Abstract
An ad hoc mobile network is a collection of mobile nodes that are dynamically and arbitrarily located in such a manner that the interconnections between nodes are capable of changing on a continual basis. In order to facilitate communication within the network, a routing protocol is used to discover routes between nodes. The primary goal of such an ad hoc network routing protocol is correct and efficient route establishment between a pair of nodes so that messages may be delivered in a timely manner. Route construction should be done with a minimum of overhead and bandwidth consumption. This article examines routing protocols for ad hoc networks and evaluates these protocols based on a given set of parameters. The article provides an overview of eight different protocols by presenting their characteristics and functionality, and then provides a comparison and discussion of their respective merits and drawbacks.

A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks

Elizabeth M. Royer, University of California, Santa Barbara
Chai-Keong Toh, Georgia Institute of Technology

Since their emergence in the 1970s, wireless networks have become increasingly popular in the computing industry. This is particularly true within the past decade, which has seen wireless networks being adapted to enable mobility. There are currently two variations of mobile wireless networks. The first is known as the infrastructure network (i.e., a network with fixed and wired gateways). The bridges for these networks are known as base stations. A mobile unit within these networks connects to, and communicates with, the nearest base station that is within its communication radius. As the mobile travels out of range of one base station and into the range of another, a “handoff” occurs from the old base station to the new, and the mobile is able to continue communication seamlessly throughout the network. Typical applications of this type of network include office wireless local area networks (WLANs).

The second type of mobile wireless network is the infrastructureless mobile network, commonly known as an ad hoc network. Infrastructureless networks have no fixed routers; all nodes are capable of movement and can be connected dynamically in an arbitrary manner. Nodes of these networks function as routers which discover and maintain routes to other nodes in the network. Example applications of ad hoc networks are emergency search-and-rescue operations, meetings or conventions in which persons wish to quickly share information, and data acquisition operations in inhospitable terrain.

This article examines routing protocols designed for these ad hoc networks by first describing the operation of each of the protocols and then comparing their various characteristics. The remainder of the article is organized as follows. The next section presents a discussion of two subdivisions of ad hoc routing protocols. Another section discusses current table-driven protocols, while a later section describes those protocols which are classified as on-demand. The article then presents qualitative comparisons of table-driven protocols, followed by demand-driven protocols, and finally a general comparison of table-driven and on-demand protocols. Applications and challenges facing ad hoc mobile wireless networks are discussed; and finally, the last section concludes the article.

Existing Ad Hoc Routing Protocols
Since the advent of Defense Advanced Research Projects Agency (DARPA) packet radio networks in the early 1970s [1], numerous protocols have been developed for ad hoc mobile networks. Such protocols must deal with the typical limitations of these networks, which include high power consumption, low bandwidth, and high error rates. As shown in Fig. 1, these routing protocols may generally be categorized as:

- Table-driven
- Source-initiated (demand-driven)

Solid lines in this figure represent direct descendants, while dotted lines depict logical descendants. Despite being designed for the same type of underlying network, the characteristics of each of these protocols are quite distinct. The following sections describe the protocols and categorize them according to their characteristics.

Table-Driven Routing Protocols
Table-driven routing protocols attempt to maintain consistent, up-to-date routing information from each node to every other node in the network. These protocols require each node to maintain one or more tables to store routing information, and they respond to changes in network topology by propagating updates throughout the network in order to maintain a consistent network view. The areas in which they differ are the number of necessary routing-related tables and the methods by which changes in network structure are broadcast. The following sections discuss some of the existing table-driven ad hoc routing protocols.

Destination-Sequenced Distance-Vector Routing — The Destination-Sequenced Distance-Vector Routing protocol (DSDV) described in [2] is a table-driven algorithm based on the classical Bellman-Ford routing mechanism [3]. The improvements made to the Bellman-Ford algorithm include freedom from loops in routing tables. Every mobile node in the network maintains a routing table in which all of the possible destinations within the net-
work and the number of hops to each destination are recorded. Each entry is marked with a sequence number assigned by the destination node. The sequence numbers enable the mobile nodes to distinguish stale routes from new ones, thereby avoiding the formation of routing loops. Routing table updates are periodically transmitted throughout the network in order to maintain table consistency. To help alleviate the potentially large amount of network traffic that such updates can generate, route updates can employ two possible types of packets. The first is known as a full dump. This type of packet carries all available routing information and can require multiple network protocol data units (NPDUs). During periods of occasional movement, these packets are transmitted infrequently. Smaller incremental packets are used to relay only that information which has changed since the last full dump. Each of these broadcasts should fit into a standard-size NPU, thereby decreasing the amount of traffic generated. The mobile nodes maintain an additional table where they store the data sent in the incremental routing information packets.

New route broadcasts contain the address of the destination, the number of hops to reach the destination, the sequence number of the information received regarding the destination, as well as a new sequence number unique to the broadcast [2]. The route labeled with the most recent sequence number is always used. In the event that two updates have the same sequence number, the route with the smaller metric is used in order to optimize (shorten) the path. Mobiles also keep track of the settling time of routes, or the weighted average time that routes to a destination will fluctuate before the route with the best metric is received (see [2]). By delaying the broadcast of a routing update by the length of the settling time, mobiles can reduce network traffic and optimize routes by eliminating those broadcasts that would occur if a better route was discovered in the very near future.

