Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers

Charles E. Perkins
IBM, T.J. Watson Research Center
Hawthorne, NY 10562

Pravin Bhagwat
Computer Science Department
University of Maryland
College Park, MD 20742

Abstract

An *ad-hoc* network is the cooperative engagement of a collection of Mobile Hosts without the required intervention of any centralized Access Point. In this paper we present an innovative design for the operation of such ad-hoc networks. The basic idea of the design is to operate each Mobile Host as a specialized router, which periodically advertises its view of the interconnection topology with other Mobile Hosts within the network. This amounts to a new sort of routing protocol. We have investigated modifications to the basic Bellman-Ford routing mechanisms, as specified by RIP [5], to make it suitable for a dynamic and self-starting network mechanism as is required by users wishing to utilize ad-hoc networks. Our modifications address some of the previous objections to the use of Bellman-Ford, related to the poor looping properties of such algorithms in the face of broken links and the resulting time dependent nature of the interconnection topology describing the links between the Mobile Hosts. Finally, we describe the ways in which the basic network-layer routing can be modified to provide MAC-layer support for ad-hoc networks.

1 Introduction

Recently, there has been tremendous growth in the sales of laptop and portable computers. These smaller computers, nevertheless, can be equipped with hundreds of megabytes of disk storage, high resolution color displays, pointing devices, and wireless communications adapters. Moreover, since many of these small (in size only) computers operate for hours with battery power, users are free to move about at their convenience without being constrained by wires.

This is a revolutionary development in personal computing. Battery powered, untethered computers are likely to become a pervasive part of our computing infrastructure. As people begin to have mobile computers handy, for whatever purposes, sharing information between the computers will become a natural requirement. Currently, such sharing is made difficult by the need for users to perform administrative tasks and set up static, bidirectional links between their computers. However, if the wireless communications systems in the mobile computers support a broadcast mechanism, much more flexible and useful ways of sharing information can be imagined. For instance, any number of people could conceivably enter a conference room and agree to support communications links between themselves, without necessarily engaging the services of any pre-existing equipment in the room (i.e, without requiring any pre-existing communications infrastructure). Thus, one of our primary motivations is to allow the construction of temporary networks with no wires and no administrative intervention required. In this paper, such a interconnection between the mobile computers will be called an *ad-hoc* network, in conformance with current usage within the IEEE 802.11 subcommittee [4].

Ad-hoc networks differ significantly from existing networks. First of all, the topology of interconnections may be quite dynamic. Secondly, most users will not wish to perform any administrative actions to set up such a network. In order to provide service in the most general situation, we do not assume that every computer is within communication range of every other computer. This lack of complete connectivity would certainly be a reasonable characteristic of, say, a population of mobile computers in a large room which relied on infrared transceivers to effect their data communications.

Currently, there is no method available which enables mobile computers with wireless data communications equipment to freely roam about while still maintaining

---

0

connections with each other, unless special assumptions are made about the way the computers are situated with respect to each other. One mobile computer may often be able to exchange data with two other mobile computers which cannot themselves directly exchange data. As a result, computer users in a conference room may be unable to predict which of their associates' computers could be relied upon to maintain network connection, especially as the users moved from place to place within the room.

Routing protocols for existing networks have not been designed specifically to provide the kind of dynamic, self-starting behavior needed for ad-hoc networks. Most protocols exhibit their least desirable behavior when presented with a highly dynamic interconnection topology. Although we thought that mobile computers could naturally be modeled as *routers*, it was also clear that existing routing protocols would place too heavy a computational burden on each mobile computer. Moreover, the convergence characteristics of existing routing protocols did not seem good enough to fit the needs of ad-hoc networks. Lastly, the wireless medium differs in important ways from wired media, which would require that we make modifications to whichever routing protocol we might choose to experiment with. For instance, mobile computers may well have only a single network interface adapter, whereas most existing routers have network interfaces to connect two separate networks together. Besides, wireless media are of limited and variable range, in distinction to existing wired media. Since we had to make lots of changes anyway, we decided to follow our ad-hoc network model as far as we could and ended up with a substantially new approach to the classic distance-vector routing.

2 Overview of Routing Methods

In our environment, the problem of routing is essentially the distributed version of the shortest path problem [11]. Each node in the network maintains for each destination a preferred neighbor. Each data packet contains a destination node identifier in its header. When a node receives a data packet, it forwards the packet to the preferred neighbor for its destination. The forwarding process continues until the packet reaches its destination. The manner in which routing tables are constructed, maintained and updated differs from one routing method to another. Popular routing methods, however, attempt to achieve the common objective of routing packets along the optimal path. The next-hop routing methods can be categorized into two primary classes: *link-state* and *distance-vector*.

Link-State The link-state approach is closer to the centralized version of the shortest path computation method. Each node maintains a view of the network topology with a cost for each link. To keep these views consistent, each node periodically broadcasts the link costs of its outgoing links to all other nodes using a protocol such as flooding. As a node receives this information, it updates its view of the network topology and applies a shortest-path algorithm to choose its next hop for each destination. Some of the link costs in a node's view can be incorrect because of long propagation delays, partitioned network, etc. Such inconsistent views of network topologies might lead to formation of routing loops. These loops, however, are short-lived, because they disappear in the time it takes a message to traverse the diameter of the network [8].

