Accommodating Transient Connectivity in Ad Hoc and Mobile Settings

Radu Handorean, Christopher Gill, and Gruia-Catalin Roman

Department of Computer Science and Engineering
Washington University in St. Louis
Campus Box 1045, One Brookings Drive
St. Louis, MO 63130-4899, USA
{radu.handorean,cdgill,roman}@wustl.edu

Abstract. Much of the work on networking and communications is based on the premise that components interact in one of two ways: either they are connected via a stable wired or wireless network, or they make use of persistent storage repositories accessible to the communicating parties. A new generation of networks raises serious questions about the validity of these fundamental assumptions. In mobile ad hoc wireless networks connections are transient and availability of persistent storage is rare. This paper is concerned with achieving communication among mobile devices that may never find themselves in direct or indirect contact with each other at any point in time. A unique feature of our contribution is the idea of exploiting information associated with the motion and availability profiles of the devices making up the ad hoc network. This is the starting point for an investigation into a range of possible solutions whose essential features are controlled by the manner in which motion profiles are acquired and the extent to which such knowledge is available across an ad hoc network.

1 Introduction

Conventional wired networks are increasingly being extended to include wireless links. It is important to distinguish between wireless links that are essentially stable in time and space, and those that are not. With wireless technology claiming an ever increasing share of the market, the interaction patterns between hosts need to be re-examined to address additional issues raised by this new environment. Former assumptions that made things easy or even possible in the wired setting may simply cease to hold in a wireless world. Wireless technology allows devices to become mobile. The topology of the network can change while applications are running and interactions between such mobile hosts are in progress.

1.1 Conventional Wired Networks

Modern wired computer networks provide a high degree of reliability and stability. The interaction between two different hosts in a wired network is expected

A. Ferscha and F. Mattern (Eds.): PERVASIVE 2004, LNCS 3001, pp. 305–322, 2004.

to succeed and it is not believed that the normal operation of a host can make it, even temporarily, unavailable to the rest of the network. Security and other considerations may cause an *application* residing on a host to become unavailable during its normal functioning but even such intervals tend to be reasonably well bounded in duration and frequency. Furthermore, the interaction between hosts is stable: the party a host interacts with is always found at the same IP address or under the same name, even when the name is resolved to different addresses at different times (*e.g.*, obtain its address from a DHCP server).

The stability of such an environment allows for efficient configuration of the traffic in the network. The total set of *possible* routes is mostly static, "hard-coded" in configuration files and scripts that do not change very often. While on-the-fly adjustment of information about the set of *available* routes is possible, such adjustments generally have a reactive flavor, *e.g.*, in response to network congestion. Fundamentally, modern routing protocols in wired networks depend at least indirectly on a stable set of physically reliable routes.

1.2 Mobile Wireless Networks

Nomadic networks, like the cellular telephony infrastructure or the pervasive wireless networks at universities, ensure communication between mobile devices such as phones, laptops, and PDAs. Nomadic networks do this by maintaining connectivity between each device and some part of a fixed infrastructure that acts as a communication liaison between such devices. Wireless devices connect to the infrastructure via access points connected to other access points connected to other wireless devices as they travel through physical space. Connectivity among devices is maintained by routing algorithms that deliver messages between mobile devices over the fixed infrastructure. The physical position of each device accessing the fixed infrastructure is important for routing packets to the closest access point to the device. The devices are always connected, but move in space.

In ad hoc networks the infrastructure for communication is made entirely of mobile hosts that interact in a peer-to-peer manner and route traffic for each other. There is no fixed infrastructure on which the mobile hosts can rely for message delivery. As long as two devices are connected (in direct contact or via multiple hops), they can exchange information and interact as if they were wired. In ad hoc networks, however, a stable multi-hop connection has severe temporal limitations, as well as spatial limitations due to the ad hoc routing algorithms that have to be carried out by hosts themselves.

1.3 Transition from Wired to Wireless Networks

Communication in ad hoc networks is significantly more challenging than in wired settings. Issues like multicast and routing, solved in wired networks, become difficult in ad hoc settings. Much effort has been dedicated to addressing these issues, to achieve a level of functionality comparable to the wired networks.

Multicast: In [1] multicast communication is approached from the perspective of ad hoc environments. The shared-tree mechanism [2], [3], [4] is analyzed and upgraded to exhibit an adaptive behavior, better suited for the changing environment. Flooding is considered in [5] as a possible approach to implementing multicast for highly dynamic environments. Multicast session ids can be assigned dynamically to hosts [6] and their ordering used to direct the multicast flow, organizing hosts in a tree structure rooted at the source of the multicast.

Routing: The loop-free, distributed, and adaptive routing algorithm presented in [7] is designed to operate in highly dynamic environments. This source-initiated, temporally ordered, routing protocol focuses its route maintenance messages to only the few nodes next to the occurrence of a topological change. Destination-sequenced distance-vector routing [8] and its extension ad-hoc on-demand distance vector routing [9] are improvements to the Bellman-Ford algorithm to avoid loops in routing tables. Sequence numbers help mobile nodes distinguish obsolete routes from the new ones. The wireless routing protocol (WRP) [10] addresses asymmetric links and uses piggybacking for route information updates.

This paper presents an approach to increasing communication capabilities in mobile ad hoc networks. It makes three main contributions to the state of the art in ad hoc networks. First, it formalizes the problem of message delivery in the presence of disconnections between hosts, and illustrates the formalism with examples. Second, it gives an algorithm for reliable message passing in the face of temporally discontinuous connectivity, exploiting host motion and availability profiles. Third, it offers an analysis of and refinements to that algorithm, based on the amount of information needed about the temporal patterns of connectivity.

