# A Simulation Study of Table-Driven and On-Demand Routing Protocols for Mobile Ad Hoc Networks

**Sung-Ju Lee and Mario Gerla, University of California**
**Chai-Keong Toh, Georgia Institute of Technology**

## Abstract

Bandwidth and power constraints are the main concerns in current wireless networks because multihop ad hoc mobile wireless networks rely on each node in the network to act as a router and packet forwarder. This dependency places bandwidth, power, and computation demands on mobile hosts which must be taken into account when choosing the best routing protocol. In recent years, protocols that build routes based on demand have been proposed. The major goal of on-demand routing protocols is to minimize control traffic overhead. In this article we perform a simulation and performance study on some routing protocols for ad hoc networks. Distributed Bellman-Ford, a traditional table-driven routing algorithm, is simulated to evaluate its performance in multihop wireless networks. In addition, two on-demand routing protocols (Dynamic Source Routing and Associativity-Based Routing) with distinctive route selection algorithms are simulated in a common environment to quantitatively measure and contrast their performance. The final selection of an appropriate protocol will depend on a variety of factors, which are discussed in this article.

An *ad hoc* or *multihop mobile wireless network* is an infrastructureless network with no fixed routers, hosts, or wireless base stations. Current cellular networks use a wireless last hop architecture but rely on a wired infrastructure to interconnect different cells. In ad hoc wireless networks, a remote mobile node interconnection is achieved via a peer-level multihopping technique. This implies that the interconnection topology can change dynamically, giving rise to many challenging research issues. In this environment, ad hoc routing is critical and has to be supported before any applications can be deployed for ad hoc mobile networks.

Routing protocols used in conventional wired networks (e.g., Bellman-Ford [1, 2] and link state [3]) are not well suited to the mobile environment due to the considerable overhead produced by periodic route update messages and their slow convergence to topological changes.

Numerous ad hoc routing protocols have been proposed to the Internet Engineering Task Force (IETF) Mobile Ad Hoc Networks (MANET) Working Group [4]. These protocols share the characteristic that routes are established based on demand by the source (hence the term *on-demand routing*).

Some of these protocols have been evaluated via simulation, but in most cases they are not simulated in a common environment. This makes quantitative performance comparisons among these protocols difficult. In this article we use a common simulation platform to investigate the performance of three routing schemes. We compare Associativity-Based Routing (ABR) [5, 6], which considers multiple route selection metrics, with Distributed Bellman-Ford (DBF), which is not an on-demand scheme, and Dynamic Source Routing (DSR) [7], which still uses the shortest path as the routing metric. We have chosen these three protocols for the following reasons:

- To evaluate the performance of a conventional table-driven routing scheme (DBF) in multihop wireless networks
- To study the performance of different routing metrics in dynamic ad hoc networks

The remainder of this article is organized as follows. The next section presents a discussion on DBF, DSR, and ABR protocols, highlighting how each of them supports route setup and mobility. We then evaluate and compare the performance of DBF, DSR, and ABR protocol via simulation. The conclusion follows in the last section.


0890-8044/99/$10.00 © 1999 IEEE

## Existing Ad Hoc Routing Protocols

Recently, several proposals to support ad hoc mobile communications have evolved, including DSDV [8], LMR [9], WRP [10], DSR [7], ABR [6], AODV [11], SSA [12], TORA [13], ZRP [14], and CEDAR [15]. Surveys of routing techniques for mobile wireless networks and ad hoc networks have been reported in [16, 17]. In this article we focus our attention and discussion on DBF, DSR, and ABR.

### Distributed Bellman-Ford

The DBF algorithm was developed originally to support routing in the ARPANET. A version of it is known as Routing Internet Protocol (RIP) [18] and is still being used today to support routing in some Internet domains. It is a table-driven routing protocol,that is, each router constantly maintains an up-to-date routing table with information on how to reach all possible destinations in the network. For each entry the next router to reach the destination and a metric to the destination are recorded. The metric can be hop distance, total delay, or cost of sending the message. Each node in the network begins by informing its neighbors about its distance to all other nodes.

