International Journal of Wireless Information Networks, Vol. 14, No. 2, June 2007 (© 2007)
DOI: DOI: 10.1007/s10776-006-0055-9

Density-Independent, Scalable Search in Ad Hoc Networks

Zygmunt J. Haas[1,2] and Rimon Barr[1]

We analyze the asymptotic cost of discovering a route within a flat ad hoc network and we show that one can discover a route with cost that is proportional only to the area of the network, which is independent of the number of network nodes. Furthermore, we show that this is optimal and that *bordercasting* (a query propagation protocol where a node retransmits a query to a set of nodes at some hop-distance away) possesses this density-independence property. We present the design of bordercast and the associated maintenance protocols, and we evaluate their performance. In particular, we highlight that the aggregation of local information by bordercasting at each network node is a fundamental building block for the construction of scalable protocols in flat ad hoc networks.

KEY WORDS: Ad hoc networks; routing; flooding; bordercasting; scalable routing; route discovery; service discovery; scalability; zone routing protocol; ZRP; sensor networks

1. INTRODUCTION

Research into dynamic, self-organizing, multi-hop wireless networks, the *ad hoc networks*, attempts to improve network performance, for instance, the *efficiency* and *coverage* of wireless communication, by increased network connectivity. The idea is well known; rather than tether wireless devices to the wired world via a *single* wireless hop to a base station, a device within an ad hoc network may be connected via *multiple*, shorter hops across other wireless devices that function as routers (and, in some cases, as repeaters). In general, the devices may also communicate with each other (as peer-to-peer) and even operate in isolation; i.e., without global connectivity (e.g., through a base station). Regardless of the traffic pattern, since in most practical cases the power strength of a radio signal is attenuated more than the square of the distance traveled, while supporting the same throughput as a single long wireless hop, multiple shorter wireless links use overall less power, a scarce resource in mobile settings. Furthermore, the overall effective capacity of the airspace may be increased, because these lower power, shorter range transmissions generate less interference. (For a study associated with such tradeoffs, the reader is referred to [1, 2, 3].) Alternatively, with the same power, one can achieve greater capacity or coverage.

The vision of extending the reach of wireless communication in this manner is enticing. However, routing and transmitting packets efficiently become increasingly difficult as the ad hoc network grows. In general, increasing the scale of ad hoc networks to sizes greater than a few hundred nodes still remains an important research challenge. One of the obstacles is the fact that as the number of nodes grows, the average path length (measured as number of hops) increases as well, resulting in a significant reduction in network capacity, as each packet is now transmitted numerous times in the network.

In this article, we address an essential component of this problem. The following are the contributions of this work. We analyze the cost of discovering a route within a flat ad hoc network in the absence of any information about the desired destination node, except for its unique address. We show that one can discover a route with cost proportional only to the area of the network and independent of the number of nodes in the network (i.e., independent of the network density). Furthermore, we show that this is optimal; i.e., that this cost

[1] Wireless Networks Laboratory, Cornell University, Ithaca, NY, 14850, USA
[2] E-mail: haas@ece.cornell.edu

is a lower bound on any possible route discovery protocol that does not rely on additional information about the destination node. Finally, we consider *bordercasting*, a query propagation protocol where a node resends the query to nodes at some distance away. We show that *bordercasting*, which was proposed as part of the *Zone Routing Protocol* (ZRP) framework [4, 5], is, indeed, *density-independent*.

A key message of this article is that the aggregation of zone-wide information at each network node can lead to more efficient and scalable network protocols. We discuss how to collect this zone-wide information efficiently, and we focus on how it can be used to perform scalable route discovery. Nevertheless, the results of this work are extendable to other protocols, such as *service discovery* and more complicated situations, such as is the case in *multicasting group* membership maintenence, for example.

2. BACKGROUND AND RELATED WORK

Prior research has produced numerous routing protocols designed for mobile ad hoc wireless networks; some examples include DSR [6], AODV [7], TBRPF [8], and OLSR [9].

Each of these protocols is often designed and optimized under different networking assumptions, such as the expected traffic pattern or the rate of topological changes in the network. For example, the OLSR protocol is proactive in its route discovery, which is a feature suitable for more static networks, while AODV is reactive, and thus is more efficient in highly mobile settings. The differences in applicability of these protocols to various networking settings motivated the introduction of hybrid protocols, such as ZRP [4,5], ZHLS [10], SHARP [11], and HARP [12]. Additionally, the various protocols also differ in their use of route caching techniques, their mechanisms for detecting route failures, and their capabilities for maintaining routes.

At one time or another, either due to limited cache sizes, changes in the network that invalidate existing information, or the arrival of a new query for an unknown destination, each of these protocols is forced to send a query into the network in search of a node or of information about a node, information that might be available at some other, nearby nodes. Because it is such a fundamental operation, efficient query propagation is, therefore, of significant importance when performance is considered. Flooding is a frequently used, albeit naïve, query propagation protocol, whereby each node, upon receiving a query for the *first* time, merely rebroadcasts it to all its neighbors, possibly with some jitter to reduce the probability of congesting the network. Reception of a previously seen query is ignored. Excluding failures, every node with a path to the source receives the query *at least* once and transmits it exactly once. Truncating the flood by using an expanding ring (such as TTL-based ring [13]), as is done, for example, in AODV, is merely a stop-gap measure that is useful only when the destination node or cached information happens to exist nearby. In general, as the number of network nodes increases, the cost of the flood increases in proportion as well.

In the absence of any information about the destination node, including cached information or even general location information, a route discovery query should be propagated to all nodes that are connected to the source of the query. But the basic observation is that it is certainly not necessary for every node to transmit the query, as is the case with flooding. This observation allows for significant improvements of many existing ad hoc network routing protocols. When operating in dense networks, the flooding-based propagation component of these protocols causes unnecessary transmissions, frequently referred to as "broadcast storms" [14].

Various ideas have been proposed to address the unnecessary transmissions, including probabilistic broadcast protocols [15], gossip-based schemes [16], planar-geometric optimizations [17], and distributed algorithms that compute cluster-heads or connected dominating sets [18, 19], among many others. Excellent surveys on this topic could be found in [14], [20], and [21]. We submit that each of these protocols shares the same basic density-independent property. Indeed, this density-independent property has been introduced by some of the earlier protocols; for example, by the *Zone Routing Protocol* (ZRP) and more specifically, by its *bordercast* operation. Furthermore, we emphasize that aggregation of zone-wide state at each network node is a fundamental building block upon which efficient and scalable network protocols can be built. In the following section, we briefly describe this zone-wide information and how it could be used for scalable route discovery.

