Marketplaces as Communication Patterns in Mobile Ad-Hoc Networks[⋆]

Daniel Görgen, Hannes Frey, Johannes K. Lehnert, and Peter Sturm

University of Trier
Department of Computer Science
54286 Trier, Germany
E-mail: {goergen|frey|lehnert|sturm}@syssoft.uni-trier.de

Abstract. This paper proposes a novel communication pattern for mobile multihop ad-hoc networks which is based on a marketplace metaphor. In order to substantially increase the probability that negotiating peers sucessfully reach an agreement, communication is focused on a static geographic area, called the marketplace. Users are not constrained to be at the marketplace physically, but are allowed to utilize other ones mobile devices located at the marketplace to let a software agent or a service installed on each device negotiate with others on their behalf. The forwarding and negotiation protocols needed to implement the marketplace solution are described in this work. Additionally, a prototypical implementation of the protocols is evaluated in a simulation environment. Since simulation results strongly depend on the mobility model, three realistic models based on an extension of the random waypoint model are used. Their movement patterns are resulting from persons on a music festival, a university campus, and an exhibition.

1 Introduction

Mobile distributed systems are formed by PDAs, Pocket PCs, and even smaller systems that communicate with nearby devices using wireless transmission technologies such as IEEE 802.11 or Bluetooth. Wireless communication is characterized by low bandwidth, a high probability for packet loss due to interference, short interaction periods with generally yet unknown neighbors (ad-hoc), and the additional constraint to conserve energy in low powered devices. It is state of the art to realize so-called single-hop mobile systems, where wireless communication is only used in order to utilize an otherwise traditional network infrastructure such as IPv4 – and any middleware on top of it – via a stationary access point. In contrast, multi-hop networks are characterized by the absence of such a stable and dependable backbone network. In order to manage these networks successfully and to execute mobile applications efficiently, self-organization techniques have to be deployed instead. The underlying principle of this kind of

[⋆] This work is funded in part by DFG, Schwerpunktprogramm SPP1140 "Basissoftware für selbstorganisierende Infrastrukturen für vernetzte mobile Systeme".

self-organization is to base all decisions of a mobile device on its local knowledge, to cooperate (sometimes altruistically) with immediate neighbors only, and to achieve the overall goals primarily through synergy.

A fundamental communication pattern for many self-organizing distributed applications is to identify one or more mobile peers satisfying a given set of requirements as defined by some client. This general pattern can be used e.g., to implement a mobile auction system in which one or more bidders have to be found for offered goods. Other examples are digital ride boards to arrange possible lifts among drivers and travelers or electronic blackboards where students seek for tutors assisting in exercises. The simple solution for a client to just wait until a matching peer device comes into communication range has to fail, since the probability to find such a peer by random might be too low. In contrast, flooding the entire ad-hoc network with all active client requests is doomed to fail as well because of the immediate network saturation arising from broadcast storms [16].

A working solution for this communication pattern is based on a marketplace metaphor. A marketplace is a fixed geographical location where information is traded at given times. Marketplaces should be located were high device density could be expected. Information about time and location of marketplaces is assumed to be distributed within the network by means of an underlying basic information dissemination protocol[6]. Client requests or agents acting on behalf of the client travel to the marketplace by infecting promising nearby devices. This decision to infect another device within communication range is based primarily on the relative geographical positions of the device actually carrying the request resp. the agent, the candidate device, and the marketplace itself. When arriving at the marketplace, the device actually hosting the request or agent is searching for matching peers by periodically announcing the set of requirements. These infrequent broadcasts are limited to a given perimeter around the geographical center of the marketplace. Hosting devices are changed if they are going to leave the marketplace. When the market is closed, any data and agents involved in the marketplace stick to their actual host device and travel back at the next opening time. At a given deadline or if a sufficient number of matching peers is found, the response resp. the successful agent will travel back to the coordinates of the home zone, defined by the initiator. Depending on the application, some condensed data may remain at the marketplace for a limited time to identify and eliminate duplicates that are likely to occur in this highly dynamic environment.

