# Sophisticated Route Calculation Approaches for Microscopic Traffic Simulations

Karl Hübner
Technische Universität Berlin
Daimler Center for Automotive Information Technology Innovations
Ernst-Reuter-Platz 7, 10587 Berlin, Germany
karl.huebner@tu-berlin.de

Björn Schünemann
Technische Universität Berlin
Daimler Center for Automotive Information Technology Innovations
Ernst-Reuter-Platz 7, 10587 Berlin, Germany
bjoern.schuenemann@dcaiti.com

Ilja Radusch
Fraunhofer FOKUS
Automotive Services and Communication Technologies
Kaiserin-Augusta-Allee 31, 10589 Berlin, Germany
ilja.radusch@fokus.fraunhofer.de

## ABSTRACT

In order to implement meaningful microscopic traffic simulations, a sophisticated calculation of the vehicle routes is essential. In this paper, different route calculation approaches for microscopic traffic simulators are analysed and compared. Various simulations are performed to detect traffic distribution and traffic flow resulting from the used route calculation approaches. To have simulations as realistic as possible, real OD matrices of the city of Fulda (Germany) were used for the evaluation of the different approaches. The results show that simple route calculation algorithms are not appropriate for microscopic traffic simulations. However, approaches that are more sophisticated and that integrate additional heuristics produce good results regarding traffic distribution and traffic flow and can improve iterative techniques to find a user equilibrium.

## Categories and Subject Descriptors

I.6.7 [**Simulation and Modeling**]: Simulation Support Systems

## General Terms

Route Calculation

## Keywords

Route Calculation Approaches, Microscopic Traffic Simulations, Route Assignment, Trip Generation, Shortest-path Search, Choice Routing, Dynamic Traffic Assignment, User Equilibrium

## 1. INTRODUCTION

The generation of realistic traffic is an essential requirement for the execution of meaningful traffic simulations. Since the movements of all vehicles are simulated individually in microscopic traffic simulations, the place and the time of the departure, the destination, and the route through the traffic network have to be defined for each vehicle. The process of this data generation can be split into four steps, which are described by the traditional traffic prediction model: *Trip generation* and *trip distribution* are responsible for determining origin and destination points and the number of trips between them, *mode choice* detects which transportation mode is used for each trip, and *route assignment* computes a route for each trip and defines the point in time when a trip is started [9, 12].

If realistic background traffic for a whole city or a certain area is needed, data based on real measurements or empirical studies are helpful, such as predefined origin-destination (OD) matrices which can be obtained from local traffic authorities (e.g. [14]). If such matrices are available, only *mode choice* and *route assignment* have to be carried out for traffic generation. While *mode choice* is a rather simple task (using statistics about modal splits for example) [12], *route assignment* is more difficult to perform since the aim is to find routes which result in balanced traffic which is close to reality.

A trivial approach of *route assignment* would be to assign the shortest route for each trip (all-or-nothing). However, this approach would often result in traffic bottlenecks and heavy congestion. In order to prevent this, the routes

SIMUTOOLS 2015, August 24-26, Athens, Greece

DOI 10.4108/eai.24-8-2015.2261359

might be distributed among the road network, which can be achieved by applying Wardrop's principle of user equilibrium (UE) where each driver chooses a route in a way that he/she cannot reduce his/her own travel time by changing to another route [16, 5]. In order to calculate the user equilibrium in a time-dependent environment, *dynamic traffic assignment* methods are used. Such methods solve the UE problem either mathematically, or - for microscopic simulations more suitable - by iterative simulations until the equilibrium is reached [4]. An iterative approach, however, requires much computing time due to the excessive amount of simulation runs needed. Therefore, it might be helpful to reduce the number of iterations by putting more effort into route calculation, which is presented in the following.

### 1.1 Paper Structure

This paper is structured as follows: In Section 2, different approaches for the route calculation will be presented. An introduction of methods used to create vehicular traffic from existing OD matrices will follow in Section 3. Moreover, our simulation setup and the used evaluation methods and measures will be introduced in Section 4. Finally, the achieved results will be analysed in Section 5, and a conclusion will be given in Section 6.

## 2. IMPROVEMENTS IN ROUTE ASSIGNMENT

In order to prevent traffic bottlenecks and heavy congestion caused by the generated vehicle routes, sophisticated methods for the route calculation have to be identified. In the following section, several approaches are discussed for improving the single route search and for calculating alternative routes and distributing the traffic among these routes.

### 2.1 Improving Shortest-Path Search

Two mechanisms seem to be promising to improve the shortest-path search: Avoid the use of roads with low capacity and avoid time-consuming turns.

The use of roads with low capacity can be avoided by increasing the costs of the relevant roads for the path search, for example by adapting the costs of each road segment according to its defined road type or capacity.