The receiving nodes extract this information and modify their routing table if any route measure has changed. For instance, a different route may have been chosen as the best route or the metric to the destination may have been altered. The node uses the following formula to calculate the best route:

$$D(i, j) = \min_{k} [d(i, k) + D(k, j)]$$

where $D(i, j)$ is the metric on the "shortest" path from node $i$ to node $j$, $d(i, k)$ is the cost of traversing directly from node $i$ to node $k$, and $k$ is one of the neighbors of node $i$. After recomputing the metrics, nodes pass their own distance information to their neighbor nodes again. After a while, all nodes/routers in the network have a consistent routing table to all other nodes.

This protocol does not scale well to large networks due to a number of reasons. One is the so-called count-to-infinity problem. In unfavorable circumstances, it takes up to $N$ iterations to detect the fact that a node is disconnected, where $N$ is the number of nodes in the network [19]. Another problem is the increase of route update overhead with mobility. RIP uses time-triggered (periodic, about a 30-s interval) and event-triggered (link changes or router failures) routing updates. Mobility can be expressed as rate of link changes and/or router failures. In a mobile network environment, event-triggered routing updates tend to outnumber time-triggered ones, leading to excessive overhead and inefficient usage of the limited wireless bandwidth.

### Dynamic Source Routing

*Protocol Characteristics* — DSR was developed at Carnegie Mellon University [7]. It is a direct descendant of the source routing scheme used in bridged LANs [19]. It uses source routing instead of hop-by-hop packet routing. Each data packet carries the list of routers in the path. The main benefit of source routing is that intermediate nodes need not keep route information because the path is explicitly specified in the data packet. DSR does not require any kind of periodic message to be sent, supports unidirectional and asymmetric links, and sets up routes based on demand by the source. DSR consists of two phases, route discovery and route maintenance, which are explained in the following sections.

*Route Discovery* — When a source has a data packet to send but does not have any routing information to the destination, the source initiates a route discovery. To establish a route, the source floods a `route request` message with a unique request ID. When this request message reaches the destination or a node that has route information to the destination, it sends a `route reply` message containing path information back to the source. The *route cache* maintained at each node records routes the node has learned and overheard over time to reduce overhead generated by a route discovery phase.

When a node receives a `route request` packet, this message is forwarded only if all of the following conditions are met:

- The node is not the target (destination) of the `route request` packet.
- The node is not listed in source route.
- The packet is not a duplicate.
- No route information to the target node is available in its route cache.

If all are satisfied, it appends its identification to the source route and broadcasts the packet to its neighbors. If the second or third condition is not met, it simply discards the packet. If a node is the destination of the packet or has route information to the destination, it builds and sends a route reply to the source, as described above.

*Route Maintenance* — The main innovation of DSR with respect to bridged LAN routing is in route monitoring and maintenance in the presence of mobility. DSR monitors the validity of existing routes based on the acknowledgments of data packets transmitted to neighboring nodes. This monitoring is achieved by passively listening for the transmission of the neighbor to the next hop or by setting a bit in a packet to request an explicit acknowledgment. When a node fails to receive an acknowledgment, a `route error` packet is sent to the original sender to invoke a new route discovery phase. Nodes that receive a `route error` message delete any route entry (from their route cache) which uses the broken link. Note that a `route error` message is propagated only when a node has a problem sending packets through that link. Although this selective propagation reduces control overhead (if no packets traverse a link), it yields a long delay when a packet needs to go through a new link.

*Information Stored in Each Node*

- *Route cache:* Each node stores routing information it has learned and overheard in its route cache. Routing information can be obtained while processing `route reply` messages and the source route list of a data packet header. More than one route for each destination can be stored in the cache. When a `route error` message is received or overheard, routes that use the broken link specified in the `route error` are removed from the route cache.
- *Route request table:* Nodes producing a `route request` packet store information in the route request table. Recorded information includes the destination node of a `route request`, the time when the node last sent a `route request` to the destination, and the time the node has to wait until it can send the next `route request` to the destination. The purpose of maintaining this table is to restrict frequent `route request` transmissions to the same destination.

*Optimizations* — To improve performance and reduce overhead, a few optimizations can be achieved in DSR. Some of the optimizations are:

- *Nonpropagating route requests:* When originating a `route request`, senders set the time to limit (TTL) to zero hop, thus allowing only the neighbors to receive the packet. If a neighbor is the destination or has route information to the destination in its cache, it sends a reply to the originator. If no reply is received within a timeout period, an ordinary (propagating) `route request` is flooded by the sender.