By assigning a marketplace to a specific geographical location and by requiring the client requests or agents to move to this place, the probability to identify matching peer devices can be increased substantially. Devices within the marketplace perimeter may even host more than one request or agent at a time and may broadcast accumulated data periodically. The number of devices at the marketplace and the number of active requests impose a communication load within the marketplace. This load is observed locally by special caretaker agents that may decide to split the place into two separate markets if the load exceeds a given threshold or to join two independent marketplaces in case of low utilization. How to partition the problem space for two separate markets is defined by the application, e.g. in case of a digital ride board the zip codes of the destinations can be subdivided or in case of the auction system categories for goods can be introduced and distributed among the marketplaces.

In the following section, the required protocols to implement marketplaces are presented in detail. Due to space limitations, protocols to balance the load among several marketplaces and further issues concerning the caretaker agents are not presented. Section 3 discusses simulation results with respect to agent mobility and negotiation. These simulations have been evaluated with three realistic mobility models: (a) the festival model with a single big hotspot of mobile devices, (b) the campus model as defined by populated buildings with well-defined paths in between, and (c) the exhibition model where mobile devices randomly move from booth to booth. Section 4 discusses related work. In the last section, the conclusion that marketplaces are well-suited communication patterns for certain application domains in the area of self-organized mobile systems is drawn and work in progress is pointed out.

2 Protocol description

This section describes the protocols used to implement the marketplace-based communication patterns. The first subsection outlines the protocol used to move to and from the marketplace. The second subsection describes the protocol used to negotiate at the marketplace.

2.1 Moving to and from the marketplace

Agents are required to be location-aware in order to use location information to reach the marketplace resp. their home zone. By using trajectory information of its current host and the hosts in its direct neighborhood, the agent *jumps* on that host, which makes it most likely reach the destination. The trajectory information of adjacent devices could be gathered by periodical broadcasts or requests after the discovery of a new neighbor. This work proposes three strategies for moving to a desired geographic coordinate.

– *D-method* is the easiest way to reach the geographic destination by using the greedy method as proposed in [14]. By using this method, the neighbor having the least distance to the desired geographical position is chosen.
– By using *DC-method*, first a set of best devices regarding distance to destination is determined. From this set the device with the best course to the destination is selected.
– *CD-method* is analogous to *DC-method* but the other way round.

Due to frequent topology changes in a mobile ad-hoc network, there is no guarantee for two devices remaining connected during an agent transmission. In particular, there is no guarantee that disconnected devices will subsequently be reconnected, so that an error recovery can take place in case of a link failure during agent transmission. Thus, agents might get lost during transmission. To

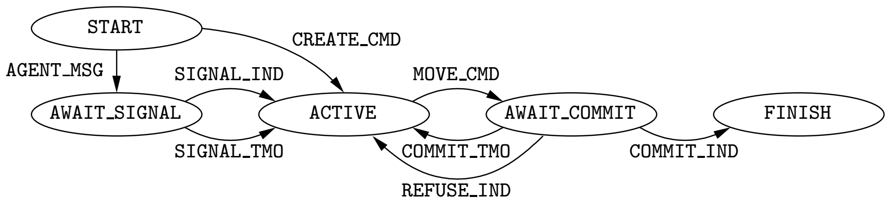

Fig. 1. Life cycle of an agent on a particular device.

cope with this problem, an agent remains on the sending device until it is sure, that its copy arrived at the receiving device. This prevents agent losses but might lead to agent duplicates.

An agent is tagged with a duplicate flag indicating that duplicates of itself may exist. This flag is examined upon arrival at the marketplace. The following algorithm uses such a flag. It can easily be proved, that the algorithm assures that if duplicates exist of a particular agent, each incarnation of this agent is tagged with the duplicate flag. The opposite does not hold. That is, if an agent incarnation is tagged with the duplicate flag, it does not always hold that there really exists a second incarnation of itself. As a consequence, a recovery procedure has to be started at the marketplace to delete possible additional instances of an agent.

The finite state machine of figure 1 depicts the lifecycle of a particular agent and its possible duplicates on the agent platform of a mobile device.