The rest of this paper is organized as follows. Section 2 discusses issues of transient connectivity in mobile ad hoc networks, which motivate this work. Section 3 formalizes the problem of exploiting knowledge of host motion and availability for message delivery in ad hoc networks. Section 4 describes our solution and Section 5 analyzes it in terms of message, storage and computation complexity. Section 6 discusses the implications of our work. Section 7 presents related work, and Section 8 offers concluding remarks.

2 Transient Connectivity

In mobile ad hoc networks, communication paths are created, changed and destroyed at a much greater rate than in conventional or nomadic networks. The message routing infrastructure is composed of mobile participants, and therefore is itself subject to disconnections. A new approach is needed to describe the connectivity and communication paths between mobile hosts in ad hoc networks. In ad hoc networks, the mobility of hosts introduces the question of whether two hosts can be connected at all. The attention shifts from maintaining stable *connectivity* paths (*i.e.*, paths that route the traffic between two hosts that are connected to the network at the same time), to whether two hosts can *communicate* in an environment with transient connectivity (*i.e.*, a message can be

delivered from one host to another over a set of hosts which may or may not form a *continuous* path from source do destination at all times during the delivery).

In conventional networks, two hosts can exchange messages if and only if a stable path exists between them in the network topology. The ability to exchange messages in that setting is reflexive, symmetric and transitive, *at each hop and along the entire path*. However, in mobile ad hoc networks paths are not stable over time, as the connectivity between any two hosts can vary due to the motion of those hosts or due to their availability (*e.g.*, due to power-saving modes).

When two hosts are in direct contact, messages can flow between them symmetrically, *i.e.*, as peer-to-peer communication. Ad hoc routing extends this symmetry across multiple hops, but requires all hosts along the path to be connected throughout some continuous interval of time.

However, there are reliable message delivery paths in ad hoc networks that are not symmetric, and thus cannot be exploited by either peer-to-peer or ad hoc routing approaches. In particular, messages that traverse a link before it goes down may be delivered on that path, while messages that reach that link after the disconnection will not. Therefore disconnection introduces non-symmetric message delivery semantics for any path longer than one hop.

We leverage characteristic profiles of the motion and availability of the hosts as functions of time to accomplish disconnected message delivery. Thus, we do not need to rely on the availability of a reliable end-to-end communication route during the process of communication. The only thing we can rely on is pair wise connectivity of hosts, which can be inferred from the characteristic profiles.

2.1 Connectivity Intervals and Message Paths

Figure 1 shows an example of temporally discontinuous intervals of direct connectivity between hosts a, b, and c. At time t_0, hosts a, b, and c are disconnected. From t_1 to t_2, hosts a and b are connected and can exchange messages. From time t_2 to t_3 all hosts are disconnected, etc.

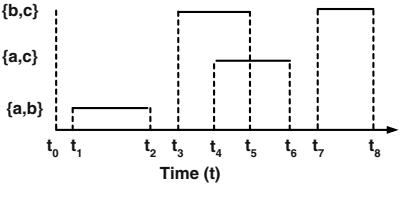

Fig. 1. Connectivity Intervals

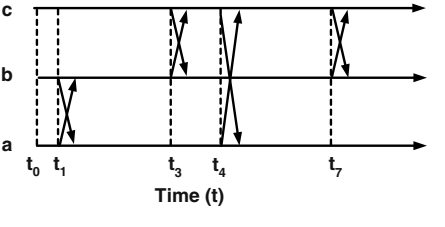

Fig. 2. Possible Message Paths

We define a *connectivity interval* for any two hosts to be a continuous time interval they can use for effective communication. In practice, a connectivity interval is likely to be shorter than the total time the two hosts can communicate directly, since the beginning and the end of the direct connectivity period may

incur overhead spent to establish contact and to disconnect without loss of data or inconsistency of state. However, because such overhead is a function of the connection and disconnection protocols used, for the sake of generality in this paper we assume the connectivity interval for two hosts is identical to the interval during which they can communicate. It is relatively simple to extend the formalism in Section 3 and the algorithm in Section 4 to consider such sources of overhead in specific applications, as we describe in Section 6.

Figure 2 shows the possible message exchanges that can take place between hosts a, b, and c in intervals in which the connectivity is available as shown in Figure 1. When two hosts are connected, a message can hop from one to the other, and the sequence of connectivity intervals over time can be transformed into a directed graph.

We define a *message path* between any two hosts to be any sequence of hops that a message can follow over time from the source host to the destination host. This should be understood differently from the ad hoc routing definition of a path. In ad hoc routing the entire path has to be defined from start to end at a given moment in time. Our approach allows ad hoc routing to occur when hosts are directly connected but also allows message delivery, even when (potentially all) hosts are disconnected during an interval between when a host receives a message and when it transmits it to the next host.

2.2 Discontinuities and Non-symmetric Paths

Two hosts that are directly connected have symmetric message passing capabilities. For example, in Figure 1 hosts a and b can exchange messages during interval $[t_1,t_2]$, as b and c can during $[t_3, t_5]$, and so on. However, if no direct connection exists between two nodes then messages may be able to pass between them in one direction but not the other.