3. ZONE MAINTENANCE

Our network model is that of a flat ad hoc network. Our network consists of n independently moving nodes within an area of A[m^2]. Nodes can directly communicate with other nodes that are located within their transmission radius of r [m]. (We assume here the *protocol reception model*. Extension of this work to the *physical reception model* is left for future work.) All nodes are equal in their capabilities, trusting one another, co-operating participants in the network, and there is no central "coordinator" node. (Indeed, the lack of a central entity is one of the biggest advantages of the ad hoc networking technology, as this prevents a single point of failure.) For the communication medium, we assume a single, shared channel and, for simplicity, that links are bi-directional.

A *zone* is defined per every node; the zone of a particular node A consists of the set of nodes that are within R network hops from A. R is referred to as the *zone radius*. Each node knows what are the nodes within its zone, as well as the links among those nodes. We assume in this article that R is a network-wide constant. However, in principle, the zone radius may change over time and need not be homogeneous across the entire network (see, for example [5]). The *border* of a zone is defined as the set of nodes that are *exactly* R hops away.

Zone-wide information is collected and maintained at each node by a protocol called IARP (*I*ntr*A*-zone *R*outing *P*rotocol). The result of executing an IARP protocol is local knowledge of the zone; i.e., the identity of all the nodes, N_z, that are within R hops as well as the state of the links, E_z, among them. For a sparse network, $|E_z| = O(|N_z|)$. However, as the network density increases the size of links' set grows quadratically, – i.e. $|E_z| = O(|N_z|^2)$ – as does the cost of the zone maintenance protocol. This places a limit on the size of the zone, particularly when the membership of the zone and its links' states change rapidly, such as is the case with high node mobility. Therefore, it is important for the zone maintenance protocol to be as efficient as possible. We propose here two possible zone maintenance protocols: *IARP-node* and *IARP-zone*.

The zone maintenance protocol receives information about its immediate neighbors and link states from the *Neighbor Discovery Protocol* (NDP). NDP can be either a simple heartbeat-based node discovery protocol running at the network layer, or some other mechanism operating lower on the protocol stack, such as the MAC layer. It provides information about the first hop with *L*ink-*U*p and *L*ink-*D*own notifications whenever a link to a neighbor is discovered or is lost, respectively.

The IARP-node protocol broadcasts these *local* link state changes in update packets with the TTL set to R hops. Each such update packet is sequenced at its source. Every node that receives such a packet for the first time, simply decrements the TTL and rebroadcasts it to all its neighbors, unless the remaining TTL has reached zero. In this manner, the entire zone learns of the zone's topological changes. The protocol could also incorporate a jitter delay, so as to lower the probability of congestion; such congestion could lead to collisions at the MAC layer and to buffer overflow at the network layer. Furthermore, addition of the jitter delay can also allow accumulation of a few additional changes at the source node before sending the update packet. For nodes that join the network, there is a mechanism to acquire zone state from a neighbor. Finally, there is also a *periodic broadcast* of the full links' state at a larger time interval, which allows to update the topology with changes that otherwise would have not been propagated, such as, for example, expiration of links whose both endpoints have either left the network or have failed.

When the zone is stable, the IARP-node protocol is silent, except for the infrequent periodic broadcasts. However, in the worst case, for a zone of k nodes and l links that are changing all the time, each of the k nodes will transmit a packet containing $O(l)$ link changes that will be rebroadcast by $k-1$ other nodes in the zone. In other words, the worst case of the IARP traffic load per "update period" is $O(k^2l)$ information and $O(k^2)$ packets per zone, or $O(kl)$ information and $O(k)$ packets per node, where k and l are functions of the zone radius R, the average network density n/A, and the transmission radius r. (Note that although there is no set "update period" in IARP-node, we refer here to this term as the average time between two conscutive updates, which may be restricted by the delay jitter mentioned above.)

The IARP-zone protocol takes a different approach. Every node broadcasts only a *single* packet at every update period, if there has been any change to the link state. Each packet has a TTL of 1, but contains the known changes of the entire *zone-wide* link state since the last transmission. The same packet is received by neighbors from multiple directions, but each one prunes out the information relevant for its zone, based on shortest network distance calculations. As in the IARP-node protocol, each link update is sequenced at the link source, there is a forced periodic update at a longer time interval that permits link expiration, and the protocol is otherwise silent if the zone is stable. In the worst case of an all-changing zone, there is still $O(kl)$ information sent per node, but now only in a single packet. This larger packet may be readily sub-divided into a few independent, smaller packets, if the information happens to exceed the link MTU. However, the advantage of transmitting zone-wide changes in batches is retained: multiple links updates contain common endpoints, permitting a more efficient encoding. More specifically, within each packet, we build a table of endpoints, which contains IP addresses, link source sequence numbers, and any query-specific node state. The packet then contains a list of source endpoint indices, each with a sub-list of destination endpoint indices, representing all of the links. Bi-directional links are encoded using a reversal bit. We will evaluate the gain of this efficient encoding in this article.

Finally, though we have not studied it in this article, it may be possible to further improve the performance by intentionally omitting some links in dense networks. Beyond a certain threshold, additional links will, with high probability, not affect the connectivity within the zone, nor appreciably degrade the bordercast performance.

4. BORDERCAST-BASED ROUTE DISCOVERY

In this section, we overview the bordercast protocol, which can be used in place of the flooding-based query propagation found in many existing ad hoc network routing protocols. We show how bordercasting uses the zone information to efficiently propagate a route discovery query. Our discussion here is based on [5].

Like flooding, the bordercast protocol propagates the query across the entire network. However, while flooding attempts to iteratively relay the query to any neighbors that have not heard it yet, the bordercast protocol seeks to iteratively relay the query to any of its *border* nodes that have not seen the query yet. Thus, while all the neighbor nodes receive the query broadcast, not all of them need to retransmit it on its way to the border nodes. If we consider the nodes within the zone to be a micro-ad hoc network with approximate area of $A_z = \pi(Rr)^2$, it is clear that the cost of propagating the query across the zone is a function of its area and not of the number of nodes within it. Therefore, because of the broadcasting nature of wireless communications, the bordercast protocol broadcasts the query to all its neighbors, but selects only a few to re-bordercast the message. The other neighboring nodes are silent recipients.

