RENDEZVOUS-PROBLEM IN LOCAL SHARED-RIDE TRIP PLANNING

Radoslaw Rudnicki, Karl-Heinrich Anders, Monika Sester

Institute of cartography and geoinformatics
Leibniz University of Hanover
Appelstraße 9a, 30167 Hanover, Germany -
(radoslaw.rudnicki, karl-heinrich.anders, monika.sester)@ ikg.uni-hannover.de

Commission IV, WG IV/6

KEY WORDS: Mobile GIS, Network Analysis, Aerial Survey, Simulation, Navigation, Optimization, Location based Service

ABSTRACT:

In today's society mobility plays an important role. The impulse to locomotion arises from the wish to participate in fundamental social systems, like education and work. One challenge for the future will be to handle the problems resulting from traffic. For the near future enhancing the efficiency of already existing traffic can be part of the solution and gives time to search for further solutions. This paper represents a concept for local shared ride trip planning and shows the results preserved by a simulation software that implements this ideas. It explains how a scalable solution can look like, which requirements exist for a communication protocol and which algorithms can be used for the routing problems within a shared ride planning system.

1. INTRODUCTION

Traffic is a growing problem in cities and on motorways. Cars are the preferred vehicles of transportation, they are used in 73.9% for urban journeys (Bratzel & Tellermann, 2007). The travellers enjoy a high level of comfort, they can reach nearly every place and save time compared to public transportation. But this entails negative ecological, economic and social effects: Traffic is jointly responsible with about 20% for the carbon dioxide emissions worldwide. The pollution has a direct negative influence on quality of life. Economical problems result from traffic jams and accidents. Beside the personal tragedy, costs are generated due to loss of production of injured people, emergency medical services, hospital expenses, material loss, courts of law and insurances. The costs caused by accidents were 114 billion Euros in Germany in 2004. Another 100 billion Euros financial loss was caused by overloaded traffic infrastructure and resulting traffic jams (Bratzel & Tellermann, 2007). Other than that mentioned problems noise, for example, is a further social burden. Nowadays 550 million vehicles are registered, thereof 150 million just in the USA. China expects 190 and India 80 million cars on their roads in 2035 (Rauch, 2007). So, in the near future the problems will increase.

This increasing demand for mobility leads to an increasing amount of traffic. The paper presents a concept that exploits the current infrastructure in terms of public transport and also includes private cars into that system. Individual cars are considered as additional means of transport available for everyone in the spirit of a shared ride trip. Both the challenge and the advantage of this system is its *locality*: a traveller (client) with an ad-hoc plan to go to a certain place needs a car (host) that is in the vicinity and offers a ride. Thus he or she has to communicate the travel plan in the local environment, and only hosts that are there will be able to make an offer. We present results achieved with a simulation software in terms of economy and efficiency of the system.

The idea of shared ride trip planning can be considered a special case of mobile geosensor network. It has been addressed on a conceptual level by Winter & Nittel (2006). In a recent work, Raubal et al. (2007) used a multi-agent simulation software to implement and test the system with realistic environments. In Wu et al. (2008) different types of agents are studied in a peer-to-peer based shared ride trip planning system.

2. APPROACH

The idea to reduce the problems made by traffic is to increase the efficiency of the current used transportation system. On the one hand there is the *space efficiency*, which is an indicator for how good the capacity of roads is used. On the other hand the *time efficiency* reflects the best possible time for a journey in relation to the real needed value. In the majority of cases drivers travel alone although an average car has five seats. So the space efficiency is only 20%. Getting the travellers to drive together could enhance the space efficiency enormously. More difficult is to get a better time efficiency. By using carpools the number of cars would decrease, which results in less traffic jams and improving the time efficiency, but there is no guarantee for every person to get quicker to his or her destination, even though the average case is typically better than without this ride sharing.

Both, host and client can profit by sharing rides. A host can share its costs or even make some money when it transports enough clients on a route. The client does not need to use its own vehicle, but nevertheless can make the journey in comparatively short time. Public transportation can be incorporated in this system, so even though a client cannot find a good host, the use of a bus or tram is still given. In the concept public transportation has some specific characteristics: the routes are fixed both in connection and time; and also the possible meetings points are limited to the bus/tram stops.