Considering turn costs is a more challenging task. Turn costs can lead to P-turns, e.g. a detour around a city block containing three right turns instead of one left turn in order to avoid a turn restriction [17]. P-turns, however, would violate Bellman's optimality condition. This condition indicates that if the shortest path between origin and destination passes through two arbitrary nodes A and B, also the chosen path between A and B must be the shortest possible path [11, 2]. This condition implies that a shortest path must not contain the same node twice [6]. Since turn restrictions are a special case of turn costs, this problem needs to be addressed as well. In order to avoid the violation of the principle, there are two main methods: Firstly, junctions within the graph can be modelled as subgraphs with additional edges representing all turning possibilities. With this approach, any existing shortest path algorithm can be applied without violating Bellman's principle [17] (see fig. 1). Another method is using an edge-based path search algorithm. Instead of storing weights for each visited node, weights for each visited edge are stored during traversal. This approach allows applying turn restrictions and turn costs since the previous edge is known and can be used to calculate turn costs (see fig. 2). Thus, turn costs are considered and P-turns are possible without violating Bellman's principle [6].

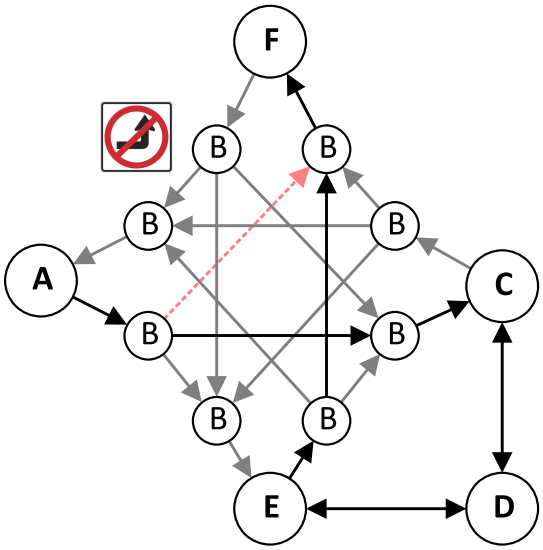



**Figure 1: To avoid the violation of Bellman's principle, junctions within the graph are modelled as subgraphs with additional edges representing all turning possibilities. In this way, any node-based routing algorithm (e.g. Dijkstra) finds allowed P-turns.**

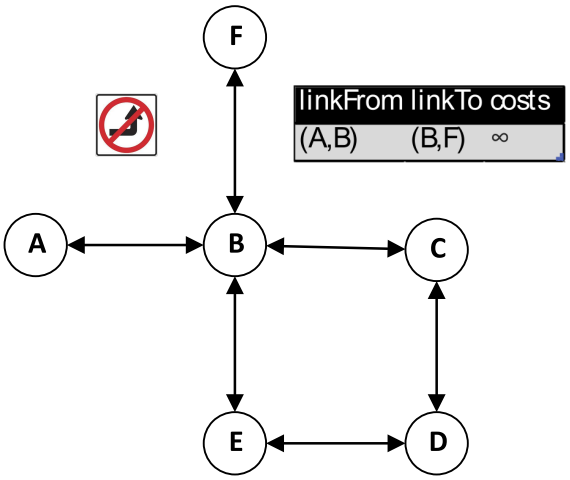



**Figure 2: An edge-based routing algorithm and turn-cost tables are used to enable turn restrictions and turn costs without editing the underlying graph.**

### 2.2 Determining Turn Costs

To calculate turn costs, Geisberger et al. [7] propose to consider the deceleration, the acceleration and the maximum turning speed to achieve a realistic presentation of the time needed for a turn. Considering the maximal tangential acceleration $max_a$, the speed limits $v$ and $v'$, the edge lengths

$l$ and $l'$, and the angle $\alpha$ in between, the maximum turning speed between both edges can be calculated by equation 1 [7].

$$v_{turn} := \min(v, \sqrt{max_a \cdot \tan(\alpha/2) \cdot \min(l, l')/2}) \quad (1)$$

Furthermore, considering $a_{acc}$ as the maximum acceleration and $a_{dec}$ as the maximum deceleration of a vehicle, the approximated time a turn costs $c_{turn}$ can be calculated by equation 2 [7].

$$c_{turn} := \frac{(v - v_{turn})^2}{2 \cdot a_{dec} \cdot v} + \frac{(v' - v_{turn})^2}{2 \cdot a_{acc} \cdot v'} \quad (2)$$

In our experiments, we used these equations to determine the turn costs. We assume the accelerations $a_{acc} = 2{,}6\,\mathrm{m/sec^2}$ and $a_{dec} = 4{,}5\,\mathrm{m/sec^2}$. These are the values the traffic simulator SUMO applies for the default acceleration and the default deceleration.[1]

## 2.3 Finding Alternative Paths

In general, drivers have different preferences and experiences, and therefore choose different routes to reach the same destination. Consequently, several approaches and algorithms exist to find paths with similar costs but different road sections.