- *Piggybacking on route discoveries:* To eliminate the route acquisition latency, data can be piggybacked on `route request` packets. If, however, a route is replied by an intermediate node which has route information to the destination in its cache, that node needs to construct a data packet and forward it to the destination node in order not to lose any data.
- *Gratuitous route replies:* When receiving a packet not addressed to itself, a node refers the listed source route that has not been traversed yet. If the unprocessed part contains the identification of the node, it realizes that a shorter route can be achieved by not visiting the preceding hops in the source route. This node sends a gratuitous `route reply` to the sender to inform a shorter route.
- *Gratuitous route errors:* When a source of the broken route receives a `route error`, it piggybacks the received `route error` on the next `route request` packet for route rediscovery. This piggybacking prevents nodes from replying with stale routes.
- *Salvaging:* If an intermediate node of a route detects that the next-hop node cannot be reached, it searches its route cache for an alternate route. If such a route is found, it substitutes this available route for the stale route in the data header and forwards it. The intermediate node is still required to send a `route error` back to the sender.
- *Snooping:* When processing data, a node examines the unvisited nodes in the source route and inserts those routes into its route cache. This snooping enables nodes to have multiple alternate routes for each destination.

### Associativity-Based Routing

*Protocol Characteristics* — Developed at Cambridge University, ABR [5, 6] is a protocol designed for an ad hoc mobile network environment. Routes are established based on demand. The uniqueness of this scheme is the route selection criteria. By exploiting the spatial and temporal relationship of mobile hosts, ABR introduces the following new routing metrics:

- Longevity of a route based on associativity
- Route relaying load of intermediate nodes supporting existing routes
- Link capacities of the selected route

By "associativity" or "affinity" we mean the spatial, temporal, and connection relationship of a mobile host with its neighbors. Associativity is measured by recording the number of control beacons received by a node from its neighbors. For example, assume that each mobile host has a transmission/reception range of 10 m in diameter and there are two mobile hosts, A and B. Initially, A and B are not in radio connectivity with each other, but each sends a control beacon to signify its presence once every 2 s. If A is migrating at 1 m/s, and it starts to enter B's radio range and move through it diagonally, both A and B record at most five beacons each. Hence, this is the associativity threshold. Namely, if only five or less beacons are recorded, one can assume that the other mobile host is migrating past it, and this situation is viewed as *associatively unstable*. Otherwise, if the mobile host is moving but is constantly within the radio coverage of its neighbors, more than five beacons will be recorded, and hence the node is regarded as *associatively stable*. Note that associativity has an interlocking characteristic since a node's associativity stability with its neighbors depends on the mobility profile of the neighbors. By selecting nodes with high associativity counts/ticks, the route is expected to have a long-lived characteristic. This stability could result in a route with a nonshortest path, but the route can be maintained with less chance of having to perform route recovery. The detailed algorithm for route selection in ABR can be found in [6].

The following sections shall elaborate further on route discovery and route reconstruction.

*The Route Discovery Phase* — The route discovery process consists of broadcast query (`BQ`) and `BQ-REPLY` cycles. When a source demands a route, it floods a `BQ` message. Any intermediate node (IN) that receives the `BQ` packet checks if the message has already been processed by looking up the seen table, which will be explained later. If the `BQ` packet has not been seen before, it appends the following to the `BQ` packet:

- Its identifier
- Associativity ticks with its neighbors
- Route relaying load
- Link propagation delay
- Hop count information

The IN then broadcasts the packet to its neighbors.

When the destination node receives `BQ` packets, it knows all the possible routes and their qualities. The destination node then selects the best route based on longevity and other qualities (route load, minimum hop, etc.) and sends a `BQ-REPLY` control packet (which contains a list of INs' addresses/IDs and a summary of selected route QoS) back to the source node via the selected route. When INs of the selected route receive the `BQ-REPLY` packet, they update their routing tables with this new route.

