Time geography for ad-hoc shared-ride trip planning in mobile geosensor networks

Martin Raubal[1,3], Stephan Winter[2], Sven Teßmann[3], Christian Gaisbauer[2]

[1]Department of Geography, University of California at Santa Barbara, CA 93106-4060, U.S.A.

[2]Department of Geomatics, The University of Melbourne, VIC 3010, Australia

[3]Institute for Geoinformatics, University of Münster, 48149 Münster, Germany

Ad-hoc shared-ride trip planning in an urban environment is a complex planning task within a non-deterministic transportation network. Mobile geosensor networks provide the technical environment for realizing ad-hoc shared-ride trip planning: Network nodes are autonomous agents that interact locally by ad-hoc short-range communication and arrange for shared rides. In a mobile geosensor network, communication costs are critical because of constraints regarding bandwidth, available energy, and memory. This paper introduces spatio-temporal concepts from time geography, which can be employed during the planning process to significantly reduce communication costs. We will integrate network-based algorithms and different wayfinding strategies to assist both shared-ride clients and hosts in finding optimal travel assignments. Multi-agent geosimulation in a real street network is used to demonstrate the applicability of the approach and quantitatively confirm the theoretically foreseen reduction in communication costs.

Keywords: trip planning, mobile geosensor networks, time geography, network algorithms, agent-based simulation.

1. Introduction

Shared-ride systems match a specified travel demand of clients (e.g., pedestrians) and transportation supply by hosts (e.g., private cars, taxis, or public transportation vehicles) such that clients find rides offered by hosts to their destinations. Current shared-ride systems exist in a large variety of forms, be it by social conventions in informal groups {Resnick, 2004 #1055}, or provided by centralized services {Colorni, 2001 #1043;Wash, 2005 #1057;EuropeAlive, 2006 #1047}. In general, centralized services require service pre-notification by hosts as well as pre-booking by clients, and thus, are more popular for nationwide than for local traveling. Today's centralized services cannot cope with instantaneous transportation supply and demand on a large scale, since this concerns monitoring and managing a complex, non-deterministic transportation network in real time.

Advances in technology allow envisioning an *ad-hoc*, peer-to-peer shared-ride system. This decentralized system will be based on a mobile geosensor network {Winter, 2006 #1031}. In this mobile geosensor network each node is represented by a software agent on a mobile device, either attached to a client or to a host. In the remainder we will no longer refer to the human roles, and hence, can call the agents shortly *client* and *host*. Clients and hosts have positioning sensors on board, are moving, and can communicate with each other via radio within a limited range. Clients can negotiate directly with nearby hosts for rides. If only nearby hosts are needed, then this system architecture is fully scalable.

However, an ad-hoc shared-ride system poses some challenges in optimizing its communication and trip planning strategies. An optimal trip, for example, the quickest trip, can take any route and can involve rides with multiple stops, where clients can change hosts. Consequently, an agent planning a trip seems to require knowledge of the complete current transportation network before coming up with an optimal trip. Since this agent must collect such knowledge ad-hoc from other agents, it becomes clear that one critical question is how the communication effort to collect this knowledge can be reduced. The motivation for any reduction is manifold {Zhao, 2004 #1064}:

- Communication costs in terms of energy consumption are a major concern in any mobile sensor network. Nodes are battery-powered and radio communication is the most energy-consuming activity of a node. In our scenario, energy is a concern at least for clients.
- Bandwidth in the communication channel is another concern. Mobile sensor networks communicate in relatively short communication windows (to save power), and this limits the number of messages to be exchanged.
- The memory of mobile devices is another limited resource in this type of application. In this work we assume that all agents have a copy of the street network available in memory and also routing algorithms for their own purposes. But the transportation network in shared-ride trip planning is dynamic, with the number of edges equalling the number of hosts times the lengths of their travel plans. For an inner-urban traffic situation, this number can easily exhaust the storing and analyzing capacity of any planning agent.

In this paper we are interested in the optimal trip for a client at the time of planning. The hypothesis is that *the transportation network knowledge required to find the optimal trip, and therefore the necessary communication in the geosensor network, can be restricted significantly*. This directly implies a significant reduction in signaling and computational overhead, as well as power consumption of the mobile devices {Küpper, 2005 #1022}. Our focus is therefore on enriching the reasoning capabilities of the agents in the geosensor network, and we are not concerned so far with communication routing strategies or protocols.

Specifically, a heuristics based on spatio-temporal criteria is developed, which enables the planning clients to identify potentially relevant transportation hosts *before* any communication occurs, and also enables contacted hosts to decide whether their routes may contribute to an optimal trip *before* responding. The theoretical framework that underlies this filtering approach is *time geography*. It was introduced by Hägerstrand {, 1970 #840} and focuses on the question of how people's locations in space at given times affect their abilities to be at other locations at other times.

This paper extends previous work {Winter, 2006 #1023} that developed the theoretical model of this paper. Here we present additionally an implementation of the heuristics in a multi-agent simulation and on a realistic street network. This requires the transfer of time-geographic elements to dynamic transportation networks. The simulation allows testing the hypothesis on a large scale including two different wayfinding strategies—a reference strategy and a multi-stop strategy—, and results of this simulation are shown and discussed.

The next section (Section 2) presents an overview of previous and related work, before we collect principles from time geography that enable us to limit the exchange of messages to relevant ones (Section 3). These theoretical ideas are tested in a multi-agent simulation, which is specified in Section 4 and results presented and discussed in Section 5. The paper closes with conclusions and directions for future work (Section 6).

2. Previous work

In this section we introduce the ideas behind shared-ride trip planning, discuss the relevant theory on network algorithms, and present an overview of simulating geospatial phenomena.

2.1 Shared-ride trip planning

Ad-hoc shared-ride trip planning has recently been proposed as a promising application for mobile geosensor networks {Winter, 2006 #1031}. Mobile geosensor networks {Stefanidis, 2005 #1020} are wireless peer-to-peer networks, which provide an effective, efficient, and elegant design alternative to the non-scalable centralized shared-ride services. This approach is *effective* because it can provide near-optimal trips (note that the globally optimal trip can be determined in a non-deterministic system only in hindsight). It is *efficient* because it does so for low communication costs. And it is *elegant* because it requires low computational effort. Details depend on the chosen combination of communication and wayfinding strategies.

Ad-hoc shared-ride trip planning is a problem defined on a complex, non-deterministic transportation network: hosts come and go, and do not follow schedules, and ad-hoc formulated demand of a client is also non-deterministic. Therefore the approach to study the properties of trip planning in a mobile geosensor network was by simulation. This simulation is specified in {Winter, 2005 #1059} and results are presented in {Winter, 2006 #1031}. The reality of urban traffic was thereby simplified to a grid world, in which hosts travel at constant velocities and clients look for rides along a predefined route in the center of the simulated world. A suitable protocol allowed bi-directional communication between agents that negotiate over shared rides. A negotiation process consisted of three steps:

1. A client broadcasts a *request message* with the route the client wants to travel.
2. Hosts with travel plans overlapping this requested route broadcast an *offer message* for this particular overlap.
3. Clients, after collecting all offers and selecting the best, broadcast *booking messages* to the selected hosts.

In this simulation it was investigated how different communication strategies—different depths of communication into the transportation network—affect the average trip lengths. Since the communication links in this network of moving agents are fragile, all three negotiation steps must happen within one communication window, which in practice radically limits the communication depth, and thus also the knowledge of the planning client agent. Hence, it was a significant result that network knowledge from a local set of hosts allows for finding trips nearly as quickly as from the full set of current hosts. The result is plausible if we consider the waiting times for hosts that are currently not near to the client. While this simulation was suited to investigate the effect of different communication strategies, it did not provide a way to compute optimal solutions for the shared-ride trip problem. Communication depth thresholds were chosen arbitrarily, not based on relevance criteria, and the chosen wayfinding strategy—following a predefined route—does not necessarily deliver the optimal (e.g., quickest) trip.