Clusterhead Gateway Switch Routing — The Clusterhead Gateway Switch Routing (CGSR) protocol differs from the previous protocol in the type of addressing and network organization scheme employed. Instead of a "flat" network, CGSR is a clustered multi-hop mobile wireless network with several heuristic routing schemes [4]. The authors state that by having a cluster head controlling a group of ad hoc nodes, a framework for code separation (among clusters), channel access, routing, and bandwidth allocation can be achieved. A cluster head selection algorithm is utilized to elect a node as the cluster head using a distributed algorithm within the cluster. The disadvantage of having a cluster head scheme is that frequent cluster head changes can adversely affect routing protocol performance since nodes are busy in cluster head selection rather than packet relaying. Hence, instead of invoking cluster head reselection every time the cluster membership changes, a Least Cluster Change (LCC) clustering algorithm is introduced. Using LCC, cluster heads only change when two cluster heads come into contact, or when a node moves out of contact of all other cluster heads.

CGSR uses DSDV as the underlying routing scheme, and hence has much of the same overhead as DSDV. However, it modifies DSDV by using a hierarchical cluster-head-to-gateway routing approach to route traffic from source to destination. Gateway nodes are nodes that are within communication range of two or more cluster heads. A packet sent by a node is first routed to its cluster head, and then the packet is routed from the cluster head to a gateway to another cluster head, and so on until the cluster head of the destination node is reached. The packet is then transmitted to the destination. Figure 2 illustrates an example of this routing scheme. Using this method, each node must keep a "cluster member table" where it stores the destination cluster head for each mobile node in the network. These cluster member tables are broadcast by each node periodically using the DSDV algorithm. Nodes update their cluster member tables on reception of such a table from a neighbor.

In addition to the cluster member table, each node must also maintain a routing table which is used to determine the next hop in order to reach the destination. On receiving a packet, a node will consult its cluster member table and routing table to determine the nearest cluster head along the route to the destination. Next, the node will check its routing table to determine the next hop used to reach the selected cluster head. It then transmits the packet to this node.

The Wireless Routing Protocol — The Wireless Routing Protocol (WRP) described in [5] is a table-based protocol with the goal of maintaining routing information among all nodes in the network. Each node in the network is responsible for maintaining four tables:
- Distance table
- Routing table
- Link-cost table
- Message retransmission list (MRL) table

Each entry of the MRL contains the sequence number of the update message, a retransmission counter, an acknowledgment-required flag vector with one entry per neighbor, and a list of updates sent in the update message. The MRL records which updates in an update message need to be retransmitted and which neighbors should acknowledge the retransmission [5].

Mobiles inform each other of link changes through the use
of update messages. An update message is sent only between neighboring nodes and contains a list of updates (the destination, the distance to the destination, and the predecessor of the destination), as well as a list of responses indicating which mobiles should acknowledge (ACK) the update. Mobiles send update messages after processing updates from neighbors or detecting a change in a link to a neighbor. In the event of the loss of a link between two nodes, the nodes send update messages to their neighbors. The neighbors then modify their distance table entries and check for new possible paths through other nodes. Any new paths are relayed back to the original nodes so that they can update their tables accordingly.

Nodes learn of the existence of their neighbors from the receipt of acknowledgments and other messages. If a node is not sending messages, it must send a hello message within a specified time period to ensure connectivity. Otherwise, the lack of messages from the node indicates the failure of that link; this may cause a false alarm. When a mobile receives a hello message from a new node, that node is added to the mobile’s routing table, and the mobile sends the new node a copy of its routing table information.

Part of the novelty of WRP stems from the way in which it achieves loop freedom. In WRP, routing nodes communicate the distance and second-to-last hop information for each destination in the wireless networks. WRP belongs to the class of path-finding algorithms with an important exception. It avoids the “count-to-infinity” problem [6] by forcing each node to perform consistency checks of predecessor information reported by all its neighbors. This ultimately (although not instantaneously) eliminates looping situations and provides faster route convergence when a link failure event occurs.

Source-Initiated On-Demand Routing

A different approach from table-driven routing is source-initiated on-demand routing. This type of routing creates routes only when desired by the source node. When a node requires a route to a destination, it initiates a route discovery process within the network. This process is completed once a route is found or all possible route permutations have been examined. Once a route has been established, it is maintained by a route maintenance procedure until either the destination becomes inaccessible along every path from the source or until the route is no longer desired.

Ad Hoc On-Demand Distance Vector Routing — The Ad Hoc On-Demand Distance Vector (AODV) routing protocol described in [7] builds on the DSDV algorithm previously described. AODV is an improvement on DSDV because it typically minimizes the number of required broadcasts by creating routes on a demand basis, as opposed to maintaining a complete list of routes as in the DSDV algorithm. The authors of AODV classify it as a pure on-demand route acquisition system, since nodes that are not on a selected path do not maintain routing information or participate in routing table exchanges [7].

When a source node desires to send a message to some destination node and does not already have a valid route to that destination, it initiates a path discovery process to locate the other node. It broadcasts a route request (RREQ) packet to its neighbors, which then forward the request to their neighbors, and so on, until either the destination or an intermediate node with a “fresh enough” route to the destination is located. Figure 3a illustrates the propagation of the broadcast RREQs across the network. AODV utilizes destination sequence numbers to ensure all routes are loop-free and contain the most recent route information. Each node maintains its own sequence number, as well as a broadcast ID. The broadcast ID is incremented for every RREQ the node initiates, and together with the node’s IP address, uniquely identifies an RREQ. Along with its own sequence number and the broadcast ID, the source node includes in the RREQ the most recent sequence number it has for the destination. Intermediate nodes can reply to the RREQ only if they have a route to the destination whose corresponding destination sequence number is greater than or equal to that contained in the RREQ.