The initial state `START` denotes that the agent is not yet existing on that device. There are two possible events leading to an incarnation of an agent on a device, creation of a new agent (`CREATE_CMD`) and receipt of an agent (`AGENT_MSG`) which has to be transmitted to this platform. The first state of a newly created agent is `ACTIVE`.

An agent in state `ACTIVE` is allowed to change its hosting device by the `MOVE_CMD`. This decision might follow one of the strategies described above. Once a `MOVE_CMD` occurs, an agent sends a copy of itself to the new device and switches into state `AWAIT_COMMIT`. Additionally, a timeout `COMMIT_TMO` is set, which is noticed in state `AWAIT_COMMIT`.

State `AWAIT_COMMIT` means, that the agent still remains on its old platform, since it is not sure, whether its transmission was successful. There are three possible events noticed in this state. The event `COMMIT_IND` occurs if agent transmission was successful and a notification from the receiving platform was received on the sending platform and thus the agent can be deleted there. The sender switches to state `FINISH` and notifies the receiving device with a `SIGNAL_IND` message. The `REFUSE_IND` will occur if the receiving device is not capable to host an additional agent due to memory or processor limitations. The third event `COMMIT_TMO` is noticed if the time to wait for a `COMMIT_IND` or `REFUSE_IND` expired. In this case it is not certain, whether during the transmission to the new platform the agent itself or the notification from the receiving platform was lost. Thus, the agent is not removed from the sending device and is tagged with a duplicate flag and switched back to state ACTIVE. From now on the agent and its possible duplicates will remain tagged as duplicated.

When a mobile device receives an agent in an AGENT_MSG and there are enough resources, the agent is placed on the agent platform in state AWAIT_SIGNAL and the SIGNAL_TMO timeout is set. Subsequently, it replies with a COMMIT_IND message. If there are not enough resources to host the new agent, a REFUSE_IND message is sent instead. An agent will remain in AWAIT_SIGNAL state until a timeout SIGNAL_TMO expires or SIGNAL_IND is received from the sending device. If the timeout SIGNAL_TMO occurs in this state it is not certain, whether the reply message COMMIT_IND from the receiving device or the reply message SIGNAL_IND from the sending device got lost during the agent transmission protocol. Thus, it is not clear if an agent copy remains on the sending device. As a consequence, the agent switches into state ACTIVE but is tagged with the duplicate flag and remains tagged as described above. If an agent receives a SIGNAL_IND in state AWAIT_SIGNAL, it knows that its former incarnation on the sending device was removed. Hence, it can switch into state ACTIVE without being tagged.

2.2 Negotiating at the marketplace

Agents that arrive at the marketplace can negotiate with other agents. An agent may have two roles in such a negotiation. Either it announces its own offer or it looks for suitable offers.

Marketplaces may be bigger than half of the sending radius of the devices. Therefore RegionCast, a geographically limited form of flooding, is used to distribute offers and responses over the whole marketplace. Here RegionCast is used to address agents. Each device receiving a RegionCast message must decide if the addressed agent is locally available and then forward the message to it. Thus, even agents moving away from the center of a marketplace or changing between agent platforms can be reached.

The basic negotiation protocol as depicted in figure 2 works as follows. All agents start in state IDLE. An agent switches to state AWAIT_RPL and starts a new offer when it reaches the marketplace. The agent repeatedly sends DEAL_REQ messages to the marketplace and waits for DEAL_RPL messages from other agents interested in its offer. As soon as the agent accepts one DEAL_RPL message by sending a DEAL_S_COMMIT message to its originator, it switches to state AWAIT_C_COMMIT, waiting for the final DEAL_C_COMMIT message from its negotiating party. Any DEAL_RPL messages arriving from other agents are refused by sending a DEAL_REFUSED message. When the agent receives the DEAL_C_COMMIT message, it switches to state DEAL_OK, accepts the deal and starts to move to its home zone. An agent looking for new and suitable offers listens to DEAL_REQ messages from other agents. If it is interested it sends back a DEAL_RPL message and switches to state AWAIT_S_COMMIT. In the AWAIT_S_COMMIT state it waits for an acknowledgement by the other agent; when it receives the DEAL_S_COMMIT message it responds with a DEAL_C_COMMIT message, switches to state DEAL_OK, accepts the deal and starts to move back to its home zone.