To realize such a system, there are some requirements to be considered. First of all, it should be easy to use, cheap, reliable, and scalable and can be used spontaneously. There should not be the need for much lead time to join this special service. Each traveller can be a host, which offers a transportation opportunity, or a client, which is looking to go with a host. These two types are, however, not disjoint: It is conceivable that a participant transports some clients and can be taken along by a bigger vehicle. Figure 1 shows an example.

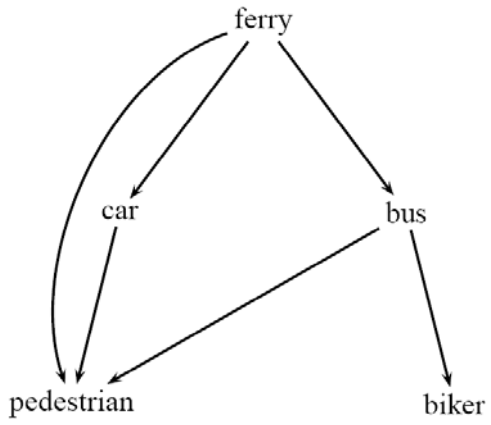

Figure 1: Hosts and clients

To meet all requirements, the idea is to decentralise the communication. There is no need for an expensive centralised infrastructure. A further advantage is that there is no communication bottleneck, which leads to a better scalability. The hosts and clients use radio communication in an ad-hoc manner. Due to the limited range of transmission, the communication can only take place in a local area. The communication soft- and hardware can be integrated into a mobile or smart phone. So every participant brings an own communication node into the net. These nodes are dynamic, they can move, appear or disappear.

With the help of a rule-based constraint system, the travellers have the opportunity to formulate their conditions to travel with somebody or when they are not going to do this. Once these conditions are initiated, the communication devices can exchange them automatically during a communication process to negotiate whose views of a travelling partner fit together. Among these social criterions, of course the routes have to match at least to some degree.

The attractiveness of a host grows with the time the client can save by going with it. If traffic density is high enough, the client can assume to find a matching host for its journey quickly. In the simulation we will show, which density of hosts and clients has to be given in order to make this system attractive.

3. COMMUNICATION

As already mentioned in section 2 there is no central node for communication. This means that the whole net intelligence has to be put in the nodes, i.e. in our case the clients and the hosts. All necessary algorithms must be implemented in the end devices. The knowledge about the clients and hosts information is distributed over all nodes. No single instance has an overall picture of the situation. The communication burden is also distributed over all nodes and not bundled in one point.

Connections between nodes can only exist when the distance to each other is smaller than the radius of the communication range. To assure the process of communication the range is assumed to be constant for all nodes, otherwise the bi-directional communication may not work. A multihop communication process is not needed. Winter and Nittel (2006) showed that the probability to find some new interesting connections is very small, but the number of messages explodes rapidly and floods the net by using multihoping. In summary, the limitation of the range because of the restricted power resources is not a serious constriction. For a client all relevant hosts are close to it or will cross the communication area in the course of time as long as the range $r > minRange$, whereas *minRange* depends on the traffic density and infrastructure. In most cases, other hosts are too far away and moving in a non-interesting region from the client's point of view.

Clients willing to use the service are periodically sending a query to their neighbourhood as long as they did not find a host for transportation. If a client finds a host, it stops sending queries and begins again when it reaches the drop position, where the client gets off and leaves the host. From this position the client again looks for an optimisation of the travel time for the rest of the distance to the client's destination.

If two travellers, a client and a host, have already done the negotiation without finding a good solution, the host stops to answer the queries from this client. This way they reduce communication traffic and precious processing time. There is no need to negotiate again only when the host changes its route. If host and client agree in all points, the host keeps on answering to queries from other clients as long as it has free transportation capacities. The host stops communicating when all seats are booked and restarts it as soon as one client gets off. No traveller has a picture of the whole situation, so it is possible that a client finds only a local best solution. But in the moment of booking a host's seat this is the best known host and so the booking is binding for both sides.

