available on the side of the operator to cover necessary infrastructural expenses.

The final conclusion is that AVs without administrative regulation are likely to attract public transport users rather than private car owners. Further research needs to be done on how AV usage can be incentivized for private car owners to reach at a beneficial traffic situation.

Finally, it remains to state that the proposed model is built on a manifold of assumptions in the initial Sioux Falls scenario, in the operation of the AV agents and on the financial model for the operator. While for the latter future predictions will become better and better, it would be highly interesting to apply the developed model to a real-world scenario with a higher confidence in the estimated utility parameters and relative factors such as current taxi pricing.

7 Outlook

During the course of this thesis a versatile and extentable basis model for the simulation of autonomous vehicles has been developed. Many topics in the field which would be interesting to investigate were not part of the scope of this thesis. The following sections will present the most important and interesting ones and give directions on how to simulate them with the framework. Likewise, these sections also give an overview about critical points, which are not taken into account in the previous results, but could change the overall picture significantly.

Four different aspects will be put into focus:

- Extensions of the infrastructure in the model, which would account for what needs to be done to maintain the AV service in a more realistic and/or efficient way,
- the interaction with the AV service, widening the possibilities that one or multiple agents have to take advantage of the new travel mode, and
- extensions to the current demand simulation.

7.1 Infrastructure Extensions

Since autonomous vehicles in general will take great advantage of the ongoing electrification in the automotive industry, the availability of the respective infrastructure is one factor which can determine how fast the adaptation of AVs will progress (Burmeister et al., 2016). Given the right amount of resources to create such an infrastructure, a big question that arises is how many recharging facilities are needed and where they should begin located in order to serve certain sizes of AV fleets (Chen, 2015). Furthermore, how can a combined infrastructure for ordinary owned EVs, private AVs and publicly provided AV services be designed?

As a first step one could assume that AVs recharge at dropoff locations, where they are artificially put to rest for a certain minimum idle time in order to simulate the recharging process. This would already lead to results on the customer acceptance side, but would ignore effects on congestion if AVs would actually need to take long trips to the next charging facility. Previous studies using MATSim already gave interesting results on the implementation of eletrified taxis in Berlin with distributed charging facilities over the network (Bischoff and Maciejewski, 2014). Such research could be combined with the AV framework developed here. The modular AgentFSM component would make it easy

to add specific new points in the state chains of an AV to drive to a charging facility. Then again, the simulation framework could be used to get insights in which scheduling strategies would be optimal for an AV fleet if recharging has to be taken into account.

Another idea to improve the simulation is to introduce means of simulating maintenance and parking. So far it has been assumed that autonomous vehicles will reside where the last dropoff has taken place. This assumption, especially in heavily packed cities, is quite optimistic, since parking space might be rare. In consequence, the driven kilometers per AV in the unoccupied state could be quite off the actual distance. More mileage would be needed to find a parking space in between tasks. This rises a whole range of questions on how an optimal AV scheduling would look like, maybe depending on the demand level, it might be even interesting to run studies on whether the search for parking space might be inferior to roaming around.

This could be done in combination with intelligent repositioning, where in between peak hours taxis could be intelligently moved to likely pickup positions and thus minimize the waiting time for customers, increasing the acceptance and reducing operator costs at the same time because less unoccupied miles might be travelled.

With the parallelization of customer trips the presented framework already shows by example how such an algorithm could be incorporated into the existing infrastructure without having too much impact on the computation times. In general, doing "some" intelligent repositioning should always be more beneficial than doing none. In this regard, the repositioning could be computed in parallel to the ongoing traffic simulation, while still offering the ability to restrict it if it slows down the main loop of MATSim.

7.2 Usage and interaction with the AV service

The AV service in the developed model so far is very basic in the way customers are able to interact with it. One obvious advantage of AV fleets is, that up to five people could be transported in ordinarily shaped cars and even more in autonomous minibusses or full-sized busses. Intelligent routing and scheduling algorithms would make it possible to pick up passengers at arbitrary locations, not being bound to a fixed public transport schedule.

Due to the extensible structrue of the AV extension, it would be easy to add such behaviour in principle. However, the problem of optimizing the trips of more than one passenger, probably while already on a ride, is a highly complex problem and acceptable heuristics are still an important research subject. Adding such functionality to the AV framework

would make it possible to test such strategies in near-realistic scenarios and measure the performance of different heuristics.

Additionally, autonomous vehicles are likely to relax the last mile problem, where people are not motivated to opt for public transport, because the last mile from the transport facility to their home is too long. An AV could bridge this distance, maybe being prescheduled to pick up the passenger according to the current expected arrival times.