Yen developed an algorithm to find the k-shortest, loopless paths in a network [18]. However, the k-shortest paths are not feasible in road traffic scenarios since they contain many similar and unrealistic paths that users in reality would not use [1]. One method to solve the problem of similarity is to remove similar and unrealistic paths out of the shortest-paths set, e.g. by applying the Minimax method by Kuby et al. [10]. Those improvements, however, increase the calculation time of routes even more.

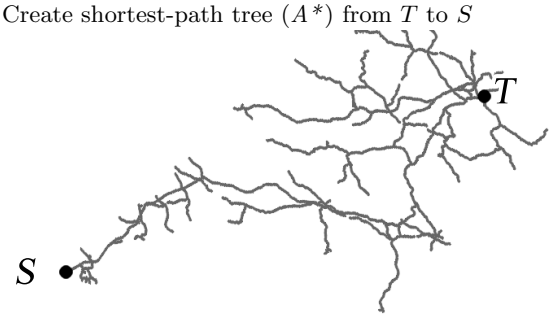
A method to identify potential dissimilar paths is *Iterative Penalty Method (IPM)* proposed by Johnson [8]. As soon as the best path has been found by any arbitrary single path search algorithm, the weights of all edges on the resulting path are penalized. A second search is done to find a different route which costs are quite similar to the best one. This can be performed several times until the desired number of alternative paths is found. Furthermore, Bader et al. penalized not only the edges of a path but all edges which are leaving from nodes visited by the path. This *tube* avoids small detours of already found routes [1].

*Choice Routing* is a relatively new method [3] which operates in four steps. Firstly, two shortest path algorithms are executed simultaneously, one from the source node towards the target and one from the target node towards the source; resulting in two shortest-path trees. In the third step, both shortest-path trees are intersected with each other, which results in a set of edges traversed by both path searches. The connected edges in this set are called *plateaus*. In the fourth step, plateaus are ranked by a quality criterion and paths are generated by following the weighted trees. Figure 3 shows the different stages of Choice Routing in an example. Bader et al. showed that this simple and fast method results in very good alternative routes. It was also shown that Choice Routing is able to match about 80% of the routes which drivers would choose in reality [1].

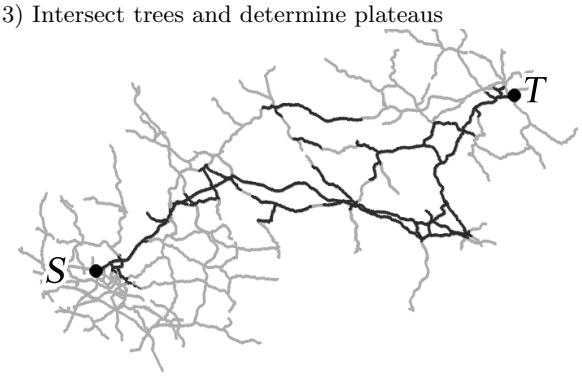
[1] `http://sumo.dlr.de/wiki/Definition_of_Vehicles,_Vehicle_Types,_and_Routes`

1) Create shortest-path tree ($A*$) from $S$ to $T$


2) Create shortest-path tree ($A*$) from $T$ to $S$




3) Intersect trees and determine plateaus




4) Build routes based on plateaus

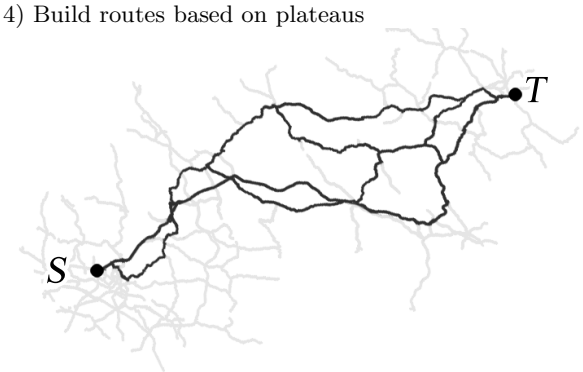



**Figure 3: Finding alternative routes in four steps by applying the Choice Routing algorithm.**

## 2.4 Route Selection

To assign different routes with the same origin and destination to the trips resulting from the corresponding OD matrix pair, decision models can be used. These decision models detect a probability for using a particular route. One example is the logit model [13]. Here, for each route $r$, a *utility function* $u_r$ is used to calculate the probability $P(r)$ to choose this particular route. Considering a set of $R$ different routes and a scaling parameter $\beta$, the logit decision model is described by equation 3 [13].

$$P(r) = \frac{\exp(\beta \cdot u_r)}{\sum_{s=1}^{R} \exp(\beta \cdot u_s)}, r = (1, \ldots, R) \quad (3)$$

# 3. GENERATING INITIAL TRAFFIC

In order to generate the initial routes for a simulation, different methods can be applied. In this chapter, we give a brief overview about approaches and methods used by us to create traffic from existing OD matrices.