It is important to understand that bordercasting does *not* actually attempt to deliver the query to every node within its zone. Rather, its objective is to relay the query only to all *border* nodes that have not yet received the query. The protocol still works correctly, because each node in the network maintains information about all the nodes within its zone and can answer queries about them or, at the very least, forward the query directly to the desired node. Thus, we say that a node is *covered* if the query has been received by any node within its zone. The bordercasting protocol ensures that every node in the network is covered by the query. With larger zone radii and at larger network densities, a significant fraction of the network may be covered without actually receiving the propagating query.

We present in Figure 1 the proposed bordercast algorithm, which is executed independently in every node. A bordercast query propagation is initiated with a call to BORDER-CAST, and incoming messages are processed by RECEIVE-BORDER-CAST. The ZONE and BORDER functions return the sets of zone and border nodes, respectively, around a given node, based only on the local information.

The local state of each node consists of the zone-wide information as well as information regarding which nodes within the zone have already been covered by a query coverage. This query coverage state maintained by each node lies at the heart of the protocol and directs its behavior. As with flooding, a bordercasting node should transmit a query at most once. To ensure this property, the protocol marks all the nodes of its zone, including the border nodes, as covered after a query has been relayed. (Note that this is a local operation, which updates coverage in the local node state.)

When all the border nodes are covered, there is no need to relay the query. Similarly, when a node receives a bordercast message from some neighbor we mark all the locally known nodes in the zone of that neighbor as covered. In other words, we update our local query coverage state to indicate that all nodes within our zone that are within R hops of the sender node have been covered by the query. Thus, the coverage state directs the query outward, toward an expanding border.

Each bordercast message contains the query to be relayed, a source address, and a list of target addresses. The bordercast message is always broadcast and the list of target addresses is always selected from among the neighbors of the recipient. If the

```
type
  msg := {
    src: address,
    targets: set of address,
    query: query
  }
  info := { addr: address, ... }
state
  local: info
  zone: {
    nodes: set of info
    links: set of { src: address, dst: address }
  }
  covered: map query to set of address

RECEIVE-BORDERCAST(m: msg):
  ▷ accumulate query coverage information
  covered[m.query] $\overset{\cup}{\longleftarrow}$ ZONE(m.src)
  if local $\notin$ m.targets then
    covered[m.query] $\overset{\cup}{\longleftarrow}$ ZONE(local.addr)
  ▷ process query and wait to avoid collision
  ▷   and hear other broadcasts
  PROCESS-QUERY(m.query)
  sleep JITTER-TIME()
  ▷ relay query to any uncovered border nodes
  border ← BORDER(local.addr)−covered[m.query]
  if border $\neq \emptyset$ then
    msg ← msg {
      src ← local.addr,
      targets ← SELECT-NEIGHBORS(border),
      query ← m.query
    }
    BROADCAST(msg)
    covered[m.query] $\overset{\cup}{\longleftarrow}$ ZONE(local.addr)

BORDERCAST(q: query):
  msg ← msg {
    src ← NULL,
    targets ← { local.addr }
    query ← q
  }
  RECEIVE-BORDERCAST(msg)
```

Fig. 1. The bordercast query propagation protocol.

receiving node is not one of the selected targets, it is implied that it is not required for the query to reach the border nodes of the recipient. Furthermore, by the definition of a zone and the manner by which targets are selected, each of receiver's border nodes must be a border node of one of the targeted neighbors. Thus, when a node receives a bordercast message and is not in the target set, the protocol marks its entire zone as covered. The effect is, as above, to ensure that a non-targeted node will remain silent, since it is not required for query propagation.

In contrast, a targeted neighbor node that receives a query is responsible for re-bordercasting it to any of its uncovered border nodes. The protocol pauses for a short random interval before doing so to lower the chances of congestion. It also allows the node to receive other bordercasts that may be occurring at neighboring nodes during this time. Learning that the query was processed at other nodes, may partially or completely cover the remaining uncovered border nodes and either reduce the number of targets required or perhaps eliminate the need to relay the query entirely.

Finally, before broadcasting the query, the protocol must select the target neighbors: SELECT-NEIGHBORS(). For each uncovered border node, there must be at least one neighbor chosen in the direction of that border node. In other words, the network distance between the selected target neighbor and the uncovered border node must be $R-1$ hops. There may be many sets of nodes that meet this criterion, and we would like to find the smallest such set. Since this matching problem between closest neighbors and their uncovered border nodes is NP-complete, we implement a greedy approximation. The neighbor node that covers the greatest number of uncovered border nodes is chosen first. We cover the border nodes closest to the chosen target and iterate until all the border nodes have been covered. The query is then broadcast with this list of targets.

Figure 2 depicts a bordercast in progress. The zone radius is equal to 2. Our query was initiated from node S, and it arrived via G and B to node A. Note that node B targeted only nodes A and C with its query broadcast, in order to reach its border node set $\{H, D, E\}$. Node F is not targeted since node A already covers it, in addition to covering D. Thus, when F receives the query from B, it processes it, notes that its entire zone is covered (since F is not a target of the query), and, therefore, remains silent. Similarly, G receives the query from B and remains silent, since G has already forwarded the query (to B, in this case) and marked all the nodes of its zone as covered. When A receives the query from B, it locally marks all the nodes of B as covered. Covered nodes are represented as black nodes in the figure. S is not marked, since A does not know about it. I, J, and K are also not marked, since they lie outside the zone of B. Thus, out of A's border set, which is $\{G, H, I, J, K\}$, only $\{I, J, K\}$ remain uncovered. Assuming that no other bordercasts are heard while A pauses, A will broadcast the query with $\{D, E\}$ as targets.