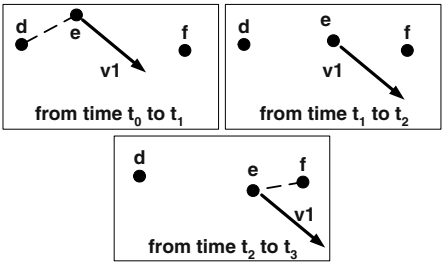

Fig. 3. Disjoint Mobile Connectivity

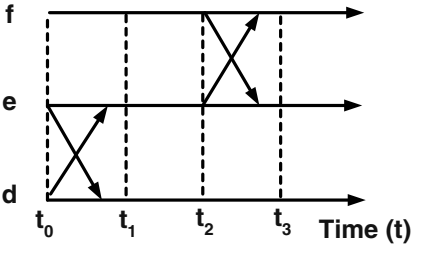

Fig. 4. Non-symmetric Message Paths

Figure 3 illustrates two disjoint connectivity intervals between three hosts d, e, and f, where d and e are connected over $[t_0, t_1]$, and e and f are connected over $[t_2, t_3]$. Figure 4 shows the non-symmetric end-to-end message passing capability

resulting from the connectivity intervals shown in Figure 3, *i.e.*, messages can be delivered from d to f but not from f to d.

3 Problem Statement and Formalization

The problem addressed by this paper is to determine whether given

- a set of hosts N,
- a source host $p \in N$,
- a destination host $q \in N$,
- an initial moment t_0, and
- *characteristic profiles* (here, of mobility and availability) over all hosts in N,

we can construct a path so as to ensure message delivery from p to q. In Section 4 we provide an algorithm based on this formalization, to decide whether for two given elements of the set, a message can travel from the first to the second under the restrictions imposed by the characteristic profiles of all nodes in the set. We also seek to build the set of paths a message could follow from source to destination, if any.

We express the *mobility* of a host in terms of a function that maps time $t \geq t_0$ to its physical position: $m_p(t)$ characterizes the mobility of host p. We formalize the *availability* of a host (*e.g.*, due to power saving modes) as a boolean function over time: $\alpha_p(t)$ is true if and only if host p is willing to communicate. Finally, we formalize a *characteristic profile* $\pi_p(t)$ for host p as a function from time to the pair consisting of p's mobility and its availability: $\pi_p(t) = < m_p(t), \alpha_p(t) >$.

We formalize the direct connectivity of any two hosts p and q in N as a time-dependent boolean relation $\Gamma_{pq}(t)$ abstracted from their characteristic profiles $\pi_p(t)$ and $\pi_q(t)$. $\Gamma_{pq}(t)$ is true if and only if at time t both hosts are available and their Euclidian distance is less than R_c, the communication range of a host (assumed to be known, constant, and the same for all hosts), and false otherwise. We express this as:

$$\Gamma_{pq}(t) = |m_p(t) - m_q(t)| < R_c \wedge \alpha_p(t) \wedge \alpha_q(t)$$

We now formalize an end-to-end path relation κ between hosts p and r over the intervals of connectivity *generated* by the relation $\Gamma(t)$. In doing so, we remove the requirement that the entire communication path be defined from source to destination uninterruptedly. We say that hosts p and r are in relation κ if and only if a message leaving from p can be delivered to r (directly or over a path accounting for disconnections). The formal expression of this definition is:

$p \ \kappa \ r \Leftrightarrow \exists \ t_0 ... t_k$ monotonically non-decreasing moments in time, with t_0 representing the current moment, and $\exists \ n_1 ... n_k$ hosts chosen [1] from the set of hosts N, such that $n_1 = p$, $n_k = r$ and $\Gamma_{n_1,n_2}(t_1) \wedge ... \wedge \Gamma_{n_i,n_{i+1}}(t_i) \wedge ... \wedge \Gamma_{n_{k-1},n_k}(t_{k-1})$.

[1] A given host in N may be chosen repeatedly in the sequence.

At any given instant t, the direct connectivity relation $\Gamma(t)$ is reflexive and symmetric, in that (1) a host is always connected to itself and retains messages across intervals of unavailability, and (2) if two hosts are connected they can communicate bidirectionally. As Figure 4 illustrates, the end-to-end path relation κ is reflexive but is neither symmetric nor transitive.

4 Solution

Figure 5 shows a more complex scenario involving eight hosts, a, b, c, d, e, f, g, and h. The discussion begins with the moment t_0 when host a wants to send a message to host h. The top half of Figure 5 depicts the connectivity intervals graphically, while the bottom half shows the pairs of hosts in $\Gamma(t)$ during each interval (for readability we show only one of each two symmetric pairs). The timeline in Figure 5 is divided into discrete intervals according to the times when at least one pair of hosts connects or disconnects. We assume sufficient time during each interval for ad hoc routing to occur, *i.e.*, messages can be exchanged among all pairs of hosts in the transitive closure of Γ during that interval - the widths of the intervals appearing in any of the figures in this section are not proportional to the duration of the interval.