Since this negotiation protocol is used in a multihop ad-hoc network, the additional DEAL_C_COMMIT message is necessary: both DEAL_S_COMMIT and DEAL_RPL

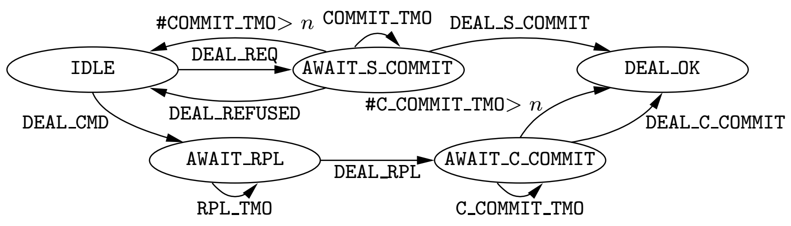

Fig. 2. Negotiation between offerer and bidder.

messages may be lost. Without the DEAL_C_COMMIT message the offering agent could already be on its way home and the responding agent would have no information about its state if the DEAL_S_COMMIT message was lost. A lost DEAL_RPL message would force the offering agent to negotiate with other agents, while the corresponding agent would accept the deal. The introduction of the DEAL_C_COMMIT message does not completely solve the problem since it may be lost, too, but the resulting undefined status is more easily resolvable.

An offering agent registers a timeout C_COMMIT_TMO when it switches to state AWAIT_C_COMMIT. Whenever this timeout occurs, it sends the DEAL_S_COMMIT message and registers the timeout again. This is repeated until the timeout has occured more than n times (e.g. $n = 10$ was used in the simulations). The repeated retransmission will fix the problem of lost DEAL_S_COMMIT messages, but will not help for lost DEAL_C_COMMIT messages. If the timeout has occured more than n times, the agent switches to DEAL_OK state and accepts the deal because it can assume that the DEAL_C_COMMIT message was lost.

An agent responding to an offer uses the same technique. It registers a timeout COMMIT_TMO when it enters state AWAIT_S_COMMIT. If this timeout occurs, it resends the DEAL_RPL message and registers the timeout again. This repeated sending of the DEAL_RPL message helps if either the DEAL_RPL message or DEAL_S_COMMIT message are lost. If nonetheless the timeout occurs more than n times, the probability of a lost DEAL_REFUSED message is high and the agent may safely switch back to state IDLE and wait for new offers.

All these additional measures cannot guarantee that no incorrect negotiations take place, but they minimize the probability a lot.

3 Simulation

The simulation environment and the simulation results are presented in this section. The first subsection describes the mobility models and common simulation parameters. The two following subsections present the simulation results regarding the mobility and negotiation protocols.

3.1 Simulation environment

Three different mobility patterns extending the random waypoint model [3] as shown in figure 3 are used to model the movement of devices. The mobility

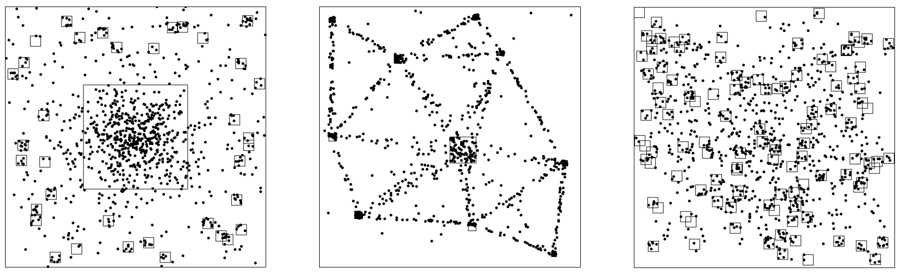
(a) Festival model (b) Campus model (c) Exhibition model

Fig. 3. Screenshots of the mobility models.

patterns use hotspots to influence the movement of devices. Hotspots have a higher probability of being a destination than the surrounding area. The figures show hotspots as rectangles and devices as dots.

