SICS Technical Report
T99/07

ISSN 1100-3154
ISRN:SICS-T–99/07-SE

A Taxonomy for Routing Protocols in Mobile Ad Hoc Networks [1]

Laura Marie Feeney

Swedish Institute of Computer Science
Box 1263, SE-164 29 Kista, Sweden
http://www.sics.se/∼lmfeeney

October, 1999

Abstract

A Mobile Ad hoc NETwork (manet) is a mobile, multi-hop wireless network which is capable of autonomous operation. It is characterized by energy-constrained nodes, bandwidth-constrained, variable-capacity wireless links and dynamic topology, leading to frequent and unpredictable connectivity changes.

In the absence of a fixed infrastructure, manet nodes cooperate to provide routing services, relying on each other to forward packets to their destination. Routing protocols designed for the fixed network are not effective in the dynamic and resource-constrained manet environment; many alternative routing protocols have been suggested.

This report provides an overview of a number of manet routing protocols. More importantly, it defines a taxonomy that is suitable for examining a wide variety of protocols in a structured way and exploring tradeoffs associated with various design choices. The emphasis is on practical design and implementation issues rather than complexity analysis.

Keywords: mobile ad hoc networks, routing protocols

[1] This report was prepared in context of the HP/SICS Internet Research Insititute, Bristol, UK.

1 Introduction

A mobile ad hoc network (or manet) is a group of mobile, wireless nodes which co-operatively and spontaneously form a network independent of any fixed infrastructure or centralized administration. In particular, a manet has no base stations: A node communicates directly with nodes within wireless range and indirectly with all other destinations using a dynamically-determined multi-hop route though other nodes in the manet. Such networks have potential value in areas such as tactical communication, disaster response, conferencing applications and various forms of ubiquitous computing. They are generally envisioned as moderate size networks, consisting of tens or hundreds of nodes.

The manet environment is characterized by energy-constrained nodes, bandwidth-constrained, variable-capacity wireless links and dynamic topology, leading to frequent and unpredictable connectivity changes. Traditional link-state and distance vector routing algorithms are not effective in this environment. Numerous manet routing protocols have been proposed, both inside and outside the IETF Manet Working Group [17].

The goal of this report is to survey the state of the art in manet routing protocols. However, the number and variety of protocols (from ABR to ZRP!), limits the usefulness of a catalog of protocol descriptions or point-by-point comparisons. Therefore, section 2 is devoted to creating a taxonomy that is suitable for examining protocols in a structured way and exploring various aspects of the design space. Compared to another excellent survey work [23], this report includes a slightly different set of protocols, with an emphasis on design and implementation issues rather than compexity analysis.

Sections 4 - 5 contain relatively high-level descriptions of a number of protocols. These are intended to emphasize key design elements, rather than describe the many subtleties of the various algorithms.

It must be emphasized that the protocols discussed in this report **vary significantly in the extent to which their performance has been investigated**. Some have been the subject of quite limited simulations, while the behavior of others has been has been the subject of intensive study. Until recently, there has been relatively little work that allows for direct comparison of protocol performance. The development of common simulation environments [6, 1] has begun to make such comparisons possible [3, 14]. Much work remains to be done in understanding the most appropriate mobility and traffic scenarios, as well as the key performance metrics [7].

Performance encompasses effectiveness, efficiency and route properties. A protocol's effectiveness is the reliability with which a source node obtains routing information for a desired destination. Efficiency refers to the resource consumption required to provide a high level of effectiveness.

Describing resource consumption is a complex task: although costs are MAC-layer dependent, it is clear that measuring raw bandwidth use is not sufficient. The high cost of channel acquisition in wireless media makes it important to consider the number of packets as well as the bytes transmitted. The proportions and patterns of broadcast and unicast traffic are also important. Protocols that lead to bursty or chain-reaction behaviors will have problems due to increased collisions and contention. Energy considerations may apply: all nearby nodes spend energy to process a broadcast message, this may not be true for a unicast message.

Routing performance metrics also include route optimality, route latency and route diversity. Route optimality, e.g. shortest-path routes, is a secondary objective of some protocols. Because routes are often short-lived, it may be preferable to use a sub-optimal route while it is available, than to spend time and network resources finding optimal routes. Route latency refers to the time it takes for a source to obtain initial routing information for a new destination. Route diversity is desirable due to both bandwidth and energy constraints. If the protocol tends to focus many flows along the same path, those nodes may suffer excessive battery consumption. Spatial diversity of routes optimizes the use of available bandwidth.

2 Taxonomy

Manet routing protocols may be classified according to several criteria, reflecting fundamental design and implementation choices.

communication model What is the wireless communication model?

structure Are all nodes treated uniformly? How are distinguished nodes selected?

state information Is network-scale topology information obtained at each node?

scheduling Is route information continually maintained for each destination?

2.1 Communication Model

The underlying wireless communication model creates the most basic division in this taxonomy, separating protocols designed for multi-channel and single channel communication.

Multi-channel protocols are low-level routing protocols which combine channel assignment and routing functionality. Such protocols are generally used in TDMA or CDMA-based networks. Examples include Clusterhead Gateway Switched Routing[5].

A larger class of protocols assumes that nodes communicate over a single logical wireless channel. Though generally CSMA/CA-oriented, these protocols vary in the extent to which they rely on specific link-layer behaviors. Some leverage link-level failure detection to avoid other forms of failure detection such as beaconing or higher-level acknowledgements. Other procotols use information extracted from data traffic observed while operating the network interface in promiscuous mode.

