Wearable Computers as Packet Transport Mechanisms in Highly-Partitioned Ad-Hoc Networks

James A. Davis Andrew H. Fagg Brian N. Levine

Department of Computer Science
University of Massachusetts
Amherst, Massachusetts 01003
{jdavis, fagg, brian}@cs.umass.edu

Abstract

The decreasing size and cost of wearable computers and mobile sensors is presenting new challenges and opportunities for deploying networks. Existing network routing protocols provide reliable communication between nodes and allow for mobility and even ad-hoc deployment. They rely, however, on the assumption of constant end-to-end connectivity in the network. In this paper, we address routing support for ad-hoc, wireless networks under conditions of sporadic connectivity and ever-present network partitions. This work proposes a general framework of agent movement and communication in which mobile computers physically carry packets across network partitions. We then propose algorithms that exploit the relative position of stationary devices and non-randomness in the movement of mobile agents in the network. The learned structure of the network is used to inform an adaptive routing strategy. With a simulation, we evaluate these algorithms and their ability to route packets efficiently through a highly-partitioned network.

1 Introduction

As sensing, computing and data storage devices become increasingly small, inexpensive and pervasive, we can expect such devices to support new applications in our environments. These devices allow for greater interaction between wearable users and the objects in their environments (rooms, appliances, robots, pets, plants, etc.). For many of these applications only low-cost networking support is likely to exist. Furthermore, we expect that these devices will be more mobile, sparsely concentrated, and randomly placed than most local networks can support today.

As an example, we can imagine a system of inexpensive beacons with only local RF or IR communication abilities scattered throughout an office building (e.g., a locust swarm [5]). Sensors attached to plants will allow them to call out for water; conference rooms will broadcast meeting schedules; and soda machines and supply cabinets will keep lists of current inventory and make restocking requests. In order to pass packets between these *stationary isolated* agents, packets must be transmitted to passing mobile devices carried by a human, robot, mail cart or elevator. From there, the packet might make several hops between mobile and stationary agents as it makes its way to its destination. If these devices require little or no administrative overhead, they can be placed with impunity, thus allowing us to create and place devices for very specific applications.

To allow such devices to cooperate with one another and for the framework to be scalable, new types of network support must be available. Existing *ad-hoc routing* protocols support communication among a group of nodes without a pre-deployed infrastructure, including routers, policy or administration. However, these protocols assume that a series of cooperative nodes connects any pair of nodes end-to-end in the network. Otherwise, no route between the two hosts exists and packets from one to the other are dropped. Traditionally these networks are referred to as *store and forward.* Unless one assumes a sufficiently open environment and a dense scattering of devices, such assumptions of end-to-end connectivity do not hold. In a *highly-partitioned* ad-hoc network, routing protocols must exploit the resources and mobility of *agents* in the network (e.g., wearable computers, palmtops, or laptops) to *store, carry, and forward* packets between devices that are physically isolated from one-another. In other words, under conditions of high partitioning, a protocol is needed that forwards packets during times of connectivity, and prioritizes and stores packets when agents are isolated.

Work on a store-and-carry version of flooding called *epidemic routing* [7] proposed a method of propagating data through a network. Epidemic routing does not exploit the relative position of stationary devices, nor the patterns of user and agent movement that allow for more efficient packet passing.

In order to serve the development and modeling of protocols for these highly partitioned networks, this work proposes a framework for expressing agent movement and communication. Within this framework, we examine how mobile and stationary isolated agents can gain performance advantages by encoding and exploiting *structure* in the network. Structure can be found in agent location and any patterns that exist in agent movement. By discovering and encoding a representation of network structure, we show how the system can increase efficiency and reliability in the network.

2 Background

A number of routing protocols have been proposed previously for non-partitioned ad-hoc networks, e.g., AODV [4] and DSR [3]. The major difference between these protocols and traditional routing protocols for wired networks is the on-demand nature of the routing. Due to the mobility of the routing infrastructure, routes are created on-demand as users require them, and commonly timeout after a few seconds. When a path between two nodes does not exist through the network, no route can be created.