5. THEORETICAL BOUNDS ON OPTIMAL PROPAGATION

Given a network, $G = \langle N, E\rangle$, where N denotes the set of nodes distributed uniformly across an area A, and E denotes the links between them, we wish to

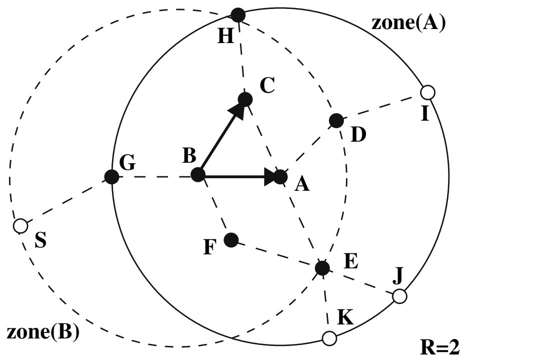

Fig. 2 A bordercast in progress: dark circles represent nodes that node A believes are covered, nodes within $Zone(A) \bigcap Zone(B)$. The query will now proceed towards the non-covered border nodes of A, namely I, J, and K.

determine the optimal number of transmissions, t, required to propagate a route query from a source node, $n_0 \in N$, to all the other n_0–connected nodes, $N^c_{n_0} \subseteq N$. Note that we refer to the term "connected" in the graph-theoretic sense, i.e., possibly through multiple network hops. Clearly, $t \leq |N^c_{n_0}|$, since this is the cost of flooding; i.e., each n_0–connected node transmits the query once.

As network nodes are placed at random locations, in the worst case, we need to cover the entire area with query broadcasts. A single wireless broadcast can be received over an area $a = \pi r^2$ around the transmitter, where r is the transmission radius, as defined above. Thus, even if we could place our transmitters at will, we still would need at least $t \geq (1 + \epsilon)A/a$ transmitters, where ϵ accommodates for the packing inefficiency of the transmission coverage circles. For convenience, let $\rho = (1 + \epsilon)A/a$.

As we increase the size of the network, by adding nodes at random locations within A, a number of things occur. The density of the network increases in proportion. The network becomes fully connected with high probability, so that $N^c = N$. And, the probability of having a node within δ distance of any chosen transmitter point, $P_\delta(t) = 1-(1-\pi\delta^2/A)^{|N|}$, approaches 1. To propagate our query over the entire area A, we select transmitter points uniformly across A, such that the distance between any two is at most r. We can already cover the entire area with ρ circles of radius r. Now place nodes at the centers of those circles and connect those nodes with an additional $\rho-1$ intermediate nodes. Thus, even as the number of nodes in the network increases, we can still flood the entire network at a cost of $t \leq 2\rho - 1$ transmissions! Thus, $t = \Theta(A)$.

The upper bound we have just derived may not be achievable, since we selected transmission points that may not actually correspond to location of nodes within the original network. Therefore, consider the *dominating set* over G, $\tilde{N}$. (A set of nodes, $\tilde{N}$, is dominating over G, if and only if every node in G is either an element of $\tilde{N}$ or is a neighbor of some node in $\tilde{N}$.) The *dominating number* of our network, or the size of the minimum dominating set, is $\gamma(G) \leq \rho$. If there are any more nodes in a dominating set than this upper bound, then at least one member node covers an area around it that is already completely covered by the remaining nodes in the set, and that node can be removed to form a smaller dominant set. We then construct a minimum *connected dominating set*, $\tilde{N}^+$. A connected dominating set is a dominating set with nodes from G added to connect the existing dominating nodes, whenever those nodes are connected in G. Again, as stated above, we refer to the term "connected" in the graph-theoretic sense, i.e., possibly through multiple network hops. Thus, by construction, the number of connected components in the connected dominating set will be equal to the number of connected network components in G. Note, also, that by the definition of the dominating set, we will need to add at most two nodes for every dominating node to form the connected dominating set: the neighbors of the two dominating nodes we are connecting. Thus, $|\tilde{N}^+| \leq 3|\tilde{N}|$, and the minimum connected dominating number of G is $\gamma^+(G) \leq 3\rho$. Finally, since the minimum connected dominating set contains all the nodes that must transmit the query in order to completely propagate it through G, we retain that $t = \Theta(A)$.

To summarize, the minimal number of transmissions required to propagate a query is proportional to the area, A, of the network. It is independent of the number of network nodes, or equivalently, of the network density. Adding a node in an area that is already covered by an existing set of propagating nodes, should not increase the number of transmissions required. Next, we show that the performance of the bordercast (Figure 1) protocol is optimal.

6. PERFORMANCE EVALUATION

The evaluation of bordercast was done using the SWANS simulator [22, 23], because of its scalability property of being able to simulate very large networks. To the best of our knowledge, the scale of the simulations required for this work exceeds the capability of most other publicly available general purpose simulation tools by at least an order of magnitude.

In the first experiment, we measure the relative unit cost of each of the zone maintenance and query propagation protocols discussed in Section 3 for networks of different sizes, but at constant density. We generate the network by placing wireless nodes randomly within a square area and increase the area size in proportion to the number of nodes. Each network node is turned on at time $t = 0$ with no information other than its own unique address. The protocol stack at each node comprises a wireless radio, the 802.11b MAC, IPv4 network, UDP[1] transport, and our test application components that generate traffic. Other relevant protocols, such as NDP, for example, have also been implemented as part of the simulation. Note that since the various protocols perform link-level broadcasts, the 802.11 collision avoidance and retransmission mechanisms do not play a role in these simulations. However, each of the simulated protocol incorporates jitter to reduce the probability of congestion and is already resilient to point failures, either due to repetition (NDP) or due to a flooding-like behavior (IARP and BRP). The simulator accounts for signal interference, but neither for shadow fading nor for Raleigh fading.

We measure the *unit* packet cost of a protocol, which is defined as the number of packets sent throughout the network to perform a single round or operation. The unit cost of the IARP protocols is the number of packets for the protocol to quiesce, such that every node has learned its complete zone state. Since the nodes begin with no information, this measurement represents the worst case (or, alternatively, the largest mobility case) for the protocol, which is when the information about *all* the zone links must be communicated. The unit cost of a bordercast operation is the number of packets transmitted to cover the entire network with a query. For any fixed density, both of these protocols grow linearly with the area of the network or, equivalently, linearly with the number of nodes in the network, since we keep the density constant.