Some manet routing protocols are based on more specific link-layer properties, such as the RTS/CTS control sequence used by the popular IEEE 802.11 (and other) MAC layers to avoid collisions due to hidden and exposed terminals. There are also a few protocols which incorporate "physical layer" information, such as received signal strength or geographic position into the routing algorithm.

This report discusses single channel, network-layer protocols, some of which make use of link-layer information.

2.2 Structure

Routing protocols may be categorized as uniform or non-uniform protocols.

2.2.1 Uniform Protocols

In a uniform protocol, none of the nodes take on a distinguished role in the routing scheme: each sends and responds to routing control messages the same way. No hierarchical structure is imposed on the network. Although such a protocol avoids the resource costs involved in maintaining high-level structure, scalability may become an issue in larger networks.

2.2.2 Non-uniform Protocols

Non-uniform protocols attempt to limit routing complexity by reducing the number of nodes participating in a route computation. Such an approach can improve scalability and reduce communication overhead; alternatively, it can support the use of algorithms of greater computational or communication complexity than is possible in the full ad hoc network. In addition, higher-level topology information can facilitate load balancing and QoS support. Non-uniform protocols fall into two categories: protocols in which each node focuses routing activity on a subset of its neighbors and protocols in which the network is topologically partitioned.

In neighbor-selection protocols, each node selects some subset of its neighbors to take a distinguished role in route computation and/or forwarding traffic on its behalf. Each node makes its selections independently; there is no negotiation process in which nodes must achieve consensus, nor is a node's selections affected by non-local topological changes. However, protocol performance may be sensitive to the actual neighbor-selection.

In partitioning protocols, nodes negotiate a topological partitioning of the network. This is a distributed operation, as there is no central topology manager. Generally, nodes are partitioned into clusters, whose membership changes as network connectivity changes. Some nodes take on a distinguished role in the routing process, possibly acting as a “cluster-head” or as a “gateway” between two clusters. A variant of some uniform protocol, e.g. source routing, is used for inter-cluster routing.

Significant resources are needed to impose topological structure on a highly dynamic ad hoc network. Partitioning models are often based on clique-finding or minimum dominating set – NP-hard problems requiring approximative techniques. In addition, some clustering algorithms are based on “least cluster change” principle, incorporating a topology change so as to minimize its scope rather than maintain near-optimal structure. A “bad” partitioning can have a large negative performance impact.

2.3 State Information

Protocols may be described in terms of the state information obtained at each node and/or exchanged among nodes.

2.3.1 Topology-based Protocols

Nodes participating in topology-based protocols maintain large-scale topology information. The best known such protocols are “link-state” protocols. In link-state protocols, every node advertises its connectivity with each of its immediate neighbors to all other nodes in the network. The shortest path to each destination is computed using, for example, Dijkstra’s algorithm and maintained in a routing table. Each node makes routing decisions based on complete topology information.

Such protocols are effective for routing in the fixed Internet, but the amount of data and the frequency with which it must be distributed throughout the network are a significant disadvantage in the resource-constrained, highly dynamic manet environment.

For this reason, variants of the link-state protocol which require less data exchange or only apply the expensive link-state computation to a few nodes have been introduced for use in the manet environment. Other topology-oriented manet routing protocols provide nodes with large-scale topology information, but only for “active” areas of the network. For example, nodes participating in source routing protocols have access to complete topology information about the path along which they are sending, forwarding or receiving traffic.

Large-scale topology information can be used not only for basic routing functionality, but also improve route selection, load balancing and QoS management.

2.3.2 Destination-based Protocols

Nodes participating in destination-based protocols do not maintain large-scale topology information, although some maintain local topology information (e.g. 1 or 2-hop neighborhood). The best known such protocols are ”distance-vector” protocols, which maintain a distance (hop count or other metric) and vector (next hop) to a destination.

Every node exchanges its distance estimates for all other network nodes with each of its immediate neighbors. Such algorithms are known to behave poorly - leading to routing loops and slow convergence - in a dynamic environment.

Several proposed protocols adapt the distance vector approach for operation in mobile ad-hoc networks. Techniques include the use of sequence numbers and next-to-last-hop in the distance-vector information to ensure freedom from long-lived routing loops.

Other destination-based protocols entirely avoid the exchange of distance information. Nodes only maintain distance vector routing Information for "active" destinations - those to which they are sending or forwarding traffic.

2.4 Scheduling

Finally, protocols can be be considered in terms of when a source obtains route information as it initiates traffic flow to a destination.

