ABSTRACT
Traffic routing is a well established optimization problem in traffic management. We address here dynamic routing problems where the load of roads is taken into account dynamically, aiming at the optimization of required travel times. We investigate ant-based algorithms that can handle dynamic routing problems, but suffer from negative emergent effects like road congestions. We propose an inverse ant-based routing algorithm to avoid these negative emergent effects. We evaluate our approach with the agent-based traffic simulation system MAINS²IM. For evaluation, we use a synthetic and two real world scenarios. Evaluation results indicate that the proposed inverse ant-based routing can lead to a reduction of travel time.

Categories and Subject Descriptors
I.2.11 [Distributed Artificial Intelligence]: Multiagent systems; I.2.8 [Artificial Intelligence]: Problem Solving, Control Methods, and Search

General Terms
Algorithms, Management, Measurement, Experimentation

Keywords
Traffic simulation, Multiagent-Based Simulation (MABS), routing, Ant-inspired

1. INTRODUCTION
Traffic routing is a well established research and optimization problem in traffic management [6]. Most research has been done for static problems, i.e., settings where the problem structure does not change. In static problems the routing decision boils down to find the shortest path between the start and the goal point. Once a solution has been found for all routes the optimal ones can be used whenever needed. These algorithms typically are based on shortest path algorithms, like the well known A* algorithm.

The situation becomes more complex if we regard dynamic problems. In a dynamic problem, the problem structure changes while solving the problem. For routing decisions this implies that not the traveling distance has to be optimized but the traveling time [1]. Of course, a simple approach is to assume a fixed average speed that can be used for every road and to use this in the calculation of the weights of the graph. It has turned out that this simple approach often can help, but it also turned out not to be sufficient for roads which load changes in time [13, 19].

One approach to handle this limitation is to gather data to enrich the routing graph with time dependent traveling information. Based on acquired data the traveling speed on a road is extracted and can be used during route planning. In this approach, the dynamic problem is reformulated to a static one which has a larger complexity than the initial static one, as for each edge in the routing graph, time dependent traveling speed information is available. But it turned out that the data acquisition takes a lot of effort [13, 19].

Using current trends and technologies like car-2-car communication [14] and autonomic road transportation support systems [8] cars can be enabled to communicate with each other, and also with their environment. Therefore, each car can be seen as an autonomous entity, that has computing
engines, so if a road is congested the following cars will take a
different route if the road ahead of them is blocked. But due
to the principle of following other ants in an ant-optimizing
algorithm, these situations will emerge regularly, given a sit-
uation with heavy traffic, as it can be often found in urban
areas. The emergence of congestions is a negative emergent
behavior [28].

Heavy traffic is typical for urban areas nowadays. But we
can also make another observation in these areas. There of-
ten exists a number of alternative routes, as well. In this pa-
per, we investigate how to handle dynamic routing problems
in urban areas, with high traffic and a number of alternative
routes. The goal of our research is to avoid the negative
effects of these emerging congestions while preserving the
positive emergent behavior of ant algorithms for routing in
dynamic environments. Therefore, we will change the inter-
nal reasoning of the cars to navigate to their destinations.
For the evaluation of our approach we use simulation stud-
ies based on an agent-based traffic simulation system, called
MAINS²IM. In our evaluation, we are comparing our ap-
proach with existing alternative routing approaches.

The rest of the paper is structured as follows. In the next
section we discuss related work especially from the fields of
traffic simulation and ant-based routing algorithms. Then,
in Section 3, we give a brief introduction into MAINS²IM.
Since we are focusing on routing problems, we will especially
highlight routing methods of MAINS²IM in Section 4. Our
approach of adapting the ant algorithm to make it more
suitable for urban areas, by avoiding negative emergent be-
havior, is described in Section 5. In the subsequent section
we evaluate our approach in a series of simulation studies
and discuss our results. Finally, we conclude and outline
potential future research.

2. RELATED WORK

In the first part of this section, we give an introduction to
related work in the field of traffic simulation. In the second
part we discuss current research in the field of ant-based
routing algorithms. Thereby, we focus in both subsections
especially on agent-based approaches.