The optimal trip can take any route. Hence, a client should request not a route, but any contribution to a travel between start and destination. But with this poorly specified request, it seems difficult for the hosts to determine whether their own travel plans are potentially relevant for the client. This problem will be addressed in the current work by utilizing elements of time geography during the planning process and applying them to a street network.

2.2 Network algorithms

A client's shared-ride trip planning corresponds to searching for an optimal trip. The client can always determine an optimal trip within a collected set of travel offers. But what if the client has not collected any attractive offers? Hence, we define an optimal trip as being the optimal trip computed from all hosts present at the time of trip planning. The interesting question is then for a client to collect all the offers from the current set of hosts that can contribute to this optimal trip. A trip can be optimal with respect to any single- or multi-criteria cost function. Without loss of generality, we assume in the remainder of this paper that travel time is the criterion to be minimized.

At this moment it is helpful to distinguish the sorts and characteristics of networks in the shared-ride system. Their characteristics and representation are decisive for the design of efficient algorithms. We distinguish three different networks:

- The *street network*: The street network is a directed and weighted network that can be considered static during the time of a trip. We can assume that all agents, clients and hosts, know the street network and its static properties.
- The *communication network*: This network is created by connectivity, which exists between all pairs of agents that are within radio range to each other. Note that a mobile sensor network can consist of disconnected components.
- The *transportation network*: This network is the dynamic (space-time) network of host trajectories and waiting times at transfer points. Since multiple hosts can go along the same street segment at different times, the travel time weight of the street segment becomes time-

dependent, with waiting times at the street intersections. This structure is well known for time-dependent transportation networks {Pallottino, 1998 #1054}, and relevant for schedule-based trip planning. Depending on the ability and willingness of clients to walk, additional space-time-segments can be created in the form of walking trajectories along any street segment and without waiting delay.

Classic shortest path algorithms, such as Dijkstra's {, 1959 #112}, or A* {Hart, 1968 #1049}, expect full network knowledge. This is also the case for their time-dependent versions {Cooke, 1966 #1044;Orda, 1990 #1053;Ziliaskopoulos, 1993 #1065;Chabini, 2002 #1041;Chon, 2003 #1042}. But in the decentralized planning process the clients have only local network knowledge: they learn about the trajectories of nearby hosts. This knowledge is both *spatially* (not contacting all hosts in the client's communication network component) and *temporally* limited (not accessing any information from future hosts—information which might not exist at all). This sort of planning problem requires an adaptive algorithm. Winter and Nittel {, 2006 #1031} propose to let the client regularly revise its travel plans: by reaching different hosts with each negotiation, they catch some of the unpredictability. However, Winter and Nittel were not primarily interested in the algorithm, so their clients can simplify the planning problem by requesting a specific route. Broadening their approach to an adaptive A* strategy allows clients finding quicker routes if that requires detours. This algorithm was first proposed by {Koenig, 2004 #1051} and then adapted by {Wu, 2005 #1063}.

All these algorithms cannot find the overall optimal route—the one that considers future opportunities as well. Furthermore, as long as clients request rides only from nearby hosts, in most trip planning cases the transportation network constructed from current offers will not contain any trip to the destination. Hence, an adaptive A* algorithm will necessarily mix information from two different networks. It will use the transportation network for determining the shortest path tree of rides from the current position, and it will then evaluate each node in this tree with a heuristics on the remaining part to the destination, which can be based on the street network. Since weights in the transportation network change over time, any decision may be outdated after a short time. This means that the iterative application of this algorithm does not guarantee a global optimum, only the best decision at each time.

An inverse problem to the given one has been approached by Wolfson and Xu {, 2004 #1062}. They investigate the dissemination of reports about available resources in mobile (geo-)sensor networks. Translated to the shared-ride system scenario, hosts would be able to disseminate free seat capacity. However, the types of resources considered by Wolfson and Xu, such as a parking spot or an accident, are not mobile.

2.3 Geosimulation

Simulation of agent behavior in space is a powerful research method to advance our understanding of the interaction between agents and their environment. It allows for both the examination and testing of models and their underlying theory as well as the observation of the system's behavior {Gimblett, 1997 #743}. *Geosimulation* is a term for simulation modeling of spatial phenomena using concepts from computer and geographic information science. It is used to represent complex, adaptive, and dynamic systems based on a generative (bottom-up) approach {Benenson, 2004 #1014}. Geosimulation is suitable for representing self-organizing and emergent systems, such as urban environments or transportation systems. While having much in common with traditional simulation approaches (microsimulation, cellular automata, agent-based simulation), geosimulation explicitly captures the characteristics of individual spatial units and spatial relationships. Spatial analysis and remote sensing data can be used as input for geosimulation environments.

The most common types of geosimulation are cellular automata, geographic automata systems, and agent-based systems. *Cellular automata* (CA) are based on a regular grid of cells where each cell has an internal state {Wolfram, 1984 #1033}. For spatial simulations two dimensions with finite boundaries are often chosen for reasons of simplicity. During simulation each cell's state is updated in discrete time intervals, depending on the state of its neighboring cells. Cellular automata are used to simulate systems, in which the behavior of elements can be described by rules, for example, to study the spatial growth of populations. Recently, *geographic automata* have been introduced by Benenson and Torrens {, 2004 #1014}. This approach focuses on capturing a typology of entities, space and spatial relationships, and representing the change of spatial properties over time. In particular, a set of georeferencing rules defines the geographic location of automata within the system. Contrary to cellular automata, neighborhood relationships can vary in space and over time. The concept of neighborhood in geographic automata systems covers concepts known from traditional GIS, such as connectivity or proximity.

Multi-agent systems (MAS) depict systems as a combination of multiple autonomous and independent agents. Agents extend the representation of automata by adding behavior that governs the internal state change {Maes, 1995 #1034}. They are situated in some environment and capable of autonomous action {Wooldridge, 1999 #682}. Categories such as intelligent, distributed, and mobile agents exist. Formally, the term *multi-agent system* refers to a system consisting of an environment including objects, agents, and locations {Ferber, 1999 #1036}. Objects and agents are linked by relations. Various operations represent the agent's actions, for example, an agent can perceive and manipulate an object. Finally, there are operators to represent the application of the operations and the reactions of the world to this attempt of modification.

Agents have been mainly dealt with in artificial intelligence but have recently also gained popularity in other fields such as geography {Frank, 2000 #698}. MAS are of interest to geosimulation due to their ability to reflect human behavior {Frank, 2001 #777}. For example, wayfinders {Raubal, 2001 #757}, car drivers, or shoppers in a mall {Ali, 2005 #1040} can be modeled as agents with internal states and transition rules that drive their actions. Using this concept self-organizing and complex urban systems can be studied well with MAS. Examples include the study of pedestrian's walking paths, cars in urban traffic, and city development.

In previous work on time geography for ad-hoc shared-ride trip planning, cellular automata were used for a proof of concept {Winter, 2006 #1023}. However, in this work, a multi-agent system is the preferred approach for implementing the simulation environment. Cellular automata are too static for the modeling of moving nodes and network structures. The MAS approach is well suited for modeling the dynamics of shared-ride trip planning scenarios: Clients and hosts can be modeled as agents with their own identity and behavior.

3. Optimizing shared-ride planning through time geography

This section introduces time geography and describes those elements, which are utilized to minimize the number of hosts relevant for a declared demand of a client. This method is then applied to networks.