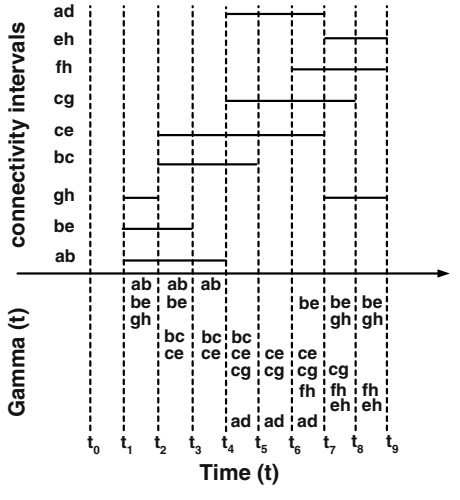

Fig. 5. Example Connectivity

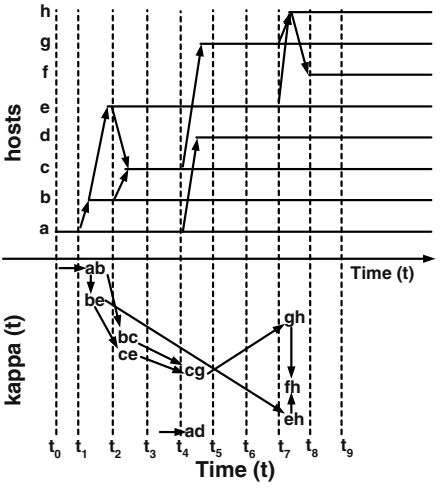

Fig. 6. Building the Paths Tree

4.1 Global Oracular Algorithm

The first problem we solve is to show that if a source host a has full knowledge of the characteristic profiles for all hosts, it can compute a message delivery path

from a to some destination host h if such a path exists. Our first algorithm for message passing in ad hoc networks depends on knowledge that is global with respect to space and time, *i.e.*, the motion and availability profiles for all hosts.

This strong requirement restricts the applicability of this first algorithm. Nonetheless, there are many situations in which the assumption of global knowledge holds. For example, we can easily imagine the connectivity segments illustrated in Figure 5 being generated by the motion profiles of a set of robots in an industrial factory, whose paths bring them temporarily (and possibly repeatedly) into wireless radio contact. Each robot might need to send messages to other robots telling them to slow down or speed up steps of a common task, to communicate failures, or to pass along sensor information about their environment. While in some settings it would be possible to connect the robots with a wired network, in others the range of motion or the cost of deploying and maintaining a wired network over numerous hosts may make a wired solution less attractive. Other applications with potentially well defined motion and availability profiles in which wired networks are impossible or costly to integrate include satellites, light rail systems, and ground-controlled unmanned aerial vehicles.

Using the connectivity intervals depicted in Figure 5, the following algorithm describes how to build a tree of message delivery paths from source a to destination h. In the path tree, the root is the source host for the message and any path from the root to a node in the tree represents a path a message *could* follow until it is delivered to the destination or gets stuck without any chance for progress.

We start building the tree from the root. Each node contains the ordered sequence of hosts the message has visited. We add children to each node for each previously unvisited host the message can reach next from that node until there are no more children to add to any node. Using the previously defined formalization, we express this as follows. For example, assume a node contains the marking $< a, b, e >$, because nodes a, b, and e, have already received the message. Because during the interval $[t_2, t_3]$ Γ_{bc} holds, we add a child node to the tree, attached to the $< a, b, e >$ node: $< a, b, e, c >$. We avoid adding a previously seen host (which could result in undesirable looping of messages or unbounded recursion of the algorithm itself) by recording the sequence of hosts seen so far. We also label each edge in the tree with the earliest point in time when that transition can be taken, again based on the ordering of connectivity intervals in relation κ.

Once the tree is built, a depth-first search ending in a leaf of the tree containing the destination node will reveal the path the message needs to follow from source to destination. Further analysis on the entire set of paths the tree may reveal can allow for optimizations in terms of the number of hops or the set of hosts the message will visit (*e.g.*, the latter can be important for security reasons). If we label each edge of the tree with the earliest moment when the extension can take place (when the child node is added to the tree), then the end-to-end delivery time or the time spent on each host can also be considered as constraints in the path selection decision.

Figure 6 illustrates how our algorithm runs on the example input data, which consists of an initial time t_0, a set of hosts a, b, c, d, e, f, g, h, a source host a, a destination host h, and the characteristic profiles of the hosts, $\pi_a(t)$, $\pi_b(t)$, $\pi_c(t)$, $\pi_d(t)$, $\pi_e(t)$, $\pi_f(t)$, $\pi_g(t)$, and $\pi_h(t)$. The top half of Figure 6 shows which hosts store and forward the message, and when they do so. The bottom half of Figure 6 shows how the connectivity intervals in Γ are concatenated to form the end-to-end path relation κ.

During the interval $[t_1, t_2]$ hosts a, b, and e form a temporary ad hoc routing network, and the message can be passed from a to b and from b to e. During the interval $[t_2, t_3]$ the message can be passed from either b or e to c. During the interval $[t_4, t_5]$ the message can be passed from a to d and from c to g. During the interval $[t_7, t_8]$ the message can be passed from either e or g to h and then from h to f.

The resulting tree of all message delivery paths from source host a is shown in Figure 7. If we specify a particular destination host such as h, the subtree of interest is one in which all leaves are labelled with a sequence of host names ending in h, illustrated by the dashed line in Figure 7.

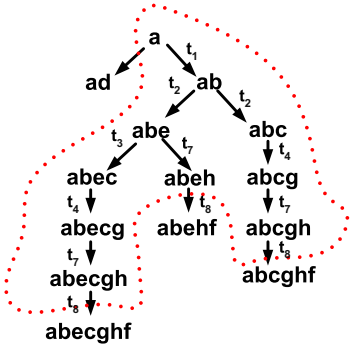

Fig. 7. Computed Paths Tree

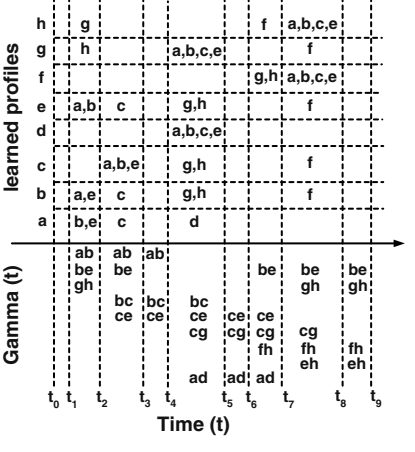

Fig. 8. Learning Profile Information

4.2 Learning the Future

In this section, we relax the assumption that any given node starts with global knowledge, and consider how well a node can do if nodes accumulate and exchange profiles opportunistically each time they meet. In particular, we describe how partial information held by each node can still allow nodes to construct message paths, and show that any path so constructed is valid.