2.4.1 Proactive Protocols

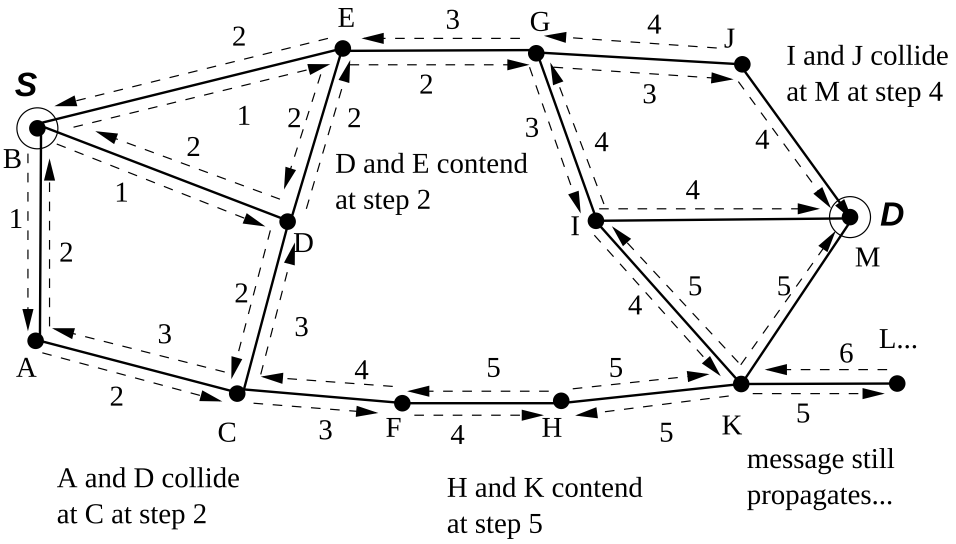
Proactive, or table-driven, routing protocols attempt to maintain routing information for all known destinations at every source. Fixed network routing protocols are proactive, as are some manet protocols.

In proactive routing protocols, nodes exchange route information periodically and/or in response to topology change. This has the advantage of minimizing delay in obtaining a route when initiating traffic to a destination and quickly determining whether a destination is reachable. This process can also consume significant network resources. Moreover, the resources used to establish and re-establish unused routes is entirely wasted.

2.4.2 On-demand Protocols

In the resource-constrained, highly dynamic manet environment, the cost of maintaining unneeded routing information is a more serious problem than it is in a fixed network. Therefore, a class of reactive protocols, which discover needed routes on-demand, has been proposed.

These protocols consist of a "route discovery" process and a "route maintenance" process. The route discovery process is initiated when a source needs a route to a destination, for which it broadcasts a route_request. Each intermediate node receiving the request records the link over which it was received and re-broadcasts it (silently ignoring duplicates). When the request reaches the destination, a route_reply is sent back to the source, instantiating routing information at the appropriate intermediate nodes. (The destination eventually receives the request over each viable route and can



Graph: 13 nodes, avg. degree 2.6

Figure 1: Broadcast Storm Problem

select one based on metrics (e.g. hop-count or latency) included in the request.) Once the reply reaches the source, data traffic can be sent to the destination.

On-demand protocols do not spend resources maintaining unneeded routes, but the route discovery process is potentially both expensive and unpredictable. In particular, route latency is much more variable than the constant-time table-lookup associated with proactive protocols.

In order to ensure that the route_request reaches the destination, it must be disseminated throughout the network. Flooding the network with route_request messages leads to the "broadcast storm problem"[18], Figure 1. Flooding is highly redundant: Each node receives the route_request *degree* times and the request can propagate far beyond the destination. Because nearby nodes will receive and re-broadcast messages at roughly the same time, contention (when senders can hear each other) and collision (when senders cannot hear each other) will be common.

Techniques to improve the efficiency of broadcast flooding include:

- Adding random delay to re-broadcasts to reduce collisions.

- Using a sequence of hop-limited route_requests rather than a single, pervasive request.

- Trading the reduced traffic load obtain by using probabilistic re-broadcast against the risk that the request does not reach the destination.

- Using location or signal strength heuristics for determining most "productive" re-broadcasters.

In addition, many on-demand protocols specify that an intermediate node that has a route to the destination may send a route_reply on its behalf. This makes cache correctness and resistance to faulty cache data especially important. Non-uniform protocols often leverage the structure imposed on the network in order to improve the efficiency of the route discovery process.

The route maintenance process deletes failed routes and re-initiates route discovery in the case of topology change. Route maintenance depends on the failure detection model provided by lower layers. If only upper layer (i.e. end-to-end) failure detection is available, then source discovery must be reinitiated at the source node. If hop-by-hop failure detection, based on link-layer or passive acknowledgments, is used then it may be possible to do a localized route discovery to repair the broken route. However, repeated localized repairs may lead to the formation of sub-optimal routes. Some protocols incorporate proactive "hello messages" into the route maintenance process.

3 Protocol Overviews

The protocol overviews below are intended to emphasize key design and how they relate to the taxonomy described above, rather than describe every subtlety of the protocol specification.

ABR associativity based routing

AODV ad-hoc on-demand distance vector routing

CBRP cluster based routing protocol

CEDAR core-extraction distributed ad-hoc routing

DSDV destination sequenced distance vector

DSR dynamic source routing

GSR global state routing

OLSR optimized link state routing

TORA temporally ordered routing algorithm

ZRP zone routing protocol

It should be noted that while some of these protocols are no longer in active development, all of them illustrate the tradeoffs associated with various regions in this protocol taxonomy.