In this paper, we examine routing for ad-hoc mobile environments when partitions are commonplace. In these environments, nodes store and hold packets even when a route does not exist. Later, the packet may be passed to another node that has recently come into range. Little previous work that we are aware of has examined this problem.

Vahdat and Becker have proposed a flooding-like algorithm called *epidemic routing* [7]. According to the algorithm, whenever two agents meet, they synchronize with each other by requesting transmission of packets they do not already possess. After the exchange, they carry the same set of packets. Vehdat and Becker showed for a highly-partitioned network epidemic routing delivers all packets given unbounded resources of time and buffer size at each node. Given the assumption that no node is forced to drop packets due to buffer constraints, epidemic routing further guarantees that the packet will reach the destination in the shortest amount of time possible. In this paper, we address the effect of bounded memory at nodes.

Epidemic routing and the protocol we present in this paper make use of the mobility of the nodes themselves to increase the availability of paths. Grossglauser and Tse have shown that the mobility of nodes generally represents an opportunity for increasing the capacity of ad-hoc, wireless networks, even when not highly partitioned [2].

The rest of this paper is organized as follows. We discuss our framework for expressing movement and communication in Section 3. We detail our approach to exploiting mobility in a highly partitioned ad-hoc network in Section 4. We present an evaluation of our routing algorithms in Section 5, and conclude and discuss future work in Section 6.

3 Network Model Framework

In this section, we present the details of our model framework including our models of agent mobility and agent communication.

3.1 Agent Movement

Our framework is designed to model scenarios in which both fixed-location and mobile computing units exchange packets. It captures the locality of communication, a distribution of fixed-location agents, and random but structured movement of mobile devices. Environments are modeled as graphs. Each agent occupies a single graph node at any one time; mobile agents may move between neighboring nodes at each time unit. Communication between agents is limited to those which occupy the same node in the graph at the same time.

The movement model for a single agent is implemented as a matrix of goal selection probabilities between each source/destination pair of graph nodes. To simulate agent mobility, we use a variation of the *random waypoint* model [1], which selects a destination point for an agent. The agent then moves to that point at a set velocity. Our goal selection matrix is not restricted to a uniform distribution, but allows for selection based on any discrete probability distribution. Since our model is discrete in nature, velocity is expressed by pauses between moves and variation of the timestep.

When at rest, a mobile agent selects a new goal based on its movement matrix. In our model, agents move at their designated velocity along the shortest path towards their selected goals with ties broken by random selection. Upon reaching the destination, the agent pauses for a random amount of time and again chooses a new destination.

Our movement model is concerned with locations in space of unspecified volume that are isolated from adjacent spaces; agents occupying the same node are able to communicate. This is the worst case for traditional ad-hoc routing protocols as no connectivity ever exists between graph nodes and such protocols would fail completely. Our model does not use continuous space like most other ad-hoc network models, as it is meant to capture highly-partitioned network scenarios such as IR or local RF-based communication in a building.

3.2 Communication Model

In our framework, new packets that are to be transported are produced randomly. At any particular simulation time step, each agent generates a packet with a specified probability. The packet is addressed to another agent chosen uniformly at random. Each agent maintains a packet buffer of a fixed size, K, that holds all packets to be forwarded to other agents. No priority is given to packets that an agent itself has generated or those it is carrying on behalf of other agents. A newly generated packet is assigned a timeout, after which it is dropped. The timeout counter is not reset when packets are passed.

At every time step, agents occupying the same node exchange a list of all stored packets. First, deliverable packets are delivered and removed from the queue. The agents individually sort the remaining packets according to the given *drop strategy* (described in Section 4) and keep the top K packets in their queue. The two agents then request transmission of the packets that they do not currently have. If more than two agents share the same location, they conduct this packet exchange in a pairwise fashion in random order.

With infinite buffer size and timeout settings, this communication model is a pure epidemic routing style protocol. In that scenario, no packets are dropped. We want to evaluate scenarios where resources are limited, and it is necessary to introduce a drop strategy metric by which to prioritize packets. Ultimately, the metric used determines if packets are dropped before their delivery or timeout expiration.