3.1 Time geography

People and resources are available only at a limited number of locations and for a limited amount of time. The ability to be present at a particular location in time is therefore an essential human requirement. Time geography defines the space-time mechanics by considering different constraints for such presence—the capability, coupling, and authority constraints {Hägerstrand, 1970 #840}. The possibility of being present at a specific location and time is determined by people's ability to trade time for space, supported by transportation and communication services.

Space-time paths depict the movement of individuals in space over time. Such paths are available at various spatial (e.g., house, city, country) and temporal granularities (e.g., decade, year, day) and can be represented through different dimensions. Fig. 1 shows a person's space-time path during a day, representing her movements and activity participation at three different locations. The tubes depict *space-time stations*—locations that provide resources for engaging in particular activities, such as sleeping, eating, and working. The slope of the path represents the travel velocity. If the path is vertical then the person is engaged in a stationary activity.

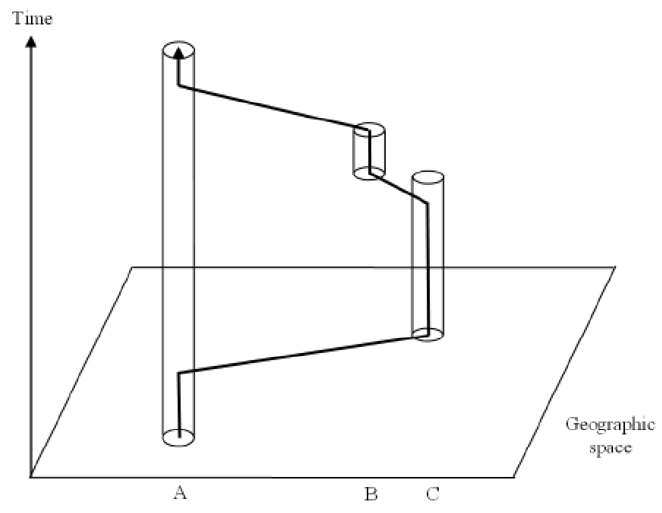


Fig. 1: Space-time path of a person's daily activities—based on {Miller, 1991 #881}.

Time geography defines different constraints that limit a person's activities in space and time. Fundamental physical restrictions on abilities and resources are summarized as *capability constraints*. Not having access to a car in order to trade time for space efficiently is one example for this type of constraint. *Coupling constraints* refer to the requirement for a person to be at a specific location at a certain time or for a fixed time duration. For example, if two persons want to meet at a Cafe, then they have to be there at the same time. Certain domains in life are controlled by *authority constraints*: A person can only shop at a mall, when the mall is open, such as between 9*am* and 8*pm*.

All space-time paths must lie within *space-time prisms* (STP). These are geometrical constructs of two intersecting cones {Lenntorp, 1976 #876}. Their boundaries limit the possible locations a path can take based on people's abilities to trade time for space. Fig. 2 depicts a space-time prism for a scenario where origin and destination have the same location. The time budget is defined by $\Delta t = t_2 - t_1$ in which a person can move away from the origin, limited only by the maximum travel velocity. In a street network, movement is limited by the network geometry and the maximum travel velocity, which can vary for different edges and times. This will be further discussed in Section 3.3. The interior of the prism defines a *potential path space* (PPS), which represents all locations in space and time that can be reached by the individual during Δt. The projection of the PPS onto geographical space results in the *potential path area* {Miller, 1991 #881}.

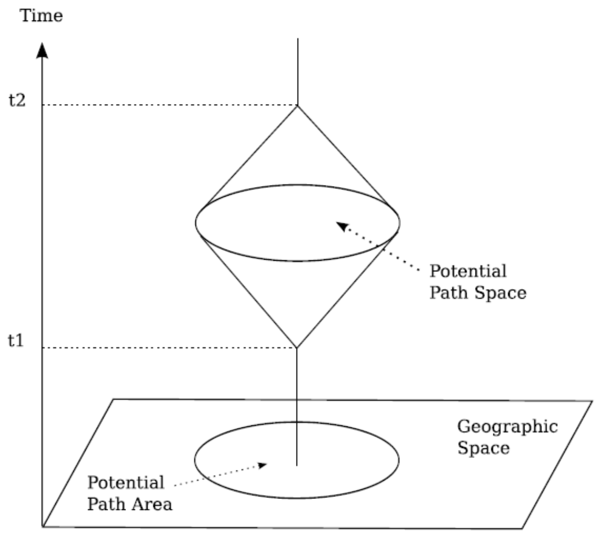


Fig. 2: Space-time prism as intersecting cones—based on {Miller, 1991 #881}.

Time geography has been applied in the area of GIS regarding transportation networks to model and measure space-time accessibility {Miller, 1999 #882;Miller, 2000 #883;Wu, 2001 #885}. It has also been advocated to integrate time geography with both GIS and Location-Based Services to achieve more user-centered systems {Raubal, 2004 #898;Miller, 2005 #830}. Further applications in the geo-domain concern the structuring of dynamic wayfinding environments {Hendricks, 2003 #1010} and the modeling of geospatial lifelines {Hariharan, 2000 #1013}. Analytical formulations of basic entities and relationships from time geography can be found in {Miller, 2005 #1000}.

3.2 Defining basic cones and prisms

In {Winter, 2006 #1023} filtering techniques based on space-time prisms were formally specified and utilized to reduce communication in a geosensor network during shared-ride trip planning. The developed heuristic is based on spatio-temporal criteria and enables the planning clients to identify potentially relevant transportation hosts before any communication. It also enables contacted hosts to decide whether their routes may contribute to an optimal trip before responding.

Using an arbitrary (i.e., walking time) latest arrival time t_1, a cone centered at the destination going backward in time to the start time t_0 can be constructed. This forms one half of a space-time prism. The slope of the cone is defined by walking speed in this case. The base circle of this so-called *dl*–cone (destination, latest) contains all hosts that can potentially contribute to the client's request at time t_0, because they can reach the destination within the given time interval $\Delta t = t_1 - t_0$ (Fig. 3). All hosts outside cannot contribute to the client's travel plan, thus applying this element already reduces the communication effort through elimination of messages from hosts outside the *dl*-cone.

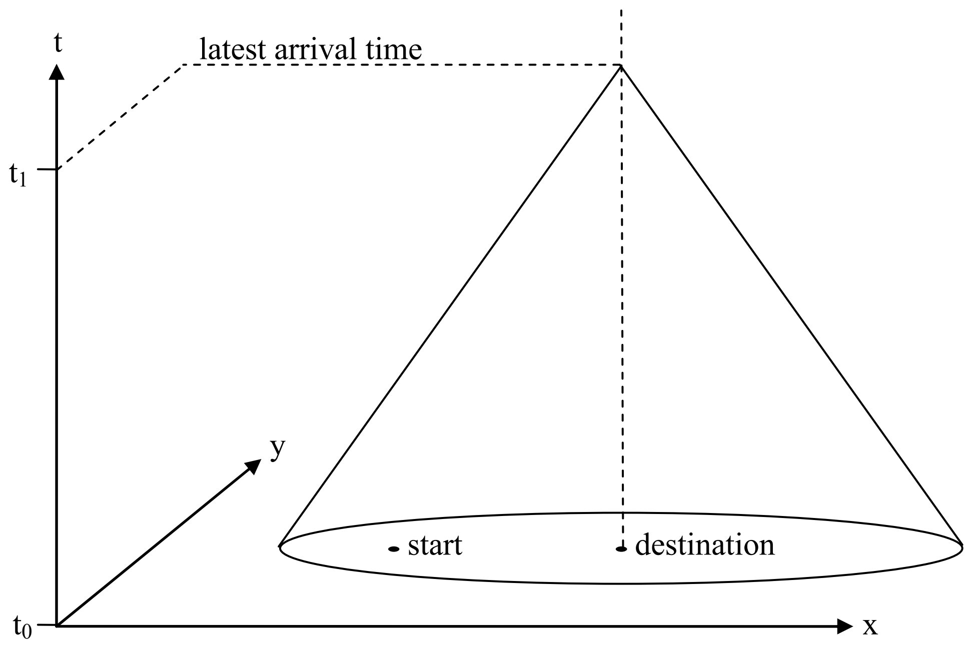


Fig. 3: Example of a *dl*-cone.