Implementing this behaviour would need only small changes in the MATSim framework. Generally, a public transport trip is generated as three legs: One `transit_walk` leg in order to get to the stop facility, one `pt` leg for the actual residual time in the bus and another `transit_walk`. A first attempt would be to replace some of the `transit_walk` legs during the planning phase with `av` trips, which could already lead to a convincing simulation of AVs feeding the public transport network. A probable requirement for this to work well would be to have prescheduled AVs to give a higher reliability on these connections.

Prescheduled autonomous vehicles would be easy to implement in the existing infrastructure. In fact, test have already been done, but abandoned due to the fact that in a first approximation delays from the scheduling can be modeled using worse utility values for the waiting times. In the current state, the AV framework fully supports such prescheduled trips, though their impact has not been investigated in the scope of this thesis. As shown in figure fig. 15, where the state diagram of the AV is depicted, a "Waiting" state is already included, which would make an AV which arrives early at a pickup location wait for the passenger.

7.3 Demand Analysis

An intereting point in the simulation is the spatial dependence. On one side, it would be interesting to investigate where AV users live, maybe indicating that on specific pricing strategies, people from the suburbs prefer AVs, while other strategies might encourage people living in the center to use them.

Heavily related to that is the initial distribution of AVs at the beginning of the daily simulation. As described before, AVs are currently distributed dependent on the population density, though that might not be the best approach. Especially if one wants to encourage suburbians to use AVs, the density there should be higher.

In that regard it would be beneficial to investigate how the initial conditions in the MATSim simulation influence the result on an abstract level. For the case of AVs, having initial AV plans mainly assigned to agents from a city center, but not for people from the suburbs, the relaxed state might settle down in exactly this condition. However, if AVs are mainly distributed in the suburbs in the initial plans, the main user group might stay there, just because during the simulations the waiting times are shorter in either case and therefore these people might stick to their initial plan decisions. Therefore, a thorough investigation of the distribution behavior of the algorithm would be very interesting.

Furthermore, the research in this thesis has shown that without any incentives, AVs might lead to adverse effects, which should be corrected by intelligent policy decisions. Chen and Kockelman (2016) suggests interesting approaches of incentivizing AV usage. In order to "free" the city center from too much congestion one could for instance in the Sioux Falls scenario put high monetary fees on using the streets within the highway belt for private cars, while in parallel increasing the likelihood for AVs to use the highways. This way one could try to move traffic to the highways, then taking a direct trip to the workplaces in a perpendicular way.

Experiments like these are ready to be done with the existing simulation framework by adding custom scoring functions for private cars and autonomous vehicles.

Finally, efforts have been made recently to diversify the population in MATSim simulations (Chakirov, 2015). This means that people might be constrained or inclined due to age or income to use certain means of transport. Combining the simulation of AVs with the introduction of that heterogeneity could give insights on how the availability and pricing of AVs affect the distribution of users in terms of a richer set of social variables.

8 References

Carles Amat, Javier Ortigosa Marin, and Miquel Estrada-Romeu. Assessment of the taxi sector efficiency and profitability based on continuous monitoring and methodology to review fares. *Transportation Research Board 93rd Annual Meeting*, 2014.

James M. Anderson, Kalra Nidhi, Karlyn D. Stanley, Paul Sorensen, Constantine Samaras, and Oluwatobi A. Oluwatola. *Autonomous vehicle technology. A guide for policymakers*. RAND Corporation, Santa Monica, 2016.

Joschka Bischoff and Michal Maciejewski. Agent-based Simulation of Electric Taxicab Fleets. *Transportation Research Procedia*, 4:191–198, 2014.

Joschka Bischoff and Michal Maciejewski. Simulation of city-wide replacement of private cars with autonomous taxis in Berlin. *The 7th International Conference on Ambient Systems, Networks and Technologies*, 2016.

Patrick M. Boesch and Francesco Ciari. Agent-based simulation of autonomous cars. *Proceedings of the American Control Conference*, July 2015:2588–2592, 2015.

Patrick M. Boesch, Francesco Ciari, and Kay W. Axhausen. Required autonomous vehicle fleet sizes to serve different levels of demand. *TRB 95th Annual Meeting Compendium of Papers*, 2016.

Geoff Burmeister, Sommer Engels, Ben Hamm, and Andrea Kraus. Automated Vehicles for a Sustainable City. Technical report, 2016.

Artem Chakirov. *Urban Mobility Pricing With Heterogeneous Users*. PhD thesis, 2015.

Artem Chakirov and Pieter J. Fourie. Enriched Sioux Falls Scenario with Dynamic And Disaggregate Demand. 2014.