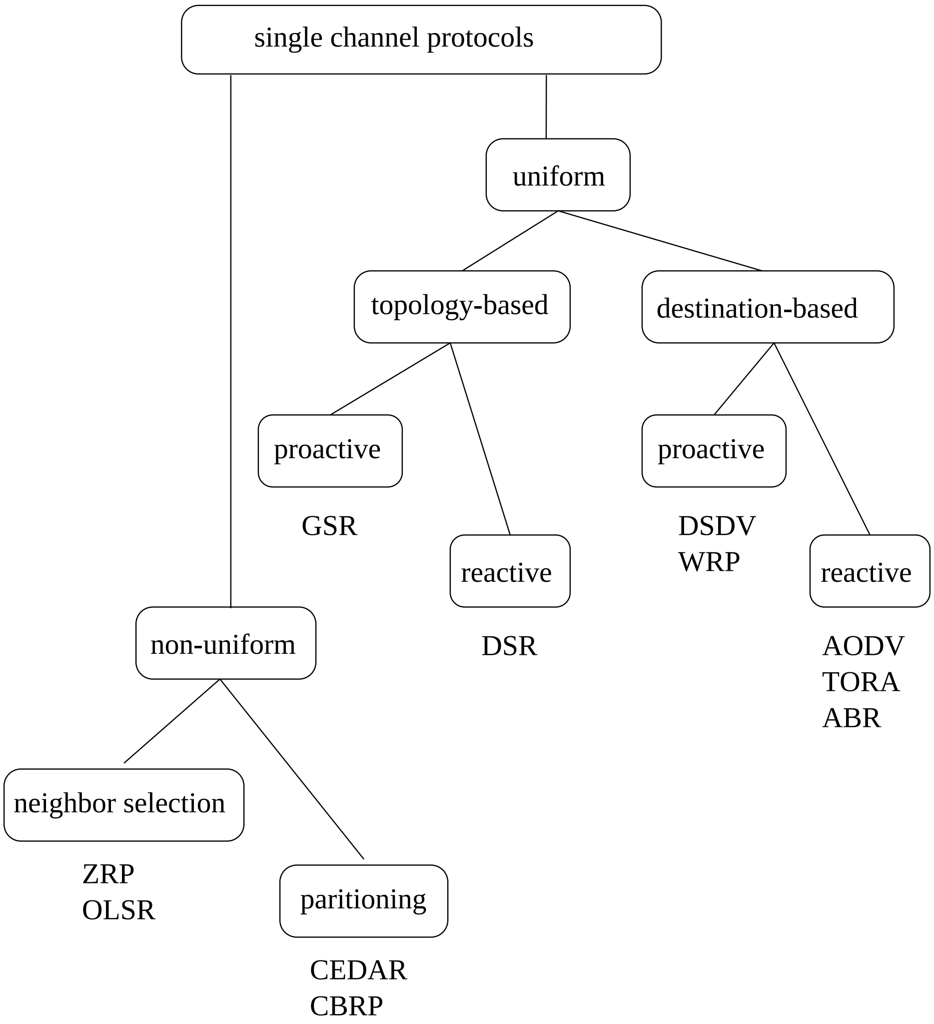


Figure 2: Protocol Taxonomy

4 Uniform Structure Protocols

4.1 Topology-based Protocols

4.1.1 DSR

Dynamic Source Routing (DSR) [15, 2] is a uniform, topology-based, reactive protocol. It emphasizes aggressive caching and deduction of topology information extracted from source routing headers.

Route discovery is as follows: When a source has no route to a destination, it broadcasts a route_request. Each intermediate node that receives the request appends its ID to the request and re-broadcasts it (silently ignoring duplicate requests and any request in which its ID already appears). When the route_request reaches the destination (or an intermediate node with a source route to the destination), it determines a complete source route to the destination.

The destination (or intermediate node) sends a route_reply containing the route back to the source. This can be sent along the reverse source route (assuming bi-directional links) or piggybacked onto a route_request initiated by the destination. When the source receives a route_reply, it caches the source route and includes it in the header of each data packet. Intermediate nodes forward the packet according to the route specified in the header.

Many optimizations based on aggressive caching and analysis of topology information are incorporated into this scheme. From the source route included in each data packet, each intermediate node can trivially extract routes to all downstream nodes. Additional topology information can be deduced by combining information about several routes. Still further topology information may be obtained by nodes operating their network interfaces in promiscuous mode, eavesdropping on routes being used by adjacent nodes.

Thus, nodes on and near active routes will incorporate more and more of the large-scale topology of the "interesting" parts of the network into their route caches. A high cache hit rate means that the expensive route discovery is used less frequently and finds routes more quickly. However, aggressive caching also increases the risk of stale route information being injected into the network [16, 11].

Route maintenance also uses cached information: If link failure is detected on a hop-by-hop basis, the packet may be salvaged by an intermediate node which replaces the broken route with a better one. A route_error message is sent to the source. Like other routing traffic, route_errors may be eavesdropped and failed routes removed from the intermediate nodes' caches. This minimizes the effects of faulty cached information. In the case of route failure, the source reinitiates the route discovery process.

Topology changes can also lead to the formation of shorter routes. A node that overhears a data packet with its own ID as a downstream node in the route can send an unsolicited route_reply back to the source, indicating the more direct route.

4.1.2 GSR

Global State Routing (GSR) [4] is a uniform, topology-oriented, proactive routing protocol. It is a variant of traditional link-state protocols, in which each node sends link-state information to every node in the network each time its connectivity changes. GSR reduces the cost of disseminating link-state information by relying on periodic exchange of sequenced data rather than flooding.

In GSR, each node periodically broadcasts its entire topology table to its immediate neighbors. The topology table includes the node's most recent assessment of its local connectivity and its current link-state information for the whole network topology. Each entry is tagged with a sequence number. A destination's link-state entry is replaced only if the received entry has a larger sequence number.

Based on the complete topology information in the topology table, any shortest-path algorithm can be used to compute a routing table containing the optimal next-hop information for each destination. GSR defines a variant of Dijkstra's algorithm for this purpose.