During the process of forwarding the RREQ, intermediate nodes record in their route tables the address of the neighbor from which the first copy of the broadcast packet is received, thereby establishing a reverse path. If additional copies of the same RREQ are later received, these packets are discarded. Once the RREQ reaches the destination or an intermediate node with a fresh enough route, the destination/intermediate node responds by unicasting a route reply (RREP) packet back to the neighbor from which it first received the RREQ (Fig. 3b). As the RREP is routed back along the reverse path, nodes along this path set up forward route entries in their route tables which point to the node from which the RREP came. These forward route entries indicate the active forward route. Associated with each route entry is a route timer which will cause the deletion of the entry if it is not used within the specified lifetime. Because the RREP is forwarded along the path established by the RREQ, AODV only supports the use of symmetric links.

Routes are maintained as follows. If a source node moves, it is able to reinitiate the route discovery protocol to find a new route to the destination. If a node along the route moves, its upstream neighbor notices the move and propagates a link failure notification message (an RREP with infinite metric) to each of its active upstream neighbors to inform them of the erasure of that part of the route [7]. These nodes in turn
propagate the link failure notification to their upstream neighbors, and so on until the source node is reached. The source node may then choose to reinitiate route discovery for that destination if a route is still desired.

An additional aspect of the protocol is the use of hello messages, periodic local broadcasts by a node to inform each mobile node of other nodes in its neighborhood. Hello messages can be used to maintain the local connectivity of a node. However, the use of hello messages is not required. Nodes listen for retransmission of data packets to ensure that the next hop is still within reach. If such a retransmission is not heard, the node may use any one of a number of techniques, including the reception of hello messages, to determine whether the next hop is within communication range. The hello messages may list the other nodes from which a mobile has heard, thereby yielding greater knowledge of network connectivity.

**Dynamic Source Routing** — The Dynamic Source Routing (DSR) protocol presented in [8] is an on-demand routing protocol that is based on the concept of source routing. Mobile nodes are required to maintain route caches that contain the source routes of which the mobile is aware. Entries in the route cache are continually updated as new routes are learned.

The protocol consists of two major phases: route discovery and route maintenance. When a mobile node has a packet to send to some destination, it first consults its route cache to determine whether it already has a route to the destination. If it has an unexpired route to the destination, it will use this route to send the packet. On the other hand, if the node does not have such a route, it initiates route discovery by broadcasting a route request packet. This route request contains the address of the destination, along with the source node’s address and a unique identification number. Each node receiving the packet checks whether it knows of a route to the destination. If it does not, it adds its own address to the route record of the packet and then forwards the packet along its outgoing links. To limit the number of route requests propagated on the outgoing links of a node, a mobile only forwards the route request if the request has not yet been seen by the mobile and if the mobile’s address does not already appear in the route record.

A route reply is generated when the route request reaches either the destination itself, or an intermediate node which contains in its route cache an unexpired route to the destination [9]. By the time the packet reaches either the destination or such an intermediate node, it contains a route record yielding the sequence of hops taken. Figure 4a illustrates the formation of the route record as the route request propagates through the network. If the node generating the route reply is the destination, it places the route record contained in the route request into the route reply. If the responding node is an intermediate node, it will append its cached route to the route record and then generate the route reply. To return the route reply, the responding node must have a route to the initiator. If it has a route to the initiator in its route cache, it may use that route. Otherwise, if symmetric links are supported, the node may reverse the route in the route record. If symmetric links are not supported, the node may initiate its own route discovery and piggyback the route reply on the new route request. Figure 4b shows the transmission of the route reply with its associated route record back to the source node.

Route maintenance is accomplished through the use of route error packets and acknowledgments. Route error packets are generated at a node when the data link layer encounters a fatal transmission problem. When a route error packet is received, the hop in error is removed from the node’s route cache and all routes containing the hop are truncated at that point. In addition to route error messages, acknowledgments are used to verify the correct operation of the route links. Such acknowledgments include passive acknowledgments, where a mobile is able to hear the next hop forwarding the packet along the route.

**Temporally Ordered Routing Algorithm** — The Temporally Ordered Routing Algorithm (TORA) is a highly adaptive loop-free distributed routing algorithm based on the concept of link reversal [10]. TORA is proposed to operate in a highly dynamic mobile networking environment. It is source-initiated and provides multiple routes for any desired source/destination pair. The key design concept of TORA is the localization of control messages to a very small set of nodes near the occurrence of a topological change. To accomplish this, nodes need to maintain routing information about adjacent (one-hop) nodes. The protocol performs three basic functions:

- **Route creation**
- **Route maintenance**
- **Route erasure**

During the route creation and maintenance phases, nodes use a “height” metric to establish a directed acyclic graph (DAG) rooted at the destination. Thereafter, links are assigned a direction (upstream or downstream) based on the relative height metric of neighboring nodes, as shown in Fig. 5a. This process of establishing a DAG is similar to the query/reply process proposed in Lightweight Mobile Routing (LMR) [11]. In times of node mobility the DAG route is broken, and route maintenance is necessary to reestablish a DAG rooted at the same destination. As shown in Fig. 5b, upon failure of the last downstream link,
a node generates a new reference level which results in the propagation of that reference level by neighboring nodes, effectively coordinating a structured reaction to the failure. Links are reversed to reflect the change in adapting to the new reference level. This has the same effect as reversing the direction of one or more links when a node has no downstream links.

Timing is an important factor for TORA because the “height” metric is dependent on the logical time of a link failure; TORA assumes that all nodes have synchronized clocks (accomplished via an external time source such as the Global Positioning System). TORA’s metric is a quintuple comprising five elements, namely:

- Logical time of a link failure
- The unique ID of the node that defined the new reference level
- A reflection indicator bit
- A propagation ordering parameter
- The unique ID of the node

The first three elements collectively represent the reference level. A new reference level is defined each time a node loses its last downstream link due to a link failure.