2.1 Traffic Simulation

The modeling of traffic is a well established field, ranging
back to early work in the first half of the preceding century,
e.g., [17]. Since then, different models and (later) traffic
simulations have been proposed. The focuses of those range
from the simulation of huge scenarios, e.g., the road traffic
in Switzerland [27], using a cellular automaton based model
proposed in [22], to the simulation of very small areas (e.g.,
[5]) with high fidelity traffic models like, e.g., [30].

Traffic simulation systems consist of models for road user be-
havior, as well as traffic demand models and routing meth-
ods. The road model is typically encoded in form of an
notated graph. The users’ behavior is often described by
their capabilities, their goal(s), and their behavior patterns,
e.g., acceleration patterns. In this work, we focus on routing
methods.

Gehrke and Wojtusiak present an approach for inductive
learning of traffic predictions in relation to day, time and
weather which leads to a higher traffic flow in an agent-
based traffic simulation with agents using the predictions
for route planing [16].

Vasirani and Ossowski present an agent-based approach for
efficient allocation of road traffic network with help of an
artificial market [29]. The approach could be shown to be
efficient in a small scenario but has not yet been tested in a
traffic simulation system.

A crucial point is the explicit modeling of decision making
of simulated cars [2], respectively the reasoning in the sim-
ulated agents. Gawron presents an iterative algorithm for
route plan optimization towards an equilibrium [15]. In a
first simulation run, each simulated car chooses an optimal
route. Before the next run, a portion of agents re-plan with
help of the travel time knowledge gained from the previously
used roads. This is done repeatedly, until an equilibrium
condition is established. An analogous approach is used by
Raney et al. for TRANSIMS [26]. The simulation has to be
run for about fifty times in order to reach the desired state
in their scenario. Thus, this approach is very time-consuming.

Bazzan and Klügl present an agent-based approach for dy-
namic re-planning [3]. When a car agent perceives the
occupancy of the next road on its route plan to be higher
than $\tau$, a re-routing mechanism enables a to drive around
the potentially jammed area. The approach leads to a better
overall performance. For simulation, frequently re-routing is
time consuming. Another important aspect is at what time
of the travel the re-planning has to be done. When the next
road on the current route of a is jammed, a re-routing may
have been much better, if done earlier.

2.2 Ant-based Routing Algorithms

As pointed out before, ant-based routing algorithms have
been already investigated for traffic management [1, 4, 19].
In current work, researchers try to adapt the basic algorithm
to avoid the negative emergent effects outlined before. In
the following we discuss those approaches.

The approach by Alves et al. [1] is based on the equilibrium
theory of traffic networks, and therefore grounded in game
theory. In this approach, the cars are routed by a central-
ized traffic control management system. The central traf-
ic management system collects information about the load
dependent changes of the traveling times which are com-
puted by an ant colony optimization method. Therefore,
the ant-based routing is used to create information about
the expected load of roads, to adapt expected traveling time
information in the routing algorithm. In their approach,
cars are controlled; they have no abilities to make decisions.

The need for decentralized decision making in dynamic rout-
ing problems has been pointed out by Narzt et al. [24]. Simi-
lar to our approach it is assumed that cars become smart en-
tities that can communicate with each other and with their
environment. Cars act as ants and leave a pheromone trace
along the way they travel. In the presented approach the cars use the intensity of the pheromone trace to infer the traffic density on this particular section of the road ahead. Thus, based on this pheromone information the cars can adjust their local planning model and adapt the weights of the edges, used for their routing algorithm, i.e., A*.

The idea of inferring the current traffic situation from the pheromone trace has also been suggested by Bedi et al. [4]. In their approach, Bedi et al. discuss an approach for picking the next edge to travel towards the destination. For calculating the probability for an edge they combine three factors: the traveling distance, the pheromone strength and a random factor. Based on their description, the usage of their approach focuses the individual routing problem for a specific road user, who specifies its starting and end point. With this information and the traffic infrastructure different routes are computed iteratively, trying to minimize the probability that the car will get stuck in a traffic jam.