Distance-Vector In distance-vector algorithms, every node i maintains, for each destination x, a set of distances $\{d_{ij}^x\}$ where j ranges over the neighbors of i. Node i treats neighbor k as a next-hop for a packet destined for x if d_{ik}^x equals $min_j\{d_{ij}^x\}$. The succession of next hops chosen in this manner lead to x along the shorest path. In order to keep the distance estimates up-to-date, each node monitors the cost of its outgoing links and periodically broadcasts, to each one its neighbors, its current estimate of the shortest distance to every other node in the network.

The above distance-vector algorithm is the classical Distributed Bellman-Ford (DBF) algorithm [2]. Compared to link-state method, it is computationally more efficient, easier to implement and requires much less storage space. However, it is well known that this algorithm can cause the formation of both short-lived and long-lived loops [3]. The primary cause for formation of routing loops is that nodes choose their next-hops in a completely distributed fashion based on information which can possibly be stale and, therefore, incorrect. Almost all proposed modifications to DBF algorithm [6, 7, 9] eliminate the looping problem by forcing all nodes in the network to participate in some form of internodal coordination protocol. Such internodal coordination mechanisms might be effective when topological changes are rare. However, within an ad-hoc mobile environment enforcing any such internodal coordination mechanism will be difficult due to the rapidly changing topology of the underlying routing network.

Simplicity is one of the primary attributes which makes any routing protocol *preferred* over others for implementation within operational networks. RIP [5] is a classical example. Despite the *counting-to-infinity* problem it has proven to be very successful within small size internetworks. The usefulness of RIP within ad-hoc environment, however, is limited as it was not designed to handle rapid topological changes. Furthermore, the techniques of *split-horizon* and *poisoned-reverse* [5] are not useful within the wireless environment due to the broadcast nature of the transmission medium. Our design goal therefore has been to design a routing method for ad-hoc networks which preserves the simplicity of RIP, yet at the same time avoids the looping problem. Our approach is to tag each route table entry with a sequence number so that nodes can quickly distinguish stale routes from the new ones and thus avoid formation of routing loops.

3 Destination-Sequenced Distance Vector (DSDV) Protocol

Our proposed routing method allows a collection of mobile computers, which may not be close to any base station and can exchange data along changing and arbitrary paths of interconnection, to afford all computers among their number a (possibly multi-hop) path along which data can be exchanged. In addition, our solution must remain compatible with operation in cases where a base station is available. By the methods outlined in this paper, not only will routing be seen to solve the problems associated with ad-hoc networks, but in addition we will describe ways to perform such routing functions at Layer 2, which traditionally has not been utilized as a protocol level for routing.

Packets are transmitted between the stations of the network by using routing tables which are stored at each station of the network. Each routing table, at each of the stations, lists all available destinations, and the number of hops to each. Each route table entry is tagged with a sequence number which is originated by the destination station. To maintain the consistency of routing tables in a dynamically varying topology, each station periodically transmits updates, and transmits updates immediately when significant new information is available. Since we do not assume that the mobile hosts are maintaining any sort of time synchronization, we also make no assumption about the phase relationship of the update periods between the mobile hosts. These packets indicate which stations are accessible from each station and the number of hops necessary to reach these accessible stations, as is often done in distance-vector routing algorithms. It is not the purpose of this paper to propose any new metrics for route selection other than the freshness of the sequence numbers associated with the route; cost or other metrics might easily replace the number of hops in other implementations. The packets may be transmitted containing either layer 2 (MAC) addresses or layer 3 (network) addresses.

Routing information is advertised by broadcasting or multicasting the packets which are transmitted periodically and incrementally as topological changes are detected – for instance, when stations move within the network. Data is also kept about the length of time between arrival of the *first* and the arrival of the *best* route for each particular destination. Based on this data, a decision may be made to delay advertising routes which are about to change soon, thus damping fluctuations of the route tables. The advertisement of routes which may not have stabilized yet is delayed in order to reduce the number of rebroadcasts of possible route entries that normally arrive with the same sequence number.

The DSDV protocol requires each mobile station to advertise, to each of its current neighbors, its own routing table (for instance, by broadcasting its entries). The entries in this list may change fairly dynamically over time, so the advertisement must be made often enough to ensure that every mobile computer can almost always locate every other mobile computer of the collection. In addition, each mobile computer agrees to relay data packets to other computers upon request. This agreement places a premium on the ability to determine the shortest number of hops for a route to a destination; we would like to avoid unnecessarily disturbing mobile hosts if they are in sleep mode. In this way a mobile computer may exchange data with any other mobile computer in the group even if the target of the data is not within range for direct communication. If the notification of which other mobile computers are accessible from any particular computer in the collection is done at layer 2, then DSDV will work with whatever higher layer (e.g., Network Layer) protocol might be in use.