In a second step not only locations of hosts are considered, but also their future travel plans. By constructing the so-called *se*-cone (start, earliest) originating in the start of the client at time t_0 and going upward until t_1 the travel possibilities of the client from the start are considered. Intersecting the *dl*-cone and the *se*-cone results in a space-time prism that defines the possible movements of the client between t_0 and t_1 (Fig. 4). This can be used to further limit the number of hosts to be considered, because only hosts with trajectories intersecting the space-time prism are relevant. In order to compute the space-time prism, the hosts need to know only the street network, but nothing about other agents in the network. Therefore no communication among hosts is needed.

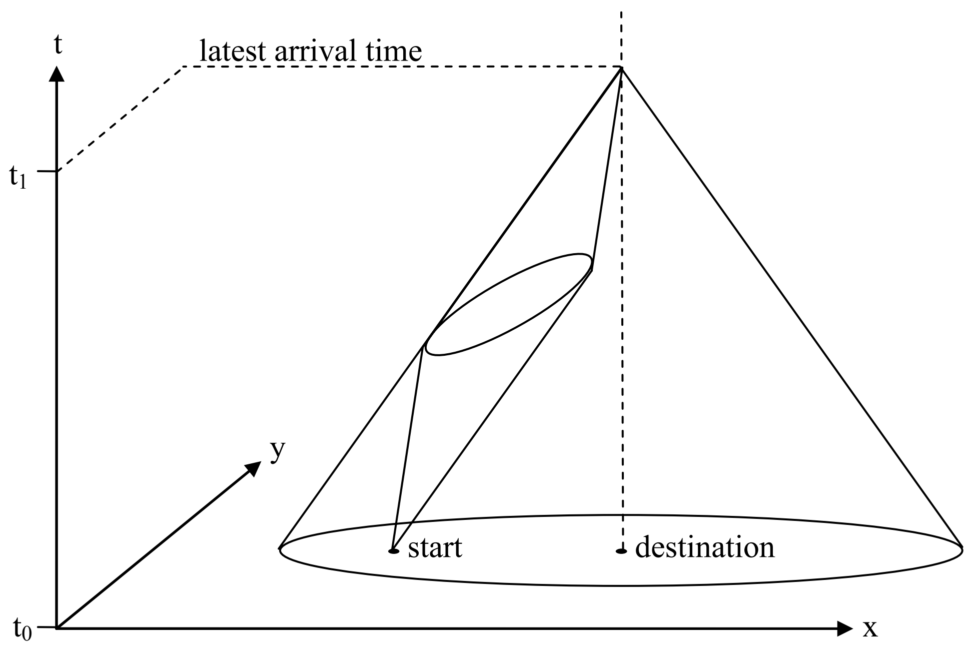


Fig. 4: Example of a client's space-time prism.

Regarding the gain in efficiency of communication it was theoretically shown that the space-time prism of clients imposes a clear criterion for hosts to determine whether their travel plans are potentially relevant for a client. As demonstrated by Winter and Raubal {, 2006 #1023} up to 97% of all hosts that are irrelevant for the planning process can be filtered out. This theoretical estimate was also supported by an example using cellular automata.

As stated by Winter and Raubal {, 2006 #1023} the utilization of time-geographic elements was accomplished with the assumption of continuous space and constant travel velocities in all directions and at all times. This results in perfect cones and space-time prisms with simple geometries. In realistic urban settings a large portion of space is not available for participation in activities or travel {Miller, 1991 #881}. Cones still exist, but they have irregular shapes due to the travel time geometry. They can be determined, though, by average travel time costs along street edges. In the following, the network equivalents of time-geographic elements are defined.

3.3 Time geography for networks

Space-time paths of individuals in networks are limited to movement along edges and nodes. Projecting a space-time path onto geographic space therefore maps to edges in the network. Space-time stations coincide with locations along edges or node locations. Fig. 5 shows an example of a network space-time path including three nodes and two edges. Note that the geometry of the transportation network represents a space-time constraint for individuals using the network.

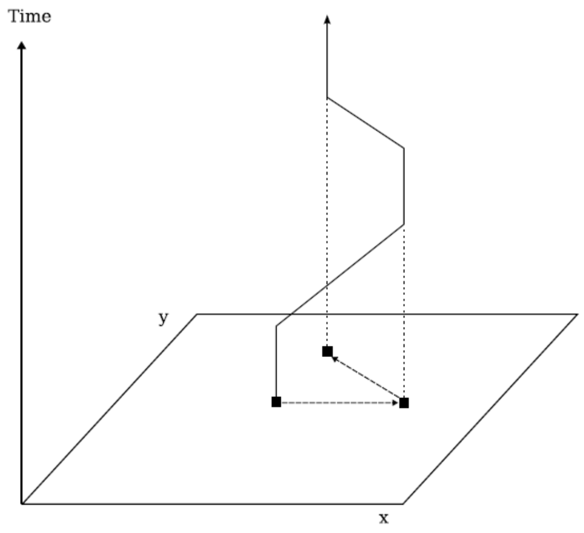


Fig. 5: Space-time path in a network.

The impact of the network structure on the geometry of the STP can be large. Since movement within a transportation network is limited by its geometry and the travel velocity may vary for each edge (and also in time), the geometry of the STP in a network is not a cone, but forms an irregular shape. Miller {, 1991 #881} developed methods and procedures for implementing network-based space-time prisms. The *network time prism* (NTP) consists of edges and nodes in transportation networks. The *potential path tree* (PPT) is a subgraph of the network, which consists of edges and nodes reachable by an individual, given fixed activity locations and a time budget {Wu, 2001 #885}. For networks this represents a more realistic geometry of accessibility for individuals. Miller {, 1991 #881} also presented an algorithm for computation of the PPT, which works based on the following inputs: locations of origin and destination; locations and characteristics of relevant activities; travel environment. The generic procedure consists of two steps:

1. Calculation of shortest paths from the travel origin up to a cumulative impedance (travel time) along each path.
2. Testing, for each edge, whether traveling from the origin over the edge and to the destination is possible within the cumulative impedance.

This approach covers all properties of a network, including edge directions and turn costs. Relaxing the constraints on these properties, we introduce a simpler approach in Section 4.3 for calculating the PPT. Miller {, 1991 #881} further discussed implementation issues for query and visualization applications. Research was also done on incorporating cognitive constraints into the network-based space time prisms {Kwan, 1998 #869}.

4. Agent-based simulation of shared-ride trip planning

In the following, the method of reducing computational and communication overhead by using elements from time geography is applied to an urban environment by simulating the behavior of clients and hosts in a street network. Two trip assignment strategies are tested and the necessary algorithms specified.