The less frequent exchange of large topology tables is much more efficient than flooding, especially in a wireless environment, where channel acquisition cost is significant. It also ensures that routing overhead does not depend on node mobility. However, traffic will be dropped until new connectivity information has been disseminated, an operation that can take several update intervals.

4.2 Destination Based Protocols

4.2.1 DSDV

Destination Sequenced Distance Vector (DSDV) [20] is a uniform, destination-based proactive protocol. DSDV is a variant of traditional distance-vector algorithms. It avoids routing loops by tagging route information for each destination with a sequence number originated by the destination. DSDV also prevents routing fluctuations by delaying advertisement of possibly unstable routes.

Each node periodically broadcasts its current routing table, containing the distance and highest known sequence number for each destination. The message also includes the sender's own sequence number, which is incremented for each new broadcast. Each receiving node compares the broadcast sequence number for each destination with the one in its routing table. If the sequence number is higher, the receiver updates its routing table entry, naming the sender as the next-hop and incrementing the distance by one hop. If the sequence numbers are equal and incremented distance is smaller, the receiver also updates its routing table.

When a node detects link failure, it sets the distance to each destination routed via the failed link to infinity and increments the sequence numbers associated with these entries. Because the sequence number for each of these destinations has increased, the change will be propagated though the network. Each of these destinations is thus effectively disconnected from the network, until it generates for itself a new routing message, containing a new sequence number.

In addition to periodic broadcasts of complete routing tables, nodes also send incremental updates. In order to reduce the amount of routing traffic and to avoid route fluctuations, nodes have some latitude in determining what information is included in an incremental update. Some information is advertised immediately, such as new destinations, failed routes (distance infinity) and repaired routes (updating an entry that had distance infinity).

Other updates, such as an improved distance metric for a route may not be "important enough" to advertise immediately. Updates with equal sequence numbers, but different metrics (that is, received via different neighbors) can arrive in arbitrary order. To prevent this, nodes delay advertising the new route, based on weighted average history of the time between the arrival of the first route and the best one. Similarly, a new sequence number for existing route data may not justify immediate advertisement. However, simulation [3] suggests that prompt advertisement may help preempt unnecessarily wide dissemination of route failure information.

4.2.2 AODV

Ad hoc On-demand Distance Vector (AODV) [21, 22] is a uniform, destination-based, reactive protocol. It incorporates the destination sequence number technique used in DSDV into an on-demand protocol.

Route discovery is as follows: When a source has no route to a destination, it broadcasts a route_request. Nodes receiving the request record a "reverse" destination vector back toward the source, using the node from which the broadcast was received as the next-hop, and re-broadcast the request (silently ignoring duplicate requests).

When the route_request reaches the destination, the destination sends a route_reply back to the source, using the path defined by the reverse destination vectors. If an intermediate node has an up-to-date route to the destination, it may send a route_reply on behalf of the destination. As the route_reply follows the reverse path back to the source, the corresponding "forward" destination vector is created at each intermediate node. Once the route_reply reaches the source, data traffic can flow along the newly established route. Reverse destination vectors not activated by the route_reply are quickly expired.

Destination vector algorithms are subject to routing loops. Like DSDV, AODV uses a destination-generated sequence number to ensure the freshness of each route. Each route_request is tagged with highest sequence number the source has seen for that destination. An intermediate node may send a route_reply on behalf of a destination only if its highest sequence number for the destination is at least as high as the one in the route_request and the route has not expired (i.e. recently used to successfully forward traffic). If the route_reply is sent by the destination, it is tagged with a sequence number reflecting the last topology change known to the destination.

The route maintenance process is as follows: When a node detects a link failure, it sends an unsolicited route_reply, setting the distance to infinity and incrementing the destination sequence number, to each neighbor for which it is forwarding traffic through the link. This route_reply is thus propagated to each source for which traffic is being routed through the failed link, causing the route discovery process to be reinitiated. When the incremented destination sequence number reaches the destination (via the new route_request), this is reflected in the new destination sequence number sent in the route_reply. The destination node also increments the destination sequence number on detecting a change in its neighbor set, but does not generate an unsolicited route_reply.

4.2.3 TORA

Temporally Ordered Routing Algorithm(TORA) [19] is a uniform, destination-based, reactive protocol. TORA is based on earlier "link reversal" algorithms. A destination-oriented directed acyclic graph is built for each destination. If connectivity changes result in a node losing all of its outbound links, the node "reverses" the direction of some or all of its inbound links.

TORA assumes that each node is informed of link-status changes for any of its immediate neighbors. If this is not provided by the link-layer, beaconing is required.

When a source has no route to a destination, it broadcasts a route_request for the destination. The request is rebroadcast until it reaches the destination, which is defined to have zero height with respect to itself. The destination broadcasts an update_message, indicating its height. Each node that receives the update_message updates its height to be one higher than the height in the update_message and broadcasts an update_message, indicating its new height. The updates must be broadcast reliably and ordered by a synchronized clock or logical timestamp in order to prevent long-lived loops. This process creates a DAG from the source to the destination, which is used for hop-by-hop routing.

If a link failure is detected, the node adjusts its height to a local maximum with respect to the failed link ("link reversal") and broadcasts an update_message to its neighbors. A route failure is propagated only when a node loses its last downstream link. TORA distinguishes nodes whose height already reflects a link reversal ("reflected"). Partitions can therefore be explicitly detected and only in this case is a clear_route message propagated and the expensive route discovery reinitiated. Again, reliable, ordered broadcast is required in order to prevent long-lived routing loops.