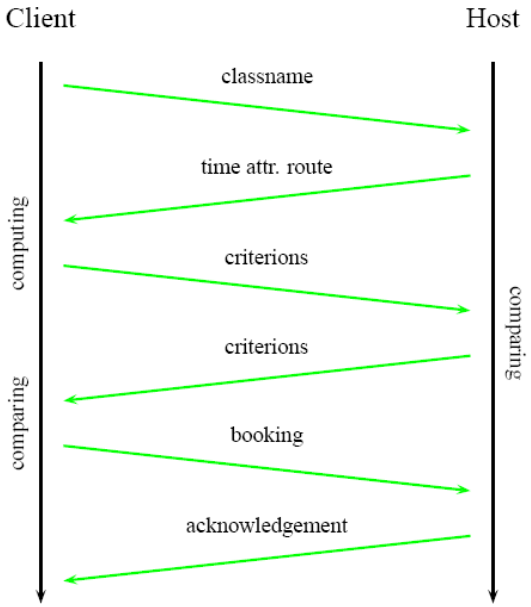

Figure 2: Successful booking

A communication protocol describes which information are send by the client to start a query looking for a host and which information the host gives back (Figure 2). The aim is to send as few as possible information and give the nodes just few important calculations to save resources. To achieve this aims a query begins just with a simple information: a unique id of a traveller and its class name is transmitted by the client. The client that has less battery power than a typical host, therefore does not send its full routing information. That is important, because transmitting information needs considerable more power than calculating data. Therefore the clients have the task to analyse the routing information from the hosts (section 5). Only hosts which are able to transport the requested class need to answer the client. The host sends time attributed routing information the client, which includes approximations about when the host reaches which route point. So the client has the chance to see if it is possible to reach the host's route in time to go with it. According to our system design, a host does never needs to change its route for a client. If the client finds a point to get in which is reachable in time, it looks for a point to get off and calculates the saved time. If there is no point to jump in or get off, the client cancels the conversation with these hosts. With the remaining hosts an exchange of criteria for a common tour takes place. After calculating and comparing all information for all leftover hosts, the client can book the best host's seat, by transmitting time and rendezvous point and cancels all other hosts.

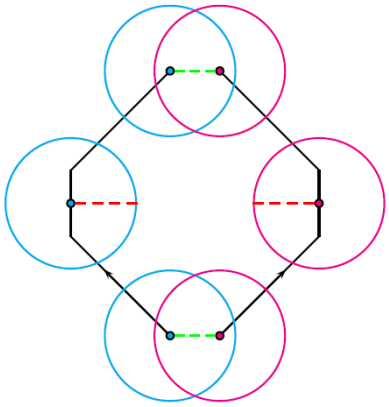

Figure 3: Communication time problem

Because hosts and clients are moving while communicating, it is thinkable that they leave the communication range during conversation (Figure 3). So a timeout is required to protect the communication against dead locks. In such a case, both partners delete their information about each other, because it is not sure if they come closer again. Hence it is not necessary to save the intermediate results and the memory can be saved. When host and client meet again in a communication process, they start the negotiation from scratch.

4. GEOSPATIAL DATA

For a correct functionality of the system all travellers need the same data. Due to the fact that this is not realistic, perhaps they have maps (route information) from different segments of a map or different map dates, a peer-to-peer spreading of time stamped maps is conceivable. During the communication process the conversational partners can exchange their map information. So the travellers keep themselves up to date.

5. ROUTING

This section presents the way finding algorithm of the travellers and how clients find out which host helps them to save most time. Of course all calculations should be as fast as possible for efficient use.

The Dijkstra algorithm calculates the shortest paths between a start node and any end node (Dijkstra, 1959). Actually it is possible to use different cost models as long as the values are positive. The only problem is the algorithm searches in all directions, which can cost needless time even though Dijkstra is applicable. The A*-algorithm extends Dijkstra with a heuristics for the search direction (Hart et al., 1968, Hart et al., 1972 Pearl, 1984). If the approximation of the left distance from one node to the destination is smaller or equal than the real distance, this algorithm finds also the optimal route. For example, when using air distances this requirement is achieved. So A* is appropriate for the efficient way finding. Because of the knowledge of every traveller about its own speed, it is even possible to compute a time approximation for every node of the route, if nothing unpredictable happens. For the simulating (section 6) this approximation is reliable, in reality it is much harder to do it right and the approximations should be updated from time to time.

Of course a client takes advantage of the higher speed of a host. A model with hosts which are slower than the clients brings hardly a benefit. The only way to profit in this case is that the client can take an abbreviation which was unreachable by itself, i.e. to pass a river in the case the host is a boat.