As previous authors Krömer et al. [19] also point out the need for handling dynamic routing problems for traffic routing. They also point out that collecting real world data, can be a complex task, that often has limitations in the amount of data and the area covered. Krömer et al. present an approach to avoid the negative emergent behavior in their routing algorithm. They modify the original ant colony optimization algorithm slightly, by introducing a probability threshold. If a probability of taking an edge becomes larger than the threshold, e.g., because it has a high pheromone mark, the probability will be cut to the threshold. Thus, the probability cannot become larger than the threshold. By adding this threshold, the authors are able to gain the advantages of the ant-based routing, but also have a mean to avoid that the shortest path gets blocked due to heavy usage. In their experimentation they used realistic road infrastructure and car behavior patterns. As expected the original ant algorithm found the shortest path faster, i.e., it could converge faster to a solution, but traffic jams occurred. This effect could be softened using the ant-based routing with the probability threshold.

3. SIMULATION SYSTEM

The routing algorithms discussed in this paper have been integrated into the simulation system MAINS²IM (Multimodal INnercity SIMulation). The simulation uses cartographic material from the OpenStreetMap¹ initiative in order to automatically generate a simulation graph, leading to an executable traffic simulation. The system is built on the base of the free geographical information system (GIS) toolkit GeoTools².

In order to set up a simulation, an OpenStreetMap (.osm) file is clipped into a user defined map section. The new file is split into logical GIS layers in relation to their type of geometry or for rendering purposes (e.g., landscape polygons, waterways, buildings, points, railways, routes and roads). In a first step, a basic graph data structure is calculated, which then is refined by several analysis and correction steps. The result of this transformation process is a graph with Edge-Informations (EI) representing roads and NodeInformations (NI) representing the connections between roads, taking into account urban traffic circumstances like, e.g., cross-walks, traffic lights, roundabouts, speed limits, numbers of lanes and priorities for the determination of the right of way.

MAINS²IM provides microscopic traffic models for cars (passenger cars, trucks and buses), as well as bicycles and pedestrians. The models are discrete in time and continuous in space. One simulation iteration corresponds to one second real time.

The road users in the simulation system are modeled by simple reflex agents. Each driver-car-entity has its individual driving capabilities, e.g., different dallying behavior, acceleration, maximum velocity or rating for safety distances. When a situation occurs where multiple cars prevent each other from passing a crossing due to the right of way rules, the involved agents are able to abstain from their right of way and let another one pass the crossing. When a driver-car-entity has to wait for a certain time in front of a crossing, it may re-plan and use another route to its original destination. This is done via usage of the A* search algorithm with prohibition of the next road of the original route.

The simulation is written in Java and can be executed on a workstation computer. Detailed descriptions of the simulation system as well as case studies can be found in, e.g., [20, 9, 10] and on the corresponding website www.mainsim.eu.

This work deals with a modification of current routing methods for traffic simulations. Thus, the next section describes the current routing approaches in MAINS²IM.

4. ROUTING METHODS

Currently, MAINS²IM implements three different routing methods. Two of them can be used to identify a specific route from a starting point to a goal and one can be used to generate probabilistic routing behavior from a specific starting point. The following subsections 4.1 to 4.3 describe the approaches and subsection 4.4 discusses the presented approaches.

4.1 Precalculated Routes

In order to precalculate all possible routes in the simulation graph, an all-pairs shortest path problem has to be solved. A reasonable approach would be the Floyd-Warshall algorithm [7, pp. 629-635]. It solves the problem in \( O(|V|^3) \). A distance function \( d(NI_a, NI_b) \) estimates the duration of travel on the edge \( EI \) between nodes \( NI_a \) and \( NI_b \). The algorithm has one shortcoming for the problem of traffic routing: It is not able to take account of the preceding edge on a path. Consider the situation shown in figure 1.

The Floyd-Warshall algorithm is not capable of suppressing u-turns or turns with an over sized turning angle \( \alpha \), which may be unrealistic. Another method is the repeated calculation of the Dijkstra algorithm [11], once for each \( NI \). Overall, this method also leads to computational complexity of \( O(|V|^3) \). During computation of the Dijkstra algorithm, our method not only stores the preceding \( NI \) on a way, but also the preceding \( EI \) and thus, overcomes the aforementioned problem.