4.2.4 ABR

Associativity Based Routing (ABR)[25] is a uniform, destination-based, reactive protocol. ABR uses end-to-end topology information in route selection, preferring routes that reflect long-lived associations. However, only destination-vectors are maintained during routing.

Route discovery is as follows: When a source has no route to a destination, it broadcasts a route_request. When an intermediate node receives the request, it appends its ID to the route_request and re-broadcasts it (silently ignoring duplicates). Each route_request received by the destination node contains a different complete source route to the destination. The destination learns many feasible routes, eventually selecting an "optimal" one.

Each node broadcasts a periodic heartbeat and counts heartbeats received from its neighbors. This measures the associativity, or time during which the link has been stable, between nodes. The associativity of each hop is accumulated in the route_request. Routes with high threshold and aggregate associativity are considered superior, even if there are shorter routes. ABR is therefore oriented toward a mobility model characterized by large intervals (or regions) of little connectivity change, interspersed with brief episodes of high mobility.

The destination sends a route_reply back to the source along the selected route. Each intermediate node activates the appropriate forwarding information in its routing table.

The route maintenance process is quite complex. Response to link failure is a combination of local and source initiated repair, depending on the position of the failed link in each route for which it was used. Nodes downstream of the link failure send route_error messages toward the destination, deleting invalid route entries. A local_request (hop-limited route_request) is initiated by the upstream node. If the query fails to find a new partial route, the next node upstream is so informed and initiates a local_request. If the process traverses too much of the distance back to the source, it is abandoned and a route_error is sent to the source, which reinitiates the route discovery process. This is sensitive to race conditions when there are simultaneous topology changes. Consistent behavior is dependent on the most recent request suppressing earlier attempts.

5 Non-uniform Protocols

5.1 Neighbor-selection Protocols

5.1.1 OLSR

Optimized Link State Routing (OLSR) [12] is a neighbor selection based protocol, in which each node includes only a subset of its neighbors in a link-state protocol.

In link-state protocols, each node distributes its link-state information to every other node in the network each time its connectivity changes. OLSR reduces the cost of this operation in two ways. First, multi-point relay is used to reduce redundant rebroadcasting during the flooding operation. Second, in order to reduce the size of link-state messages and the frequency with which they are initiated, each node only broadcasts the state of nodes in its multi-point relay set.

A node's multi-point relay (MPR) set is the minimal (or near minimal) subset of its one-hop neighbors which must rebroadcast a message so that it is received by all of its two-hop neighbors. When a node sends a broadcast message, all of its neighbors receive and process the data. However, only those neighbors which belong to the source node's MPR set and have not previously received the message rebroadcast it. This reduces the number of broadcast messages needed to flood a message through the network. Since each node selects its MPR set independently, it must know the topology of its two-hop neighborhood, but additional inter-nodal coordination is not required.

In the OLSR protocol, each node uses this flooding technique to distribute the link-state of its MPR set. This is done periodically, with the period decreasing to a minimum on detecting a change in the MPR and increasing to a refresh interval while the MPR set remains stable. Each node uses the topology information in the combined link-state messages to construct its routing tables. It is sufficient for nodes to advertise only their MPR sets, rather than their complete neighbor list; by definition, each of the node's two-hop neighbors is a one-hop neighbor of some node in the MPR set.

5.1.2 ZRP

Zone Routing Protocol (ZRP) [8, 10] is a neighbor selection based protocol, designed to combine the advantages of proactive and reactive routing strategies. Each node applies the former within its local neighborhood and the latter when establishing routes to further destinations.

A node defines its zone as the set of nearby nodes which can be reached within zone-radius hops. Nodes that are exactly zone-radius hops away form the border of the zone. Each node is therefore potentially located in many zones and on many borders.

The proactive intra-zone routing protocol (IARP) is a modified distance-vector algorithm. When a source has no IARP route to a destination, it invokes a reactive inter-zone routing protocol (IERP). The route discovery process is somewhat different from that of protocols operating in a uniform routing structure, as it can leverage the availability of zone-wide information at each node. The source therefore "bordercasts" a route_request to its border nodes. On receiving the request, each border node consults its IARP routing information. If the destination is not in the border node's zone, it adds its ID to the request and re-"bordercasts" it. When the request reaches a border node whose zone includes the destination, it contains a non-strict source route to the destination; that is, each listed node has a IARP route to the next and previous elements in the source route. This loose source route can be used to accumulate a complete source route, or portions of the route can be cached at intermediate nodes. It can also be used to find route optimizations.

Bordercast is more expensive than the broadcast flooding used in other reactive protocols. Nodes generally have many more border nodes than neighbors; in addition, each bordercast message has to traverse zone-radius hops to the border. Therefore, ZRP proposes a number of mechanisms to reduce the cost of bordercast route_requests [9]. Redundancy suppressing mechanisms based on caching overhead traffic include query detection, early termination and loopback termination. The IARP topology information maintained at each node can be used for backward search prevention and selective bordercasting. Selective bordercasting is similar to the MPR selection used in OLSR; each node selects a subset of its border nodes that achieves equivalent coverage.

Route failure is detected proactively, in conjunction with the IARP. Failures may be repaired locally, in which case it may not even be necessary to inform the source node. If necessary, a hop-limited local_request can be used to repair the route, or a route_error message can be sent to re-initiate the route discovery from the source.