Clients always look for hosts to reduce travel time. So there is an upper limitation for the desired arrival time at this point. If a client needs a longer time to its destination going with a host than on its own, this host is not appropriate. The easiest case is that the client finds a host that is going exactly the same route and picks up the client. Even if the host goes just a part of the client's route, it is time profitable. But in many cases it can be necessary for the client to leave its planned route to get to the host's route and wait for the opportunity to go with this host. This can go so far, that the client moves in the opposite direction of its destination to get to a host's route. This can lead to a local worsening of the travel time to the destination. If the host can recover the lost time it can still be a time profitable way. Hence, not every detour is useless. Furthermore, it is imaginable that the client has to wait at the rendezvous point for the host, but still this detour and waiting time can be profitable in global consideration.

So all combinations of waiting and changing route are allowed as long as after getting off of the host, the client is in a position to reach its destination in a shorter time than without this shared trip. Even if the host does not drop the client at its destination point, with increasing density of hosts, the client has the chance to find a new host for the rest of its route and save time again.

5.1 Rendezvous-point

Before a client can calculate how much time it can save by going with a host, the client needs to know if it is possible to reach one point of the host's route before the host itself passes this point. As shown in Section 3 the client gets a time attributed route directly from the hosts. It would be easy, but not efficient, to try all route points and see which one can be reached earlier than by the host. To avoid too much computing,

the client is only interested in the nearest points of the host's route to its own position. In addition, the client just needs to know only one potential rendezvous point and the time benefit does not depend on this point at all. For an easy comparison of the suitability of the host's route points, the client calculates the air distances to them. Then the client detects the local minima and sorts the distances according to size (see Figure 4). A local distance minimum is given when the previous and next distance are greater than this distance. These points are marked green in the illustration. The highest probability to reach the host's route is when trying to reach the nearest route point. If this point is not reachable the client tries the next point from the sorted list. If the client finds a rendezvous point it breaks the further search.
Figure 4 shows an example where the client finds a rendezvous point at t8.

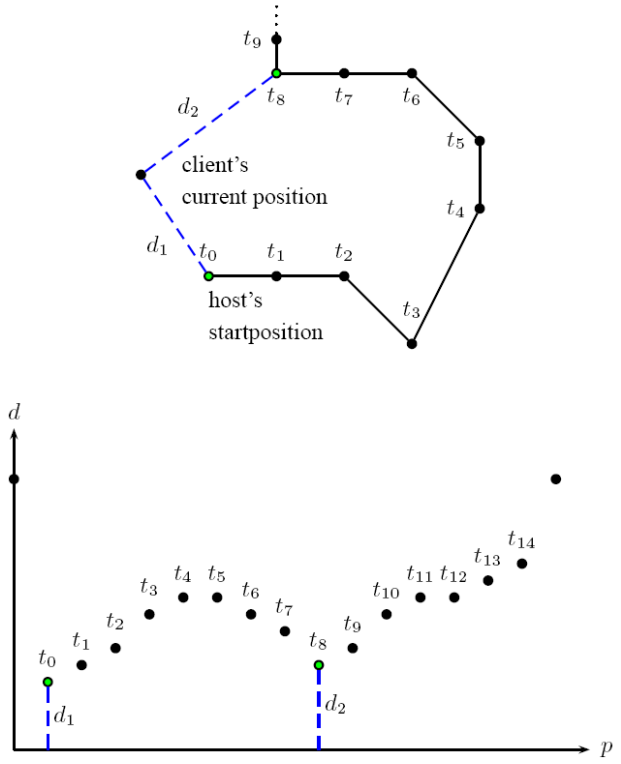

Figure 4: Determination of Rendezvous point

The point at t_0 is unreachable as the host is starting at this point and it is impossible for the client to be there before the host passes this point. So the nearest point is not necessarily the best one. The waiting time is the difference between the time when the host passes the rendezvous point and the time the client reaches it. So the client is able to find the rendezvous point easily and efficiently without trying all points.

5.2 Leave Point

To find out where to leave the host's vehicle, the client uses a similar algorithm as for the rendezvous point. Again the client calculate distances, this time from its destination point to the host's route beginning at the point the client got in. For each local minimum distance from the route to the destination point, a time attributed route is computed. Hence, for the client it is possible to qualify the destination time from every local minimum, if it does not find any other rides to go with. The difference between arrival time and original one is the saved time.

For the get out point it is not adequate just to find only one position. The distance gives no hint about the quality of the point for getting out, so it is not necessary to sort the local minimum distances. Figure 5 shows an example in which neither the nearest nor the farthest local minimum is the best get out point.