## 3.1 Preparation of the OD Matrix

In many cases, the origin and destination points given in OD matrices are based on traffic analysis zones (TAZ). Those analysis zones need to be transferred into a simulation scenario. In order to distribute departures and arrivals within a zone, we generate several origin and destination points at junctions within each TAZ randomly. Furthermore, OD matrices do not provide the number of trips per hour, but for one whole day. Since departure times for each trip are required, the trips have to be distributed over time. For this task, we use *one-day-variation curves* which describe the amount of traffic over a normal working day per hour. For each trip, the time of departure is determined by distributing all trips over time according to such curves. Within each hour, departure times are distributed equally.

## 3.2 Calculation of the Initial Routes

In the following sections, several methods for calculating initial routes are described. We used these routes to generate the traffic for our evaluations.

**F** - For each OD pair, the fastest route is calculated. No additional routes are provided or calibration techniques are applied.

**F*** - For each OD pair, the fastest route is calculated by considering turn costs and avoiding streets in residential areas. No additional routes are provided or calibration techniques are applied.

**CR4*** - For each OD pair, four alternative routes are calculated by Choice Routing. The route calculation considers turn costs and avoids streets in residential areas. No calibration techniques are applied.

## 3.3 Calibrating Traffic

In addition to the previous methods, the following approaches use an iterative dynamic traffic assignment approach (DTA) in order to calculate an initial route for each trip. While the first iteration step is an all-or-nothing assignment, each following iteration uses the traffic flow of the previous simulation in order to assign routes for the vehicles.

**DTA** - The following steps are executed alternately for a specific number of iterations: 1) route calculation, 2) route selection, 3) traffic simulation. In each iteration, the route calculation calculates fastest routes based on the traffic flow of the previous iteration without considering turn costs (figure 4).

**CR4*+DTA** - An improved version of the previous approach, where a more sophisticated route calculation approach is used in order to pre-calculate a set of routes. At first, four alternative routes are calculated for each OD pair by applying Choice Routing (including turn costs and the avoidance of residential areas). Afterwards, the traffic is calibrated by executing the following steps alternately: 1) route selection, 2) traffic simulation. Here, the route selection only chooses between existing routes which were calculated once at the beginning (figure 5).

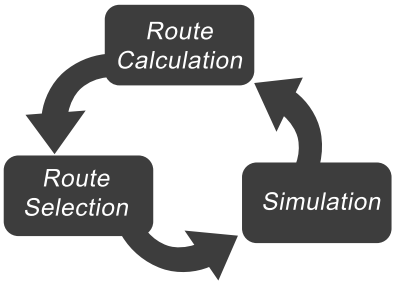



**Figure 4: Iterative Dynamic Traffic Assignment: Reaching the user equilibrium in three alternating steps.**

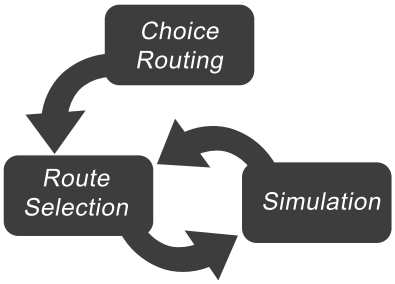



**Figure 5: Iterative Dynamic Traffic Assignment: Instead of recalculating routes every time, a set of sophisticated routes is calculated once at the beginning.**

# 4. PREPARATION OF THE SIMULATION SCENARIO

In this section, we introduce our simulation setup. We use the previously presented methods to generate vehicle routes and to simulate the resulting traffic by the SUMO simulator. Moreover, we define evaluation methods and measures to allow a comparison of the different route calculation approaches.

## 4.1 Simulation Setup

In 2008, the results of the future planning of the public transportation services of the city of Fulda (Germany) were published online [14]. These results include OD matrices and a detailed map of all traffic analysis zones. To model traffic demand close to reality, we used the published OD matrices to calculate the vehicle routes for our simulations.