5.2 Partitioning Protocols

5.2.1 CEDAR

Core Extraction Distributed Ad hoc Routing (CEDAR) [24] is a partitioning protocol, emphasizing QoS support. Each partition includes a "core node". The core nodes use a reactive source routing protocol to outline a route from a source to destination. From this directional guideline, a QoS admissible route is generated.

CEDAR partitioning uses minimum dominating set. This is the minimum subset of nodes such that all nodes are at most one-hop away from a dominating node. (MDS is NP-hard; approximative techniques must be used.) The core consists of the dominators and "tunnels", unicast paths which connect each core node with nearby core nodes. By definition of MDS, these tunnels consist of at most two intermediate non-core nodes and form a connected graph. In order to discover their core neighbors and select tunnels, core nodes advertise their presence in the three-hop neighborhood.

When a source has no route to a destination, it forwards a route_request to its dominator. Instead of using broadcast flooding to disseminate the request, CEDAR uses a unicast mechanism, the "core broadcast", in which a core node tunnels the message to each of its core neighbors. (This is made more efficient by a proposed MAC layer enhancement, in which RTS/CTS packets are monitored to suppress duplicate transmission of core broadcast packets.) This mechanism is used to discover a "core path", or source route from the dominator of the source to the dominator of the destination. A route_reply containing this route is sent back to the source.

Core broadcast is also used to disseminate QoS state information. When a link experiences a large change in available bandwidth capacity, the endpoints inform their respective dominators. Increase/decrease messages are propagated such that only the most useful information (stable, high-capacity links) is widely distributed. Thus increase waves propagate slowly and as an increasing function of the available bandwidth, while decrease waves travel faster and farther than corresponding increase waves.

When the source receives the route_reply, it uses core path and QoS state information to compute a QoS (bandwidth) admissible route. The dominator of the source computes the shortest-widest admissible path to the furthest possible intermediate node along the core path, using local state and information about remote stable high-bandwidth links where possible. The process is repeated until a QoS admissible path to the destination has been found. This complete source route is returned to the source via a route_reply and used for data traffic.

In the case of link failure, CEDAR attempts to recompute an admissible path at the point of failure, using the existing core path as a guideline. This is treated as a short term measure, to reduce the impact on packets already in-flight: the source also reinitiates the route discovery process.

5.2.2 CBRP

Cluster Based Routing Protocol (CBRP) [13] is a partitioning protocol emphasizing support for uni-directional links. Clusters are defined by bi-directional links, but inter-cluster connectivity may be obtained via a pair of uni-directional links.

Each node maintains two-hop topology information to define clusters. Each cluster includes an elected cluster-head, with which each member node has a bi-directional link. Clusters may be overlapping or disjoint; however, cluster-heads may not be adjacent. (Note that, as with CEDAR core nodes, a "good" set of cluster-heads will approximate a MDS of the graph defined by bi-directional links.)

In addition to exchanging neighbor information for cluster formation, nodes must find and inform their cluster-head(s) of the status of "gateway" nodes, cluster members which can be reached from a node belonging to another cluster. Thus, each cluster-head has knowledge of all clusters with which it has bidirectional connectivity, possibly via a pair of unrelated unidirectional links. The latter are discovered by flooding adjacent cluster heads with a request for an appropriate link.

When a source has no route to a destination, it forwards a route_request to its cluster-head. The cluster infrastructure is used to reduce the cost of disseminating the request. When a cluster-head receives a request, it appends its ID, as well as a list of (non-redundant) adjacent clusters, to the requests and broadcasts it. Each neighboring node that is a gateway to one of these adjacent clusters unicasts the request to the appropriate cluster-head.

When the request reaches the destination, it contains a loose source route specifying a sequence of clusters. When the route_reply is sent from the destination back to the source, each intermediate cluster-head writes a complete source route into the reply, optimizing that portion of the route based on its knowledge of cluster topology. Therefore, routes need not pass through cluster-heads. When the complete source route is received at the source, it is used for data traffic.

As with DSR, intermediate nodes may generate new routes to take advantage of improved routes or salvage failed routes. Unlike DSR, only cluster-level (2-hop neighborhood) information may be used for this purpose: nodes do not attempt to cache network-scale topology information.

6 Conclusion

A large number and wide variety of routing protocols have been proposed for use in mobile ad hoc networks. The framework developed above provides a structured approach for describing protocols. This approach helps show recurrent patterns and common issues. These include the route discovery process and route selection criteria associated with reactive protocols, the correctness of cached information in topology-oriented protocols and efficient maintenance various kinds of topological covers used in partitioning protocols.

It is difficult to name a "best" protocol from among the ones described here. Some are in active development, while others are very much research. In addition, intellectual property restrictions can impede independent analysis.

Most of these protocols have been investigated only in simulation. Few have been implemented and fewer still have seen realistic use. Direct performance comparisons have been limited and there is little agreement on what constitutes a typical network scenario or application load. There are also open issues in wireless MAC protocols for the ad hoc environment.

With these caveats in mind, two simulation-based performance characterizations [3, 14] indicate that AODV and DSR perform better than DSDV and TORA ([3] only) in relatively challenging (dense, high mobility, high traffic) scenarios. Both studies are based on the mobility-enhanced *ns-2* simulator developed by the Monarch Project at CMU [6].