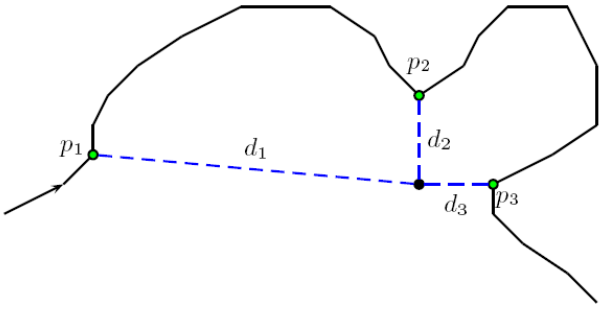

Figure 5: Determination of leave point

From Point p_1 it is a far way to the destination. If the client would get out at this position, given that no other host take this client with it, there would be no time saved. Point p_3 is the nearest point from the destination. But it is reached relatively late, as the host's route makes a big arc after p_2. Assuming the client would get off at p_2, it could use the time, comparing to stay with the host until achieving at p_3, to go into destination's direction. If $t_3 + t_3' > t_2 + t_2'$ whereas t_n is the time the host needs to achieve at point p_n and t_n' is the time the client needs to achieve the destination from t_n, p_2 is the better point to get out for the client. So in this case the algorithm does not stop when it finds a get out point in comparison to the algorithm searching for the rendezvous point.

For the detection of potential points for get in or get off the air distances are used as a heuristic to accelerate the calculation. For compact road networks the assumption that air distances and real route distances are similar is acceptable. In less compact road networks and in some special cases, the rendezvous points are not optimal, but relatively close. That is the case when an obstacle, like a river, where the client cannot go through, is between the target points. This case is called the moat problem, but can mean any other obstacle, too. The following Figure 6 illustrates the situation.

The shortest distance between the host's route and the client's destination D crosses an obstacle. Getting out at this position forces the client either to go back or to move to the next overpass. In the shown situation it would make sense to get off at overpass_1, because the distance from overpass_1 to D is the same as the distance from overpass_2 to D. So the earlier the clients get out, the earlier it reaches the destination. The algorithm for calculation of the distance minima does not consider any information about overpasses or obstacles. This tentatively leads to the result that the clients do not get off at the optimal position every time. Nevertheless in most cases they are very close. The algorithms described above can handle effectively the rendezvous problem and where to get out again even if the host's and client's route do not match.

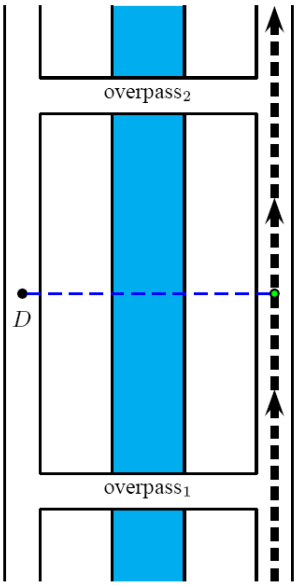

Figure 6: Moat problem

6. SIMULATION

To analyze the influence of different parameters with respect to the achievable saving time by ride sharing, a simulation software was developed. This software can handle different traffic layers, a rule set for negotiation matching partners, various numbers of different road users and a few global attributes like the maximum transmission range or the ratio of hosts and clients. The benefit is described in terms of saving time of taking a shared ride trip versus going by foot (in %).
Different scenarios have been processed in order to test the influence of different parameters. The underlying data set was road information from the city of Hanover – in total a road network of a map of nearly 400 km^2. In the experiments the hosts and clients were randomly placed on the road network, with randomly chosen start- and destination points.

6.1 Influence of communication range

In order to check the influence of the communication range on the travel benefit, 36 simulations were run, with transmission ranges between 1000 and 10000 meters and the same number of hosts and clients. In
Figure **7** a dot stand for the average case of a result. The horizontal lines mark the upper and lower limitations of results for the simulated cases. Even though in the beginning (from 1000 to 4000 meters of transmission range) it seems like the profit is increasing, the values for 5000 and 10000 meters disprove this assumption. The results are fluctuating a little bit around a time saving of 8 %, but never break out unexpectedly.

6.2 Average saving time

For most simulations we used 100 clients and 100 hosts and a communication range of 2000 meters. Because of the long calculation time of about 11-19 hours per simulation we did not try more road users in one simulation yet. For this case the average saved time was about 18-22%. In the following Figure 8 the communication range and planned route of a selected client is shown in pink color.