¹http://www.openstreetmap.org, accessed 03/02/12
²http://www.geotools.org, accessed 03/02/12
The computation of all ways in the simulation graph leads to the challenge to store them in the simulation system. Consider the simulation of a medium sized city with about 5,000 NIs, leading to about $5,000^2 = 25 \cdot 10^6$ paths. An efficient method in space and time is needed to store those paths. In MAINS a collection of all NIs (NodeInformationCollection (NIC)) is implemented in form of a list data structure. The ith element of NIC represents the NI with id i. This leads to a simple lookup when searching for a NI with a given ID. The IDs are stored as integer values.

Each NI stores a list of IDs. The ith entry of the list is the ID of the next NI_next node on the way to a given destination NI_dest with ID i. The time complexity to lookup a path in the graph is $\Theta(n)$, where n is the amount of EIs on the way.

The space complexity of this method is about $|NIC|^2 \cdot 32$ bit (integer) leading to about 95MB to store all paths. Coming back to figure 1, it is obvious that the amount of different IDs stored in the lists is small for each list. Each NI therefore stores a lookup table (LUT) for the IDs of its neighbors, as shown in figure 2.

The input of the table is the number of the neighbor and the result is the corresponding ID. The integer data type with the smallest amount of bits in Java is used: byte (8 bit). The approach leads to a compression ratio of about 25% without noticeable loss of time.

Note that the procedure is performed three times, because each basic type of road user (car, bicycle, pedestrian) has different routing characteristics and thus needs its own routes. Considering the previous example, the required space is reduced from about 286MB to about 72MB. The calculations are performed once and the entire graph data structure with its computed paths are stored in an external file.

The method of only using precalculated routes is suitable for static problem settings. But if multiple cars are simulated, the problem will become a dynamic one, as traveling times are changed due to other cars. Thus, the approach of pre-calculating routes leads to traffic jams in main streets of the road map, because the shortest paths use the fastest roads which are frequently selected by many road users. In order to obtain more heterogeneity in route choice, a probability based routing is introduced.

### 4.2 Probability-based Routing

In this approach, the routing of simulated road users is done probabilistically. Road users traveling under usage of this method, may have a defined starting position, but the destination of travel is not predetermined. Traffic of this kind is valuable for background traffic, influencing simulated road users with calculated routes from a defined source to a defined destination.

The approach proposed in this section is a refinement of the method stated in [9]. Let $\Omega_{NI}$ be the set of EIs, connected to NI. Each NI of the graph stores a turning probability function $p_t(\text{EI}_{\text{curr}}, \text{EI}_{\text{next}})$, for a road user of type $t \in \{\text{car, bicycle, pedestrian}\}$ coming from $\text{EI}_{\text{curr}}$, giving the probability to choose $\text{EI}_{\text{next}}$ as the next EI for travel. The function $p_t$ holds equation 2 and 3.

\[
p_t(\text{EI}_{\text{curr}}, \text{EI}_{\text{curr}}) = 0 \quad (2)
\]

\[
\sum_{\text{EI} \in \Omega_{NI}} p_t(\text{EI}_{\text{curr}}, \text{EI}) = 1 \quad \forall t, \text{EI}_{\text{curr}} \quad (3)
\]

With a given $\text{EI}_{\text{curr}}$, a route through the graph can be obtained via repeated random selection of the next $\text{EI}$ from $\text{EI}_{\text{curr}}$. The function $p_t$ is computed with help of the pre-calculated routes from subsection 4.1.

Each $NI \in NIC$ holds counters for all types $t$ and each combination $\text{EI}_{\text{curr}}, \text{EI}_{\text{next}} \in \Omega_{NI}$, initialized with 0. The routes $\zeta(NI_{\text{start}}, NI_{\text{dest}})$ between all non-equal pairs of $NI_{\text{start}}$ to $NI_{\text{dest}}$ are identified as lists of NIs and EIs. Each route $\zeta$ is analyzed and at each $NI \in \zeta$ the corresponding counter for the connection between the current and the next EI is incremented. This is done for all types of road users. Afterwards, all counters are normalized, resulting in $p_t$ with the conditions shown in equation 2 and 3.

The determination of $p_t$ has a complexity of $O(|NIC|^3)$, because of $O(|NIC|^2)$ paths and a maximum path length of $O(|NIC|)$. During simulation, the computation of a path with $n$ NIs takes $\Theta(n)$.