*The Route Reconstruction Phase* — In circumstances where nodes' mobility invalidate the selected route, the route reconstruction (RRC) process is invoked to discover alternate partial routes quickly. The migration of neighbor nodes can be detected when no beacon message is received within the timeout interval. When an IN of an existing route moves away from radio range of its immediate upstream or downstream, the route is invalidated. The immediate downstream node sends a route notification (`RN`) packet toward the destination to inform the invalidity of that route. Nodes that subsequently receive such a message delete their route entry. The immediate upstream of the moved node, however, performs a localized query (`LQ`) to discover a new partial route. Unlike `BQ`, an `LQ` process performs a limited scope broadcast (i.e., the flood radius is controlled by a hop count field). However, similar to `BQ`, information about route metrics is appended into `LQ` packets as they make their way to the destination. After the destination node receives several `LQ` messages, it selects the best partial route (again based on associativity stability) and sends back an `LQ-REPLY` message to the node that invoked the `LQ` process. As a result, all nodes in this partial path have their routing entry updated, allowing subsequent data packets to be forwarded via this new partial path.

When the node that sent the `LQ` message does not receive the `LQ-REPLY` message within the timeout period (i.e., when partial paths cannot be located), it sends an `RN` packet to the immediate upstream node (i.e., backtracks). When a node receives an `RN` packet from an immediate downstream node, it recognizes the backtrack and invokes an `LQ` process again. The fundamental strategy here is to localize the route discovery process to a bounded region so that other parts of the route are not affected. This localization also helps avoid unnecessary use of full broadcast. For a displacement of a node along the route, `LQ` processes can be performed on at most half the route hop distance. Thereafter, if no partial path can be located, an `RN` message is sent back to the source node of the route to invoke a `BQ` process. This quick abort mechanism is to shorten route recovery time (avoiding the possibility of backtracking all the way to the source) by limiting the number of `LQ` processes.

| Protocols | DBF | DSR | ABR |
|---|---|---|---|
| Route establishment | Proactive | On-demand | On-demand |
| Routing metric | Shortest path | Shortest path | Associativity, load, delay, etc. |
| Periodic messages | Route tables | None | Beacons |
| Loop-free | No | Yes | Yes |

Table 1. *A summary of DBF, DSR, and ABR.*

*Data Transmission* — To utilize the channel efficiently, ABR uses a simple and short packet header. Each data packet header contains only the neighboring node information rather than all the nodes in the route.

Similar to DSR, flow control is achieved by monitoring passive acknowledgments. When node A receives a packet and forwards it to the next-hop node B, A hears B's transmission when B relays the packet to another node. This is known as *passive acknowledgment* and is a technique used in packet radio [20]. Active acknowledgment is used by the destination node (since it has no more neighbors to which to relay the packet) where an explicit message is sent to the upstream node. If a node does not receive a passive acknowledgment within the timeout period after forwarding a packet, it retransmits the data packet for an appropriate number of times. If an acknowledgment is not received after a few attempts, a mobile host is considered to have moved out of radio range or powered down, and an RRC phase is therefore invoked.

*Information Stored in Each Node*

- *Routing table:* If a node is part of an active route in the network, it stores the route information in its routing table. Not only are the source and destination IDs of the route recorded, but also the incoming and outgoing node IDs are kept so that incoming packets can be forwarded accordingly. Information on the hop count to the destination and the total number of active routes the node is currently supporting are maintained in the routing table as well. Unlike distance-vector-based routing protocols, the ABR routing table contains only routing information for routes that are actually required by the source, not every possible destination in the network.
- *Neighbor table:* Each node maintains a neighbor table that records its associativity relationship with surrounding neighbors. An associativity counter is incremented when a beacon message transmitted by a neighboring node is received. If no beacon message is received from a neighboring node within the timeout interval, the corresponding associativity counter field is reset to zero (to reflect the associativity instability).
- *Seen table:* A seen table is used to prevent a mobile host from processing and forwarding the same BQ or LQ message multiple times. When receiving a BQ or LQ message, a node looks up its seen table and checks if the received message has been processed before. If an entry matches the type (BQ or LQ), source ID, destination ID, and sequence number, the received packet is discarded. Note that entries in the seen table need not be maintained permanently. Schemes such as Least Recently Used (LRU) [21] can be employed to expire and remove old entries and prevent the size of a seen table to be extensive.

Key characteristics and properties of DBF, DSR, and ABR are summarized in Table 1.