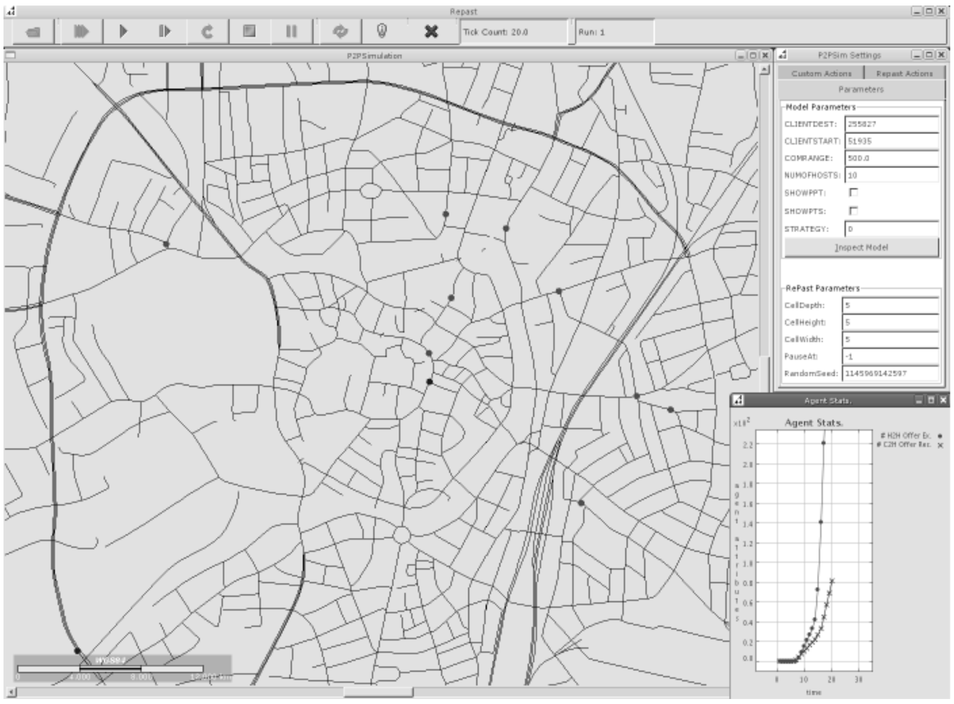
4.1 Environment

As a simulation environment, the software *P2PSim* is developed, which simulates shared-ride trip planning scenarios with one client requesting a ride and multiple hosts moving in a street network, possibly offering transportation supply[1]. P2PSim allows for the simulation of both the communication and the computations of client and hosts, and their real-time visualization. The application supports import of GIS data and the representation of these data as a consistent network structure. Various parameters, such as number of hosts, the trip assignment strategies, and communication range are adjustable, and the recording and analysis of measurement variables over time is possible. The

[1] The software including the code can be downloaded from http://srtp.dvrdns.org.

software is based on open-source frameworks, i.e., *RepastJ*—Java port of the Recursive Porous Agent Simulation Toolkit {North, 2006 #1037}—is used for multi-agent modeling and the *GeoTools*[2] framework for basic GIS data handling.

As a database a street network of the city of Münster (Germany) is used. The data is available in an ESRI® shapefile containing polylines, which were manually digitized from aerial photographs. The network contains 127,264 nodes and 160,803 edges. Fig. 6 shows a screenshot of the simulation environment. As a communication strategy we use *flooding* {Nittel, 2004 #1038}: If a geosensor receives a message it rebroadcasts it to every other node within its communication range. The receiving nodes repeat this process.



Tick Count: 20.0
Run: 1

Model Parameters
CLIENTDEST: 255827
CLIENTSTART: 51935
COMRANGE: 500.0
NUMOFHOSTS: 10
SHOWPPT: ☐
SHOWPTS: ☐
STRATEGY: 0

RePast Parameters
CellDepth: 5
CellHeight: 5
CellWidth: 5
PauseAt: -1
RandomSeed: 1145969142597

Fig. 6: Screenshot of a simulation run.

4.2 Trip assignment strategies

For the simulation runs, the client is placed at a location in the city center with a destination on the inner street circle. This choice represents a realistic medium-length inner-urban trip offering various

[2] http://www.geotools.org

alternatives to reach the destination. Different numbers of hosts are created with randomly chosen start and destination locations. The trips are created such that host routes fall into one of three possible categories (random assignment with adjustable probabilities for each category):

- *Random route*: the complete route is chosen randomly.
- *Random start and destination*: start and destination location are chosen randomly, but the route includes the current location of the client.
- *Random start location*: starting from a random location, the host route includes the current client location and its final destination.

These choices are made in order to give the client a chance to be picked up by a host, as well as keeping the simulation realistic. The time budget is limited to walking time to the destination, therefore in the worst case of not having any rides from hosts, the client reaches its destination by foot. In order to verify our hypothesis that communication between clients and hosts can be reduced significantly by using spatio-temporal criteria during the planning process, two trip assignment strategies are defined and implemented, and then compared with each other—a simplistic *reference strategy* without and a *multi-stop strategy* with the use of time-geographic elements as defined in Section 3.

4.2.1 Reference strategy

For the reference strategy, hosts do not implement any specific algorithm to decide for relevance. This strategy serves as a brute-force approach in order to demonstrate the full scope of possible optimization based on the application of spatio-temporal criteria. If a host receives a request from the client (i.e., start and destination locations) an offer is returned instantly, consisting of the host's route ahead—specified as a sequence of nodes. Messages are rebroadcasted without any restrictions. The client collects all travel plans and decides for the quickest trip to the destination. Once a matching offer—hosts reaching the client's destination—is received the trip is booked. This strategy is implemented for comparing the results to the multi-stop strategy described in Section 4.2.2. For this strategy the following probabilities for creating host routes are used: 20% random route, 60% random start and destination location, and 20% random start location, but at least one from the last category. It is expected that the reference strategy leads to a high number of offers being created, as well as a fast increase in the number of exchanged messages between the client and hosts, and between hosts, because no filtering techniques are applied.

4.2.2 Multi-stop strategy

For this strategy, the NTP of the client's travel possibilities is used by hosts to decide whether to make an offer or not. We expect a significant reduction of the number of offers created by hosts, therefore also significantly reducing communication in the network, because all hosts with routes outside the NTP will not make an offer. With the multi-stop strategy hosts also make offers for partial routes, therefore it is not required that the client's destination is part of an offered route. The client chooses the one that gets it closest to the destination. Shared-ride trip planning using the multi-stop strategy involves the following steps:

1. The client calculates the shortest path from its current location to the destination (for walking).
2. The client sends a request and thereby communicates start and destination as two network nodes.
3. Hosts receiving such request calculate the client's NTP. They create an offer (i.e., a sequence of nodes including arrival times) if their route includes the client's current position and a part of their future route intersects the NTP.
4. The client collects all offers and iterates over these host routes to identify the most promising node. The ranking of nodes is done by calculating for each reachable location *i* the travel time *a(l,i,t)*[3] to reach *i* from the client's current location *l* and the expected minimal travel time *b(i,d)* from *i* to the destination *d* using the Euclidean distance as a heuristic. The final score *r* for each node being part of an offer is calculated by *r(l,i,d,t) = a(l,i,t) + b(i,d)*. If any offer contains the destination it is preferred over other offers due to client preferences.

[3] *a(l,i,t)* stands for the travel time of all available host routes. For our calculations, all travel distances in the space-time network are matched to travel times. The network itself is different for every pair *(l,i)* depending on the time *t*.

5. After identifying the most promising node from all received offers, the client compares this value to the *r*-value of the next walkable node along its precalculated shortest path and decides whether to book the offer or walk.
6. The client sends a booking message to the corresponding host, containing the determined node *i* as the drop-off location. Once this location is reached the whole process is repeated until the client has reached its destination.