TORA’s route erasure phase essentially involves flooding a broadcast clear packet (CLR) throughout the network to erase invalid routes.

In TORA there is a potential for oscillations to occur, especially when multiple sets of coordinating nodes are concurrently detecting partitions, erasing routes, and building new routes based on each other. Because TORA uses inter-nodal cooperation, its instability problem is similar to the “count-to-infinity” problem in distance-vector routing protocols, except that such oscillations are temporary and route convergence will ultimately occur.

**Associativity-Based Routing** — A totally different approach in mobile routing is proposed in [12]. The Associativity-Based Routing (ABR) protocol is free from loops, deadlock, and packet duplicates, and defines a new routing metric for ad hoc mobile networks. This metric is known as the degree of association stability. In ABR, a route is selected based on the degree of association stability of mobile nodes. Each node periodically generates a beacon to signify its existence. When received by neighboring nodes, this beacon causes the associativity tables to be updated. For each beacon received, the associativity tick of the current node with respect to the beaconing node is incremented. Association stability is defined by connection stability of one node with respect to another node over time and space. A high degree of association stability may indicate a low state of node mobility, while a low degree may indicate a high state of node mobility. Associativity ticks are reset when the neighbors of a node or the node itself move out of proximity. A fundamental objective of ABR is to derive longer-lived routes for ad hoc mobile networks.

The three phases of ABR are:

- Route discovery
- Route reconstruction (RRC)
- Route deletion

The route discovery phase is accomplished by a broadcast query and await-reply (BQ-REPLY) cycle. A node desiring a route broadcasts a BQ message in search of mobiles that have a route to the destination. All nodes receiving the query (that are not the destination) append their addresses and their associativity ticks with their neighbors along with QoS information to the query packet. A successor node erases its upstream node neighbors’ associativity tick entries and retains only the entry concerned with itself and its upstream node. In this way, each resultant packet arriving at the destination will contain the associativity ticks of the nodes along the route to the destination. The destination is then able to select the best route by examining the associativity ticks along each of the paths. When multiple paths have the same overall degree of association stability, the route with the minimum number of hops is selected. The destination then sends a REPLY packet back to the source along this path. Nodes propagating the REPLY mark their routes as valid. All other routes remain inactive, and the possibility of duplicate packets arriving at the destination is avoided.

RRC may consist of partial route discovery, invalid route erasure, valid route updates, and new route discovery, depending on which node(s) along the route move. Movement by the source results in a new BQ-REPLY process, as shown in Fig. 6a. The RN[I] message is a route notification used to erase the route entries associated with downstream nodes. When the destination moves, the immediate upstream node erases its route and determines if the node is still reachable by a localized query (LQ[I]) process, where I refers to the hop count from the upstream node to the destination (Fig. 6b).

---

**Figure 5.** a) Route creation (showing link direction assignment); b) route maintenance (showing the link reversal phenomenon) in TORA.
the destination receives the LQ packet, it REPLYS with the best partial route; otherwise, the initiating node times out and the process backtracks to the next upstream node. Here an RN[0] message is sent to the next upstream node to erase the invalid route and inform this node that it should invoke the LQ/FA process. If this process results in backtracking more than halfway to the source, the LQ process is discontinued and a new BQ process is initiated at the source.

When a discovered route is no longer desired, the source node initiates a route delete (RD) broadcast so that all nodes along the route update their routing tables. The RD message is propagated by a full broadcast, as opposed to a directed broadcast, because the source node may not be aware of any route node changes that occurred during RRCs.

**Signal Stability Routing** — Another on-demand protocol is the Signal Stability-Based Adaptive Routing protocol (SSR) presented in [13]. Unlike the algorithms described so far, SSR selects routes based on the signal strength between nodes and a node’s location stability. This route selection criteria has the effect of choosing routes that have “stronger” connectivity. SSR can be divided into two cooperative protocols: the Dynamic Routing Protocol (DRP) and the Static Routing Protocol (SRP).

![Figure 6. Route maintenance for source and destination movement in ABR.](image)

The DRP is responsible for the maintenance of the Signal Stability Table (SST) and Routing Table (RT). The SST records the signal strength of neighboring nodes, which is obtained by periodic beacons from the link layer of each neighboring node. Signal strength may be recorded as either a strong or weak channel. All transmissions are received by, and processed in, the DRP. After updating all appropriate table entries, the DRP passes a received packet to the SRP.

The SRP processes packets by passing the packet up the stack if it is the intended receiver or looking up the destination in the RT and then forwarding the packet if it is not. If no entry is found in the RT for the destination, a route-search process is initiated to find a route. Route requests are propagated throughout the network, but are only forwarded to the next hop if they are received over strong channels and have not been previously processed (to prevent looping). The destination chooses the first arriving route-search packet to send back because it is most probable that the packet arrived over the shortest and/or least congested path. The DRP then reverses the selected route and sends a route-reply message back to the initiator. The DRP of the nodes along the path update their RTs accordingly.

Route-search packets arriving at the destination have necessarily chosen the path of strongest signal stability, since the packets are dropped at a node if they have arrived over a weak channel. If there is no route-reply message received at the source within a specific timeout period, the source changes the PREP field in the header to indicate that weak channels are acceptable, since these may be the only links over which the packet can be propagated.