The approach presented in the previous subsection assumes global and accurate knowledge of the information used to compute the end-to-end paths. While

there are situations where the assumption of global knowledge is reasonable, in this section we relax that assumption while still assuming perfect information in the characteristic profiles that are known. This allows our algorithm to be used in applications where hosts are highly mobile and global information may be difficult to obtain.

The message delivery paths that *precede* the moment when a host wants to send a message are especially important, as they represent prior opportunities for hosts to have exchanged information about their future behavior. At the point of wanting to send a message, the source host can then exploit the information available up to that point and plan its message delivery path accordingly, if possible. Several optimizations can be done by a host learning information that is not known to other hosts, including the ability to shortcut a path planned by another host when a better plan is discovered – we call this particular optimization a *path update*, and describe its use in greater detail in Section 4.3.

A key feature of this version of our algorithm is that a host can send not only information about itself, but information that it has learned from other hosts. We extend the algorithm in Section 4.1 so that when two hosts meet, in addition to exchanging messages whose message delivery paths go from one to the other, each host also sends profiles learned since their last encounter. If two hosts have not met previously, each sends the other all of the profiles it knows. At any point in time, a host can compute message delivery paths over the set of hosts whose characteristic profiles it knows at that point. The algorithm to construct those paths is thus the same as in Section 4.1, although the domain of hosts over which the algorithm operates is restricted to the subset of hosts whose profiles are known locally.

Assuming the same intervals of connectivity depicted in Figure 5, Figure 8 illustrates how the hosts learn each others' profiles. For the sake of this discussion, we start at time t_0 with each host knowing only its own profile, and show how characteristic profiles can be learned as time passes and interactions between hosts occur. Profiles are passed between hosts during all intervals from t_1 to t_8 except $[t_3, t_4]$ and $[t_5, t_6]$. It is also interesting to observe how the non-symmetric message delivery paths in κ result in non-symmetric profile information across the hosts. For example, f, g, and h all learned the profile for a, but a did not learn their profiles.

It is particularly interesting to note that in Figure 8 even if a message can start from a at time t_0 and be delivered to h as early as t_7, a is not able to discover that information. Thus, it is important to consider not only which message delivery paths exist between the hosts, but which hosts learn those paths and when. Only paths that are learned prior to when a message needs to be sent can be exploited by this version of our algorithm. The partial information held by the source node when it needs to send a message determines which paths the source node can compute. If a source node cannot compute a valid path due to incompleteness of its information, even if that path should exist, then the only alternative is for that source node to flood messages speculatively.

4.3 Recording the Past for Future Efficiency

With only local information, a source host may fail to compute any message delivery path to a particular destination. In that case, the source host may resort to Epidemic Routing [11], passing the message labelled only with the desired destination host (and not a planned path to that destination host) to each new host it encounters.

For this reason, in addition to maintaining and exchanging sets of characteristic profiles, it can be useful to record the history of hosts each message has visited. In this section we extend the algorithm presented in Section 4.2 as follows. We store the sequence of hosts a message has transited within the message itself. When a host forwards a message that does not have a planned message delivery path, it adds its identity to the set of hosts recorded in the message. When the message is delivered, the receiver also learns its history. All hosts share a common decision function to rank such paths. A host that knows of a better path than the one in a message they receive (we assume a planned path is better than not having one) will always perform a path update, recording the new plan in the message and sending it to the next host in the plan when it meets it.

Three kinds of information can be provided by recording a message's history. First, if we record the set of hosts a message has transited, a host that receives the message can learn of other hosts having the message even though it knows nothing of their characteristic profiles. Second, from the same information, a host can infer at least partial information about which other hosts have been in contact previously, *i.e.*, if combined with hosts exchanging profiles the message history can give each host that receives it at least a partial representation of the information each previous host knows about the others. Third, if we also record the time at which messages traverse each host, and similarly stamp other information such as when host profiles were exchanged, then information about the past can be correlated directly with information about the future. Recording the past history of messages can thus increase the scope of information at each host, and offer future performance improvements, particularly in message complexity.

As Figure 8 illustrates, hosts may know different subsets of the global set of characteristic profiles. Using additional information which it has but which the sending host does not have, an intervening host may be able to route a message along a better path it computes, instead of the path originally given to it by the sender. Hosts using only local information can therefore select paths that are closer to optimal if, as Section 4.2 describes, they are allowed to update path plans for messages sent from other hosts.

For example, assume the governing path selection heuristic is "fewest hops" and source host a computes its best path to destination host e through the set of hosts whose profiles it knows, $\{b, c, d, e\}$, as $abcde$. Suppose that just after receiving the message from host a and then disconnecting from host a, host b encounters a new host, f, which knows it can deliver the message directly to e. In that case, host b could update the planned path to be $abfe$ and route the message through host f. Note that an intervening host can only learn of a better

path by meeting a host whose profile was unknown to the sender: otherwise the sender would have already computed that better path.