All the computers interoperating to create data paths between themselves broadcast the necessary data periodically, say once every few seconds. In a wireless medium, it is important to keep in mind that broadcasts are limited in range by the physical characteristics of the medium. This is different than the situation with wired media, which usually have a much more well-defined range of reception. The data broadcast by each mobile computer will contain its new sequence number and the following information for each new route:

- The destination's address;
- The number of hops required to reach the destination; and
- The sequence number of the information received regarding that destination, as originally stamped by the destination;

The transmitted routing tables will also contain the hardware address, and (if appropriate) the network address, of the mobile computer transmitting them, within the headers of the packet. The routing table will also include a sequence number created by the transmitter. Routes with more recent sequence numbers are always preferred as the basis for making forwarding decisions, but not necessarily advertised. Of the paths with the same sequence number, those with the smallest metric will be used. By the natural way in which the routing tables are propagated, the sequence number is sent to all mobile computers which may each decide to maintain a routing entry for that originating mobile computer.

Routes received in broadcasts are also advertised by the receiver when it subsequently broadcasts its routing information; the receiver adds an increment to the metric before advertising the route, since incoming packets will require one more hop to reach the destination (namely, the hop from the transmitter to the receiver). Again, we do not explicitly consider here the changes required to use metrics which do not use the hop count to the destination.

One of the most important parameters to be chosen is the time between broadcasting the routing information packets. However, when any new or substantially modified route information is received by a Mobile Host, the new information will be retransmitted soon (subject to constraints imposed for damping route fluctuations), effecting the most rapid possible dissemination of routing information among all the cooperating Mobile Hosts. This quick re-broadcast introduces a new requirement for our protocols to converge as soon as possible. It would be calamitous if the movement of a Mobile Host caused a storm of broadcasts, degrading the availability of the wireless medium.

Mobile Hosts cause broken links as they move from place to place. The broken link may be detected by the layer-2 protocol, or it may instead be inferred if no broadcasts have been received for a while from a former neighbor. A broken link is described by a metric of ∞ (i.e., any value greater than the maximum allowed metric). When a link to a next hop has broken, any route through that next hop is immediately assigned an ∞ metric and assigned an updated sequence number. Since this qualifies as a substantial route change, such modified routes are immediately disclosed in a broadcast routing information packet. Building information to describe broken links is the only situation when the sequence number is generated by any Mobile Host other than the destination Mobile Host. Sequence numbers defined by the originating Mobile Hosts are defined to be even numbers, and sequence numbers generated to indicate ∞ metrics are odd numbers. In this way any "real" sequence numbers will supersede an ∞ metric. When a node receives an ∞ metric, and it has a later sequence number with a finite metric, it triggers a route update broadcast to disseminate the important news about that destination.

In a very large population of Mobile Hosts, adjustments will likely be made in the time between broadcasts of the routing information packets. In order to reduce the amount of information carried in these packets, two types will be defined. One will carry all the available routing information, called a "full dump". The other type will carry only information changed since the last full dump, called an "incremental". By design, an incremental routing update should fit in one network protocol data unit (NPDU). The full dump will most likely require multiple NPDUs, even for relatively small populations of Mobile Hosts. Full dumps can be transmitted relatively infrequently when no movement of Mobile Hosts is occurring. When movement becomes frequent, and the size of an incremental approaches the size of a NPDU, then a full dump can be scheduled (so that the next incremental will be smaller). It is expected that mobile nodes will implement some means for determining which route changes are significant enough to be sent out with each incremental advertisement. For instance, when a stabilized route shows a different metric for some destination, that would likely constitute a significant change that needed to be advertised after stabilization. If a new sequence number for a route is received, but the metric stays the same, that would be unlikely to be considered as a significant change.

When a Mobile Host receives new routing information (usually in an incremental packet as just described), that information is compared to the information already available from previous routing information packets. Any route with a more recent sequence number is used. Routes with older sequence numbers are discarded. A route with a sequence number equal to an existing route is chosen if it has a "better" metric, and the existing route discarded, or stored as less preferable. The metrics for routes chosen from the newly received broadcast information are each incremented by one hop. Newly recorded routes are scheduled for immediate advertisement to the current Mobile Host's neighbors. Routes which show an improved metric are scheduled for advertisement at a time which depends on the average settling time for routes to the particular destination under consideration.

Timing skews between the various Mobile Hosts are expected. The broadcasts of routing information by the Mobile Hosts are to be regarded as somewhat asynchronous events, even though some regularity is expected. In such a population of independently transmitting agents, some fluctuation could develop using the above procedures for updating routes. It could turn out that a particular Mobile Host would receive new routing information in a pattern which causes it to consistently change routes from one next hop to another, even when the destination Mobile Host has not moved. This happens because there are two ways for new routes to be chosen; they might have a later sequence number, or they might have a better metric. A Mobile Host could conceivably always receive two routes to the same destination, with a newer sequence number, one after another (via different neighbors), but always get the route with the worse metric first. Unless care is taken, this will lead to a continuing burst of new route transmittals upon every new sequence number from that destination. Each new metric is propagated to every Mobile Host in the neighborhood, which propagates to their neighbors and so on.

One solution is to delay the advertisement of such routes, when a Mobile Host can determine that a route with a better metric is likely to show up soon. The route with the later sequence number must be available for use, but it does not have to be advertised immediately unless it is a route to a destination which was previously unreachable. Thus, there will be two routing tables kept at each Mobile Host; one for use with forwarding packets, and another to be advertised via incremental routing information packets. To determine the probability of imminent arrival of routing information showing a better metric, the Mobile Host has to keep a history of the weighted average time that routes to a particular destination fluctuate until the route with the best metric is received. We hope that such a procedure will allow us to predict how long to wait before advertising new routes.