When a failed link is detected within the network, the intermediate nodes send an error message to the source indicating which channel has failed. The source then initiates another route-search process to find a new path to the destination. The source also sends an error message to notify all nodes of the broken link.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>DSDV</th>
<th>CGSR</th>
<th>WRP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time complexity (link addition/failure)</td>
<td>O(d)</td>
<td>O(d)</td>
<td>O(h)</td>
</tr>
<tr>
<td>Communication complexity (link addition/failure)</td>
<td>O(x = N)</td>
<td>O(x = N)</td>
<td>O(x = N)</td>
</tr>
<tr>
<td>Routing philosophy</td>
<td>Flat</td>
<td>Hierarchical</td>
<td>Flat*</td>
</tr>
<tr>
<td>Loop-free</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes, but not instantaneous</td>
</tr>
<tr>
<td>Multicast capability</td>
<td>No</td>
<td>No*</td>
<td>No</td>
</tr>
<tr>
<td>Number of required tables</td>
<td>Two</td>
<td>Two</td>
<td>Four</td>
</tr>
<tr>
<td>Frequency of update transmissions</td>
<td>Periodically and as needed</td>
<td>Periodically and as needed</td>
<td>Periodically and as needed</td>
</tr>
<tr>
<td>Updates transmitted to</td>
<td>Neighbors</td>
<td>Neighbors and cluster head</td>
<td>Neighbors</td>
</tr>
<tr>
<td>Utilizes sequence numbers</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Utilizes hello messages</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Critical nodes</td>
<td>No</td>
<td>Yes</td>
<td>No (cluster head)</td>
</tr>
<tr>
<td>Routing metric</td>
<td>Shortest Path</td>
<td>Shortest Path</td>
<td>Shortest Path</td>
</tr>
</tbody>
</table>

**Table 1. Comparisons of the characteristics of table-driven routing protocols.**

```
<table>
<thead>
<tr>
<th>Parameters</th>
<th>DSDV</th>
<th>CGSR</th>
<th>WRP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time complexity (link addition/failure)</td>
<td>O(d)</td>
<td>O(d)</td>
<td>O(h)</td>
</tr>
<tr>
<td>Communication complexity (link addition/failure)</td>
<td>O(x = N)</td>
<td>O(x = N)</td>
<td>O(x = N)</td>
</tr>
<tr>
<td>Routing philosophy</td>
<td>Flat</td>
<td>Hierarchical</td>
<td>Flat*</td>
</tr>
<tr>
<td>Loop-free</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes, but not instantaneous</td>
</tr>
<tr>
<td>Multicast capability</td>
<td>No</td>
<td>No*</td>
<td>No</td>
</tr>
<tr>
<td>Number of required tables</td>
<td>Two</td>
<td>Two</td>
<td>Four</td>
</tr>
<tr>
<td>Frequency of update transmissions</td>
<td>Periodically and as needed</td>
<td>Periodically and as needed</td>
<td>Periodically and as needed</td>
</tr>
<tr>
<td>Updates transmitted to</td>
<td>Neighbors</td>
<td>Neighbors and cluster head</td>
<td>Neighbors</td>
</tr>
<tr>
<td>Utilizes sequence numbers</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Utilizes hello messages</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Critical nodes</td>
<td>No</td>
<td>Yes</td>
<td>No (cluster head)</td>
</tr>
<tr>
<td>Routing metric</td>
<td>Shortest Path</td>
<td>Shortest Path</td>
<td>Shortest Path</td>
</tr>
</tbody>
</table>

Abbreviations:
N = Number of nodes in the network
d = Network diameter
h = Height of routing tree
x = Number of nodes affected by a topological change

* While WRP uses flat addressing, it can be used hierarchically [15].
** The protocol itself currently does not support multica; however, there is a separate protocol described in [16], which runs on top of CGSR and provides multicast capability.
```
Comparisons

The following sections provide comparisons of the previously described routing algorithms. The next section compares table-driven protocols, and another section compares on-demand protocols. A later section presents a discussion of the two classes of algorithms. In Tables 1 and 2, time complexity is defined as the number of steps needed to perform a protocol operation, and communication complexity is the number of messages needed to perform a protocol operation [11, 14]. Also, the values for these metrics represent worst-case behavior.

Table-Driven Protocols

The discussion here is based on Table 1. As stated earlier, DSDV routing is essentially a modification of the basic Bellman-Ford routing algorithm. The modifications include the guarantee of loop-free routes and a simple route update protocol. While only providing one path to any given destination, DSDV selects the shortest path based on the number of hops to the destination. DSDV provides two types of update messages, one of which is significantly smaller than the other. The smaller update message can be used for incremental updates so that the entire routing table need not be transmitted for every change in network topology. However, DSDV is inefficient because of the requirement of periodic update transmissions, regardless of the number of changes in the network topology. This effectively limits the number of nodes that can connect to the network since the overhead grows as $O(n^2)$. In CGSR, DSDV is used as the underlying routing protocol. Routing in CGSR occurs over cluster heads and gateways. A cluster head table is necessary in addition to the routing table. One advantage of CGSR is that several heuristic methods can be employed to improve the protocol’s performance. These methods include priority token scheduling, gateway code scheduling, and path reservation [4].

The WRP protocol differs from the other protocols in several ways. WRP requires each node to maintain four routing tables. This can lead to substantial memory requirements, especially when the number of nodes in the network is large. Furthermore, the WRP protocol requires the use of hello packets whenever there are no recent packet transmissions from a given node. The hello packets consume bandwidth and disallow a node to enter sleep mode. However, although it belongs to the class of path-finding algorithms, WRP has an advantage over other path-finding algorithms because it avoids the problem of creating temporary routing loops that these algorithms have through the verification of predecessor information, as described in an earlier section.