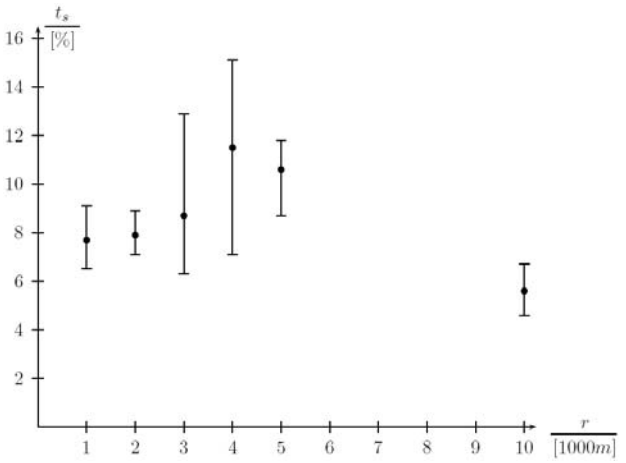

Figure 7: Time benefit per client and second for various transmission ranges

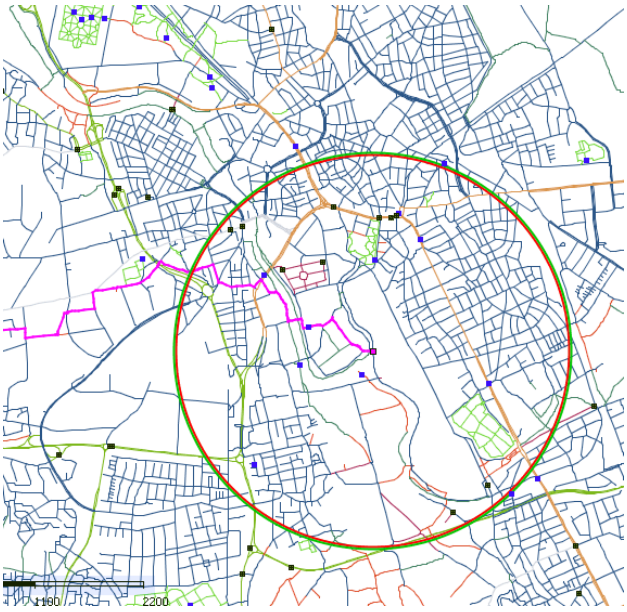

Figure 8: Screenshot of simulation program

6.3 Influence of number of clients on saving time

Another experiment was also done to quantify the influence of the number of clients on the average saving time. For this we used a fixed number of ten hosts. The number of clients was increased in several steps from 10 up to 200. The data shows a similar graph as in the transmission range experiment. For this experiment even the average resulted values are all very densely close (Figure 9). Apparently, neither the transmission range nor the number of clients has an appreciable influence for time benefit per client. The relatively low benefit in time saving results from the fact that only 10 hosts were available on a considerably large road network. This means, that only for a limited number of clients there was the possibility of a shared ride trip at all, and thus a benefit versus going by foot.

6.4 Influence of number of hosts

More interesting is the saving possibility when increasing the number of hosts. In an experiment with a fixed transmission range of 2000 meters the number of clients was set to ten.

Obviously, the availability of more hosts leads to a rapidly increasing benefit for the clients, as shown in Figure 10.