For simulation runs using the multi-stop strategy the probabilities of host creation are slightly adjusted. Since the client uses multiple stops on its trip to the destination, a new set of random hosts is generated after the client has completed a partial trip. After *n* runs the probability of a host route to reach the destination is increased with each run. This way the client can reach the destination after a few runs. We expect that the overall trip length using the multi-stop strategy increases because the client most likely does not take the shortest path, but rather takes detours.

4.3 Algorithms and data structures

Different algorithms are needed for the implementation of the shared-ride trip planning simulation. First, the calculation of the network time prism is described, and then we present the specifications for the host decision algorithm and the client choice algorithm.

4.3.1 Calculation of NTP

As described in Section 3.3 the NTP forms the equivalent to the space-time prism for networks. The NTP must be calculated for the multi-stop strategy. Using the same parameters as for the *dl*-cone, a *dl*-subtree can be constructed by running a shortest-path algorithm originating at the destination location that stores the cumulative travel times along the shortest path in each node. The algorithm discards subtrees from nodes where the cumulative travel time exceeds the upper time limit, i.e., the latest arrival time. This solution assumes that traveling along an edge is possible in both directions at the same cost and turn costs are not considered. The reason is that the client travels to the destination in the opposite direction.

For calculating the NTP, the *se*-subtree must also be constructed. This is done by running a shortest-path algorithm originating at the current client location and iterating up to the travel time limit. The NTP of the client's travel possibilities is then calculated by intersecting both subtrees. It consists of all nodes that are part of both the *dl*- and the *se*-subtree. Fig. 7 summarizes the construction process: (1) a *dl*-subtree originating at the destination location starts to grow, (2) a growing *se*-subtree is added, and (3) the two subtrees overlap creating the network time prism. An intersection results only if the upper time limit is sufficient. Calculating the NTP consists of running an impedance-aware shortest-path algorithm twice. For a graph with n nodes and m edges Dijktra's algorithm using a Fibonacci heap implementation has a logarithmic average time complexity of $O(n*logn+m)$. The NTP can be stored using a heap-based structure, because it is used for individual node lookup only, therefore the topology of the network does not need to be preserved. Regarding space complexity two factors have to be considered: (1) the nodes and edges of the travel network are stored external to the algorithm with one upper bound of $O(m+n)$ and (2) Dijktra's algorithm behaves similar to other graph search algorithms in terms of memory consumption with one upper bound of $O(b^m)$, where b refers to the branching factor at a given node and m to the maximum search along a given path. With regard to mobile devices (1) is much more relevant, since typical street network can become very large.

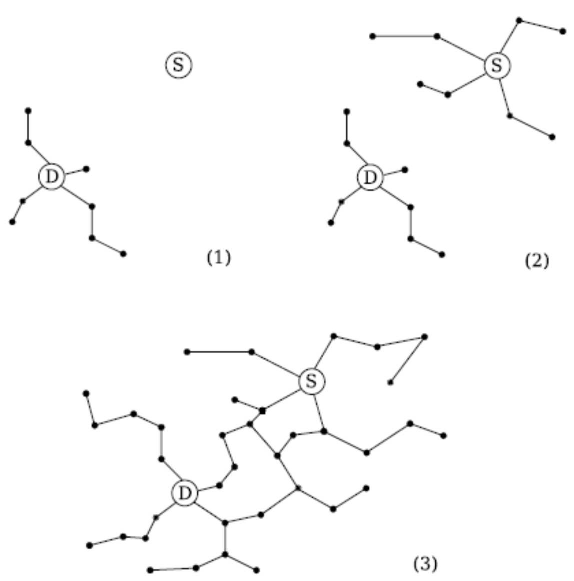


Fig. 7: Construction of NTP.

4.3.2 Host decision algorithm

Hosts have to store their own travel route. This can be done with an abstract data type *stack* that has entries for each node to be visited. At the beginning of the trip the host calculates its route by applying a shortest-path algorithm. Successive nodes can be obtained from the stack within constant time complexity of *O(1)*. After receiving a client's request hosts have to decide whether to make an offer or not. Algorithm 1 (Table 1) shows the specification for the multi-stop strategy. The input parameters are the client request (*request*) consisting of start and destination nodes, a cache of previous requests (*cache*), and the host's travel route stack (*route*). First, duplicate requests are eliminated to avoid resending offers (1). Assuming a hash-based implementation of the request cache, lookup can be done within constant time complexity of *O(1)*. In (2) the offer is initialized as an empty stack together with a Boolean helper variable. The *ntp* variable is initialized with a calculation of the network time prism based on the request parameters. Then iteration starts by looking at one node of the future route at a time (3). If the client's start location is found, the helper variable is set to *True*. Once the start location has been found the currently visited node is added to the offer stack if it is part of *ntp*. An empty stack indicates that the host's travel plan does not include the client's start location, thus no offer is created (4).

Algorithm 1: Host decision algorithm (for multi-stop strategy)

```
input: client request, request cache, host route
output: host offer
  function DECIDE_REQUEST(request, cache, route)
    if cache contains request then STOP          (1)
    offer = empty stack;                         (2)
    foundStart = false;
    ntp = CALCULATE_NTP(request)
    while route not empty do                     (3)
      node = pop from route;
      if node == client start location then foundStart = true;
      if foundStart == true then
        if ntp contains node then push node in offer;
        else break;
    if offer == empty stack then STOP            (4)
    return offer
```

Table 1: Host decision algorithm.

The time complexity of the host decision algorithm can be found by considering the following parts:

1. Calculation of the NTP has logarithmic average time complexity of $O(n*logn+m)$,
2. cache lookup and stack operations have constant time complexity of $O(1)$, and
3. search in the NTP has logarithmic average time complexity of $O(log\ n)$ when using a heap-based implementation.

The cost of iterating over the route has linear time complexity of $O(r)$. With growing network size, however, the input size r (number of nodes in the host route) gets much smaller on average than the input size n and m (nodes and edges in the graph) and can therefore be neglected. The host decision algorithm has therefore total linearithmic average time complexity of $O(n*logn+m)$and a space complexity of $O(r)$.

4.3.3 Client choice algorithm

The client must be able to receive, store, and evaluate offers received from hosts. Storage is necessary to identify duplicate offers and to select, at any point in time, the best of all current offers. An offer contains a route (list of nodes) and arrival times for each node. This way the client can easily identify the optimal trip. The multi-stop strategy requires the client to choose from the received offers the one that contains a node with the best *r*-value. To evaluate closeness of such node to the destination, the Euclidean distance is used as a heuristic. Algorithm 2 (Table 2) shows the pseudo code notation for this strategy.

For the list of known offers two priority queues are used: one for offers containing the destination (*cOfferQueue*) and one for offers containing only partial routes (*pOfferQueue*). The client chooses the best offer from the complete offer queue (see also Section 4.2.2); if this queue is empty the best from the partial offer queue is chosen. The best host offer can then be retrieved with constant time complexity of *O(1)*. At first, newly received complete offers are handled and compared to the currently known best offer from *cOfferQueue* (1). Time complexity for the update operation depends on the implementation of the priority queue—using a balanced tree the insert, update, and remove operations have a worst-case time complexity of *O(log n)* {Knuth, 1997 #1039}. If the received offer contains only a partial route, the drop-off location from the currently known best partial offer is extracted. The algorithm then iterates over all nodes from the received partial offer's route (2) and calculates the score *r* for the current node, which is then compared to the best known drop-off location (3). If a better location was found *pQueue* is updated accordingly (4). Two new operations must be considered for evaluating time complexity: the iteration over all nodes from the offer and the travel time calculation. The latter has a time complexity of *O(1)*, the former of *O(n)*. Total average time complexity of Algorithm 5 is therefore linear, i.e., *O(n)*. Because the input size *n* of the client choice algorithm refers to the number of nodes in the offer, which is on average much smaller than the total number of nodes in the network, the average time complexity of the client's calculations is less relevant than the host decision algorithm. This responds to increasing network size, whereas the former responds to increasing trip length of the offer. The same is true for space complexity of the algorithm with an upper bound of *O(n)*, because in the worst case, every new offer is enqueued in one of the two queues.