Having discussed the operation and characteristics of each of the existing table-driven routing protocols, it is important to highlight the differences. During link failures, WRP has lower time complexity than DSDV since it only informs neighboring nodes about link status changes. During link additions, hello messages are used as a presence indicator such that the routing table entry can be updated. Again, this only affects neighboring nodes. In CGSR, because routing performance is dependent on the status of

<table>
<thead>
<tr>
<th>Performance parameters</th>
<th>AODV</th>
<th>DSR</th>
<th>TORA</th>
<th>ABR</th>
<th>SSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time complexity (initialization)</td>
<td>$O(2d)$</td>
<td>$O(2d)$</td>
<td>$O(2d)$</td>
<td>$O(d + z)$</td>
<td>$O(d + z)$</td>
</tr>
<tr>
<td>Time complexity (postfailure)</td>
<td>$O(2d)$</td>
<td>$O(2d)$ or $0^*$</td>
<td>$O(2d)$</td>
<td>$O(l + z)$</td>
<td>$O(l + z)$</td>
</tr>
<tr>
<td>Communication complexity (initialization)</td>
<td>$O(N)$</td>
<td>$O(N)$</td>
<td>$O(M)$</td>
<td>$O(N + y)$</td>
<td>$O(N + y)$</td>
</tr>
<tr>
<td>Communication complexity (postfailure)</td>
<td>$O(N)$</td>
<td>$O(N)$</td>
<td>$O(N)$</td>
<td>$O(x + y)$</td>
<td>$O(x + y)$</td>
</tr>
<tr>
<td>Routing philosophy</td>
<td>Flat</td>
<td>Flat</td>
<td>Flat</td>
<td>Flat</td>
<td>Flat</td>
</tr>
<tr>
<td>Loop-free</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Multicast capability</td>
<td>Yes</td>
<td>No</td>
<td>No**</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Beaconing requirements</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Multiple route possibilities</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Routes maintained in</td>
<td>Route table</td>
<td>Route cache</td>
<td>Route table</td>
<td>Route table</td>
<td>Route table</td>
</tr>
<tr>
<td>Utilizes route cache table expiration timers</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Route reconfiguration methodology</td>
<td>Erase route; notify source</td>
<td>Erase route; notify source</td>
<td>Link repair; route repair</td>
<td>Localized broadcast query</td>
<td>Erase route; notify source</td>
</tr>
<tr>
<td>Routing metric</td>
<td>Freshest and shortest path</td>
<td>Shortest path</td>
<td>Shortest path</td>
<td>Associativity and shortest path and others***</td>
<td>Associativity and stability</td>
</tr>
</tbody>
</table>

Abbreviations:

- $d =$ Diameter of the affected network segment
- $z =$ Total number of nodes forming the directed path where the REPLY packet transits
- $z =$ Diameter of the directed path where the REPLY packet transits
- $*$ Cache hit
- ** Like CGSR, TORA also does not support multicast; however, there is a separate protocol, LAM [18], which runs on top of ORA and provides multicast capability.
- *** ABR also uses the route relaying load and cumulative forwarding delay as routing metrics.

Table 2. Comparisons of the characteristics of source-initiated on-demand ad hoc routing protocols.
specific nodes (cluster head, gateway, or normal nodes),
time complexity of a link failure associated with a cluster
head is higher than DSDV, given the additional time need-
ed to perform cluster head reselection. Similarly, this
applies to the case of link additions associated with the clus-
ter head. There is no gateway selection in CGSR since each
node declares it as a gateway node to its neighbors if it is
responding to multiple radio codes. If a gateway node
moves out of range, the routing protocol is responsible for
routing the packet to another gateway.

In terms of communication complexity, since DSDV,
CGSR, and WRP use distance vector shortest-path routing as
the underlying routing protocol, they all have the same degree
of complexity during link failures and additions.

Source-Initiated On-Demand Routing Protocols

Table 2 presents a comparison of AODV, DSR, TORA,
ABR, and SSR.

AODV employs a route discovery procedure similar to
DSR; however, there are a couple of important distinctions.
The most notable of these is that the overhead of DSR is
potentially larger than that of AODV since each DSR packet
must carry full routing information, whereas in AODV pack-
ets need only contain the destination address. Similarly, the
route replies in DSR are larger because they contain the
address of every node along the route, whereas in AODV
route replies need only carry the destination IP address and
sequence number. Also, the memory overhead may be slight-
ly greater in DSR because of the need to remember full
routes, as opposed to only next hop information in AODV. A
further advantage of AODV is its support for multicast [17].
None of the other algorithms considered in this article cur-
rently incorporate multicast communication. On the down-
side, AODV requires symmetric links between nodes, and
hence cannot utilize routes with asymmetric links. In this
aspect DSR is superior, since it does not require the use of
such links and can utilize asymmetric links when symmetric
links are not available.

The DSR algorithm is intended for networks in which
the mobiles move at moderate speed with respect to packet
transmission latency [8]. Assumptions the algorithm makes
for operation are that the network diameter is relatively
small and that the mobile nodes can enable a promiscuous
receive mode, whereby every received packet is delivered to
the network driver software without filtering by destination
address. An advantage of DSR over some of the other on-
demand protocols is that DSR does not make use of period-
ic routing advertisements, thereby saving bandwidth and
reducing power consumption. Hence, the protocol does not
incur any overhead when there are no changes in network
topology. Additionally, DSR allows nodes to keep multiple
routes to a destination in their cache. Hence, when a link
on a route is broken, the source node can check its cache
for another valid route. If such a route is found, route
reconstruction does not need to be reinvoked. In this case,
route recovery is faster than in many of the other on-
demand protocols. However, if there are no additional
routes to the destination in the source node’s cache, route
discovery must be reinitiated, as in AODV. If the route is
still required. On the other hand, because of the small
diameter assumption and the source routing requirement,
DSR is not scalable to large networks. Furthermore, as pre-
viously stated, the need to place the entire route in both
route replies and data packets causes greater control over-
head than in AODV.