In Figure 3, we compare the performance of query propagation of flooding and bordercasting as a function of node density. Each point represents the average of at least 10 runs. The graph shows how a flooding-based propagation grows in proportion to

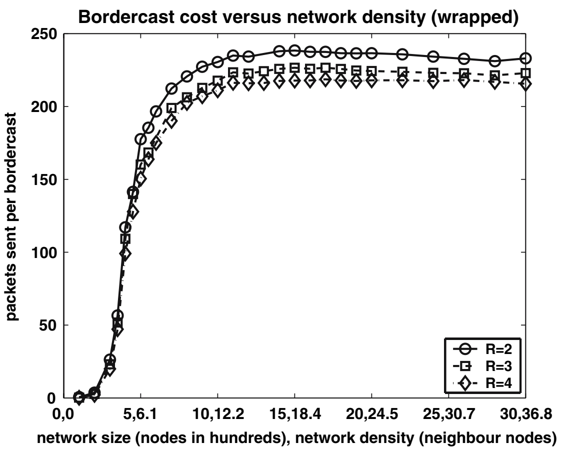

Fig. 3. Unlike flooding, for fixed network area, the bordercast cost of query propagation is independent of the network density.

the number of nodes, but that bordercast is density independent. In other words, adding more nodes to the network does not increase the cost of bordercasting. Note that the left side of the curves represents a very sparse network that is poorly connected. In this case, both flooding and bordercast are simply not able to span the area because of the lack of intermediate nodes. The x-axis shows both the total number of nodes, as well as the network density in terms of the expected average number of neighbors per node. This number of neighbors is computed from the node density and the transmission radius, i.e., $E[l/k] = \pi r^2(n/A)$. It matches the values reported by NDP in simulation. Finally, we observe that by setting the zone radius to 1, the performance of bordercast degenerates to flooding. This is expected, since with $R = 1$ the border set becomes the neighbor set. The slight advantage of bordercast over flooding is merely an edge effect: edge nodes do not retransmit the query under the bordercast protocol, because all of their neighbors are already covered.

Increasing the zone radius improves the performance of bordercast only minimally, as shown in Figure 4. To eliminate the impact of the edges, the results in the figure were obtained with opposite edges wrapped around to create a torus.

Some smaller improvements due to larger zone radii can be seen within the core of the network, where the protocol can sometimes "avoid" regions that are sufficiently sparse. These are, in some sense, "internal edges" of the network. To highlight this phenomenon, we present Figure 5, a spatial plot of one of the smaller ($n = 800$, $R = 4$) simulations plotted in Figure 4. The heavy circle at the bottom right highlights the query source. The circle's radius

[1] The results are not expected to be significantly different for other transport-layer protocols.

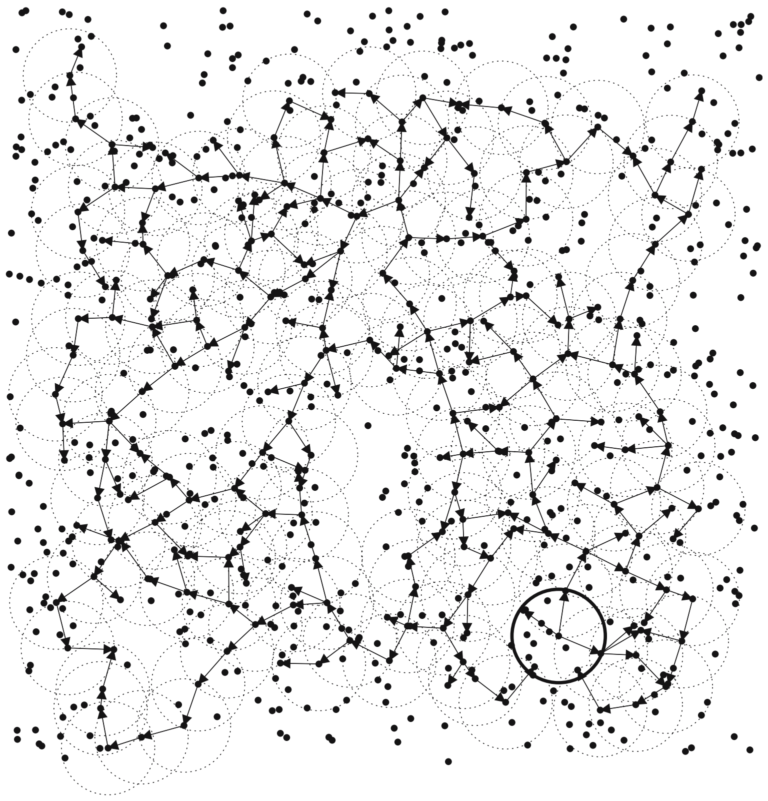

Fig. 4. Discounting edge effects, bordercast cost is not significantly affected by increased zone radius.

equals the transmission radius, and lighter circles surround nodes that transmitted the query. Thus, all nodes within circles receive the propagating query. Arrows represent the targeted neighbors, which may relay the query, if necessary. Notice that many nodes are not within these circles, which means they never receive the query. However, *all* the nodes are contained within 4 hop-sized circles around nodes that actually receive the query, a close approximation of the actual zones, indicating that the protocol is covering the whole network. (Zone circles are omitted to improve the clarity of the figure.) One can also see an example of internal nodes in the center that are covered, but without receiving the query.

Another interesting phenomenon shown by the spatial plot is that the targeted neighbors are usually found near the boundaries of the transmission circles, even though there is no location information available to the protocol. This occurs because the bordercast protocol tries to minimize the number of targeted neighbors by selecting those neighbor nodes that are closer to (i.e., $R-1$ hops from) the *largest* number of uncovered border nodes. A beneficial consequence is that the chosen targets are more likely to be found at the outer limits of the transmission range of the query relaying node.

We now turn to the cost of maintaining the required zone state at each node. Shown in Figure 6 is the effect of increased density on the number of packets sent by our two zone maintenance protocols: IARP-node and IARP-zone. We have plotted the cold-start scenario, where all the links in a zone must be discovered and added. In other words, this represents the worst case in terms of mobility in the

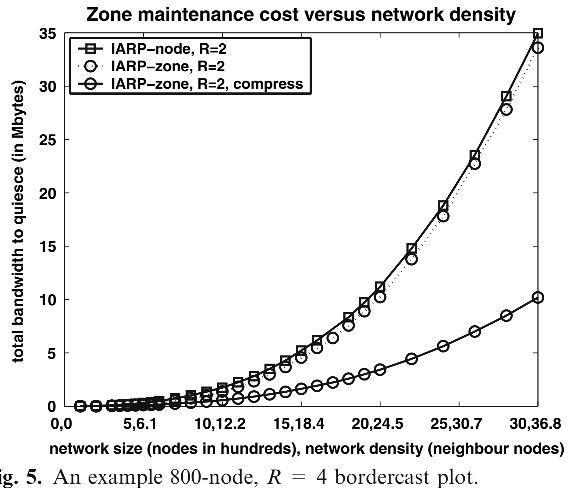

Fig. 5. An example 800-node, $R = 4$ bordercast plot.