Algorithm 2: Client choice algorithm (for multi-stop strategy)

```
input: host offer, complete offer queue (cOfferQueue), partial offer queue (pOfferQueue)
output: -
 function RECEIVE_OFFER(newOffer,cOfferQueue,pOfferQueue)
   if newOffer includes destination then
     bestOffer = dequeue from cOfferQueue;
     if newOffer is quicker than o then                    (1)
       enqueue newOffer in cOfferQueue at top;
     STOP;
     bestOffer = dequeue from pQueue;
   dropOff = null;
   best_r = bestOffer.r;
   for all nodes ∈ newOffer do                             (2)
     a = a + TIME(n-1,n);
     b = TIME(n,destination);
     r = a + b;
     if (r < best_r) then                                  (3)
       dropOff = n;
       best_r = r;
   if dropOff is not null then enqueue newOffer in pQueue;   (4)
```

Table 2: Client choice algorithm.

5. Simulation results and discussion

This section presents the results of the simulation runs for the two trip assignment strategies specified in the previous section. The outcomes are then discussed.

5.1 Results of the simulation

The software *P2PSim* was used to perform 200 simulation runs with the street network data of the city of Münster (see Section 4.1). The parameters were set in the following way: (1) *communication range*: 500 meters, (2) *travel assignment strategies*: reference and multi-stop strategy with 100 runs each, and (3) *number of hosts*: 10, 20, 30, 40, and 50 with 20 runs each. In order to evaluate communication and computation effort, six variables were recorded:

1. *Client-to-host (C2H) messages*: count of offers and requests that were exchanged directly between the client and hosts.
2. *Host-to-host (H2H) messages*: count of all messages exchanged between hosts, including messages that have been rebroadcasted.
3. *Unique H2H messages*: count of all unique messages received by hosts, i.e., which had not been previously received by the host.
4. *Offers*: count of all offers created by hosts.
5. *Computation time*: total time (in milliseconds) spent by hosts running the host decision algorithm.
6. *Time*: time (in seconds) for each run.

The simulation data was recorded in *runs*, consisting of constant time intervals (*ticks*) of one second. Details on configuring a simulation run with regard to clients, hosts, and routes are given in Section 4.2.

5.1.1 Number of exchanged messages

The number of exchanged messages between the client and hosts was recorded and aggregated. Table 3 shows the average number count per simulation run of client-to-host (C2H), host-to-host (H2H), and unique host-to-host (H2H Unique) messages for all runs. The majority of messages falls into the category H2H. Less than 0.1% of these messages were unique host-to-host messages. Only a small portion of the exchanged messages falls into the C2H category. Using the reference strategy resulted in the exchange of a significantly higher number of messages compared to using the multi-stop strategy, which leads to a reduction of 31.58%.

Strategy	**C2H (Avg.)**	**H2H (Avg.)**	**H2H Unique (Avg.)**
Reference	14,575	757,768	692
Multi-stop	31,201	497,319	412

Table 3: Average number of exchanged messages per simulation run.

5.1.2 Number of created offers

The total number of offers created by the hosts was recorded for each run. Using the reference strategy, 21 offers were created, compared to 16 offers generated when using the multi-stop strategy. This means that on average 23.81% less offers were created.

5.1.3 Effect of increasing network density

For the simulation runs different numbers of hosts were used to investigate the effect of increased network density on communication and computation for both strategies. Fig. 8 shows the average number of exchanged messages per simulation run with an increasing number of hosts (10 to 50). For the message count we used the sum of unique host-to-host (H2H Unique) and client-to-host (C2H) messages per simulation run. The values document a higher rate of increase in communication for the reference strategy after more than 30 hosts populate the network. The number of exchanged messages increases almost linearly using the multi-stop strategy.

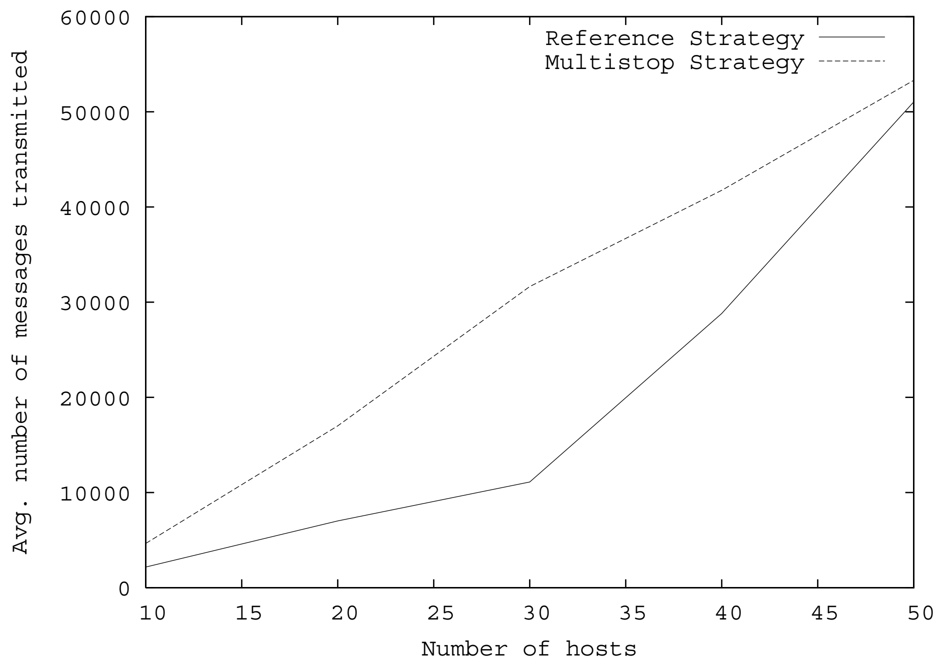


Fig. 8: Average number of exchanged messages per simulation run.

5.1.4 Increase of communication over time

During the simulation, the variables were also recorded for each tick to provide a time series of communication and computation effort. Fig. 9 illustrates the timeline of the increase in average number of messages per simulation run exchanged for both strategies. The reference strategy has a higher increase in communication, with quadratic growth between 0 and 45 ticks, settling after about 60 ticks to a logarithmic increase. With a roughly logarithmic increase the multi-stop strategy documents a smaller rate of increase in the average number of exchanged messages.