TORA is a “link reversal” algorithm that is best suited
for networks with large dense populations of nodes [10]. Part
of the novelty of TORA stems from its creation of DAGs to
aid route establishment. One of the advantages of TORA is
its support for multiple routes. TORA and DSR are the only
on-demand protocols considered here which retain multiple
route possibilities for a single source/destination pair. Route
reconstruction is not necessary until all known routes to a
destination are deemed invalid, and hence bandwidth can
potentially be conserved because of the necessity for fewer
route rebuildings. Another advantage of TORA is its sup-
port for multicast. Although, unlike AODV, TORA does not
incorporate multicast into its basic operation, it functions as
the underlying protocol for the Lightweight Adaptive Multi-
cast Algorithm (LAM), and together the two protocols pro-
vide multicast capability [18]. TORA’s reliance on
synchronized clocks, while a novel idea, inherently limits its
applicability. If a node does not have GPS or some other
external time source, it cannot use the algorithm. Addition-
ally, if the external time source fails, the algorithm will cease
to operate. Furthermore, route rebuilding in TORA may not
occur as quickly as in the other algorithms due to the poten-
tial for oscillations during this period. This can lead to
potentially lengthy delays while waiting for the new routes to
be determined.

ABR is a compromise between broadcast and point-to-
point routing, and uses the connection-oriented packet for-
warding approach. Route selection is primarily based on the
aggregated associativity ticks of nodes along the path. Hence,
although the resulting path does not necessarily result in the
smallest possible number of hops, the path tends to be longer-
lived than other routes. A long-lived route requires fewer
route reconstructions and therefore yields higher throughput.
Another benefit of ABR is that, like the other protocols, it
is guaranteed to be free of packet duplicates. The reason is
that only the best route is marked valid, while all other possible
routes remain passive. ABR, however, relies on the fact that
each node is beaconing periodically. The beaconing interval
must be short enough to accurately reflect the spatial, tempo-
ar, and connectivity state of the mobile hosts. This beaconing
requirement may result in additional power consumption.
However, experimental results obtained in [19] reveal that
the inclusion of periodic beaconing has a minute influence on
the overall battery power consumption. Unlike DSR, ABR does
not utilize route caches.

The SSR algorithm is a logical descendant of ABR. It uti-
lizes a new technique of selecting routes based on the signal
strength and location stability of nodes along the path. As in
ABR, while the paths selected by this algorithm are not neces-
sarily shortest in hop count, they do tend to be more stable
and longer-lived, resulting in fewer route reconstructions. One
of the major drawbacks of the SSR protocol is that, unlike in
AODV and DSR, intermediate nodes cannot reply to route
requests sent toward a destination: this results in potentially
long delays before a route can be discovered. Additionally,
when a link failure occurs on a path, the route discovery
algorithm must be reinvoked from the source to find a new
path to the destination. No attempt is made to use partial
route recovery (unlike ABR), that is, to allow intermediate
nodes to attempt to rebuild the route themselves. AODV and
DSR also do not specify intermediate node rebuilding. While
this may lead to longer route reconstruction times since link
failures cannot be resolved locally without the intervention of
the source node, the attempt and failure of an intermediate
node to rebuild a route will cause a longer delay than if the
source node had attempted the rebuilding as soon as the bro-
en link was noticed. Thus, it remains to be seen whether
intermediate node route rebuilding is more optimal than
source node route rebuilding.
### Table 3. Overall comparisons of on-demand versus table-driven routing protocols

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Table-driven</th>
<th>On-demand</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability of routing information</td>
<td>Available when needed</td>
<td>Always available regardless of need</td>
</tr>
<tr>
<td>Routing philosophy</td>
<td>Flat</td>
<td>Mostly flat, except for CGSR</td>
</tr>
<tr>
<td>Periodic route updates</td>
<td>Not required</td>
<td>Required</td>
</tr>
<tr>
<td>Coping with mobility</td>
<td>Use localized route discovery as in ABR and SSR</td>
<td>Inform other nodes to achieve a consistent routing table</td>
</tr>
<tr>
<td>Signaling traffic generated</td>
<td>Grows with increasing mobility of active routes (as in ABR)</td>
<td>Greater than that of on-demand routing</td>
</tr>
<tr>
<td>Quality of service support</td>
<td>Few can support QoS, although most support shortest path</td>
<td>Mainly shortest path as the QoS metric</td>
</tr>
</tbody>
</table>

### Table-Driven vs. On-Demand Routing

As discussed earlier, the table-driven ad hoc routing approach is similar to the connectionless approach of forwarding packets with regard to when and how frequently such routes are desired. It relies on an underlying routing table update mechanism that involves the constant propagation of routing information. This is not the case, however, for on-demand routing protocols. When a node using an on-demand protocol desires a route to a new destination, it will have to wait until such a route can be discovered. On the other hand, because routing information is constantly propagated and maintained in table-driven routing protocols, a route to every other node in the ad hoc network is always available, regardless of whether or not it is needed. This feature, although useful for datagram traffic, incurs substantial signaling traffic and power consumption. Since both bandwidth and battery power are scarce resources in mobile computers, this becomes a serious limitation. Table 3 lists some of the basic differences between the two classes of algorithms.