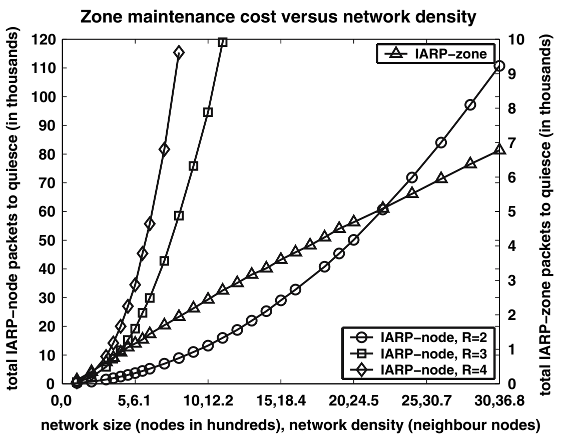

Fig. 6. Cost of zone maintenance increases dramatically with increased density and zone radius.

network. As expected, the total number of the IARP-node packets (left axis) increase quadratically with the density, i.e., $O(k^2)$, where k is the number of nodes within the zone: each node sends $O(k)$ packets and there are more nodes in the network. In turn, the number of nodes within a zone increases quadratically with the zone radius, and thus the total number of packets in the network increases with the fourth power of the zone radius! The $R = 4$ curve is actually significantly lower than expected due to the large number of collisions caused by the flurry of link state updates in the large zone. However, since IARP-node is a flooding protocol, lost packets are often retransmitted by other neighbors and, with high probability, any loss of information is local only. Nevertheless, it is interesting that the missing link state does not appreciably affect the bordercast performance discussed earlier. As expected, the number of IARP-zone packets (right axis) increases linearly with the density, since each node sends a constant number of packets, forwarding the "waves" of new information traveling in each direction. The number of IARP-zone packets is not affected by the zone radius, since it merely passes along more information within the same packet.

In Figure 7, we compare the average packet sizes of the two zone maintenance protocols. IARP-node packets (left axis) are very small, because they contain the link states of only a single node. In contrast, IARP-zone packets (right axis) contain information about changes to the entire zone. The size of these packets is proportional to the number of links, which grows quadratically with density and with the fourth power of the zone radius. However, the more efficient encoding of this information saves around 60% of the packet size at the density of 30 neighbors per node, independent of the zone radius. The compression ratio increases with increasing network density, since the proportion of common link endpoints increases with the density.

In Figure 8, we show the total bandwidth consumed by each of the zone maintenance protocols to quiesce starting from cold-start state. As before, this represents the worst case in terms of network mobility. This figure integrates the two trends:

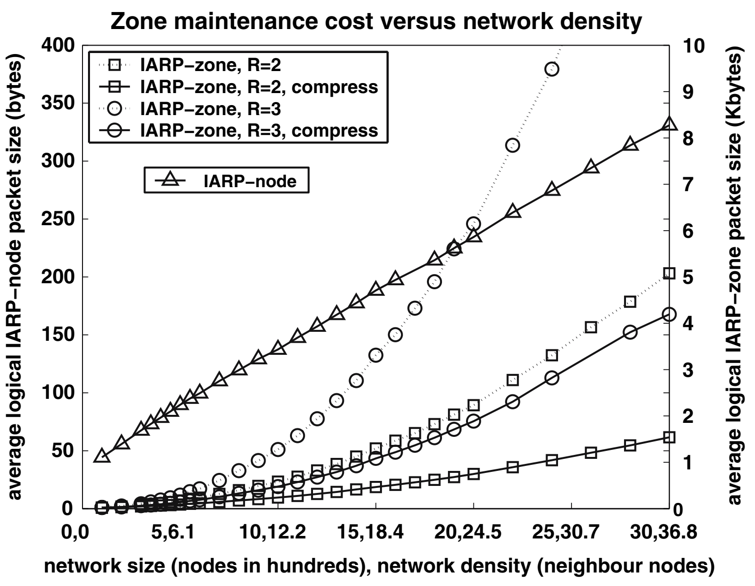

Fig. 7. Aggregated link state can be encoded efficiently to reduce the average size of update packets.

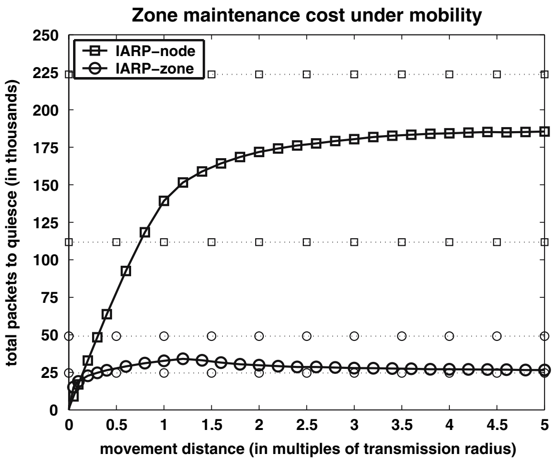

Fig. 8. Comparing the two zone maintenance protocols shows that zone-widelink update aggregation and efficient encoding can be highly beneficial.

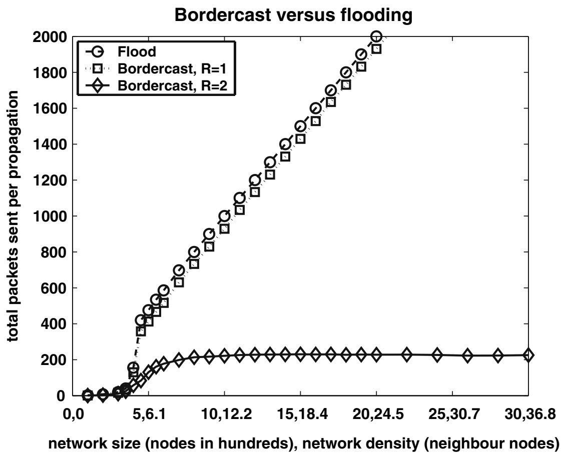

Fig. 9 Mobility increases the cost of zone maintenance.

smaller number of packets but larger average packet size of the IARP-zone protocol. The two protocols transmit the same amount of information, but IARP-zone outperforms IARP-node through efficient encoding of the update packets. The plotted data show the packet payloads, *not* including packet headers, to allow a meaningful comparison. Including packet header overheads would further benefit the IARP-zone results, since it transmits fewer, larger packets.