4 Exploiting Mobility

The exploitation of node mobility by epidemic routing comes at a cost. The main tradeoff is the additional storage at nodes as packets are stored, carried, and forwarded to the destination. However, the *exposure* of packets along paths other than the shortest path to the destination exacerbates the storage requirements. Additionally, the power costs in transmitting packets can be problematic for devices with small power stores.

Epidemic routing is designed to maximize exposure by distributing packets as widely as possible, while requiring buffers of infinite length. This paper imposes finite buffer sizes on agents and proposes the selective dropping of packets when the buffer limitations are reached. These *drop strategies* form an implicit global routing algorithm by deciding which packets to drop from the buffer.

We explored a variety of drop strategies, a few examples are detailed here:

- Drop-Random (DRA): the packet to be dropped is chosen at random. In many scenarios, this is the most equitable drop strategy since agents that have generated a lot of traffic are more likely to have their packets dropped than agents that send packets more sparingly.
- Drop-Least-Recently-Received (DLR): the packet that has been in the agent's buffer longest is dropped. The logic behind this heuristic is that packets that have been in the buffer longest are the most likely to have been passed to other agents.
- Drop-Oldest (DOA): the packet that has been in the network longest is dropped. This heuristic is based on the idea that the globally oldest packets are the ones most likely to have already been delivered.
- Drop-Least-Encountered (DLE): the packet is dropped based on the estimated likelihood of delivery. This is our adaptive strategy that seeks to drop packets based on information about agent location and movement.

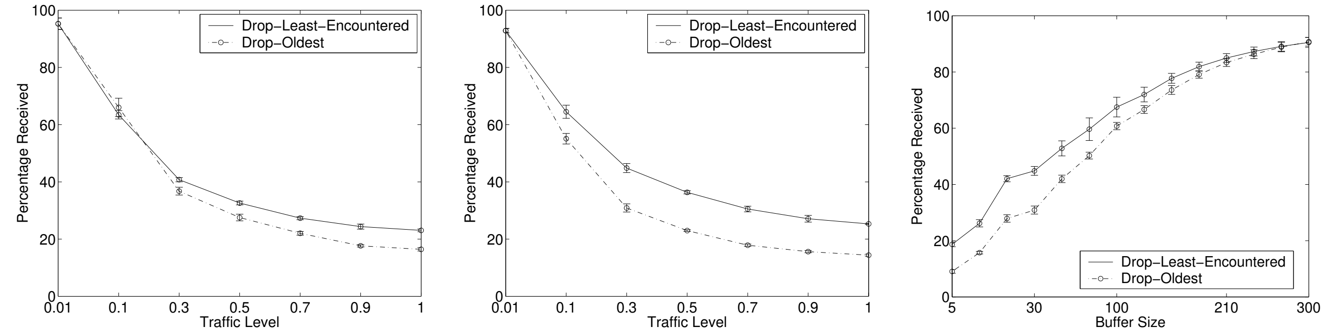
In practice, the two drop strategies that performed best (and the two that we present in this paper) are Drop-Oldest and Drop-Least-Encountered. The details of our adaptive strategy, Drop-Least-Encountered, are discussed below.

4.1 Estimation of Meeting Likelihood

The Drop-Least-Encountered algorithm utilizes a per-agent metric that estimates the likelihood of meeting each of the other agents. Each agent maintains a table indexed by agent address. The value associated with another agent encodes the likelihood of a pair-wise meeting path to that agent. At each time step, agent A updates its meeting value for every other agent, C, with respect to co-located agent, B, according to the following rule:

$$M_{t+1}(A,C) = \begin{cases} \lambda\, M_t(A,C) & \text{if none co-located} \\ \lambda\, M_t(A,C) + 1 & \text{if } C = B \\ \lambda\, M_t(A,C) + \alpha\, M_t(B,C) & \text{for all } C \neq B \end{cases},$$

where $M_t(A,C)$ is the meeting value at time t that agent A associates with agent C, $\alpha = 0.1$ is the parameter dictating the portion of agent B's meeting value that agent A should add to its own, and $\lambda = 0.95$ is the decay rate of the meeting value. Initially $M_{t=0}(A,C) = 0\ \forall A, C$.