The festival model (3a) imitates the movement pattern of people at a music festival. Most people move to a central place and stay for a while to listen to the music. This central place is surrounded by lots of smaller places where people relax. The campus model (3b) simulates the movement of students at a university campus with a set of hotspots (buildings). In order to model the use of paths, devices are only allowed to choose nearby destinations. The exhibition model (3c) represents the movement of people at an exhibition. People randomly move from booth to booth with short pauses to look at the displayed products. This pattern is modeled with 100 small hotspots representing the booths.

Common parameters for evaluating the mobility strategies are the size of the simulated area (500m $\times$ 500m), a wireless communication facility with a sending radius of 25 meters, a transmission rate of 100 kBytes/s, and a simulation duration of five hours. The negotiation protocol is evaluated with a smaller festival model (200m $\times$ 200m) and a duration of two hours. All measured values are averaged over 10 simulation runs using independently chosen seed values.

3.2 Evaluation of the mobility strategies

The evaluation of agent mobility is primarily focused on the time needed to move to the marketplace and return back to the home zone. Additionally, the number of hops and the number of duplicates are examined. All values are investigated depending on the movement strategy and the mobility model. During one simulation run 100 agents are produced on the first 100 devices, all of them targeting the marketplace and thereafter a dedicated home zone. Agents use one of the three moving methods mentioned in section 2.1. Furthermore, the number of devices (250, 500, 1000, 2000) and the device speed (1, 2, 4 and 8 m/s) are varied.

Mobility models have less effect on movement strategies, when simulating with a high device population (figure 4). This is not limited to the depicted example, but is observed in all performed simulation runs with a high device population. An increasing number of mobile devices leads to a less partitioned

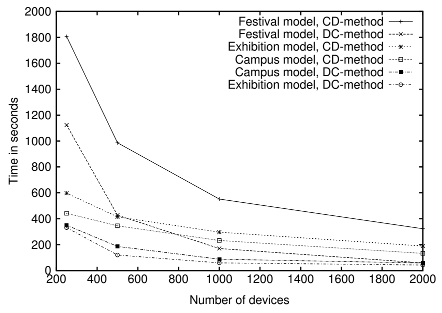

Fig. 4. Mobility models: time for varying device numbers (speed 1 m/s).

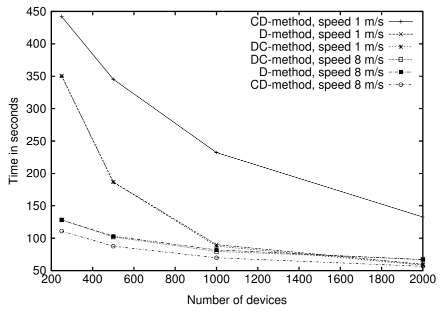

Fig. 5. Campus model: movement strategies for varying device numbers.

network. Thus, selecting a device with the best course to the destination is less significant, as there exists a possible path from source to destination. Due to space limitations only a part of the results and only the campus model is imaged.

Concerning duration and hops, D- and DC-method produce nearly the same results. Both are fast strategies but increasing speed, particularly in campus and exhibition model, causes better results for CD-method (figure 5). This is substantiated with a higher device mobility than in festival model, which is strengthened by increasing speed. Due to longer pause times the mobility in the festival model is lower and so the network is quasi static. Thus, increasing speed has barely no effects on strategy quality because of a quasi static network [3].

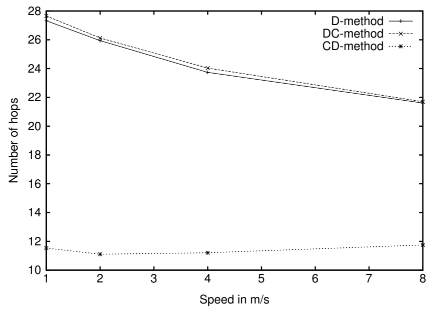

Fig. 6. Campus model, 250 devices: number of hops for varying device speeds.