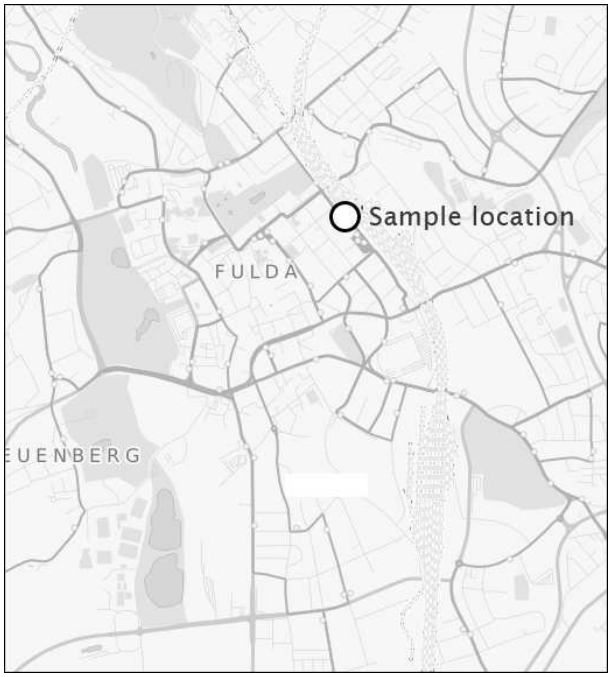



**Figure 6: This map of the city of Fulda shows the network we used for the evaluation. At important road segments, e.g. at the depicted sample location, we set up induction loops for counting traffic.**

The road network we used for our simulations is based on OpenStreetMap data of the region Hesse, Germany[2]. We removed unnecessary objects from the OSM data and fixed some errors caused by the converting process of the SUMO tool netconvert. For example, we fixed illegal turns and faulty traffic light programs. For a later analysis, we added detectors at important road segments within our simulation to analyse and identify congested roads.

## 4.2 Preparation of Traffic Demand

After the simulation scenario preparation, we set up the traffic demand. For this purpose, the following files were chosen from [14]: the OD matrix of the public transportation demand as expected in 2015 and the OD matrix of the general traffic demand as expected in 2015 (general traffic = public transportation + motorized traffic [14]). The OD matrices had to be converted into a suitable format since they were published as PDF files. In order to retrieve an OD matrix with traffic demand of motorized vehicles only, the OD matrix for public transportation was subtracted from the one for general traffic. The provided data consists of 67 traffic analysis zones (TAZ), however, half of the traffic concentrates within the first 17 zones. Therefore, we simulated traffic within these zones only. The lost traffic was compensated by raising the number of trips accordingly. For each TAZ, origin and destination points were created (according to section 3.1). As a result, 5 760 OD pairs and 160 000 trips were created and used for the route calculation approaches described in sections 3.2 and 3.3.

[2]`http://download.geofabrik.de/europe/germany/hessen.html`

## 4.3 Measures

After the vehicle routes were generated, numerous simulations were run in SUMO. At the end of each simulation, the generated trip information were used to compare the different simulation runs. For the comparison of the different route calculation approaches, we used following measures:

**Number of vehicles** which reached their target. In the case that several vehicles did not reach their target until the end of the day, probably a huge traffic congestion occurred in the network which prevented vehicles from reaching their target on time.

**Average length of trips** helps to detect which amount of detours was made by the vehicles compared to the shortest possible routes.

**Average duration of trips** shows how long vehicles drove from the origin to the destination. We assume, that a better distribution of traffic results in less congestion and therefore in lower trip durations. Therefore, the average travel time is used as one of the main characteristic for the analysis of the traffic distribution.

**Average standard deviation of duration of trips** shows how much trip duration varies in relation to the average duration. For this measure, we calculate the standard deviation of all trips which have the same source and destination. The average of these standard deviation values is used to analyze the traffic distribution in terms of fulfilling the user equilibrium. The lower the deviation for a set of trips with the same source and destination, the lower the benefit a driver would experience when changing to another route. Additionally, high values in standard deviation indicate a congested network, since drivers suddenly need more time to reach their destination.

**Average speed** of vehicles describes the average velocity of vehicles. Comparing different simulation runs, this measure offers a good indicator of free or congested traffic.

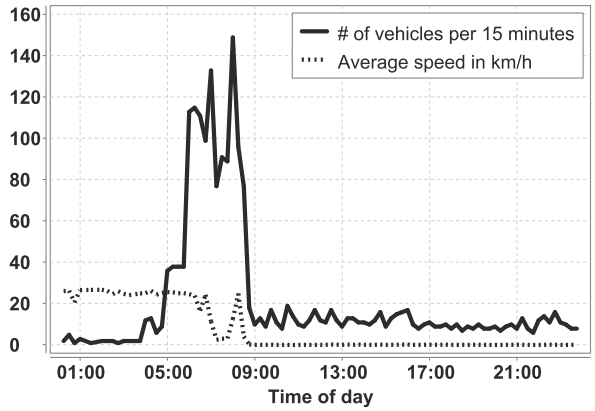
**Average waiting time** of vehicles shows how many seconds on average vehicles spent waiting during their trip, for example at traffic lights or due to traffic congestion.

In addition to these metrics, several induction loops on important roads were set up. These detectors measured the **number of vehicles**, the **average speed** and the **vehicle flow** over a period of 300 seconds. By these measures, time-depending events could be detected.