Operating DSDV at Layer 2

The addresses stored in the routing tables will correspond to the layer at which this ad-hoc networking protocol is operated. That is, operation at Layer 3 will use network layer addresses for the next hop and destination addresses, and operation at Layer 2 will use Layer 2 Media Access Control (MAC) addresses.

Using MAC addresses for the forwarding table does introduce a new requirement, however. The difficulty is that Layer 3 network protocols provide communication based on network addresses, and a way must be provided to resolve these Layer 3 addresses into MAC addresses. Otherwise, a multiplicity of different address resolution mechanisms would be put into place, and a corresponding loss of bandwidth in the wireless medium would be observed whenever the resolution mechanisms were utilized. This could be substantial since such mechanisms would require broadcasts and retransmitted broadcasts by every Mobile Host in the ad-hoc network. Thus, unless special care is taken, every address resolution might look like a glitch in the normal operation of the network, which may well be noticeable to any active users.

The solution proposed here, for operation at Layer 2, is to include Layer 3 protocol information along with the Layer 2 information. Each destination host would advertise which Layer 3 protocols it supports, and each Mobile Host advertising reachability to that destination would include along, with the advertisement, the information about the Layer 3 protocols supported at that destination. This information would only have to be transmitted when it changes, which occurs rarely. Changes would be transmitted as part of each incremental dump. Since each Mobile Host could support several Layer 3 protocols (and many will), this list would have to be variable in length.

Extending Base Station Coverage

Mobile computers will frequently be used in conjunction with base stations, which allow them to exchange data with other computers connected to the wired network. By participating in the DSDV protocol, base stations can extend their coverage beyond the range imposed by their wireless transmitters. When a base station participates in DSDV, it is shown as a default route in the tables transmitted by a mobile station. In this way, mobile stations within range of a base station can cooperate to effectively extend the range of the base station to serve other stations outside the range of the base station, as long as those other mobile stations are close to some other mobile station that is within range.

4 Examples of DSDV in operation

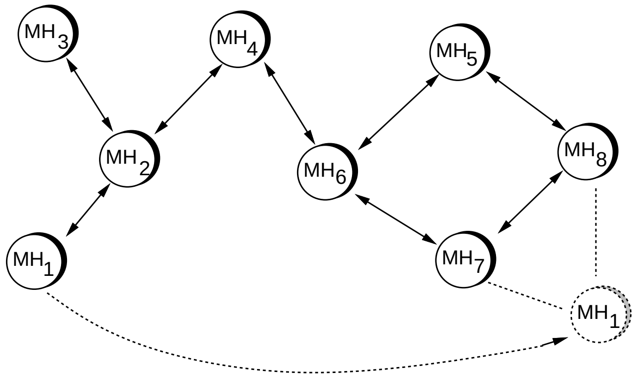

Figure 1: Movement in an ad-hoc network

Consider MH_4 in Figure 1. Table 1 shows a possible structure of the forwarding table which is maintained at MH_4. Suppose the address[1]. of each Mobile Host is represented as MH_i Suppose further that all sequence numbers are denoted SNNN_MH_i, where MH_i specifies the computer that created the sequence number and SNNN is a sequence number value. Also suppose that there are entries for all other Mobile Hosts, with sequence numbers SNNN_MH_i, before MH_1 moves away from MH_2. The install time field helps determine when

[1] If DSDV is operated at level 2 then MH_i denotes the MAC address, otherwise it denotes a level 3 address

Destination	NextHop	Metric	Sequence number	Install	Flags	Stable_data
MH_1	MH_2	2	S406_MH_1	T001_MH_4		Ptr1_MH_1
MH_2	MH_2	1	S128_MH_2	T001_MH_4		Ptr1_MH_2
MH_3	MH_2	2	S564_MH_3	T001_MH_4		Ptr1_MH_3
MH_4	MH_4	0	S710_MH_4	T001_MH_4		Ptr1_MH_4
MH_5	MH_6	2	S392_MH_5	T002_MH_4		Ptr1_MH_5
MH_6	MH_6	1	S076_MH_6	T001_MH_4		Ptr1_MH_6
MH_7	MH_6	2	S128_MH_7	T002_MH_4		Ptr1_MH_7
MH_8	MH_6	3	S050_MH_8	T002_MH_4		Ptr1_MH_8

Table 1: Structure of the MH_4 forwarding table

to delete stale routes. With our protocol, the deletion of stale routes should rarely occur, since the detection of link breakages should propagate through the ad-hoc network immediately. Nevertheless, we expect to continue to monitor for the existence of stale routes and take appropriate action.

From table 1, one could surmise, for instance, that all the computers became available to MH_4 at about the same time, since its install_time for most of them is about the same. One could also surmise that none of the links between the computers were broken, because all of the sequence number fields have times with even digits in the units place. Ptr1_MH_i would all be pointers to null structures, because there are not any routes in Figure 1 which are likely to be superseded or compete with other possible routes to any particular destination.

Table 2 shows the structure of the advertised route table of MH_4.