Both described methods are not directly capable for respecting dynamic routing features. Thus, the next subsection discusses a method based on the well-known A* search algorithm.

### 4.3 A*-based Route Determination

The A* search algorithm is suited to solve the single-pair shortest path problem in graph data structures. Nevertheless, it takes more time to calculate a way using A* search online than by the other methods described above.
A* search works similar to the Dijkstra algorithm, but uses a few heuristics to speedup computation, without loss of accuracy. The assumed distance \( d(NI_a, NI_b) \) may be adjusted arbitrarily during runtime of the simulation, enabling for dynamic modification of the routing methods. Different kinds of simulated road users may favor different types of roads, e.g., one may prefer motorways, the other country roads. This feature brings more specific characteristic behavior to road user plans on the cost of computation time.

4.4 Discussion

We have outlined three different approaches for planning routes for simulated road users, that have already been implemented in the MAINS2IM system. The first assumes static travel times for each \( EI \) and leads to traffic overloads on major roads. The second method is a more dynamic approach, but without exact steering capabilities, although the \( p_t \) may be adjusted using a point of attraction, as shown in [9]. The third approach is appropriate for dynamic routing with exact start and destination points, but will be noticeable slower in computation time in large scenarios with thousands of nodes in the graph.

The next section discusses a modification method for the edge weight function \( d(NI_a, NI_b) \) in order to overcome the overloading problems of the precalculated routes.

5. ANT-INSPIRED ROUTING

The idea of ant-colony optimization is to mimic strategies observed from real ants. Ants leave behind trails of pheromones, e.g., when looking for food. When an ant has to decide, which way to choose, it chooses the way with the higher pheromone concentration more likely than the others. These pheromones have a given, typical linear vaporization rate, i.e., the trace becomes weaker in time. This leads to the emergent effect, that short routes are found in the environment. A survey is provided by Dorigo and Blum [12]. It is obvious that this algorithm can be directly applied for routing decisions, e.g., in traffic simulation, as done here.

Our approach uses a simplified inverse ant colony optimization. It is inverse in the sense that a strong pheromone trace will influence following cars not to follow their predecessors but instead to avoid this road, taking a different route to their goals.

The basic idea of this ant-inspired routing is that each road holds a “smell intensity” \( si \). The pheromone trace is used to indicate the previous usage of a road. The higher \( si \), the slower a car will be able to drive on the corresponding road, since the usage of the road is higher, which will slow down the traffic on this road. Each \( EI \) holds two values \( EI^{si}_a \) and \( EI^{si}_b \) for the two possible directions of travel on \( EI \) (with direction a: in the course of road, direction b: contrary).

Let \( length(EI) \) be the length of the corresponding road in \( m \) and \( EI_{vMax} \) the speed limit on the road. The distance estimation function \( d(NI_a, NI_b) \) between two nodes \( NI_a \) and \( NI_b \) over \( EI \) in direction \( dir \) is adjusted by a modification of the estimated travel velocity on \( EI \), as shown in the following equation.

\[
d(NI_a, NI_b) = \frac{length(EI)}{v^*} = \left(1 - EI^{dir}_{si}\right) \cdot EI_{vMax}
\]

The domain of \( EI_{si} \) is \([0 \ldots 1]\), leading to a maximal estimated travel velocity \( v^* \) of \( EI_{vMax} \) when there is “no smell” and a minimal \( v^* = 0 \), when there is a high “smell intensity”.

In each simulation iteration, each \( EI \) has to adjust its values of \( EI_{si} \), as shown in equation 5.

\[
EI^{dir}_{si} = \min\left(\max\left(\kappa + \vartheta \cdot dens\left(EI^{dir}\right) , 0\right), 1\right)
\]

The value of \( EI^{dir}_{si} \) is absorbed by the subtrahend \( \kappa \) and increased by the traffic density in the direction \( dir \) on \( EI \), scaled by \( \vartheta \). The result is bounded to the interval \([0 \ldots 1]\).