## Performance Evaluation

### Simulation Model

The simulator for evaluating three routing protocols is implemented within the Global Mobile Simulation (GloMoSim) library [22]. The GloMoSim library is a scalable simulation environment for wireless network systems using the parallel discrete-event simulation capability provided by PARSEC [23]. The simulation models the network of 30 mobile hosts migrating within a 20 m x 20 m space with a transmission radius of 5 m. Every node in the network moves in a random fashion, with a static time of 5 s before migrating again. The channel capacity is 2Mb/s. IEEE 802.11 Distributed Coordination Function (DCF) [24] is used as the medium access control protocol. A free-space propagation model [25] with a threshold cutoff has been used in our experiments. In the free-space model, the power of a signal attenuates as $1/d^2$ where $d$ is the distance between radios. In addition to the free-space channel model, we have also implemented the Simulation of Indoor Radio Channel Impulse-Response Models (SIRCIM) [26], which considers fading, barriers, foliages, multipath interference, and so on. The SIRCIM is more accurate than the free-space model, but we have decided against using SIRCIM in our study because:

- The complexity of SIRCIM increases simulation time by two orders of magnitude.
- The accuracy of the channel model does not affect the relative ranking of the routing protocols evaluated in this study.
- SIRCIM must be "tuned" to the characteristics of the physical environment (e.g., indoor, outdoor), thus requiring a much more specific scenario than we are assuming in our experiments.

In the radio model, capture effects are taken into account.

If the capture ratio (the minimum ratio of an arriving packet's signal strength relative to those of other colliding packets) [25] is greater than the predefined threshold value, the arriving packet is received while other interfering packets are dropped. A traffic generator was developed to simulate constant bit rate sources. Source nodes and destination nodes were chosen randomly with uniform probabilities. A packet is dropped when no acknowledgment is received after retransmitting it a certain number of times. Simulation runs of 200,000,000,000 simulation ticks (which is 200 s of simulation time) were performed multiple times.

### Simulation Results

DBF, a traditional table-driven routing scheme used in wired networks, is compared with on-demand ad hoc routing schemes (ABR and DSR) in a common multihop mobile wireless network simulation platform.

Parameters of interest are:

- Control overhead
- Data throughput
- End-to-end packet propagation delay

Specifications stated in [6, 18, 27] are employed to implement DBF, DSR, and ABR, respectively. The results obtained are discussed below.

*Control Message Overhead* — Figure 1 shows the control overhead incurred by DBF, DSR, and ABR. Both ABR and DSR on-demand routing schemes have considerably less overhead (as high as 76.56 percent) than DBF. Sending route updates periodically and triggering updates when the topology changes in order to maintain an up-to-date routing table result in excessive control message overhead, which is unacceptable in a wireless environment with limited bandwidth. We can see that DSR has less overhead than ABR when the network is static. If nodes are not mobile, there is no route

breakage and control messages for route reconstruction are not required. ABR sends beacon messages to maintain the list of neighbors, thus resulting in more overhead when there is no mobility. One might expect ABR to have considerably more control overhead when nodes are stable. However, the result shows only a small difference since the size of beacon messages is very small.

We can observe from the result that increasing the mobility speed makes ABR more efficient than DSR. This efficiency is attributed to ABR's local route recovery feature. In DSR, if a node in the path becomes unreachable, a control message specifying a route error is propagated all the way back to the source to invoke a new route discovery. In contrast, in ABR the immediate upstream of a migrated node starts the LQ process to find a new partial route without intervention from the source, hence minimizing the transmission of control messages.

*Data Throughput* — Figure 2 shows the throughput comparison of DBF, DSR, and ABR. DBF's poor performance can be attributed to excessive channel usage by route update control messages. Also, as mobility speed increases, more event-triggered updates are generated. However, this is not present in on-demand routing protocols. The graph also reveals that ABR has higher throughput than DSR, resulting from the use of a different route selection process. In DSR a route is chosen based on the shortest delay at the instance of route establishment. Although this path may be the best route at that instant, it may be a route that lacks routing stability or has unacceptably high load. In contrast, ABR distinctively selects a route where nodes in the path are associatively stable (spatial, temporal, and connection-wise) and have light load. These route selection criteria enhance the longevity of the selected route, avoid bottleneck and congestion at INs, and eventually improve throughput.