Recording the past can further improve future efficiency by eliminating double delivery if the two message delivery paths intersect. The two paths will intersect at the destination but earlier intersection points (if any) can help cancel one of the redundant delivery paths before the message reaches the destination thus limiting unnecessary use of resources.

While dropping duplicates can potentially improve efficiency, it can also be a pitfall leading to deadlock in the message delivery protocol. For example, consider the scenario in Figure 9. Host a is the source of a message that leaves once through b and once through c (*e.g.*, through Epidemic Routing), on two message delivery paths towards a common destination h. The two path intersect at d and f. It is possible that the message coming on the route $a-b-d-e-f-h$ reaches e when the message coming on the route $a-c-f-g-d-h$ is at g. In this situation, d will drop the message from g because d has already seen it. Similarly, f will drop the message from e. This leads to deadlock in the message delivery procedure.

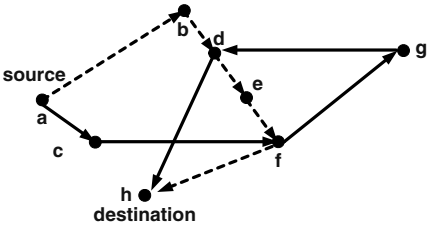

Fig. 9. Potential Deadlock in Message Delivery

It is possible to avoid the deadlock exemplified in Figure 9 by partially ordering all message delivery paths from a given source to a given destination, and dropping messages only on paths where another path from the source to the destination appears *earlier* in the partial order. The greatest potential for savings occurs when the paths are totally ordered, so that the message is delivered only along a single path, and the message coming along the highest priority route passes first through an intersection point.

It is possible to achieve a partial ordering by simply assigning a non-negative integer to each host, concatenating the host integers along each message delivery path to form an $|N|$-ary numeric value for each path with earlier hosts in the path representing higher-order digits, and having a host only drop the message if it knows a lower-valued path can deliver it reliably. If we make the host numbers unique, then the paths become totally ordered.

Unfortunately using this scheme alone unfairly causes traffic to be routed preferentially through lower-numbered hosts which could lead to overloading particular hosts or sub-networks. Such an arbitrary host numbering should therefore

only be used if needed to break ties between message delivery paths in other more suitable partial orders. Two such partial orders are reasonably evident, one based on host bandwidth and one based on times at which messages are transmitted.

First, hosts could be numbered according to decreasing bandwidth and then arbitrarily among hosts with equivalent bandwidth. The host numbers at each hop would be then concatenated as before to form the path numbers. Even though the message routing is still unfair in this case, it is distributed more proportionally according to the capabilities of the hosts. As with the arbitrary host numbering scheme, shorter paths are preferred to longer ones, and higher-bandwidth hosts are preferred to lower-bandwidth ones at each hop.

Second, path numbers could be generated by concatenating the times of the message transmissions, but with the *later* transmission times representing higher-order digits in the concatenation and the assembled digits placed to the right of the radix as a floating point fraction, rather than as an integer. Although unlikely, it is possible two message delivery paths could be temporally equivalent, in which case they can be arbitrarily distinguished to form a total order by concatenating a unique lowest order digit to the end of each. This construction does not prefer any hosts in particular, nor does it prefer paths according to the number of hops they take. Rather, this construction prefers paths where the delivery of the message to the destination is earliest, then paths where the delivery time to the penultimate host in the path is earliest, and so forth.

5 Complexity

In this section we provide an analysis of the algorithms presented in Section 4. We examine three kinds of complexity: *message* complexity, which characterizes the number of messages that are sent and the number of hops they traverse, *storage* complexity, which characterizes the amount of storage that hosts in the network must maintain, and *computational* complexity, which characterizes the cost of computing information at each node in the path tree and overall.

5.1 Message Complexity and Path Optimality

Because edges in a global path tree (such as the one shown in Figure 7) map one-to-one to message transmissions by the global algorithm, the message complexity along any distinct path in our global algorithm is bounded by one less than the height of the tree, *i.e.*, $|N|-1$. The number of transmissions needed to propagate a message from a source host to *all* other reachable hosts is also bounded by $|N|-1$ because in the presence of global information, transmissions are only made to hosts that do not have the message, and after a transmission the number of hosts that do not have the message is reduced by one.

With local information, the same worst case bounds also hold. However, best-case and average-case message complexity may be worse with local information since a source host might not find the path with the fewest possible hops due to incomplete information. This same argument applies in general to arbitrary

heuristics for selecting the best path. The profile information about other hosts that a host knows defines the subset of the global domain over which it can compute Γ and κ. Optimality of the paths it selects is thus subject to the constraints on its local information, *i.e.*, paths that a host cannot compute with its local information are not selected.

As we have seen in Section 2.1, the connectivity interval is shorter than the interval of time two hosts can communicate. In practice there is an overhead associated with the communication protocol, which we did not address in this paper. Messages carrying characteristic profiles are part of this overhead and therefore do not count (or do not influence significantly) the message complexity results derived from the analysis above.

5.2 Storage Complexity

The principal issue for storage complexity is that of maintaining characteristic profiles on the hosts. The best case occurs when a closed form can be found for equations describing the characteristic profiles. For example, the characteristic profiles of a set of small solar-powered monitoring satellites in orbit around a space station might be described as a function of their orbits and the phases of those orbits when they are in sunlight. In such a case, the storage complexity is effectively linear in the number of profiles stored.