Destination	Metric	Sequence number
MH_1	2	S406_MH_1
MH_2	1	S128_MH_2
MH_3	2	S564_MH_3
MH_4	0	S710_MH_4
MH_5	2	S392_MH_5
MH_6	1	S076_MH_6
MH_7	2	S128_MH_7
MH_8	3	S050_MH_8

Table 2: Advertised route table by MH_4

Now suppose that MH_1 moves into the general vicinity of MH_5 and MH_7, and away from the others (especially MH_2). The new internal forwarding tables at MH_4 might then appear as shown in table 3.

Only the entry for MH_1 shows a new metric, but in the intervening time, many new sequence number entries have been received. The first entry thus must be advertised in subsequent incremental routing information updates until the next full dump occurs. When MH_1 moved into the vicinity of MH_5 and MH_7, it triggered an immediate incremental routing information update which was then broadcast to MH_6. MH_6, having, determined that significant new routing information had been received, also triggered an immediate update which carried along the new routing information for MH_1. MH_4, upon receiving this information, would then broadcast it at every interval until the next full routing information dump. At MH_4, the incremental advertised routing update would have the form as shown in table 4.

In this advertisement, the information for MH_4 comes first, since it is doing the advertisement. The information for MH_1 comes next, not because it has a lower address, but because MH_1 is the only one which has any significant route changes affecting it. As a general rule, routes with changed metrics are first included in each incremental packet. The remaining space is used to include those routes whose sequence numbers have changed.

In this example, one node has changed its routing information, since it is in a new location. All nodes have transmitted new sequence numbers recently. If there were too many updated sequence numbers to fit in a single packet, only the ones which fit would be transmitted. These would be selected with a view to fairly transmitting them in their turn over several incremental update intervals. There is no such required format for the transmission of full routing information packets. As many packets are used as are needed, and all available information is transmitted. The frequency of transmitting full updates would be reduced if the volume of data began to consume a significant fraction of the available capacity of the medium.

Damping Fluctuations

The following describes how the settling time table is used to prevent fluctuations of routing table entry advertisements. The general problem arises because route updates are selected according to the following criteria:

Destination	NextHop	Metric	Sequence number	Install	Flags	Stable_data
MH_1	MH_6	3	S516_MH_1	T810_MH_4	M	Ptr1_MH_1
MH_2	MH_2	1	S238_MH_2	T001_MH_4		Ptr1_MH_2
MH_3	MH_2	2	S674_MH_3	T001_MH_4		Ptr1_MH_3
MH_4	MH_4	0	S820_MH_4	T001_MH_4		Ptr1_MH_4
MH_5	MH_6	2	S502_MH_5	T002_MH_4		Ptr1_MH_5
MH_6	MH_6	1	S186_MH_6	T001_MH_4		Ptr1_MH_6
MH_7	MH_6	2	S238_MH_7	T002_MH_4		Ptr1_MH_7
MH_8	MH_6	3	S160_MH_8	T002_MH_4		Ptr1_MH_8

Table 3: MH_4 forwarding table (updated)

Destination	Metric	Sequence number
MH_4	0	S820_MH_4
MH_1	3	S516_MH_1
MH_2	1	S238_MH_2
MH_3	2	S674_MH_3
MH_5	2	S502_MH_5
MH_6	1	S186_MH_6
MH_7	2	S238_MH_7
MH_8	3	S160_MH_8

Table 4: MH_4 advertised table (updated)

- Routes are always preferred if the sequence numbers are newer;
- Otherwise, routes are preferred if the sequence numbers are the same and yet the metric is better (lower).

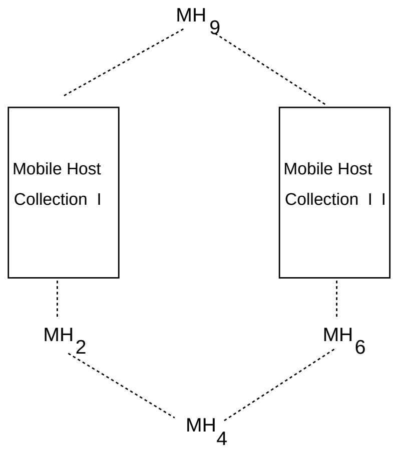

Figure 2: Receiving fluctuating routes

To see the problem, suppose that two routes with identical sequence numbers are received by a Mobile Host, but in the wrong order. In other words, suppose that MH_4 receives the higher metric next hop first, and soon after gets another next hop with a lower metric but the same sequence number. This could happen when there are a lot of Mobile Hosts, transmitting their updates not quite regularly. Alternatively, if the Mobile Hosts are acting independently and with markedly different transmission intervals, the situation could occur with correspondingly fewer hosts. Suppose, in any event, in Figure 2 that there are enough Mobile Hosts to cause the problem, in two separate collections of Mobile Hosts both connected to a common destination MH_9, but with no other Mobile Hosts in common. Suppose further that all Mobile Hosts are transmitting updates approximately every 15 seconds, that Mobile Host MH_2 has a route to MH_9 with 12 hops, and Mobile Host MH_6 has a route to MH_9 with 11 hops. Moreover, suppose that the routing information update from MH_2 arrives at MH_4 approximately 10 seconds before the routing information update from MH_6. This will occur every time that a new sequence number is issued from Mobile Host MH_9. In fact, the time differential can be drastic if any Mobile Host in collection II begins to issue its sequence number updates in multiple incremental update intervals, as would happen, for instance, when there are too many hosts with new sequence number updates for them all to fit within a single incremental packet update. In general, the larger the number of hops, the more drastic differentials between delivery of the updates can be expected in Figure 2.