*End-to-End Delay* — Figure 3 shows the end-to-end delay of data packets. DBF has larger delays than on-demand schemes due to high control overhead and thus large queuing delays. For on-demand protocols, ABR has shorter delays than DSR, and this difference becomes more obvious as mobility speed increases. The better performance of ABR can be traced to the following reasons. First, balancing the route load shortens the delay since the chance of congestion is reduced. Second, adjusting to network mobility via receiving beacon messages from neighbors yields faster convergence. In DSR a neighbor displacement is noticed only after a packet is sent explicitly to that node. The network reacts if an acknowledgment is not

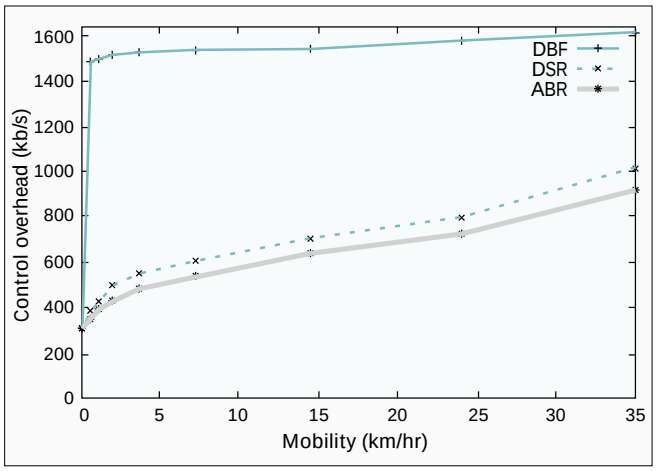


Figure 1. *Control message overhead for different mobility speeds.*

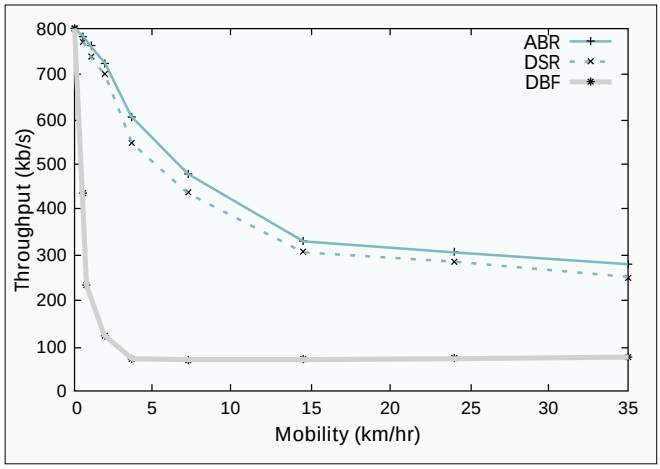


Figure 2. *Data throughput for different mobility speeds.*

received. Consequently, this increases packet delay since the packet must wait until a new route is established.

*Other Considerations* — In the previous sections we have compared routing algorithms based on the performance criteria typically measured in a simulation experiment, namely, throughput, delay, and control traffic overhead. There are other criteria, however, which must be taken into account when selecting the routing scheme for a specific application. Often, these criteria are not easily assessed via simulation. In this section we examine three such criteria: table storage overhead, probability of detection/interception, and power consumption.

Table Storage Overhead — For each route discovered by DSR, a route cache table is kept at the source as well as at each node along the route. Let $R$ be the average number of active routes a node supports and $N$ the total number of nodes in the network. Assuming a grid-like radio connection topology (consistent with optimal radio power range), the average path length is $\sqrt{N}$. Thus, the total number of route cache entries for each node is, on average, $R\sqrt{N}$. The source node of route request packets maintains a node information cache.[1] Having four fields for each destination, the average number of node information cache entries per node is $4R$. Hence, the total storage overhead for DSR is $R\sqrt{N}+4R$. Note that if there is no active traffic (i.e., $R$ is zero), the storage overhead is zero.