A similar argument can be made about the complexity of profiles that repeat over time, but for which a closed form may not exist or for which storage needed for the expression exceeds the storage needed to represent the profiles explicitly. In these cases, the profiles can be stored as tables of values with entries for each relevant segment of time in the repeated period. Although the storage requirements for these profiles are again effectively linear in the number of profiles, the overhead per profile is expected to be higher than for closed form equations.

The worst case storage complexity occurs when no temporal structure of the characteristic profiles can be used to represent them efficiently. In that case, it may not be possible to represent the entire future of the hosts, because the timeline is potentially unbounded. In some cases, *e.g.*, for factory robots that are powered down at the end of a product assembly run, it may be possible though expensive to store the profiles over a meaningful segment of time, though the storage required is linear in the number of physical movements and availability transitions. Similarly, if characteristic profiles evolve over time due to environmental influences or independent decisions by hosts, *e.g.*, for coordinated mission re-planning among mobile infantry squads, it may be appropriate for each host to store profiles for each of the other hosts only up to the time of some pre-defined future meeting.

5.3 Computational Complexity

Computing even a single reliable path between source and destination hosts a and h in our global algorithm can be factorial in the number of hosts $|N|$. Specifically, because paths may not be symmetric, the number of nodes in the

path tree that must be computed is a function of the permutations of hosts in the subset of N with a and h removed, $N - \{a, h\}$, plus one for the path tree node containing only a.

If it is necessary to compute the *best* path between each pair of hosts, an upper bound on the computational complexity to compute all paths in our global algorithm is given by the formula $(|N|-2)!+2$. Fortunately, the global nature of the information means that if it is in fact possible to obtain global and accurate information about the hosts characteristic profiles, then it is also possible to compute the path tree *off-line*, choose a path for each message, and simply give the paths to the appropriate source hosts.

In contrast, the Epidemic Routing protocol does not need to do any additional computation besides sending the message, so its computational complexity is constant. Thus, our global algorithm and the Epidemic protocol represent different points in the space of possible trade-offs between message and computational complexity for hosts in mobile ad hoc networks.

6 Discussion

Section 2 mentioned that in practice communication intervals between any two hosts will necessarily be shorter than the intervals during which they are connected. Extending the formalism described in Section 3 and the algorithm described in Section 4 to consider such overhead is mainly a matter of re-defining the relation $\Gamma(t)$ so that it holds over the segment of the communication interval not consumed by overhead. However, the overheads themselves may be non-trivial functions of the protocols and factors such as transmission latency may also need to be considered for each application. Sources of overhead must be identified and such factors must be incorporated in our message passing model. Clock synchronization is an important issue, especially for the algorithm with local information. The fact that host b tells host a that host c will be in a certain place at some moment t has real value as long as for hosts a and c moment t represents the same time. In this paper we assume clock synchronization across all hosts. As future work we will examine the implications of transient connectivity and of incomplete local information for clock synchronization protocols.

Another key area of future work is to examine the performance of the techniques described in this paper in the context of specific mobile ad hoc networked systems. We plan to conduct simulation studies based on motion profiles from a range of applications such those mentioned in Section 4 (*e.g.*, robotics, rail systems, unmanned aerial vehicles, satellites) to quantify the costs and benefits of our approach in real-world settings. We expect these studies to yield insights into further optimizations and trade-offs for the techniques described in this paper.

One important extension to the current research is to consider *message sizes.* The main question addressed in this paper was a simple decision problem: can we deliver a minimum-size message from a host to another? While this is an important issue, many real life applications handle a variety of information, and the sizes of messages have an important effect on transmission timing. If the

message can be partitioned, and a single sufficient path cannot be found, parts of the message can be sent on several paths and reassembled at the destination. Constraints related to the order in which the message parts are received pose additional challenges. If the processing of the data at the destination has a prefix property (*i.e.*, its parameters are dynamically updated such that the processing of a segment of information depends on what has happened before), ensuring in-order delivery is a must.

In practice, the quality of information, especially for motion profiles learned from other hosts, may also depend on its freshness. We assess the performance of a host in predicting the correct future paths in terms of the *completeness* of the information (for which hosts it knows motion profiles - a subset of the total graph) and the *accuracy* of each profile it has. As future work, we will examine the effects of limiting the quantity of information we can learn about the future, as well as considering a deterioration factor associated with the closeness of the upcoming events, *e.g.*, information about the near future may be more accurate than information about the far future. Repeated characteristic profile information exchanges can help update the information about a known future profile, in case the real evolution differs from the initially declared one.

7 Related Work

Many other approaches to message routing in mobile ad hoc networks have been proposed. Most of the early work concentrates on using complete route information from source to destination at a certain moment in time. Such algorithms account for mobility by adjusting the route the messages will follow. However, the asymmetry and the disconnections frequently encountered in mobile ad hoc settings are serious challenges to the applicability of many such approaches. For example, DSDV [8], its extension CGSR [12], and WRP [10], are the most prominent members of the table-driven family of routing protocols. They all maintain tables with routing information for each pair of source-destination hosts. When a host needs to send a message it will have to consult the routing table dynamically maintained and choose a complete end-to-end route for the message. The drawback of this approach is the overhead of maintaining the tables and the assumption that the routes remain stable once the transmission begins.

Another important family of protocols is the source-initiated on-demand routing protocols. Members of this family are AODV [9], DSR [13], and ABR [14]. All of them initiate route discovery on-demand, when the source of a message needs to send that message to some remote destination. The assumption is that a potential route doesn't change and doesn't break during the delivery, remaining entirely defined, end-to-end, throughout the entire process.