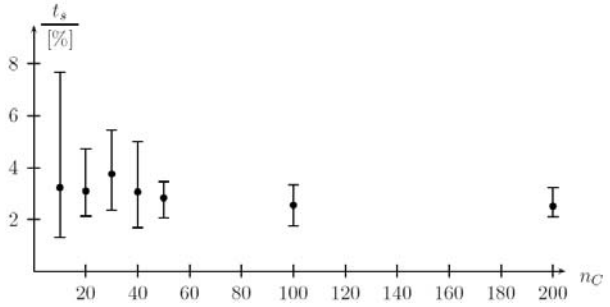

Figure 9: Time benefit per client and second for various numbers of clients

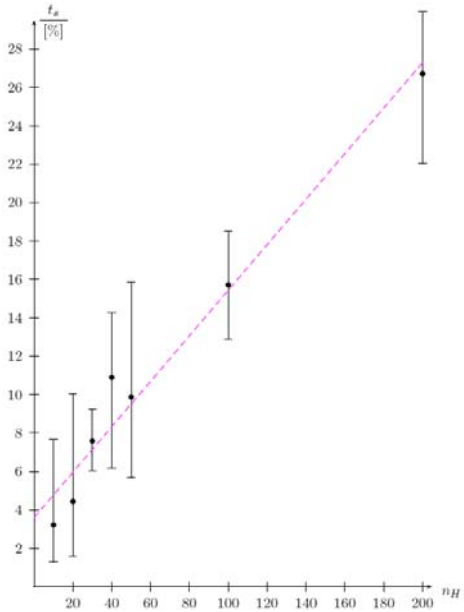

Figure 10: Time benefit per client and second for various numbers of hosts

7. CONCLUSION AND FUTURE WORK

In the simulations an average time benefit of about 20% could be achieved. Some simulations resulted even in 30% saving time. Considering the low number of total road users compared to the size of the road network, we expect a considerably higher time benefit for the clients using more than 200 hosts in a simulation for a city like Hanover.

Even though it is not possible to plan shared trips for farther places a lot of clients find quick access to new hosts. In future simulations we will try more road users. Due to the long calculation time we did not do this yet. One option is to parallelise the processes of computation the simulations, where no data dependencies exist. This could accelerate the calculation time by using multi-core CPUs or even a cluster.

Furthermore some improvements can be made in the traffic model. The density of traffic could be considered for example as well as the public transportation which is not fully implemented in the current version of the simulation. On the other hand the use of other algorithms is conceivable. The Lifelong Planning A* (short: LPA*) algorithm for instance, which is an advancement of A* but considering changing costs for the edges over time could contribute to a more realistic simulation result (Huang, et al., 2007).

In the context of realizing such a service one relevant issue are aspects of trust, reliability and security. That is a common problem in collaborative and social networks (IFIPTM 2007). Even if the compatibility of host and client has been checked by the system, there is still the possibility of misuse. Here, mechanisms of reputation and trust have to be investigated and implemented.

For a possible realization, initiatives of big cities could be supportive, e.g. to allow only cars with more than one passenger at certain times of the day. In this way, shared ride trips can get more attractive, as the host definitely takes advantages of it.

REFERENCES

Bratzel, S., Tellermann, R., 2005. Mobilität und Verkehr http://www.bpb.de/publikationen/D1AKYM,1,0,Mobilit%E4t_und_Verkehr.html (accessed 30 April 2008).

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

Hart, P. E.; Nilsson, N. J.; Raphael, B., 1968. "A Formal Basis for the Heuristic Determination of Minimum Cost Paths". In: *IEEE Transactions on Systems Science and Cybernetics SSC4* (2): pp. 100–107.

Hart, P. E., Nilsson, N. J., Raphael, B., 1972. *Correction to „A Formal Basis for the Heuristic Determination of Minimum Cost Paths“*, SIGART Newsletter, 37, pp. 28–29.

Huang, B, Wu, Q, Zhan, FB., 2007. A Shortest Path Algorithm with Novel Heuristics for Dynamic Transportation Networks. In: *International Journal of Geographical Information Science* 21(6): 625-644.

IFIPTM 2007. *Proceedings of IFIPTM 2007: Joint iTrust and PST Conferences on Privacy, Trust Management and Security, July 30– August 2, 2007, New Brunswick, Canada*. IFIP International Federation for Information Processing, Volume 238, Springer Boston, ISBN 978-0-387-73654-9, ISSN 1571-5736 (Print) 1861-2288 (Online), DOI 10.1007/978-0-387-73655-6.

Pearl, J., 1984. *Heuristics: intelligent search strategies for computer problem solving*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA.

Raubal, M., Winter, S., Teßmann, S., Gaisbauer, C., 2007. Time Geography for Ad-Hoc Shared-Ride Trip Planning in Mobile Geosensor Networks, In: *ISPRS Journal of Photogrammetry and Remote Sensing*, 62 (5): 366-381.

Rauch, J., 2007. Au!to, Traffic study for online-magazine http://rauch.twoday.net/stories/3363369/ (accessed 30 April 2008).

Winter, S., Nittel, S. 2006. Ad-Hoc Shared-Ride Trip Planning by Mobile Geosensor Networks, International Journal of Geographical Information Science, 20 (8): 899-916.

Wu, Y.-H., Guan, L.-J., Winter, S., 2008. Peer-to-Peer Shared Ride Systems. In: Nittel, S.; Labrinidis, A.; Stefanidis, A. (Eds.), Advances in Geosensor Networks. Lecture Notes in Computer Science, Springer, Berlin.