The settling time data is stored in a table with the following fields, keyed by the first field:

- Destination address
- Last settling time
- Average settling time

The settling time is calculated by maintaining a running, weighted average over the most recent updates of the routes, for each destination.

Suppose a new routing information update arrives at MH_4. The sequence number in the new entry is the same as the sequence number in the currently used entry, and the newer entry has a worse (i.e., higher) metric. Then MH_4 must use the new entry in making subsequent forwarding decisions. However, MH_4 does not have to advertise the new route immediately and can consult its route settling time table to decide how long to wait before advertising it. The average settling time is used for this determination. For instance, MH_4 may decide to delay (average_settling_time x 2) before advertising a route.

This can be quite beneficial, because if the possibly unstable route were advertised immediately, the effects would ripple through the network, and this bad effect would probably be repeated every time Mobile Host MH_9's sequence number updates rippled through the ad-hoc network. On the other hand, if a link via Mobile Host MH_6 truly does break, the advertisement of a route via MH_2 should proceed immediately. To achieve this when there is a history of fluctuations at Mobile Host MH_4, the link breakage should be detected fast enough so that an intermediate host in Collection II finds out the problem and begins a triggered incremental update showing an ∞ metric for the path along the way to Mobile Host MH_9. Routes with an ∞ metric are required by this protocol to be advertised immediately, without delay.

In order to bias the damping mechanism in favor of recent events, the most recent measurement of the settling time of a particular route must be counted with a higher weighting factor than are less recent measurements. And, importantly, a parameter must be selected which indicates how long a route has to remain stable before it is counted as truly stable. This amounts to specifying a maximum value for the settling time for the destination in the settling time table. Any route more stable than this maximum value will cause a triggered update if it is ever replaced by another route with a different next hop or metric.

When a new routing update is received from a neighbor, during the same time that the updates are applied to the table, processing also occurs to delete stale entries. Stale entries are defined to be those for which no update has been applied within the last few update periods. Each neighbor is expected to send regular updates; when no updates are received for a while, the receiver may make the determination that the corresponding computer is no longer a neighbor. When that occurs, any route using that computer as a next hop should be deleted, including the route indicating that computer as the actual (formerly neighboring) destination. Increasing the number of update periods that may transpire before entries are determined would result in more stale routing entries, but would also allow for more transmission errors. Transmission errors are likely to occur when a CSMA-type broadcast medium is used, as may well be the case for many wireless implementations. When the link breaks, an ∞ metric route should be advertised for it, as well as for the routes that depend on it.

There are additional data fields, other than those stated above, which might be transmitted as part of each entry in the routing tables which are broadcast by each participating computer (mobile or base station). These fields may depend, for instance, on higher level protocols or other protocols depending on the operation of layer 2. For instance, to enable correct ARP operation, each routing table entry must also contain an association between the Internet Protocol (IP) address and the MAC address of the destination. This would also enable an intermediate computer, when serving a routing function for its neighbors, to also issue proxy ARP replies instead of routing ARP broadcasts around. However, if packet forwarding is based on MAC addresses, hopefully such techniques will be unnecessary. And, if forwarding is done based on IP addresses, no ARP is strictly necessary, as long as neighboring nodes keep track of associations gleaned from route table broadcasts. Note also that layer 3 operation violates the normal subnet model of operation, since even if two Mobile Hosts share the same subnet address there is no guarantee they will be directly connected – in other words, within range of each other. This is compatible with the model of operation offered by the mobile-IP working group of the IETF [12, 10].

The new routing algorithm was particularly developed for enabling the creation of *ad-hoc* networks, which are most specifically targeted for the operation of mobile computers. However, the routing algorithm itself, and the operation of an ad-hoc network, can be beneficially used in situations which do not include mobile computers. For instance, the routing algorithm could be applied in any situation where reduced memory requirements are desired (compared to link-state routing algorithms). The operation of an *ad-hoc* network could be applied to wired as well as wireless mobile computers. In general, then, we provide a new destination-sequenced routing algorithm, and this algorithm is supplemented by a technique for damping fluctuations.

5 Properties of the DSDV Protocol

At all instants, the DSDV protocol guarantees loop-free paths to each destination. To see why this property holds, consider a collection of N mobile hosts forming an instance of an ad-hoc style network. Further assume that the system is in steady-state, i.e. routing tables of all nodes have already converged to the actual shortest paths. At this instant, the next node indicators to each destination induce a tree rooted at that destination. Thus, routing tables of all nodes in the network can be collectively visualized as forming N trees, one rooted at each destination. In the following discussion, we'll focus our attention on one specific destination x and follow the changes occurring on the directed graph $G(x)$ defined by nodes i and arcs (i, p_i^x) where p_i^x denotes the next-hop for destination x at node i. Operation of DSDV algorithm ensures that at every instant $G(x)$ is loop-free, or rather, it is a set of disjoint directed trees. Each such tree is rooted either at x or at a node whose next-hop is *nil*. Since this property holds with respect to each destination x, all paths induced by routing tables of DSDV algorithm are indeed loop free at all instants.