We have considered, until now, the cost of our protocols under the worst case mobility scenario. We now discuss the behavior at lower topological rate of change. Both, the node discovery and the bordercasting protocols are unaffected by the rate of topological change in the network. However, mobility can change the zone link state and its membership, which will necessitate zone maintenance. To measure this cost, we create a random network and allow the zone maintenance protocols to quiesce. We then move each node a fixed distance in a random direction and measure the number of packets required to update the zone information. This provides us with a unit cost for zone maintenance as a function of mobility. Figure 9 shows the total number of IARP-node and IARP-zone packets for a network of 10,000 nodes. We see that the number of packets grows in proportion to the number of changes in the zone, but has an upper-bound that corresponds to the total amount of zone information. The lower line below each curve is the unit cost of each respective zone maintenance protocol from cold start, shown in the previous figures. The upper line above each curve is twice this value. The additional transmissions above the lower line are due to link state *drop* notifications, which do not occur from cold start. Notice, however, that the IARP-zone cost actually tends to decrease toward the lower line. This peculiar phenomenon results from IARP-zone pruning its link state after incorporating each update packet. This pruning is essential, because under IARP-zone, nodes will receive some link state that is not relevant for their zone. And, if this new link state is forwarded on, the zone information at each node will eventually include the entire network. An added bonus of this pruning is that link failure notifications are suppressed when a node has traveled so far out of its original zone as to be irrelevant.

To summarize, there exists an upper bound on the zone maintenance cost, and this bound is independent of the rate of mobility. The incremental zone maintenance cost, albeit a function of node density, is expected to be very low. And, the route discovery cost is, as shown earlier, independent of the node density.

7. DISCUSSION OF RESULTS

In what follows, we draw certain inferences and make recommendations regarding the use of bordercasting in ad hoc networks:

7.1. Bordercast rather than flood

Bordercast propagates queries with cost proportional to the the area of an ad hoc network (or equivalently, the network diameter), regardless of the density of nodes (or equivalently, the number of nodes). It can replace the flooding-based query propagation found in many ad hoc routing protocols, as well as in other network querying operations, such as resource discovery and some sensor network data-aquisition operations.

7.2. Set the zone radius to 2 hops

Setting a larger zone radius results in little bordercast improvement and substantially increases the cost of zone maintenance, especially at higher network densities. Note that there may be other reasons to have a larger zone, including proactive route maintenance and a high rate of route requests relative to the rate of link changes (i.e., a mostly stationary network). However, one of the contributions of this article is the finding that bordercast performance is not among those reasons.

7.3. Aggregate and compress link state updates

IARP-zone outperforms IARP-node, because it aggregates link state updates, thus transmitting fewer packet headers and reducing the average packet size through efficient encoding of links.

7.4. Properly set the transmission power

The transmission power should be set to reduce the transmission radius, while maintaining network connectivity. On the one hand, shorter transmission radius settings reduce the zone membership and the cost of zone maintenance both in terms of packets and in terms of transmission power per packet. However, on the other hand, a shorter transmission radius implies a larger network diameter (in hops), which in turn implies longer latencies and more bordercast packets to cover the same area! (Note that the setting of the transmission radius also affects the capacity of the network due to interference.)

This final point raises the question of whether bordercast is necessary at all. In essence, bordercasting reduces the number of neighbors at the network level, rather at the physical level. If one can set the transmission radius small enough, such that the (physical) network degenerates into a tree (i.e., an average of two neighbors per node), then flooding would be equivalent to bordercasting in this case. Of course, one ill effect of such a network is that the average route length along such a tree may increase dramatically. In general, the benefits of bordercast stem from its ability to silence neighbors that are not required to propagate the query. If all the neighbors are required to relay the query, due to sparseness of the network, then, indeed, bordercast would not exceed the performance of flooding. However, lowering the transmission radius is not always possible for a number of reasons. These include fixed hardware power settings, increased probability of link breakage and route failure, overhead of power adaptation protocol, increased number of hidden terminals, increased network diameter, increased average route length and packet latencies, and decreased route diversity. Thus, if the number of neighbors cannot be reduced at the link level, bordercast presents a viable alternative to do so at the network level, preventing unnecessary transmissions and “broadcast storms” during propagation.

In this article, we have focused exclusively on wireless ad hoc networks. Other networking environments that utilize flooding primitives may benefit from bordercast as well. The primary requirements are only that multi-hop propagation is used and that more neighbors receive a message than are required to relay it to achieve complete network coverage.

8. CONCLUDING REMARKS

The scalability of ad hoc networks – the ability to efficiently route and transmit packets across ad hoc networks as they grow in size – is a key research challenge. In this article, we analyzed the cost of discovering a route to some desired destination node using only its unique address. We have presented the design of a bordercast protocol, a query propagation protocol that is density-independent, and have proven that it is optimal. Bordercast can improve the performance of many existing routing protocols in dense networks by replacing their flooding-based query propagation. Our results also show that: optimal bordercast performance does not require zone radii larger than two hops; the cost of zone maintenance is proportional to network mobility and is bounded; and, that aggregating and efficiently encoding link state updates can substantially reduce the overhead of zone maintenance. Finally, we have highlighted the importance of efficient zone-wide aggregation as a building block for scalable ad hoc network protocols.

ACKNOWLEDGMENTS

The work on this project was supported in part by the U.S. National Science Foundation under the grant numbers ANI-0329905 and CNS-0626751, by the DoD MURI (Multidisciplinary University Research Initiative) Program administered by the Office of Naval Research (ONR) under contract N00014-00-1-0564, and by the MURI Program administered by the Air Force Office of Scientific Research (AFOSR) under contract F49620-02-1-0217. The authors thank Robbert van Renesse, Benjamin Atkin, and Chaitanya Swamy for their ongoing help and valuable suggestions regarding this work. We also acknowledge Kelwin Tamtoro, Clifton Lin, Ben Viglietta, and Mark Fong for their work on various SWANS and JiST components. We are indebted to Alin Dobra and Paul Francis for allowing us to use their computational resources to run simulations.