A B C

Figure 1: Packet delivery as a function of traffic for : A) unstructured movement; and B) partitioned movement. Panel C shows packet delivery as a function of buffer size (traffic level of 0.3, partitioned).

This Temporal Difference [6] rule allows a recursive estimation of the likelihood that a packet held by an agent A will reach its destination C. This metric encodes three important aspects of the environment's structure: Firstly, if A encounters B, A is likely to encounter B again in the future and is a good candidate for passing packets to B. Secondly, if A encounters B and B has a high encounter value for C, then A is a good candidate for passing packets to C. Lastly, the values degrade over time, so that encounters that happen infrequently do not persist.

The DLE strategy utilizes the meeting values by ordering packets according to the *relative* ability of two agents to pass a packet to the destination. Specifically, when agents A and B meet, agent A orders the packets according to: $M_t(A, C) - M_t(B, C)$, where C is the destination for each packet. Thus, packets tend to pass from agents that are less likely to deliver a packet to agents that are more likely to deliver the packet.

5 Experiments

In this section, we present the results of our evaluation of the Drop-Oldest and Drop-Least-Encountered algorithms. We show that as network load increases the DLE algorithm more often finds routes and delivers data than DOA for a fixed buffer size. Moreover, we show that DLE is better able to localize the exposure of packets by exploiting structured agent movement and the proximity of stationary agents.

5.1 Methodology

All our simulations were based on a 10×10 grid in which we randomly placed 30 stationary agents and 10 mobile agents. We set the packet load of the network by varying from 0.01 to 1.0 the probability of generating a new packet at an agent for each time unit, which we refer to as the *traffic level.* All agent movements, initial placements, and packet generation times were based on a single random seed value. For each random seed, we executed one simulation *run* for each algorithm, which proceeded for a set number of time units before halting. All packets timed-out after 100 simulation time units, after which they were discarded by the agents.

We examined two agent movement scenarios. In the *unstructured* scenario, mobile agents selected a grid coordinate uniformly at random as a new positional goal. In the *partitioned* scenario, each mobile agent was restricted to selecting goal locations on either the left or right half of the environment; this enforced the creation of two distinct sets of mobile agents that could not interact directly during the experiment. Agents were randomly assigned sides at the beginning of the experiment. Additionally, a single mobile agent called a *relay* was restricted to movement along only a horizontal line that crossed both regions. Thus, it was necessary that packets that must be passed between regions be carried by the relay agent. We set the agent pause time between moves to 0 timesteps.

5.2 Routing Performance

Our results show that as traffic load increases, DOA and DLE degrade in their ability to deliver packets successfully for a fixed buffer size. However, DLE generally performs better, especially for high traffic levels. Figures 1A and 1B summarize the performance of both algorithms for various traffic levels for the unstructured and partitioned movement models, respectively. The buffer size for each agent was fixed at 50 packets. The points on the graphs represents the mean of five simulation runs of 1000 time units each. Error bars represent 95% confidence intervals. Note that even with sufficiently large buffers, packets in our simulations were