Potentially a loop may form each time node i changes its next-hop. This can happen in two cases. First, when node i detects that the link to its next-hop is broken, the node resets p_i^x to *nil*. Clearly, this action cannot form a loop involving i. The second scenario occurs when node i receives, from one of its neighbors k, a route to x, with sequence number s_k^x and metric m, which is selected to replace the current route it has through p_i^x. Let s_i^x denote the value of the sequence number stored at node i and d_i^x denote the distance estimate from i to x just prior to receiving route from k. i will change its next-hop from p_i^x to k only if either of the following two happens.

1. the new route contains a newer sequence number, i.e., $s_k^x > s_i^x$
2. the sequence number s_k^x is same as s_i^x, but the new route offers a shorter path to x, i.e, $m < d_{ix}$

In the first case, by choosing k as its new next-hop node i cannot close a loop. This can be easily deduced from the following observation. A node i propagates sequence number s_i^x to its neighbors only after receiving it from its current next-hop. Therefore, at all times the sequence number value stored at the next-hop is always greater or equal to the value stored at i. Starting from node i, if we follow the chain of next-hop pointers, the sequence number values stored at visited nodes would form a nondecreasing sequence. Now suppose node i forms a loop by choosing k as its next-hop. This would imply that $s_k^x \leq s_i^x$. But this contradicts our initial assumption that $s_k^x > s_i^x$. Hence, loop-formation cannot occur if nodes use newer sequence numbers to pick routes.

The loop-free property holds in the second scenario due to the theorem proved in [6], which states that in presence of static or decreasing link weights distance-vector algorithms always maintain loop-free paths.

6 Comparison with other Methods

The table 5 presents a quick summary of some of the main features of a few chosen routing protocols. The chosen set, although small, is representative of a variety of routing techniques most commonly employed in operational data networks. Except for the link-state approach, all routing methods shown in the table are a variant of the basic distance-vector approach. The comparison criteria reflects some of the most desirable features that a routing algorithm should possess for it be useful in a dynamic ad-hoc style environment. In wireless medium, communication bandwidth is the most precious and scarce resource. The formation of any kind of routing loops, therefore, is highly undesirable. In case of infrared LANS which employ pure CSMA protocol, looping packets can not only consume the communication bandwidth but they can further degrade the performance by causing more collisions in the medium. A common technique employed for loop-prevention is what we call *internodal-coordination* whereby strong constraints on the ordering of the updates among nodes is imposed. The resulting internode protocols tend to be complex. Furthermore, their update coordination may restrict a node's ability to obtain alternate paths quickly in an environment where topology changes are relatively frequent. The last criteria used for comparison is the space requirement of the routing method. Nodes in an ad-hoc network may be battery powered lap-tops, or even hand-held notebooks, which do not have the kind of memory that NSFNET dedicated routers are expected to have. Therefore, economy of space is of importance.

The primary concern with using a Distributed Bellman Ford algorithm in ad-hoc environment is its susceptibility towards forming routing loops and counting-to-infinity problem. RIP [5], which is very similar to DBF algorithm, also suffers from the same problem. Unlike DBF, RIP only keeps track of the best route to each destination, which results in some space saving at no extra performance hit. RIP also employs techniques known as *split-horizon* and *poisoned-reverse* to avoid a ping-pong style of looping, but these techniques are not powerful enough to avoid loops involving more than two hops. The primary cause of loop formation in BF style algorithms is that nodes make uncoordinated modifications to their routing tables based on some information which could be incorrect. This problem is alleviated by employing an internodal coordination mechanism as proposed by Merlin and Segall in [9]. A similar technique, but with better convergence results, is developed by Jaffe and Moss in [6]. However, we do not know of any operational routing protocols which employ these complex coordination methods to achieve loop-freedom, which leads us to the conclusion that from a practical

Routing Method	Looping	Internodal Coordination	Space Complexity
Bellman Ford [2]	s/l	-	$O(nd)$
Link State [8]	s	-	$O(n^2)$
Loop-free BF [3]	s	-	$O(nd)$
RIP [5]	s/l	-	$O(n)$
Merlin Segall [9]	loop free	Required	$O(nd)$
Jaffe Moss [6]	loop free	Required	$O(nd)$
DSDV	loop free	-	$O(n)$

s - short term loop, l - long term loop
n - number of nodes, d - maximum degree of a node

Table 5: Comparison of various routing methods

point of view the usefulness of such complex methods is diminished.

Link-state [8] algorithms are also free of the *counting-to-infinity* problem. However, they need to maintain the up-to-date version of the entire network topology at every node, which may constitute excessive storage and communication overhead in a highly dynamic network. Besides, link-state algorithms proposed or implemented to date do not eliminate the creation of temporary routing-loops.

It is evident that within ad-hoc environment design tradeoffs and the constraints under which a routing method has to operate are quite different. The proposed DSDV approach offers a very attractive combination of desirable features. Its memory requirement is very moderate $O(n)$. It guarantees loop-free paths at all instants, and it does so without requiring nodes to participate in any complex update coordination protocol. The worst case convergence behavior of the DSDV protocol is certainly non-optimal but, in the average case, it is expected that convergence will be quite rapid. We are in the process of conducting simulation studies to verify this claim.