REFERENCES

1. P. Gupta, and P. Kumar, The capacity of wireless networks, *Trans. on Info. Theory,* Vol. 46, pp. 338–404, March 2000.
2. M. Grossglauser, and D. Tse, Mobility increases the capacity of wireless adhoc networks, *IEEE/ACM Transactions on Networking,* Vol. 10, pp. 477–486, Aug 2002.
3. O. Arpacioglu and Z. J. Haas, On the Scalability and Capacity of Wireless Networks with Omnidirectional Antennas. IPSN'04, Berkeley, CA, April 26–27, 2004.
4. P. Samar, M. Pearlman, and Z. Haas, Hybrid routing: The pursuit of an adaptable and scalable routing framework for ad hoc networks. in *Handbook of Ad Hoc Wireless Networks*, CRC Press, 2003.
5. P. Samar, M. R. Pearlman, and Z. J. Haas, Independent zone routing: An adaptive hybrid routing framework for ad hoc wireless networks, *IEEE/ACM Transactions on Networking,* Vol. 12, No. 4, pp. 595–608, August 2004.
6. D. B. Johnson and D. A. Maltz, Dynamic source routing in ad hoc wireless networks. in *Mobile Computing*, Kluwer Academic Publishers, 1996.
7. C. E. Perkins and E. M. Royer, Ad hoc on-demand distance vector routing, in *Workshop on Mobile Computing Systems and Applications* pp. 90–100, Feb. 1999.
8. B. Bellur and R. G. Ogier, A reliable, efficient topology broadcast protocol for dynamic networks, in *IEEE INFOCOM* '99, March, 1999.
9. T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, A. Qayyum, and L. Viennot, Optimized link state routing protocol, in *IEEE INMIC*, 2001.
10. M. Joa-Ng, I. -T. Lu, A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks, *IEEE Journal on Selected Areas in Communications*, Vol. 17, No. 8, pp. 1415–1425, August, 1999.
11. V. Ramasubramanian, Z. J. Haas, E. Sirer, SHARP: A hybrid adaptive routing protocol for mobile ad hoc networks, in *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, June 2003.
12. N. Navid, W. Shiyi, and C. Bonnet, *HARP: Hybrid Ad hoc Routing Protocol International Symposium on Telecommunications (IST).*
13. Z. Cheng and W. Heinzelman, Flooding strategy for target discovery in wireless networks, in *ACM MSWiM*, Sept. 2003.
14. S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, The broadcast storm problem in a mobile ad hoc network, in *ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*, ACM Press, pp. 151–162, August 1999.
15. Y. Sasson, D. Cavin, and A. Schiper, Probabilistic broadcast for flooding in wireless mobile ad hoc networks, in *IEEE Wireless Communications and Networking Conference*, March 2003.
16. Z. J. Haas, J. Y. Halpern, and L. Li, Gossip-based ad hoc routing, in *IEEE INFOCOM* '02, June 2002.
17. V. K. Paruchuri, A. Durresi, D. S. Dash, and R. Jain, Optimal flooding protocol for routing in ad hoc networks, in *IEEE Wireless Communication and Networking Conference*, March 2003.
18. S. Guha, and S. Khuller, Approximation algorithms for connected dominating sets, *Algorithmica,* Vol. 20, pp. 347–387, 1998.
19. J. Wu, H. Li, A dominating set based routing scheme in ad hoc wireless networks, in *International Workshop of Discrete Algorithms and Methods for Mobile Computing and Communications*, August 7–14, 1999.
20. B. Williams and T. Camp, Comparison of broadcasting techniques for mobile ad hoc networks, in *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, pp. 194–205, 2002.
21. I. Stojmenovic, M. Seddigh, and J. Zunic, Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks, *IEEE Transactions on Parallel and Distributed Systems,* Vol. 13, No. 1, pp. 14–25, 2002.
22. R. Barr, Z. J. Haas, and R. van Renesse, JiST: an efficient approach to simulation using virtual machines. *Software Practice & Experience*. Vol. 35, No. 6, pp. 539–576, John Wiley & Sons, May, 2005.
23. R. Barr, Z. J. Haas, JiST/SWANS website http://jist.ece.cornell.edu, 2004.

Zygmunt J. Haas received his B.Sc. in EE in 1979 and M.Sc. in EE in 1985. In 1988, after earning his Ph.D. from Stanford University, he joined the AT&T Bell Laboratories, Network Research Department. There he pursued research on wireless communications, mobility management, fast protocols, optical networks, and optical switching. From September 1994 till July 1995, Dr. Haas worked for the AT&T Wireless Center of Excellence, where he investigated various aspects of wireless and mobile network technologies. In August 1995, he joined the faculty of the School of Electrical and Computer Engineering at Cornell University, where he is now a Professor and the Associate Director for Academic Affairs.

Dr. Haas is an author of numerous technical conference and journal papers and holds eighteen patents in the areas of high-speed networking, wireless networks, and optical switching. He has organized several workshops, delivered numerous tutorials at major IEEE and ACM conferences, and serves as editor of several journals and magazines, including the IEEE Transactions on Networking, the IEEE Transactions on Wireless Communications, the IEEE Communications Magazine, and the ACM/Kluwer Wireless Networks journal. He has been a guest editor of IEEE JSAC issues on "Gigabit Networks," "Mobile Computing Networks," and "Ad-Hoc Networks." Dr. Haas served as a Chair of the IEEE Technical Committee on Personal Communications and is currently serving as the Chair of the Steering Committee of the IEEE Pervasive Computing magazine. Dr. Haas is an IEEE Fellow. His interests include: mobile and wireless communication and networks, performance evaluation of large and complex systems, and biologically inspired networks. His e-mail is: haas@ece.cornell.edu and his URL is: http://wnl.ece.cornell.edu.

Rimon Barr received an Honors B.Sc. from the University of Toronto combining studies in Computer Science and Immunology. At Cornell University, he received his M.Sc. for the design and development of MagnetOS, a distributed operating system for sensor networks, and his Ph.D. in Computer Science for demonstrating a novel technique that permits the transparent optimization and efficient execution of discrete event simulations over virtual machines. His research interests lie in the areas of databases, languages, distributed systems, and ubiquitous computing. In tandem with his research, Rimon completed an MBA at Cornell's Johnson Graduate School of Management, and became interested in micro-economics and quantitative finance. He currently works on Wall Street, building high-performance data analysis tools, researching predictive financial models, and trading.