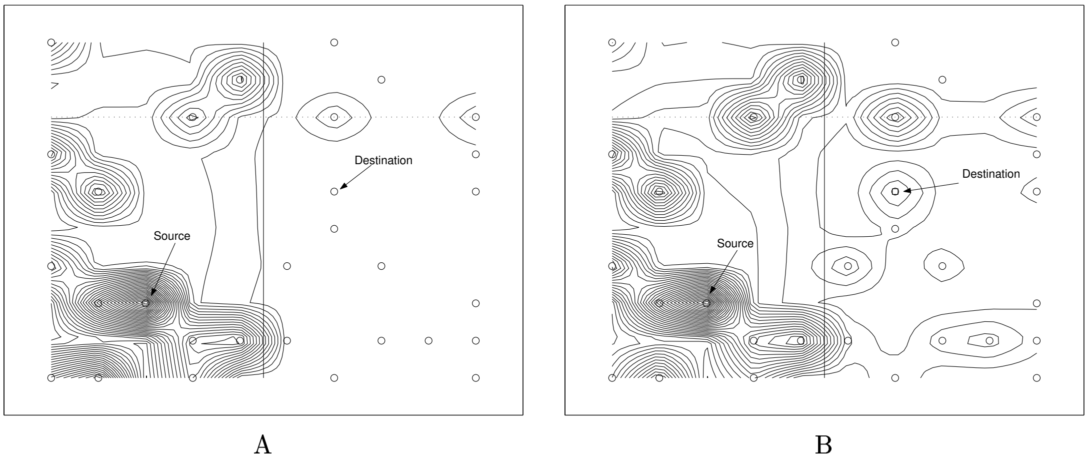


Figure 2: Contour map showing the average total time each grid point was visited by a packet for: A) DOA; and B) DLE. Small circles correspond to stationary agents.

dropped after timeouts expired, accounting for the 5% packet loss at low traffic levels in the simulations.

Figure 1A shows that even in an unstructured movement model with no patterned movement, the DLE algorithm outperforms DOA as network load increases. This is because DLE takes advantage of the relative positions of proximal stationary agents. The algorithm maintains high meeting values for a stationary agent's neighbors due to the movements of mobile agents who recently visited both in sequence. Accordingly, stationary agents serve as "mailboxes", holding on to packets that are destined for neighboring stationary agents.

A comparison of Figures 1A and 1B shows that the partitioned movement model worsens the performance of DOA. A number of packets are dropped before they reach and cross the boundary because they either timeout or are pushed out the buffer by younger packets.

In contrast, the performance of the DLE algorithm improves with the introduction of the partitioned movement model for agents. The adaptive quality of the algorithm is able to capitalize on the structure. As we discuss in more detail in the next section, packet routing is directed towards the region around the addressee and is less likely to expose the packets to regions far from their destinations. This in turn allows the available buffer space to be utilized for local packets.

Figure 1C shows the percentage of packets delivered as a function of buffer size in the restricted environment. For these simulations, the traffic level was set at 0.3. As the figure illustrates, DLE degrades more gracefully than DOA as buffer levels decrease in comparison to the network load.

5.3 Exposure of Routed Packets

A desirable routing algorithm routes a single copy of a packet along the shortest path to a destination; all other packets waste agent resources. In our second set of simulations, we evaluated the exposure of packets as they were routed between a specific stationary sender and receiver pair within the partitioned environment. The position of the source and destination are indicated in Figure 2. The partition between the halves of the environment is shown as a vertical line through the middle of the figure; the path of the relay agent is shown as a dotted horizontal line near the top.

For this set of experiments, we set the traffic level at 0.1 and fixed buffers to 5 packets in order to greatly strain agent resources. Each run was allowed to proceed for 500 time steps in order to initialize meeting values and fill buffers. We then forced the source agent to generate 50 packets, one every 100 time steps. We traced the branching path of the packets through the network, and Figures 2A and 2B plot packet exposure as contour plots for the DOA and the DLE algorithms, respectively.

For both algorithms, packets persist for the longest amount of time at the source because low traffic levels allowed stationary agents to keep packets for a long time without displacement. DOA utilizes more resources closer to the source. In contrast, DLE reduces the amount of time that the tracked packets spend in the lower-left corner. None of the 50 packets were delivered by DOA; all were displaced by younger packets.