7 Current Status and Future Work

We have implemented a preliminary version of this protocol for use with mobile computers in our lab. Currently we are making necessary modifications to the MARS simulator [1] for use in creating the appropriation simulation environment for our needs. We hope to discover good operational values via simulation for the following quantities:

- Average convergence times
- Incremental update period
- Settling time averaging method
- Full update period
- Settling time applied to triggered updates

Note that the measurement of the convergence times may depend heavily on many interesting parameters, such as the average velocity of the mobile hosts, update periods, size of the mobile host population, geographical placement of mobile hosts, existence of base stations, and average processing loads at the mobile computers.

8 Summary

Providing convenient connectivity for mobile computers in ad-hoc computers is a challenge that is only now being met. We have presented an innovative approach, DSDV, which models the mobile computers as routers, which are cooperating to forward packets as needed to each other. We make good use of the properties of the wireless broadcast medium. Our approach can be utilized at either the network layer (layer 3), or below the network layer but still above the MAC layer software in layer 2. In the latter case certain additional information should be included along with the routing tables for the most convenient and efficient operation. The information in the routing tables is similar to what is found in routing tables with today's distance vector (Bellman-Ford) algorithms, but includes a sequence number, as well as settling-time data useful for damping out fluctuations in route table updates.

All sequence numbers are generated by the destination computer in each route table entry, except for the cases when a link has been broken; the latter case is described by an ∞ metric and a sequence number which cannot be correctly generated by any destination computer. We have specified that ∞ metric route entries have sequence numbers which are odd numbers, and sequence numbers generated by each destination computer is an even number. By the natural operation of the protocol, the sequence numbers chosen to represent broken links will be superseded by real routes propagated from the newly located destination as soon as possible. Any newly propagated routes will necessarily use a sequence number greater than what was used for the broken link since the latter sequence number is chosen to be one more than the last valid route's sequence number. This allows real route data to quickly supersede temporary link outages when a mobile computer moves from one place to another.

We have borrowed the existing mechanism of triggered updates to make sure that pertinent route table changes can be propagated throughout the population of mobile hosts as quickly as possible whenever any topology changes are noticed. This includes movement from place to place, as well as the disappearance of a mobile host from the interconnect topology (perhaps as a result of turning off its power).

In order to combat problems arising with large populations of mobile hosts, which can cause route updates to be received in an order delaying the best metrics until after poorer metric routes are received, we have separated the route tables into two distinct structures. The actual routing is done according to information kept in the internal route table, but this information is not always advertised immediately upon receipt. We have defined a mechanism whereby routes are not advertised until it is likely, based upon past history, that they are stable. This measurement of the settling time for each route is be biased towards the most recent measurements for the purposes of computing an average.

We have found that mobile computers, modeled as routers, can effectively cooperate to build ad-hoc networks. We hope to explore further the necessary application-level support needed to automatically enable use of the network-layer route capabilities to provide simple access to conferencing and workplace tools for collaboration and information sharing.

9 Acknowledgment

We thank the reviewers for their many valuable comments; also, thanks to Arvind Krishna and Josh Knight for their suggestions and improvements.

References

[1] C. Alaettinoğlu, K. Dussa-Zieger, I. Matta, A. U. Shankar, and Ó. Gudmundsson. Introducing mars, a routing testbed. *ACM Computer Communication Review*, 1992.

[2] D. Bertsekas and R. Gallager. *Data Networks*, pages 297–333. Prentice-Hall, Inc., 1987.

[3] C. Cheng, R. Riley, S. P. R. Kumar, and J. J. Garcia-Luna-Aceves. A loop-free Bellman-Ford routing protocol without bouncing effect. In *ACM SIGCOMM '89*, pages 224–237, September 1989.

[4] Wim Diepstraten, Greg Ennis, and Phil Belanger. DFWMAC - Distributed Foundation Wireless Medium Access Control. IEEE Document P802.11-93/190, Nov 1993.

[5] C. Hedrick. Routing Information Protocol. RFC 1058, June 1988.

[6] J. M. Jaffe and F.H. Moss. A responsive distributed routing algorithm for computer networks. *IEEE Transactions on Communications*, COM-30(7):1758–1762, July 1982.

[7] J. J. Garcia Luna-Aceves. A unified approach to loop-free routing using distance vectors or link states. In *ACM SIGCOMM*, pages 212–223, 1989.

[8] J. M. McQuillan, I. Richer, and E. C. Rosen. The new routing algorithm for the ARPANET. *IEEE Transactions on Communications*, COM-28(5):711–719, May 1980.

[9] P. M. Merlin and A. Segall. A failsafe distributed routing protocol. *IEEE Transactions on Communications*, COM-27(9):1280–1287, September 1979.

[10] Charles Perkins. Mobile IP as seen by the IETF. *Connexions*, pages 2–20, Mar 1994.

[11] M. Schwartz and T.E. Stern. Routing techniques used in computer communication networks. *IEEE Transactions on Communications*, COM-28(4):539–552, April 1980.

[12] Bill Simpson. draft-ietf-mobileip-protocol-03.txt. Draft RFC - work in progress, May 1994.