A road with high traffic densities results in high concentrations of \( EI_{si} \) and thus influences cars to avoid the travel over \( EI \) when calculating a route with help of the A* search algorithm. This should result in more uniformly distributed traffic loads upon the road graph and thus shorter times of travel, because of less intense traffic jams. We expect that this approach is especially useful in areas with a number of different routing alternatives, and with a high traffic density. Therefore, we consider this approach in particular useful for urban areas, in which routing becomes especially important.

We believe that this approach can be valuable if the “pheromone traces” can be stored in the environment, e.g., by the underlying traffic management system. Cars that pass the roads can then read and update the pheromone traces.

6. EVALUATION

The evaluation of the routing methods described above begins with an optimization of the parameters \( \kappa \) and \( \vartheta \). Afterwards, the ant-inspired routing method is evaluated on a synthetic graph and on real world cartographic material.

6.1 Parameter Optimization

The optimization of \( \kappa \) and \( \vartheta \) is done with the method Simulated Annealing (see, e.g., [22]). We use Simulated Annealing, because of its good suitability for complex problems and its ability to avoid being stuck in local optima. The fitness of a parameter configuration is determined in the urban scenario shown in figure 3.

The performance of each parameter configuration is estimated by the average fitness of five replications. The amount of simulated cars is held constantly at 400. Whenever a car reaches its destination, a new one will be generated. Source and destination points are chosen randomly. After a settlement phase of 900 simulation iterations, a measurement phase of 3600 counts the amount of cars \( fit_r \), that have finished their travel in replication \( r \).

The fitness of a setting is

\[
fit = \frac{1}{5} \sum_{r=1}^{5} fit_r
\]

As the optimization problem is formulated as a minimization problem. This fitness function is an implicit estimation of the driving velocities of simulated cars, because the higher
the amount of finished cars, the more efficient the routing and the higher the average driving velocity.

The best parameter configuration with $fit = -4660.2$ in the described experiment was:

$$\kappa = 0.28077122867684745$$  \hspace{1cm} (6)

$$\vartheta = 2.8138856401002688$$  \hspace{1cm} (7)

6.2 Comparison of Routing Methods
The determined parameter configuration is used for a comparison of the ant-inspired approach with the method of A* search without enhancements and an iterative plan optimization approach.

The method of iterative route planning performs the simulation of identical cars several times and an amount of 10% of cars/agents is allowed to adjust its routing in each run, according to the travel times the agents have experienced in the preceding simulation runs. The method leads to a dynamic user equilibrium [15, 26], also discussed in Section 2. The training phase for this method is set to 50 replications, in order to enable the simulated cars to gain simple knowledge about the traffic conditions in the simulation area. The following experiment uses the iterative route planning approach for comparison. The first experiment for comparison is done in a synthetical scenario, followed by two experiments on real world road maps.

6.2.1 Synthetical Graph
For a first test of the obtained values, a highly dynamical experiment is used: A square lattice graph with $6 \times 6$ NIs. Each EI has a length of 250m and $E_{\text{length}} = 13,58 m \cdot s^{-1}$.

Each NI with four EIs holds a traffic light, as shown in figure 4.

For evaluation, we generate 100 different settings with randomly generated start and goal positions for road users. Due to the stochastic nature of the used behavioral model, each setting is repeated with ten replications.

A car which enters the simulation, starts with velocity $0 m \cdot s^{-1}$. It waits for a sufficient gap in traffic and then literally enters the road and begins acceleration. In the beginning, every simulated car stands still.

The result of each run is the average travel time of all cars, that have reached the destination after the settlement phase of 900 iterations. The measurement phase has a duration of 3,600 iterations and is extended until the last car has reached its destination.

Due to the stochastic nature of the used simulation model, the result of a setting is the average value of the results from its replications. The amount of cars is held constantly for the first replication and identical copies of the cars with the same start and goal positions are used for further replications with other seed values for the random number generator.

The average travel times per run $\bar{t}$ are compared. For the amounts of cars (200, 300, 400), the ant-inspired approach leads to significant reductions in travel time in comparison to the A* method and exhibits lower spreadings. The values for $\bar{t}$ increase with increasing amounts of cars. In the experiment with 500 cars, the ant-inspired approach suffers from high-value outliers for $\bar{t}$, even though it still leads to the lowest travel times for 49 out of 100 runs. This indicates an problem at very high traffic densities, potentially

\footnote{The statistical software R [25] is used for determination of significance with help of the t-test using error level $\alpha = 0.05$.}
order to deliver an insight on the distribution of traffic for (d) to (f) scales the values for each separate approach in order to show an overall comparison. The second row (parts (d) to (f)) scales the values for each separate approach in order to deliver an insight on the distribution of traffic for the individual approaches.