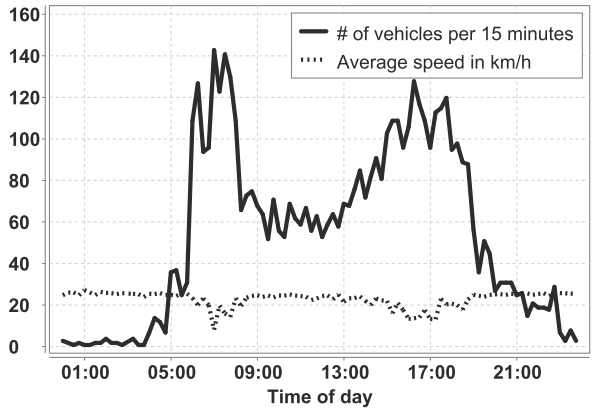
# 5. RESULTS

Table 1 shows the essential results of our simulations. Each one simulated a whole day. The simulated traffic is based on the OD matrix of Fulda, containing real traffic demand data. For the route generation, one of the discussed approaches was used. The measured results give a first insight into the differences of the approaches regarding the calculated routes and the resulting traffic flow. We assume that the traffic flow over a full day is balanced reasonably if average travel time and speed are appreciable and traffic congestion only occur locally. In order to keep track of special events which are supposed to occur in real traffic flow (e.g. peaks in morning / evening hours and local traffic congestion) and events which usually do not occur in real traffic flow (a highly congested network, frequent deadlocks on junctions), we measured, additionally, the traffic volume and speed over the time on one sample location within the network (figure 7).

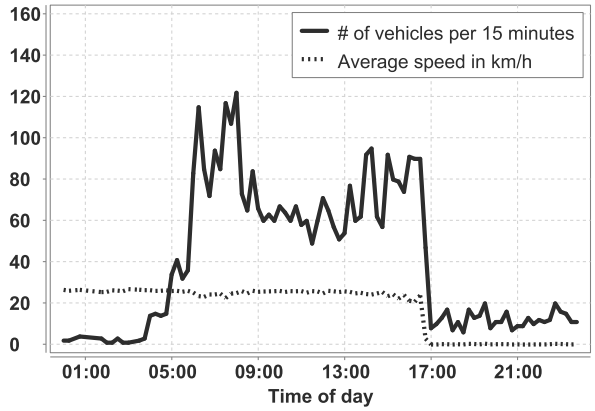
**F** (fastest routes)