Trying to relax the requirement that the route stays stable for the entire delivery period and that the communication is symmetric (e.g., the source sends discovery messages and the destination has to answer before the source sends the data), a new family of protocols is addressing disconnected communication. The main characteristic of this family of protocols is that they do not require

the entire route to be defined at a certain moment in time and to remain defined for as long as the delivery takes place. The technique used in most approaches is to store-and-forward messages from one host to the other when connectivity allows it. Our approach does the same thing, allowing for next-hop or multi-hop message delivery if connectivity allows it over a given interval.

Generally, protocols addressing disconnected communication can be separated in two groups: proactive and reactive. Reactive protocols adjust their delivery strategy to the environment. Proactive protocols take steps towards exploiting communication opportunities when and where possible, if they were not already available.

Among the reactive protocols, we have mentioned Epidemic Routing [11], one of the first efforts in this area, in which hosts broadcast a message to all nearby hosts, hoping for eventual message delivery. Hop counts and buffer size limitations may restrict the excessive use of resources. In [15], the authors add the notion of a message lifetime and also develop four strategies for dropping messages in the case of buffer overflow in the same Epidemic Routing algorithm. Partial and total ordering schemes are used in [16] to ensure message convergence towards a destination when alternate routes can be chosen on the spot. Nodes are labelled such that the destination has the lowest id and messages are sent from a node to any other node with a lower id until it reaches the destination.

From the proactive family of protocols we mention two papers. In the first one [17], the authors modify the hosts' trajectories to ensure a sequence of 1-hop communication segments from source to destination. In [18], designated message *ferries* take the messages from source and deliver them to destination.

8 Conclusions

In this paper we have shown that reliable delivery paths can be established between hosts in an ad hoc networking environment, even in the presence of disconnection. The quantity and the quality of the information known *a priori* influences the selection of message paths. Host motion and availability profiles as functions of time are two of the most important parameters that influence message delivery in mobile and ad hoc networks. These profiles can be known *a priori*, discovered at run time, or updated on the fly. The algorithms for message delivery in the face of transient connectivity that we have presented in this paper can help deliver messages when other routing algorithms fail.

Acknowledgements. This research was supported by the Office of Naval Research under MURI research contract N00014-02-1-0715. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily represent the views of the research sponsors.

References

1. Chiang, C., Gerla, M., Zhang, L.: Adaptive shared tree multicast in mobile wireless networks. In: Proceedings of GLOBECOM '98. (1998) 1817–1822
2. Carlberg, K., Crowcroft, J.: Building shared trees using a one-to-many joining mechanism. ACM Computer Communication Review (1997) 5–11
3. Ballardie, T., Francis, P., Crowcroft, J.: Core based tree (cbt) an architecture for scalable inter-domain multicast routing. In: Proceedings of the ACM SIGCOMM. (1993) 85–95
4. Deering, S., Estrin, D.L., Farinacci, D., Jacobson, V., Liu, C.G., Wei, L.: The PIM architecture for wide-area multicast routing. IEEE/ACM Transactions on Networking **4** (1996) 153–162
5. Ho, C., Obraczka, K., Tsudik, G., Viswanath, K.: Flooding for reliable multicast in multi-hop ad hoc networks. In: Proceedings of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, Seattle, WA (1999) 64–71
6. Wu, C., Tay, Y.: Amris: A multicast protocol for ad hoc wireless networks. In: Proceedings of IEEE MILCOM'99, Atlantic City, NJ (1999)
7. Park, V.D., Corson, M.S.: A highly adaptive distributed routing algorithm for mobile wireless networks. In: Proceedings of INFOCOM'97. (1997) 1405–1413
8. Perkins, C., Bhagwat, P.: Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In: ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications. (1994)
9. Perkins, C.: Ad-hoc on-demand distance vector routing. In: MILCOM '97 panel on Ad Hoc Networks. (1997)
10. Murthy, S., Garcia-Luna-Aceves: An efficient routing protocol for wireless networks. Mobile Networks and Applications **1** (1996) 183–197
11. Vahdat, A., Becker, D.: Epidemic routing for partially connected ad hoc networks. Technical Report CS-200006, Duke University (2000)
12. Chiang, C., Wu, H., Liu, W., Gerla, M.: Routing in clustered multihop, mobile wireless networks. In: IEEE Singapore International Conference on Networks. (1997) 197–211
13. Johnson, D.B., Maltz, D.A.: Dynamic source routing in ad hoc wireless networks. Mobile Computing **353** (1996)
14. Toh, C.K.: A novel distributed routing protocol to support ad-hoc mobile computing. In: Fifteenth Annual International Phoenix Conference on Computers and Communications. (1996) 480–486
15. Davis, J.A., Fagg, A.H., Levine, B.N.: Wearable computers as packet transport mechanisms in highly-paritioned ad-hoc networks. In: International Symposium on Wearable Computing. (2001)
16. Roman, G.C., Payton, J.: A termination detection protocol for use in mobile ad hoc networks. (to appear in Special Issue of Automated Software Engineering on Distributed and Mobile Software Engineering)
17. Li, Q., Rus, D.: Communication in disconnected ad hoc networks using message relay. Parallel and Distributed Computing **63** (2003) 75–86
18. Zhao, W., Amma, M.H.: Message ferrying: Proactive routing in highly-partitioned wireless ad hoc networks. In: Ninth IEEE Workshop on Future Trends of Distributed Computing Systems. (2003)