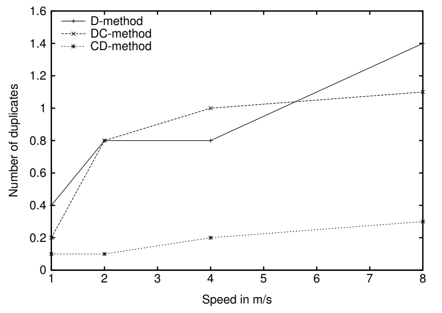

Fig. 7. Campus model, 250 devices: number of duplicates for varying device speeds.

The number of hops needed directly affects the number of messages needed and thus the network load. D- and DC-method need much more hops to reach a destination than the CD-method (figure 6). This is distinctly observable with campus and exhibition models; the festival model results show a smaller gap. Increasing speed leads to improvement in campus, caused by devices walking on paths, but to deterioration in exhibition model, caused by the fact that nearly every device in range has another direction. The more devices are in the simulation, the less hops are needed by D- and DC-method in all models, so the differences to CD-method gets smaller.

Regarding agent duplication (figure 7) CD-method performs best in all models. This is due to the fact that with D- and also DC-method an agent tries to get as far as possible in each jump and so the receiving devices are near the maximum range of the sending device so that message losses are likely.

3.3 Evaluation of the negotiation protocol

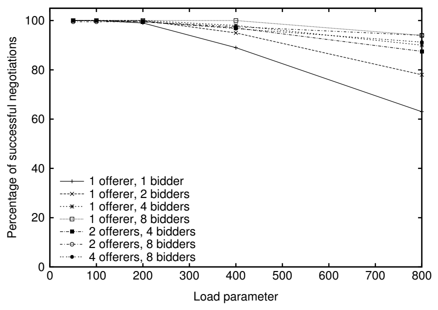

Fig. 8. Percentage of successful negotiations under varying loads.

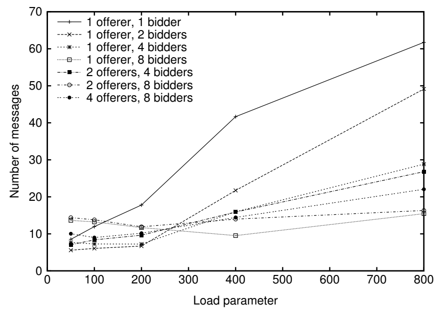

Fig. 9. Number of messages necessary for each potential negotiation.

In order to examine the negotiation protocol, two classes of agents are used, one to generate load, the other to negotiate under load. Load in the simulation is varied over an average number of 50, 100, 200, 400 and 800 negotiating agents per minute. Protocol and negotiation duration as well as successful, failed and incorrect negotiations are measured. Measurement agents consist of m offerers and n bidders with $m \in \{1, 2, 4\}$, $n \in \{1, 2, 4, 8\}$ and $m \leq n$.

The number of agents per device is restricted to 16 so that the marketplace is saturated at about 260 agents in this model. Since there are more agents wanting to reach the marketplace the measure agents are not able to enter the marketplace and are deleted after waiting up to 6 minutes. That explains the decreasing number of successful negotiations in figure 8. As described in 2.2 incorrect negotiations are possible as well but this never occurred during the simulations. The protocol after receiving a `DEAL_REQ` message takes about 1,3 msec at load 50 up to about 1,5 msec at highest load. The duration measurement starts after both agents are on the marketplace and ends when the last negotiation message is received. This takes about 5-11 secs up to 63-101 secs.

Figure 9 depicts the number of messages needed for every potential negotiation ($\widehat{=}$ number of offerers) and as supposed the number of messages increases with increasing load. The order of the results at load parameter 50 is as expected: as the amount of bidders per offerer increases, the number of messages increases. But at high load the order is rolled over. This can be explained as follows: finding only one partner is more difficult under high load than finding one out of many possible partners.