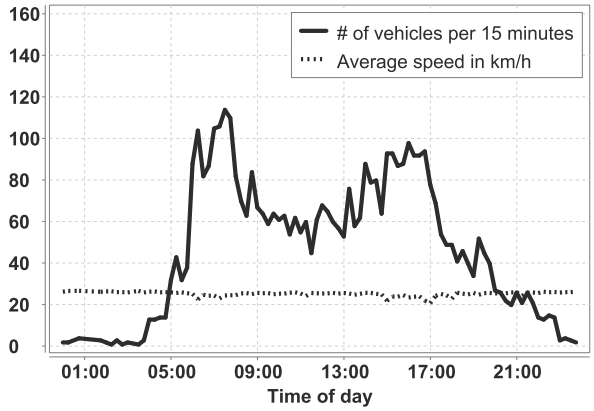

**F*** (fastest with heuristics)



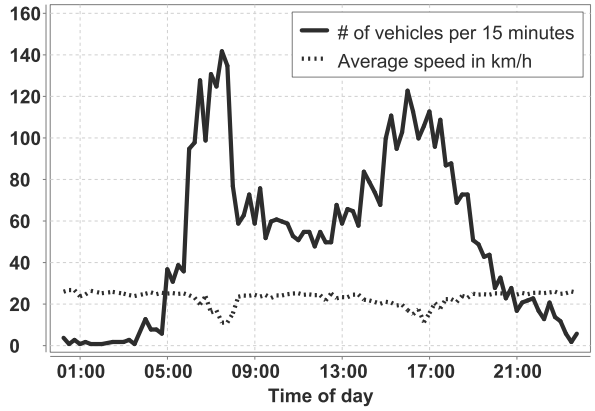

**DTA 10** (10 iterations of DTA)




**DTA 50** (50 iterations of DTA)




**CR4*** (Choice routing 4 routes)




**CR4* + DTA10** (Choice Routing and 10 DTA iterations)

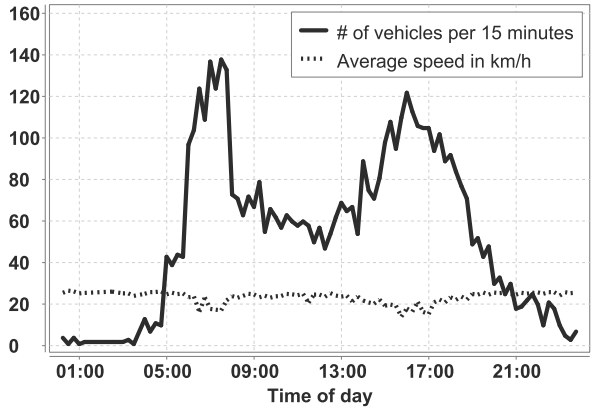



**Figure 7: Traffic flow at the defined sample location for the different route calculation approaches.**

| Approach | Vehicles Arrived | Avg. Distance | Avg. Speed | Avg. Travel Time | Avg. std. Deviation of Travel time | Avg. Waiting Time |
|---|---|---|---|---|---|---|
| F | 73,7 % | 1,58 km | 17 km/h | 2 070 sec | 1 348 sec | 1 751 sec |
| F* | 98,5 % | 1,62 km | 28 km/h | 246 sec | 52 sec | 73 sec |
| CA4* | 99,9 % | 2,03 km | 30 km/h | 249 sec | 60 sec | 61 sec |
| DTA (10 iterations) | 91,2 % | 2,46 km | 27 km/h | 1 070 sec | 870 sec | 730 sec |
| DTA (50 iterations) | 99,9 % | 2,01 km | 31,3 km/h | 245 sec | 56 sec | 42 sec |
| CR4*+ DTA (10 iterations) | 99,9 % | 1,76 km | 30,5 km/h | 215 sec | 34 sec | 51 sec |
| CR4*+ DTA (50 iterations) | 99,9 % | 1,69 km | 30,9 km/h | 203 sec | 31 sec | 45 sec |

**Table 1: Simulation results for the different route calculation approaches**

## 5.1 Fastest Routes (F)

Our evaluations show that using the simple fastest routes approach results in unbalanced traffic in our simulation scenario. Many vehicles share parts of the selected routes which quickly results in bottlenecks. Moreover, if no additional heuristics are used, vehicles are often navigated through residential areas which is rather unrealistic and causes heavy traffic congestion. In the worst case, this congestion spreads throughout the overall network which leads to long waiting times and a low average vehicle speed. As shown in table 1, each vehicle waits half an hour in average during its journey. Moreover, the average speed is very low and only 74% of all departed vehicles reach their destination within the simulation time. The overall network is congested in both morning and evening hours. Furthermore, vehicle flow and speed decrease at the sample location (figure 6 and 7) during the peak in the morning hours and do not recover due to the massive congestion in the network.

## 5.2 Fastest Routes with Heuristics (F*)

In our scenario, this method eliminates the problem of bottlenecks since the initial routes consider turn costs and avoid residential areas. In general, vehicles follow the main roads instead of taking the shortest way through residential areas, which prevents several bottlenecks. On the other hand, many vehicles still share several parts of their routes which is rather unrealistic and causes congestion on the used roads. In the analysed scenario, no persistent congestion is caused. During peak times, however, many roads still get congested due to missing alternative routes. In comparison to F, the average waiting time decreases to 73 seconds while the average speed for each vehicle increases to 28 km/h. Vehicle flow and speed at the sample location (figure 7) does not show congestion but only peaks in the morning and evening hours.

## 5.3 Choice Routing with Heuristics (CR4*)

Calculating several alternative routes reduces the problem that too many vehicles share the same road segments. Consequently, less bottlenecks occur. In the simulated scenario, the average waiting time for each vehicle decreases to 60 seconds and the average speed increases to 30 km/h. These results show that this approach improves the initial route generation even more. In this simulation scenario, only temporary local traffic congestion occur on traffic lights and the overall traffic flow is never blocked due to bottlenecks. Compared to F, vehicles have a lower average speed and no long-term congestion occurs at the sample location (figure 7).

## 5.4 Iterative Dynamic Traffic Assignment (DTA10 / DTA50)

In this approach, the route choice and the resulting traffic simulation is repeated until the user equilibrium is approximated. Consequently, the fastest route according to the traffic flow of the previous simulation is calculated for each vehicle. The higher the number of iterations of this process, the more vehicles adapt their routes to the traffic flow. Therefore, this approach shows the best results for all measures after 50 iterations. Average speed, duration and waiting time show better results than in CR4*. However, it can also be seen that the network still collapses within the tenth iteration, as seen in the deviation of travel times per trip. Also, massive congestion occurs at the sample location (figure 7) in the evening hours. These results show that iterative DTA without any optimization of initial routes requires a lot of iterations to obtain the (approximated) user equilibrium for our simulation scenario. Thus, using this approach is a time-consuming process due to the multiple runs of simulations and routing calculations.

## 5.5 Iterative Dynamic Traffic Assignment with Initial Routes (CR4* + DTA10 / CR4* + DTA50)

The main problem of the previous method is that initially, the simple fastest routes approach is used for the first route calculation which causes bottlenecks and a congested network (as seen in F). In order to obtain better results for the first iterations, this approach uses precomputed routes initially generated by Choice Routing with the same heuristics as in CR4*. Our simulation results show that this approach is more suitable for reaching the user equilibrium with less iterations. The tenth iteration already results in a route distribution which produces balanced traffic, as seen in low travel times and a low standard deviation of those. Also, there are neither huge traffic congestion nor other anomalies. Applying more than ten iterations does not produce a further improvement. Consequently, the (approximated) user equilibrium for our scenario setup seems to be reached in less iterations.