Figure 3A illustrates the spread of the the 50 packets over their lifetime under the two algorithms. Each curve shows the number of agents that are simultaneously carrying the average packet as a function of its age. We increased the traffic level to 0.2 to produce

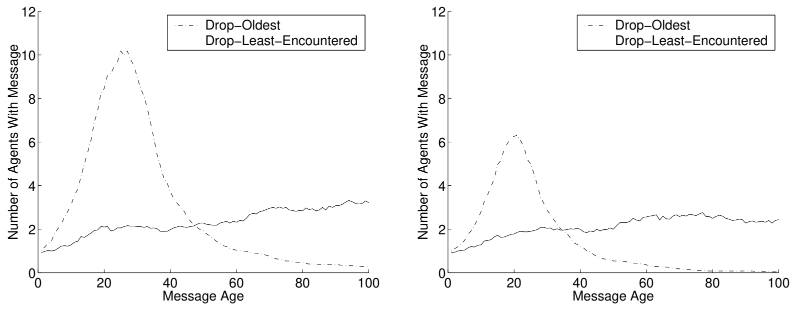


Figure 3: Average number of packet copies across the set of agents as a function of time since packet creation. A) Low traffic level (.1); and B) higher traffic level (.2). Note that packet timeout is 100 time steps.

Figure 3B. For both traffic levels, the DOA algorithm delivers no packets. The DLE algorithm delivered 7 packets and 6 packets for traffic levels of 0.1 and 0.2, respectively.

The graphs illustrate how the flooding of DOA is tightly constricted during heavier loads; packets do not travel far from the source. In contrast, DLE is slower to increase the number of copies, but sustains these copies for a much longer period of time. In fact, the limitation in this example is packet timeout and not displacement by competition from other packets. One of the primary advantages in DLE is that packets are not dropped by agents when they meet. Because the packets are sorted by the difference in the meeting values for the destination agents, the two agents select non-overlapping sets of packets to carry. The meeting of agents in DOA can actually cause old packets to be pushed out of both agents.

5.4 Meeting Values

Figure 4 illustrates the meeting values acquired by the temporal difference algorithm for several individual agents in the partitioned environment as log-contour plots. As expected, the relay agent (Figure 4A) maintains high meeting values for all stationary agents along its fixed horizontal path. In addition, the relay maintains moderate values for many of the other agents in the environment. A mobile agent restricted to the right side of the environment (Figure 4B) has acquired high meeting values for all stationary agents within the region. The somewhat higher values for agents in the top-right corner are due to the fact that this mobile agent has most recently visited this region of the environment.

Stationary agents on the path of the relay agent have slightly higher values for agents on the same side of the partition (Figure 4C), but also maintain non-trivial values for agents on the other side of the environment. The increase is due to encounters with the relay which, in turn, has encountered agents on both sides of the partition. Finally, a stationary agent in the lower-left corner of the environment (Figure 4D) maintains high values for all stationary agents in the same partition, but the highest values are concentrated in the same area.

Packets addressed to the other side of the partition find their way to a "waypoint" along the path of the relay agent. Because the waypoint is more likely than any non-relay (but mobile) agent to encounter the relay agent, the waypoint's meeting value will be higher. Hence, the packet will be passed from the mobile non-relay to the waypoint and held until the relay agent returns.

6 Discussion and Future Work

As computers move to a full-time, wearable form, users can come to expect applications and ad-hoc environments composed of numerous inexpensive computing, sensing, and data storage devices. We expect such environments to operate without pre-existing network infrastructure and have sparse connectivity among users with short communication ranges. However, network routing among users must be present to take full advantage of the opportunities offered by populations of wearable computer users.

In this paper, we have evaluated new algorithms that take advantage of the mobility of wearable computer users, who actually instantiate routes and effectively increase bandwidth by physically carrying packets between disconnected islands of stationary agents. The limited resources, sparseness, and restricted communication range of agents, as well as the structure of the environment in which agents move, make this a difficult problem. However, we have tried to evaluate the effect of all such issues in this initial study.