4 Related work

A lot of work in the area of mobile ad-hoc networks concentrates on the special case of a dense population of mobile devices, i.e. algorithms cope with the problem of frequent link failures resulting from mobile nodes, on the condition that sometimes there is a path from source to destination over one or more wireless links. In particular, adaptions and extensions [9, 19, 18, 17] of existing routing protocols known from static networks, will not deliver any packet when facing permanent network partitions. This also applies to routing protocols based on location information specially designed for ad-hoc networks [15, 11, 10, 12]. In an area with sparse device population, link failures are likely to happen and in particular there permanently might not exist a direct communication path from source to destination node. The communication paradigm proposed in this paper works well even if there are permanent network partitions. This is due to the fact that there might be intermediate nodes moving towards the location of the destination node and thus, by using the node mobility, messages can be transmitted to a not directly reachable host.

If location based routing is used to deliver packets to a certain mobile device whose position is not generally known, an additional location service is needed for tracking device positions. Proposals for location services can for example be found in [1, 13]. These proposals have in common that they put additional load on the underlying mobile network which increases the faster the network topology changes, since they have to update position information dynamically due to node mobility. It is advantageous to allow execution state migration and to define a marketplace as a well known geographic area with fixed position, since there is no additional overhead to track device positions, which increases scalability compared to existing solutions.

Since the choice of the mobility model can have significant effects on the performance investigation of an ad-hoc network protocol [3, 7, 20], the simulation environment uses a more complex model tailored to the usage scenarios where the proposed framework may be utilized. Thus, the simulation results have regarding these scenarios a more practical relevance than other performance simulations [8, 2, 5, 4] based on a general random waypoint model [3]. In paper [20], this model is restricted to positions covered by vertices and edges of a graph to achieve more realistic movement patterns. The definition of hot spots as proposed in this paper, is less restrictive for scenarios like music festival, university campus and exhibition, since devices are allowed to walk outside of such predefined paths.

5 Conclusions and future work

This paper introduces a novel communication paradigm for ad-hoc networks. It is based on a marketplace metaphor, which organizes offers and bids by using software agents or an already installed service running on each mobile device. Negotiations done on behalf of their originators are restricted to a fixed geographic area, the marketplace. The solution consists of three main parts, message delivery to and from the marketplace, negotiation at the marketplace and load balancing at the marketplace. The proposed solution scales well and is independent of a sparse device population as long as there are devices moving towards the marketplace.

Three strategies for packet forwarding to the marketplace are presented: D-method, DC-method, and CD-method. D-method is known as greedy packet forwarding [14], the others are extensions of it, additionally using information about the course of a device. Simulation results show for devices moving with walking speed that the following holds: The higher the device population the less the marketplace approach depends on the movement strategy. At a lower density there exists a tradeoff between the proposed strategies. D- and DC-method deliver packets substantially faster than CD-method, whereas CD-method needs only a fraction of messages. If device speed is increased this tradeoff between time and message complexity disappears. That is, in a highly dynamic network CD-method is the best strategy with the least time and messages complexity to deliver a packet to the marketplace. Also, the number of duplicates produced during message delivery due to link failures are least with CD-method.

No incorrect deal occurs during the negotiation protocol simulations due to a restricted number of agents per device and the introduction of a second commit message.

Furthermore, the negotiation time, when a negotiation among offerer and bidder is started is negligible even if the marketplace is under higher load. Restricting the maximum number of agents on a device avoids network congestions and leaves bandwith for other applications using the ad-hoc network. If the rate of agents moving to the marketplace is higher than the rate these agents are served on the marketplace, this solution will lead to an increasing queue of agents willing to enter the marketplace. Under this condition, the marketplace has to be split in geographically disjoint parts all serving a portion of the negotiation classes.

The next step within the scope of this work is a prototypical implementation of a marketplace solution for a distributed ride board using PDAs with a IEEE 802.11 communication facility. Since there is the need to determine the position of a device, these will additionally be equipped with a GPS receiver. In a further step this prototype will be extended to an indoor solution using an GPS-free form of triangulation technique.