Parts (a) to (c) of figure 6 show, that the maximum of road usage is dominated by the A* method and the ant-inspired iterative approach produce less intensive traffic intensities in these areas. Parts (e) and (f) show that the ant-inspired iterative methods lead to a wider absolute spreading of traffic in comparison to the pure A* based route planning, shown in part (d). Again this goes in line with our expectations, that the inverse ant-based algorithm can avoid the negative emergent behavior of road congestions, since the traffic is more balanced on the different routes of the road network. This effect gets facilitated if multiple alternative routes exist.

The comparison of the different methods took place in the same graph, the optimization was done in section 6.1. The next step is to take the methods to another road map in order to test for overfitting of parameter optimization.

### 6.2.3 Medium-sized City
The medium sized city Hanau am Main (89,000 inhabitants) with a total length of roads 548km is used for a second experiment. The resulting graph has 4,201 NIs and 5,758 EIs. The experimental setup remains identically to the preceding experiment, except that on the one hand, the amount of simulated cars is increased. On the other hand, the number of start-goal settings per traffic density is reduced to 50 due to the increased computational complexity of this scenario. Table 3 shows the experimental results.

Table 3 exhibits differences to the results of Table 2. The iterative routing mechanism leads to the lowest values of \(\bar{t}\) at high traffic densities. The ant-inspired routing approach is not beneficial for this scenario. It performs comparable to the basic A* mechanism. This could be an indicator for parameter overfitting for the parameters \(\kappa\) and \(\vartheta\) to the road network.

Table 3: Average travel times \(\bar{t}\) and standard deviation \(\sigma\) in graph for road map of Hanau.

\[
\begin{array}{|c|c|c|c|c|c|}
\hline
\text{#cars} & \text{#runs} & A^* & \text{ant} & \text{iterative} \\
\hline
500 & 50 & \bar{t} = 540.12 & \bar{t} = 554.38 & \bar{t} = 560.13 \\
& & \sigma = 4.55 & \sigma = 4.25 & \sigma = 3.62 \\
750 & 50 & \bar{t} = 561.45 & \bar{t} = 568.24 & \bar{t} = 611.66 \\
& & \sigma = 10.46 & \sigma = 16.37 & \sigma = 6.87 \\
1000 & 50 & \bar{t} = 800.22 & \bar{t} = 809.08 & \bar{t} = 683.88 \\
& & \sigma = 32.62 & \sigma = 4.48 & \sigma = 3.72 \\
1500 & 50 & \bar{t} = 1109.39 & \bar{t} = 1101.61 & \bar{t} = 948.37 \\
& & \sigma = 14.76 & \sigma = 22.05 & \sigma = 20.91 \\
\hline
\end{array}
\]

In order to determine the effects of the different routing methods, the results of one run are used for calculation of a road usage map. In each simulation time step, each car increments the road usage value for the EI it currently drives on. Figure 6 shows the comparison of the different routing methods on the map extract of figure 3. The first row (parts (a) to (c)) takes road usage values for all three approaches for determination of the minimum and maximum values in order to show an overall comparison. The second row (parts (d) to (f)) scales the values for each separate approach in order to deliver an insight on the distribution of traffic for

### 6.3 Discussion
The evaluation has shown that the ant-inspired routing method can lead to lower average travel times of simulated cars in a synthetic scenario (section 6.2.1), as well as in a real world simulation graph (section 6.2.2). Section 6.2.3 did not show advantages of the ant-inspired method over the A* based or the iterative routing approach. However, the iterative approach has been integrated as reference only.
and not for direct comparison as it is based on individual “knowledge” of agents.