Another consideration is whether a flat or hierarchical addressing scheme should be used. All of the protocols considered here, except for CGSR, use a flat addressing scheme. In [20] a discussion of the two addressing schemes is presented. While flat addressing may be less complicated and easier to use, there are doubts as to its scalability.

### Applications and Challenges

Akin to packet radio networks, ad hoc wireless networks have an important role to play in military applications. Soldiers equipped with multimode mobile communicators can now communicate in an ad hoc manner without the need for fixed wireless base stations. In addition, small vehicular devices equipped with audio sensors and cameras can be deployed at targeted regions to collect important location and environmental information which will be communicated back to a processing node via ad hoc mobile communications. Ship-to-ship ad hoc mobile communication is also desirable since it provides alternate communication paths without reliance on ground- or space-based communication infrastructures.

Commercial scenarios for ad hoc wireless networks include:
- Conferences/meetings/lectures
- Emergency services
- Law enforcement

People today attend meetings and conferences with their laptops, palmtops, and notebooks. It is therefore attractive to have instant network formation, in addition to file and information sharing without the presence of fixed base stations and systems administrators. A presenter can multicast slides and audio to intended recipients. Attendees can ask questions and interact on a commonly shared whiteboard. Ad hoc mobile communication is particularly useful in relaying information (status, situation awareness, etc.) via data, video, and/or voice from one rescue team member to another over a small handheld or wearable wireless device. Again, this applies to law enforcement personnel as well.

Current challenges for ad hoc wireless networks include:
- Multicast [21]
- QoS support
- Power-aware routing [22]
- Location-aided routing [23]

As mentioned above, multicast is desirable to support multiparty wireless communications. Since the multicast tree is no longer static (i.e., its topology is subject to change over time), the multicast routing protocol must be able to cope with mobility, including multicast membership dynamics (e.g., leave and join). In terms of QoS, it is inadequate to consider QoS merely at the network level without considering the underlying access control layer [24]. Again, given the problems associated with the dynamics of nodes, hidden terminals, and fluctuating link characteristics, supporting end-to-end QoS is a nontrivial issue that requires in-depth investigation. Currently, there is a trend toward an adaptive QoS approach instead of the "plain" resource reservation method with hard QoS guarantees. Another important factor is the limited power supply in handheld devices, which can seriously prohibit packet forwarding in an ad hoc mobile environment. Hence, routing traffic based on nodes' power metrics is one way to distinguish routes that are more long-lived than others. Finally, instead of using beaconing or broadcast search, location-aided routing uses positioning information to define associated regions so that the routing is spatially oriented and limited. This is analogous to associativity-oriented and restricted broadcast in ABR.

Current ad hoc routing approaches have introduced several new paradigms, such as exploiting user demand, and the use of location, power, and association parameters. Adaptivity and self-configuration are key features of these approaches. However, flexibility is also important. A flexible ad hoc routing protocol could, for example, invoke table-driven and/or on-demand approaches based on situations and communication requirements. The "toggle" between these two approaches may not be trivial since concerned nodes must be "in sync" with toggling. Coexistence of both approaches may also exist in spatially clustered ad hoc groups, with intracluster employing the table-driven approach and intercluster employing the demand-driven approach or vice versa. Further work is necessary to investigate the feasibility and performance of hybrid ad hoc routing approaches. Lastly, in addition to the above, further research in the areas of media access control, security, service discovery, and Internet protocol operability is required before the potential of ad hoc mobile networking can be realized.

### Conclusion

In this article we provide descriptions of several routing schemes proposed for ad hoc mobile networks. We also provide a classification of these schemes according to the routing strategy (i.e., table-driven and on-demand). We have presented a comparison of these two categories of
routing protocols, highlighting their features, differences, and characteristics. Finally, we have identified possible applications and challenges facing ad hoc mobile wireless networks. While it is not clear that any particular algorithm or class of algorithm is the best for all scenarios, each protocol has definite advantages and disadvantages, and is well suited for certain situations. The field of ad hoc mobile networks is rapidly growing and changing, and while there are still many challenges that need to be met, it is likely that such networks will see widespread use within the next few years.

Acknowledgments

We would like to thank C.-C. Chiang, J. J. Garcia-Luna-Aceves, David Maltz, Vincent Park, and Charles Perkins for their assistance in ensuring accurate descriptions of the operation of their protocols.

References


Biographies

Chai-Kheong Toh (M’97) (C.K.Toh@acm.org) received his Diploma in electronics and communication engineering and a Certificate of Merit award from the Singapore Polytechnic in 1996, his B.Eng. degree in electronics engineering with first class honors from the University of Manchester Institute of Science and Technology (UMIST) in 1991, and his Ph.D. degree in computer science from the Computer Laboratory, University of Cambridge, England in 1996. He founded and chaired the Mobile Special Interest Group (Mobile SIG) from 1994 to 1996. Before joining Cambridge University, he was a network specialist, R&D engineer and technical staff member. At Cambridge, he was an Honorary Cambridge Commonwealth Trust Scholar and a King’s College Cambridge Research Scholar. He authored the book Wireless ATM and Ad Hoc Networks: Protocols and Architectures (Kluwer, 1996). He serves on the editorial board for AOM Mobile Computing and Communications Review, IEEE Network, and Personal Technologies Journal (Springer Verlag). He is a member of IEEE, USENIX, ACM, Sigma Xi Honor Society, New York Academy of Science, American Association for the Advancement of Science (AAAS), and a Fellow of the Cambridge Philosophical Society and Cambridge Commonwealth Society. He is currently an assistant professor with the School of Electrical and Computer Engineering at the Georgia Institute of Technology, Atlanta, directing the Mobile Multimedia and High Speed Networking Laboratory.

IEEE Personal Communications • April 1999