From a networking perspective, our framework presents a non-traditional view of resources. We have placed less emphasis on the bandwidth available between agents on an end-to-end path, and we have not considered the bandwidth between agents within com-

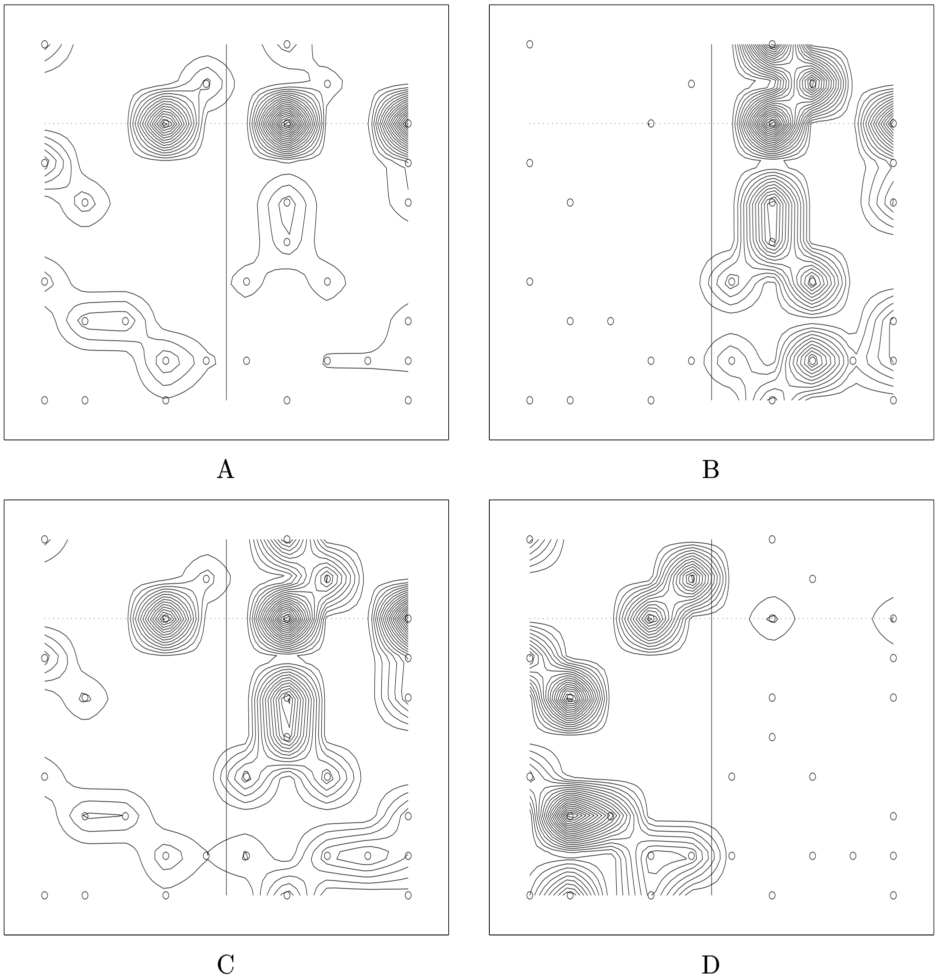


Figure 4: Log-contour map visualization of the meeting values. Each map shows the magnitude of the meeting value for a particular agent overlayed onto physical space. Circles correspond to the location of stationary agents. The four maps show the encounter values for: A) the relay agent, B) a mobile agent restricted to the right half of the environment; C) a stationary agent waypoint that encounters the relay agent; and D) a stationary agent in the lower left corner of the environment.

munications range to be a limited resource. Instead, we have focused on provision and cost of connectivity where traditional routing protocols and even ad-hoc routing protocols would find none. Our constraining resource is the number of packets or amount of data that an agent can carry.

Our contribution has been to introduce a routing algorithm based on an adaptive drop strategy that captures the likelihood with which an agent will be able to pass a packet to its destination, either directly or indirectly. In our experiments, this algorithm is able to outperform algorithms that do not adapt to the local structure and movement patterns of agents. Our DLE algorithm converges based on movements and meetings rather than data traffic analysis, and is fully distributed, depending only on pair-wise, local exchanges of information to spread knowledge about the greater network structure.

6.1 Future Work

Our evaluation of DLE has given us insight into a number of issues that must be addressed as we move beyond this initial study. We have also discovered many possibilities for developing more sophisticated drop-strategies with improved performance. We conclude with a discussion of feedback and reliability mechanisms, battery and power resources, scalability, adaptive timeouts, and networks supported by islands of Internet connectivity.