# 6. CONCLUSION

In this paper, different route calculation approaches for traffic simulators were compared regarding their generated traffic distribution and the resulting traffic flow. For our evaluations, we used an OD matrix of the city of Fulda (Germany) based on real traffic demand, and used different route calculation approaches in order to generate initial traffic for

comparable simulation scenarios. In this simulation study, the approach 'simple fastest routes without any heuristics' caused unbalanced traffic and resulted in an overall congested road network. Considering turn costs and distributing vehicles among alternative routes by using the 'Choice Routing' algorithm resulted in more balanced traffic. By using this approach, bottlenecks were avoided and, thus, no congestion occurred in the simulated scenario. However, this approach was not able to achieve a user equilibrium which represents a traffic distribution close to reality. To approximate the user equilibrium, we applied several iterations of a simple dynamic traffic assignment method. First, the initial routes were calculated by the 'simple fastest routes' approach. Here, many iterations were required until a user equilibrium was reached. In a second experiment, we used initial routes pre-calculated by 'Choice Routing' before an iterative traffic assignment approach was applied. Now, we were able to approximate the user equilibrium with less iterations.

Since the results are promising, we plan to integrate the best route calculation approaches into the simulation architecture VSimRTI [15]. Consequently, these approaches can be used for the route calculation of all traffic simulators coupled to VSimRTI.

## 7. REFERENCES


[1] R. Bader, J. Dees, R. Geisberger, and P. Sanders. Alternative route graphs in road networks. In *Proceedings of the First international ICST conference on Theory and practice of algorithms in (computer) systems*, TAPAS'11, pages 21–32, Berlin, Heidelberg, 2011. Springer-Verlag.

[2] R. Bellman and R. E. Kalaba. *Dynamic programming and modern control theory*. Academic Press New York, 1965.

[3] CAMVIT. Choice routing, 2006.

[4] M. Friedrich, I. Hofsaß, K. Nokel, and P. Vortisch. A dynamic traffic assignment method for planning and telematic applications. *PTRC-PUBLICATIONS-P*, pages 29–40, 2000.

[5] C. Gawron. An iterative algorithm to determine the dynamic user equilibrium in a traffic simulation model, 1998.

[6] R. Geisberger and C. Vetter. Efficient routing in road networks with turn costs. In *Proceedings of the 10th international conference on Experimental algorithms*, SEA'11, pages 100–111, Berlin, Heidelberg, 2011. Springer-Verlag.

[7] R. Geisberger and C. Vetter. Efficient routing in road networks with turn costs. In *Experimental Algorithms*, pages 100–111. Springer, 2011.

[8] P. Johnson, D. Joy, D. Clarke, and J. Jacobi. Highway 3. 1: An enhanced highway routing model: Program description, methodology, and revised user's manual. Technical report, Oak Ridge National Lab., TN (United States), 1993.

[9] L. Kadiyali. *Traffic Engineering and Transport Planning*. Khanna Publishers, 1983.

[10] M. Kuby, X. Zhongyi, and X. Xiaodong. A minimax method for finding the k best ŞdifferentiatedŤ paths. *Geographical Analysis*, 29(4):298–313, 1997.

[11] Y. Lim and H. Kim. A shortest path algorithm for real road network based on path overlap. *Journal of the Eastern Asia Society for Transportation Studies*, 6:1426–1438, 2005.

[12] D. Lohse. *Berechnung von Personenverkehrsströmen*. Wissenschaft und Technik im StraSSenwesen, 1977.

[13] D. Lohse. Ermittlung von verkehrsströmen mit n-linearen gleichungssystemen unter beachtung von nebenbedingungen einschlieSSlich parameterschätzung. Technical report, Technische Universität Dresden, 1997.

[14] C. of Fulda. Öpnv-nahverkehrsplan - 2. fortschreibung. `http://www.fulda.de/bauen/verkehrsplanung/oepnv-nahverkehrsplan.html`, September 2013.

[15] B. Schünemann. V2x simulation runtime infrastructure vsimrti: An assessment tool to design smart traffic management systems. *Computer Networks*, 55:3189–3198, October 2011.

[16] J. G. Wardrop. Correspondence. some theoretical aspects of road traffic research. *ICE Proceedings: Engineering Divisions*, 1:767–768(1), 1952.

[17] S. Winter. Modeling costs of turns in route planning. *GeoInformatica*, 6(4):345–361, 2002.

[18] J. Y. Yen. Finding the k shortest loopless paths in a network. *management Science*, 17(11):712–716, 1971.