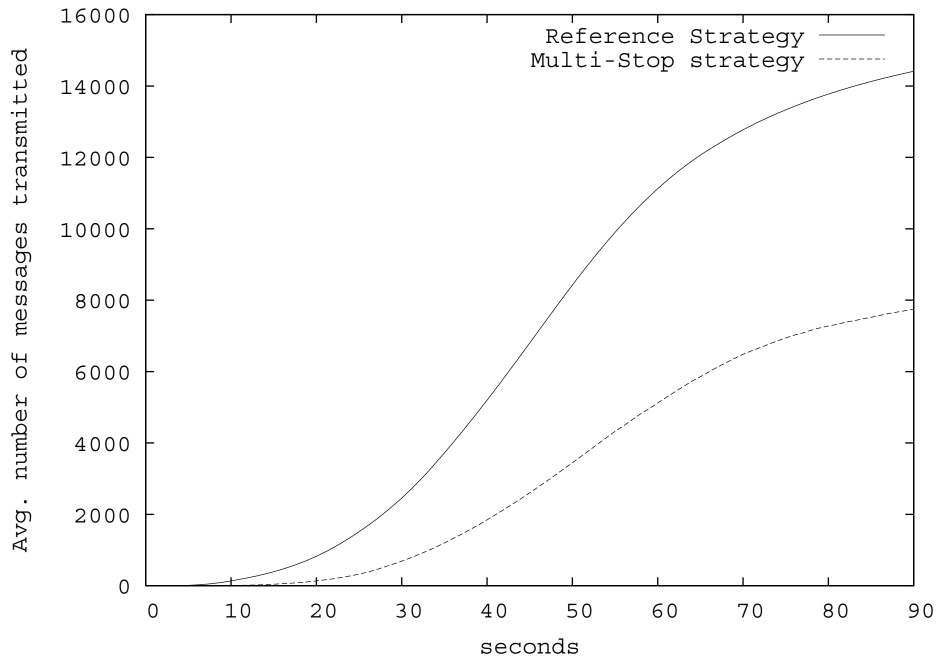


Fig. 9: Messages transmitted over time per simulation run.

5.1.5 Comparison of computation time

Computation time was also measured for the host decision algorithms performed after receiving a request. Table 4 demonstrates the average computation times[4] per simulation run using both strategies. These values give only a rough estimate, because different systems, particularly mobile devices, may lead to large difference in these values. Most valuable for comparing the computational effort are the considerations on time complexity described in Section 4. Compared to the reference strategy the multi-stop strategy has a computation time, which is more than 100 times higher.

Strategy	Avg. number of hosts	Avg. computation times (ms)
Reference	30	64
Multi-stop	30	7982

Table 4: Average computation times per simulation run.

[4] Measured on a system with the following specifications: Pentium-M 725 1.8GHz, 1GB RAM, Debian Sarge Linux (kernel 2.6.12-10-686), JRE: build 1.5.0 06-b05, mixed mode, sharing.

5.2 Discussion

When evaluating the outcome of the simulation runs, the results show a significant reduction in overall communication for the multi-stop strategy, which uses the network time prism as a filter criterion. This is extremely important for host-to-host communication, with a significant reduction of communication compared to the unfiltered approach. The larger number of client-to-host messages results from the longer trip lengths, but does not affect the overall reduction in a major way. The results reveal a large gap between the number of raw host-to-host messages and the unique host-to-host messages received. This demonstrates convincingly that the number of rebroadcasted messages has a great influence on the overall message exchange and that reducing this number of messages represents an efficient way to cut down overall communication.

For the number of created offers the results demonstrate a similar reduction. The reduction of offers using the multi-stop strategy lies within the expected amount: 80% of the generated hosts most likely did not have a valuable route for the client, yet using the reference strategy they still made an offer. With an average host count of 30 the results show that by using the reference strategy 70% of the hosts made an offer, whereas this number is only 52% when applying the multi-stop strategy. In Section 4.2.2 it was argued that the average trip duration will be longer using this strategy. This is proven by the results, which indicate that the duration of multi-stop trips is roughly twice as long compared to the reference strategy. The recorded data indicate that using the multi-stop strategy more than half of the hosts generated in the "random start and destination" category had a trip valuable for the client, because there was a node in their future travel route, which brings the client closer to its destination.

Analyzing the communication with increasing network density and over time demonstrates that by applying the filtering techniques overall message generation and rebroadcast decrease significantly while still finding optimal trips for the client. The reference strategy shows a much steeper increase in overall communication compared to the multi-stop strategy as the number of hosts in the network increases. The measured computation times reflect the increase of calculating the network time prism while using the multi-stop strategy.

6. Conclusions and future work

In this paper we introduced time-geographic concepts for networks and utilized them for ad-hoc shared-ride trip planning in mobile geosensor networks. The hypothesis was that this leads to a significant reduction of communication and computational overhead. The large-scale simulation of the model in a real street network based on the developed P2PSim software confirmed the theoretically derived results. In realistic street networks space-time prisms become network time prisms, which have irregular shapes due to the travel time geometry. They can be determined, though, by average travel time costs along street edges. The results show convincingly that the network time prism of clients imposes a clear criterion for reducing the communication effort in an ad-hoc shared-ride system. The theoretical considerations in Section 3 list the possibility to reduce the dissemination of a client's request to the client's space-time prism, and to enable hosts to determine the relevance of their travel plans for a specific request. The simulation in Section 4 realizes the filtering process of the hosts, and demonstrates its efficiency. Including more of the elements of the theoretical model into the simulation can only improve this trend.

For reasons of simplicity, the presented simulation has also introduced some deviations of the hosts from a random mobility model. These deliberate constraints directed more hosts along the client's position, and this shortened the average trip times, i.e., run times. The introduction of probability parameters kept the approach valid because chances of host routes to fall into one of the three designated classes stayed the same as with a random mobility model. Average trip times are not the focus of this investigation, and depend on many factors, among them the mobility model of hosts {Winter, 2006 #1031}. But this design has an effect on the results: It leads to higher host densities along the route of the client than elsewhere. This means we observe more communication traffic than necessary, because we have more hosts within the space-time prism of the client. With other words, the results on efficiency should only increase within a truly random mobility model.

It was argued earlier that computational costs for determining the space-time prism by hosts are comparably low {Winter, 2006 #1023}. This assumption was made for a grid street network and relatively short client trips. This paper demonstrates that the computational costs for realistic trip lengths in real street networks are effectively manageable, and low compared to the effort of unfiltered communication. The comparison with unfiltered communication is relevant not only with respect to the saved local message handling effort, but also—and in particular—with respect to the energy costs of broadcasting, which are by far higher than for computations.

The resulting model is theoretically founded, but has a heuristic component by choosing a suitable latest arrival time. In this paper we have assumed that clients are able to walk, and will walk for one time interval if no better offers were made. The walking route would be along the shortest path from the current position to the destination, and the walking speed is slower than any shared ride. After one time interval clients will revise their travel plans. With these constraints, the latest arrival time is in fact determined by walking speed: if a client does not get any shared rides, he has to walk the whole distance.

The work presented here suggests many questions and directions for future research:

1. An open question is the choice of an adequate mobility model for hosts. Hosts in urban traffic are not moving randomly in the network, and hence, violate a basic assumption of the random walker model {Camp, 2002 #1066}. Capturing the dynamic aspects of transportation networks, such as time dependent travel times, temporal peaks and congestion, is an ongoing research effort, leading to increasingly realistic space-time accessibility measures. Furthermore, if the clients have knowledge of the regular traffic patterns of hosts, they could adapt their wayfinding strategies and choose transfer points at more frequented street intersections.

2. Another question concerns the effect of competition. If clients compete for seats, hosts will introduce their own interests, and not always make an offer.
3. In the presented simulation, clients only evaluate offers at the end of a booked trip. Extending the model so that clients evaluate other nodes also during trips may lead to more efficient shared rides because additional good opportunities would not be missed.
4. Due to the dynamics of the system, both *a(l,i,t)* and *b(i,d)* do not necessarily have the same impact on the potential of an intermediate node *i*. Modeling different influences can be done through a weighted linear combination of the relevant factors $(\alpha*a + \beta*b)$. The larger the ratio α/β the more *r(l,i,d,t)* is sensitive to the time it takes to reach *i*, while the smaller the ratio the more sensitive the relevance function is to *b(i,d)* {Gaisbauer, 2006 #1067}.
5. The calculation of *b(i,d)* may be refined based on the travel time that a client could expect for a ride along the shortest path at maximum speed. Such estimation is optimistic, can never overestimate, and leads to more accurate results than the used Euclidean distance.

7. Bibliography