References

1. S. Basagni, I. Chalamtac, and V. R. Syrotiuk. A distance routing effect algorithm for mobility (dream). In *Proc. of the 4th ACM/IEEE Int. Conf. on Mobile Computing and Networking (MobiCom'98)*, pages 76–84, 1998.
2. J. Broch, D.A. Maltz, D.B. Johnson, Y.-C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proc. of the 4th ACM/IEEE Int. Conf. on Mobile Computing and Networking (MobiCom'98)*, pages 85–97, 1998.
3. T. Camp, J. Boleng, and V. Davies. Mobility models for ad hoc network simulations. *Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2002.

4. S. Das, R. Castañeda, and J. Yan. Simulation based performance evaluation of mobile, ad hoc network routing protocols. *ACM/Baltzer Mobile Networks and Applications (MONET) Journal*, pages 179–189, 2000.
5. S. Das, C. Perkins, and E. Royer. Performance comparison of two on-demand routing protocols for ad hoc networks. In *Proc. of the 2000 IEEE Computer and Communications Societies Conference on Computer Communications (INFOCOM-00)*, pages 3–12, 2000.
6. H. Frey, J.K. Lehnert, and P. Sturm. Ubibay: An auction system for mobile multihop ad-hoc networks. *Workshop on Ad hoc Communications and Collaboration in Ubiquitous Computing Environments (AdHocCCUCE'02)*, 2002.
7. X. Hong, M. Gerla, G. Pei, and C. Chiang. A group mobility model for ad hoc wireless networks. In *Proc. of the 2nd ACM/IEEE Int. Workshop on Modeling and Simulation of Wireless and Mobile Systems (MSWiM'99)*, pages 53–60, 1999.
8. P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, and M. Degermark. Scenario-based performance analysis of routing protocols for mobile ad-hoc networks. In *Proc. of the 5th ACM/IEEE Int. Conf. on Mobile Computing and Networking (MobiCom'99)*, pages 195–206, 1999.
9. D.B. Johnson and D.A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, volume 353 of *The Kluwer International Series in Engeneering and Computer Science*. Kluwer Academic Publishers, 1996.
10. Y.-B. Ko and N.H. Vaidya. Geocasting in mobile ad hoc networks: Location-based multicast algorithms. Technical report, TR-98-018, Texas A&M University, 1998.
11. Y.-B. Ko and N.H. Vaidya. Location-aided routing (LAR) in mobile ad hoc networks. In *Proc. of the 4th ACM/IEEE Int. Conf. on Mobile Computing and Networking (MobiCom'98)*, pages 66–75, 1998.
12. Y.-B. Ko and N.H. Vaidya. Geotora: A protocol for geocasting in mobile ad hoc networks. Technical report, TR-00-010, Texas A&M University, 2000.
13. J. Li, J. Jannotti, D. De Couto, D. Karger, and R. Morris. A scalable location service for geographic ad-hoc routing. In *Proc. of the 6th ACM Int. Conf. on Mobile Computing and Networking (MobiCom'00)*, pages 120–130, 2000.
14. M. Mauve, J. Widemer, and H. Hartenstein. A survey on position-based routing in mobile ad-hoc networks. *IEEE Network Magazine*, 15(6):30–39, 2001.
15. J.C. Navas and T. Imielinski. Geocast - geographic addressing and routing. In *In Proc. of the 3rd ACM/IEEE Int. Conf. on Mobile Computing and Networking (MobiCom'97)*, pages 66–76, 1997.
16. S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu. The broadcast storm problem in a mobile ad hoc network. *Proc. of the 5th ACM/IEEE Int. Conf. on Mobile Computing and Networking*, pages 151–162, 1999.
17. V.D. Park and M.S. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. In *Proc. of the Conf. on Computer Communications (IEEE INFOCOM'97)*, pages 1405–1413, 1997.
18. C. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, pages 234–244, 1994.
19. C.E. Perkins. Ad-hoc on-demand distance vector routing. In *MILCOM '97 panel on Ad Hoc Networks*, 1997.
20. J. Tian, J. Hähner, C. Becker, I. Stepanov, and K. Rothermel. Graph-based mobility model for mobile ad hoc network simulation. In *Proc. of the 35th Simulation Symposium, in cooperation with the IEEE Computer Society and ACM*, 2002.