7 Acknowledgements

Some material is derived from an earlier draft report "An Overview of Ad-hoc Routing Algorithms" by Torsten Köhler (SICS). Thomas Löfgren (SICS) provided a description of the CEDAR protocol.

References

[1] Lokesh Bajaj, Mineo Takai, Rajat Ahuja, Ken Tang, Rajive Bagrodia, and Mario Gerla. GloMoSim: A Scalable Network Simulation Environment. Technical Report, UCLA Computer Science Department - 990027.

[2] Josh Broch, David B. Johnson and David A. Maltz. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks. Internet Draft, draft-ietf-manet-dsr-*.txt. Work-in-progress.

[3] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu and Jorjeta Jetcheva. "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols". Proceedings of MobiCom'98, Dallas, TX, October 1998.

[4] Tsu-wei Chen and Mario Gerla. "Global State Routing: A New Routing Scheme for Ad-hoc Wireless Networks". Proceedings of IEEE Int'l Conference on Communications(ICC'98). Atlanta, GA, June, 1998.

[5] C.-C. Chiang, H.K. Wu, W. Liu and M. Gerla. "Routing in Clustered, Multihop, Mobile Wireless Networks with Fading Channel". Proceedings of IEEE SICON'97. Singapore, April, 1997.

[6] The CMU Monarch Project. The CMU Monarch Project's Wireless and Mobility Enhancements to *ns*. http://www.monarch.cs.cmu.edu. Work-in-progress.

[7] S. Corson and J. Macker. Mobile Ad hoc Networks (MANET): Routing Protocol Performance Issues and Evaluation Considerations. draft-ietf-manet-issues-*.txt. Work-in-progress.

[8] Zygmunt J. Haas. "A New Routing Protocol for the Reconfigurable Wireless Networks." Proceedings of ICUPC'97. San Diego, CA, October, 1997.

[9] Z.J. Haas and M.R. Pearlman. "The Performance of Query Control Schemes for the Zone Routing Protocol." Proceedings of ACM SIGCOMM'98. Vancouver, BC, August 1998.

[10] Zygmunt J. Haas and Marc R. Pearlman, The Zone Routing Protocol for Ad Hoc Networks. Internet Draft, draft-ietf-manet-zone-zrp-*.txt. Work-in-progress.

[11] Gavin Holland and Nitin Vaidya. "Analysis of TCP Performance over Mobile Ad Hoc Networks." Proceedings of MobiCom'99. Seattle, WA, August, 1999.

[12] Philippe Jacquet, Paul Muhlenthaler and Amir Qayyum. Optimized Link State Routing Protocol. Internet Draft, draft-ietf-manet-olsr-*.txt. Work-in-progress.

[13] Mingliang Jiang, Jinyan Li, Yong Chiang Tay. Cluster Based Routing Protocol Functional Specification. Internet Draft, draft-ietf-manet-cbrp-spec-*.txt. Work-in-progress.

[14] Per Johansson, Tony Larsson, Nicklas Hedman, Bartosz Mielczarek, and Mikael Degermark. "Scenario-based Performance Analysis of Routing Protocols for Mobile Ad hoc Networks." Proceedings of MobiCom'99. Seattle WA, August, 1999.

[15] David B. Johnson and David A. Maltz. “Dynamic Source Routing in Ad Hoc Wireless Networks”. In Mobile Computing, edited by Tomasz Imielinski and Hank Korth, Chapter 5, pages 153-181, Kluwer Academic Publishers, 1996.

[16] David A. Maltz, Josh Broch, Jorjeta Jetcheva, David B. Johnson. “The Effects of On-Demand Behavior in Routing Protocols for Multi-Hop Wireless Ad Hoc Networks.” IEEE Journal on Selected Areas in Communications Special Issue on Mobile and Wireless Networks. August 1999.

[17] Mobile Ad hoc Networks (MANET) Charter. Work-in-progress. http://www.ietf.org/html.charters/manet-charter.html, 1998.

[18] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen and Jang-Ping Sheu. “The Broadcast Storm Problem in a Mobile Ad Hoc Network.” Proceedings of MobiCom’99, Seattle WA, August, 1999.

[19] Vincent D. Park and M. Scott Corson. “ A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks”. Proceedings of IEEE INFOCOM ’97, Kobe, Japan (April 1997)

[20] Charles E. Perkins and Pravin Bhagwat. “Highly Dynamic Destination-Sequenced Distance-Vector (DSDV) Routing for Mobile Computers”. Proceedings of ACM SIGCOMM’94. London, UK, August 1994.

[21] C. E. Perkins and E. M. Royer. “Ad-hoc On-Demand Distance Vector Routing.” Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, February 1999, pp. 90-100.

[22] Charles E. Perkins, Elizabeth M. Royer and Samir Das. Ad Hoc On Demand Distance Vector Routing. Internet Draft, draft-ietf-manet-aodv-*.txt. Work-in-progress.

[23] Elizabeth Royer and C-K. Toh. “A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks”. *IEEE Personal Communications*, April 1999.

[24] P. Sinha, R. Sivakumar, and V. Bharghavan, “CEDAR: a Core-Extraction Distributed Ad hoc Routing algorithm.” IEEE Infocom ’99, New York, NY. March 1999.

[25] C.-K. Toh, “Associativity Based Routing For Ad Hoc Mobile Networks”, Wireless Personal Communications Journal, Special Issue on Mobile Networking and Computing Systems, March 1997.