The ant-inspired method leads to a wider distribution of traffic in the simulation area. This is the basic outcome of the iterative approach. The ant-inspired routing approach has the advantage that a distribution of traffic is done without calibration runs. The amount of about 50 calibration runs, before performing the actual simulation runs is expensive and can be avoided by the approach presented in this paper. Nevertheless, it has to be mentioned, that the calibration with 50 runs is not always necessary. It would be better to do the calibration until the agents plans do not change any more, as discussed in literature.

In contrast to the approach, presented by Narzt et al. [24] our approach is not explicitly considering re-planning during travel. This is due to the focus of MAINS²IM, namely the simulation of whole cities, taking account of multi modal traffic. For such large-scale scenarios, the simulation agents must not be very complex with respect to computational effort.

Our approach utilizes the effect that traffic jams resolve faster, when the amount of cars filling the waiting line behind the beginning of the jam decreases [21]. This is achieved by the ant-inspired routing mechanism as newly planned routes avoid road segments with high traffic.

7. SUMMARY AND PERSPECTIVES

In this paper, we have discussed the need for future routing approaches, especially in urban environments where a number of alternative routes exist and traffic is dense. In these environments a dynamic routing algorithm could help to reduce travel times. We have focused on routing approaches using the ant-optimization paradigm. While ant-based optimization algorithms offer a number of interesting features, like fast convergence to the shortest path, and self-stabilization in case of disruptions, there are some negative emergent effects. In particular this kind of optimization will produce congestions, as ants follow the most intense pheromone trail, which leads to overload situations on the road network.

To maintain the aforementioned positive effects of ant-based routing algorithms we have modified the conventional ant-based optimization approach, to have a balancing effect in the road selection. This is done with help of a simple modification of the distance function of the A* search algorithm. We have tested our modified routing algorithm in the agent-based traffic simulation system MAINS²IM. Within this simulation system we could show the positive effects of our modification in a grid network. Since MAINS²IM is able to generate a simulation model out of publicly available map information, we are able to transfer these results from a theoretical setting into settings based on real world map excerpts. The method’s parameters have been optimized for a small scenario and benefits for another scenario have not been observed. This could indicate parameter overfitting and needs to be investigated in the future.

By using an agent-based traffic simulation system, we are able to model each road user, with a different set of attributes, leading to different characteristics, which allows us to investigate richer models. In our particular research we have applied local reasoning capabilities that cars can have in taking the routing decisions locally. This goes in line with the trend that cars become smart active entities in the
Figure 6: Comparison of road usage ratios (white: low usage; black: high usage). First row: Same leveling for all methods, second row: individual leveling for each method.

overall traffic management infrastructure.

The smell intensity values of the simulated roads are dependent on the time dependent traffic densities of the corresponding roads. The transferability from density to flow needs to be investigated in the future, because it is easier to measure traffic flow than traffic densities in real traffic scenarios. One aspect for future work could be to identify a set of good measurement points to assess the traffic situation. While the simulation allows us to have an easy access to the overall traffic situation, this is not possible for real traffic management systems. Those systems often have only a limited view, defined by a set of measuring and monitoring points. Based on this local view a global view has to be estimated. Therefore, the identification of measuring points becomes of special interest, to get a good estimation of the global state by local observations.

The described approach has not lead to benefits in a medium sized city. This may have two reasons: overfitting of the parameters for the small scenario or too long-ranging agent plans. If overfitting was the actual reason, the approach would not necessarily lead to advantages over the iterative routing approach with respect to computational time as situation-dependent calibration would also be needed. All mentioned routing approaches base on the offline A* search algorithm. Future research needs to investigate its use for real-time path finding algorithms like RTA* [18] in order to use the dynamics of the pheromone concentrations during the trips of the simulated cars for a dynamic routing with replanning. This should lead to better results in huge scenarios.

The ant-based routing can be studied in the simulation setting to investigate its potentials. As we have outlined before it bases on assumptions about ongoing technology trends, and therefore could be used also for travel time optimization in real world scenarios. The method needs to be investigated for robustness against influences from pedestrians and bicycles and local public transport, in the future. The effect on gas consumption and CO₂ emissions can be identified with MAINS²IM and thus, will be a field of study for further investigations of the ant-based approach.

8. REFERENCES


Acknowledgement

This work was made possible by the MainCampus scholarship of the Stiftung Polytechnische Gesellschaft Frankfurt am Main.