T. Donna Chen. *Management of a Shared, Autonomous, Electric Vehicle Fleet: Vehicle Choice, Charging Infrastructure & Pricing Strategies*. PhD thesis, 2015.

T. Donna Chen and Kara M. Kockelman. Management of a shared, autonomous, electric vehicle fleet: Implications of pricing schemes. *Proceedings of the 95th Annual Meeting of the Transportation Research Board*, 2016.

T. Donna Chen, Kara M. Kockelman, and Josiah P. Hanna. Operations of a shared, autonomous, electric vehicle fleet: Implications of vehicle & charging infrastructure decisions. 2015.

City of Gothenburg. DriveME self driving cars for sustainable mobility, 2016. URL http://international.goteborg.se/smart-cities-and-sustainable-solutions/driveme-self-driving-cars-sustainable-mobility.

E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

Alex Erath, Pieter Fourie, Michael van Eggermond, Sergio Ordoñez, Artem Chakirov, and Kay W. Axhausen. Large-scale agent-based transport demand model for Singapore. *13th International Conference on Travel Behaviour Research*, (June), 2012.

Daniel Fagnant and Kara M. Kockelman. The travel and environmental implications of shared autonomous vehicles, using agent-based model scenarios. *Transportation Research Part C*, 40, 2014.

Geir Martin Pilskog. Statistisk sentralbyrå: Taxi transport, Q4 2015, 2015. URL http://www.ssb.no/en/transport-og-reiseliv/statistikker/drosje/kvartal/2016-03-03.

Google. Google Self-Driving Car Project, 2016. URL https://www.google.com/selfdrivingcar/.

Alexander Hevelke and Julian Nida-Rümelin. Responsibility for Crashes of Autonomous Vehicles: An Ethical Analysis. *Science and Engineering Ethics*, 21(3):619–630, 2015.

Andreas Horni, Kai Nagel, and Kay W Axhausen. The Multi-Agent Transport Simulation MATSim. 2015.

International Transport Forum. Mobility: System Upgrade, 2014.

Java API. PriorityQueue, 2016. URL https://docs.oracle.com/javase/7/docs/api/java/util/PriorityQueue.html.

Scott Le Vine, Alireza Zolfaghari, and John Polak. Autonomous cars: The tension between occupant experience and intersection capacity. *Transportation Research Part C*, 52, 2015.

Todd Litman. Autonomous Vehicle Implementation Predictions: Implications for Transport Planning. *Transportation Research Board Annual Meeting*, 2015.

Michal Maciejewski and Joschka Bischoff. Large-scale Microscopic Simulation of Taxi Services. *Procedia Computer Science*, 52:358–364, 2015.

Michał Maciejewski and Kai Nagel. Towards Multi-Agent Simulation of the Dynamic Vehicle Routing Problem in MATSim. 2012.

Thomas P Minka. Estimating a Gamma Distribution. *Microsoft Research*, 2002.

E.K. Morlok, J.L. Schofer, W.P. Pierskalla, R.E. Marsten, S.K. Agarwal, J.W. Stoner, J.L. Edwards, L.J. LeBlanc, and D.T. Spacek. Development and application of a highway network design model. Volumes 1(Final Report: FHWA Contract Number DOT-PH-11, Northwestern University, Evanston.), 1973.

Brandon Schoettle and Michael Sivak. Public Opinion About Self-Driving Vehicles in China, India, Japan, The U.S., The U.K. and Australia. 2014.

Gary Silberg, Mitch Manassa, Kevin Everhart, Deepak Subramanian, Michael Corley, Hugh Fraser, and Vivek Sinha. Self-Driving Cars: Are we Ready? *KPMG LLP*, 2013.

Kevin Spieser, Kyle Ballantyne, Rick Zhang Treleaven, Emilio Frazzoli, Daniel Morton, and Marco Pavone. Toward a Systematic Approach to the Design and Evaluation of Automated Mobility-on-Demand Systems A Case Study in Singapore. In Gereon Meyer and Sven Beiker, editors, *Road Vehicle Automation (Lecture Notes in Mobility)*. Springer, 2014.

Cheon Kheong Tan and Kwang Sheun Tham. Autonomous Vehicles, Next Stop: Singapore. *Journeys*, (November):5–11, 2014.

Tesla. Summon Your Tesla from Your Phone, 2016. URL https://www.teslamotors.com/blog/summon-your-tesla-your-phone.

Transport for NSW. Maximum taxi fares and charges, 2016. URL http://www.transport.nsw.gov.au/customers/taxis/maximum-taxi-fares-and-charges.

Transport Systems Catapult. Self-driving pods, 2016. URL https://ts.catapult.org.uk/pods.