At this stage of our work, we have assumed that agents do not receive feedback whenever packets reach — or do not reach — a destination. Adding acknowledgements from destinations will enable a reliable transport layer, much as TCP is used for IP protocols. Although the acknowledgements will themselves add traffic to the network, we believe their use may increase the overall efficiency of the wearable network. This is the case for many reasons. First, without acknowledgements, agents may unfortunately retransmit packets several times to ensure reliability with brute force. Second, with the addition of feedback from acknowledgements, retransmission decisions (and even the original transmission) can take into account the local network traffic so as to reduce the likelihood of network saturation (i.e., congestion control). Third, acknowledgements may server to notify other agents still carrying the original packet to drop their copy, freeing buffer space for other data.

Mobile devices rely on battery power, and incur a cost when transmitting information over infra-red or other channels. We believe the DLE algorithm can be modified to take into account the cost of packet swapping. For example, agents may modulate the meeting metric by a value that is proportional to the cost of transmission of that particular packet. Low-power agents may choose to carry packets for a longer period of time, selectively transmitting packets only when the advantages are clear.

In our simulations, we have only considered a constant number of mobile agents. DLE must maintain a local estimation of the meeting values for the other agents in the environment, which grows with the number of existing agents. Our future work will be to explore methods of scaling to larger populations of agents. For example, agents with relatively small meeting metric values may be dropped from the list, implicitly setting this value to zero.

We also have not considered the effects of adaptive timeouts in this study. Upon generating a packet, the sending node could set the timeout based on the meeting value it has for the destination. A high value indicates a high likelihood that the recipient is nearby in the network and that the timeout could be set low. A low value would prompt the sending agent to set the timeout high in order to increase the chance that the packet lives. This mechanism would insure that packets would not persist in the buffers of nodes in the network long after the packet was already likely to be delivered.

We have observed that many packets that are not delivered by DLE are those generated by an agent that has a low or zero value for the recipient. These packets are often dropped by the sending agent before they get a chance to be passed. A proposed solution is to modulate the meeting metric by a value that encodes the number of times the packet has been passed. If each packet had an associated counter that was incremented when it was accepted by another agent, the drop strategy could factor in the number of potential copies of the packet that exist in the network and could favor packets that had limited exposure.

In this paper, we have focused on the problem of packet routing when agents are restricted to local communication. A more general approach which we have begun to pursue is that of including a small number of agents equipped with wider range communication devices (e.g., 802.11 wireless Ethernet). In essence, these *mobile-connected* agents act as a mobile "wormholes", through which packets may quickly propagate to their destination. If the destination is another computer on the Internet, then the packet can be immediately delivered. Otherwise, the packet can be buffered on an Internet-bound machine. When a second mobile-connected agent passes by either the isolated destination or an isolated agent which is likely to visit the destination, the packet will be copied from the buffer. Through these additions to the routing protocol we hope to increase the efficiency with which information is exchanged and broaden the physical space in which interacting agents may be scattered.

7 Acknowledgements

Preparation of this manuscript was supported in part by NSF grant #EIA 9703217.

References

[1] J. Broch, D.A. Maltz, D. Johnson, Y. C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 85–97, 1998.

[2] M. Grossglauser and D. Tse. Mobility increases the capacity of ad-hoc wireless networks. In *IEEE INFOCOM 2001*, April 2001.

[3] D. Johnson and D. Maltz. Dynamic source routing in ad-hoc wireless networks. In *Computer Communications Review–Proceedings of SIGCOMM '96*, August 1996.

[4] C. E. Perkins and E. M. Royer. Ad hoc on-demand distance vector routing. In *2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, February 1999.

[5] T. Starner and D. Kirsch. The locust swarm: An environmentally-powered, networkless location and messaging system. In *Proceedings of the International Symposium on Wearable Computing*, October 1997.

[6] R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.

[7] A. Vahdat and D. Becker. Epidemic routing for partially-connected ad hoc networks. Technical report, Duke University, 2000.