ABR requires a routing table, a neighbor table, and a seen table[2] by each node in the network. The average number of routing table entries is $5R + 1$ per each node. Moreover, $4R\sqrt{N}+2RN$ entries are needed for a seen table in the worst case where each route becomes invalid and every LQ process fails. Note that this amount of storage is needed only if entries for every possible BQs and LQs are stored forever. In practice, probability of a node receiving a duplicate packet that has traversed $h$ hops decreases rapidly with $h$ since duplications are automatically filtered by neighbors after the first hop. Thus, an entry needs to be kept in the seen table only for a relatively short time and can be removed after a timeout. Therefore, maintaining a fixed number of entries for the seen table is sufficient to detect duplicates (in our comparison which we present in Fig. 4, we use a conservative value of ten entries per active route). In addition to the storage overhead

[1] *For details on the node information cache, see [27].*

[2] *See [6] for the structure of a routing table, a neighbor table, and a seen table of ABR.*

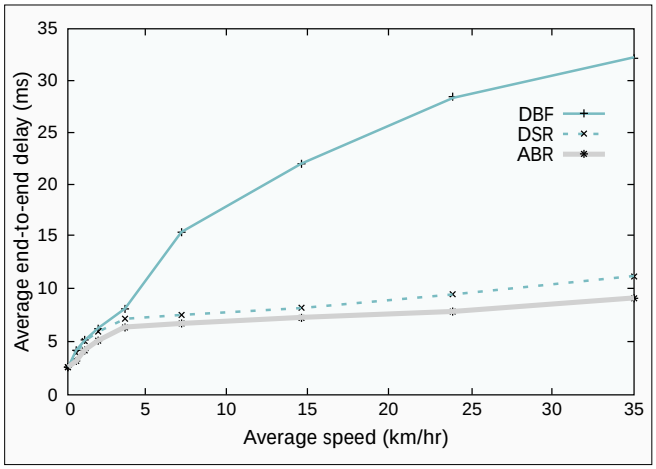


Figure 3. *Average end-to-end delay for different mobility speeds.*

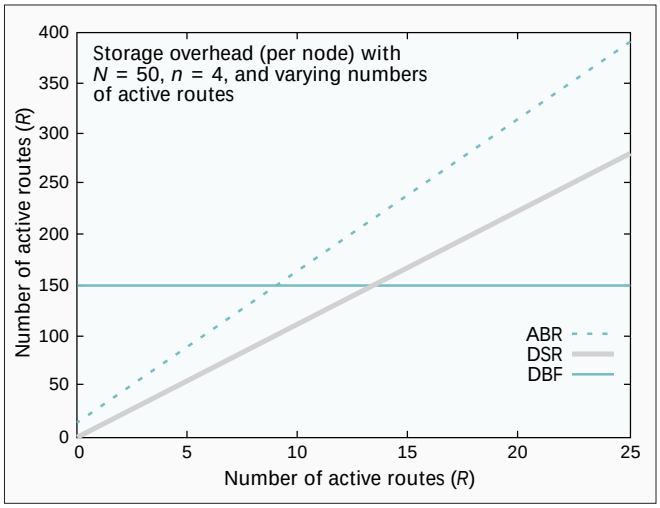


Figure 4. *Storage overhead for different numbers of active routes.*

of the routing table and seen table, neighbor table overhead of 3*n* is required for every node, where *n* is the average number of neighbors. Note that this is a constant overhead which is incurred even if there is no traffic in the network. In DSR, storage overhead is zero if there is zero traffic.

In DBF the table overhead of each network node is 3*N*, independent of traffic.[3] This overhead is higher than on-demand routing (ABR or DSR) in light traffic but lower in heavy traffic. In Fig. 4 we show the storage overhead required by each node for varying numbers of active routes in a network with 50 hosts. We can see that the storage overhead of ABR is higher than DSR, especially if the number of active routes increases. We can also see that DBF requires more storage overhead than on-demand protocols in light traffic.

Low Detection / Interception Probability — In some battlefield applications, if no packet needs to be transmitted, nodes should preferably remain silent (sleep mode) to reduce detection/interception probability. ABR sends beacon messages periodically, and this beacon may be received by an unintended receiver (e.g., an enemy).[4] Similarly, DBF nodes continuously emit update packets, which can be detected or intercepted. DSR, on the other hand, does not transmit anything if there is no user data to send. Thus, DSR has a better LDP/LIP property.

Low Power Operation — In situations where there is no data traffic, ABR demands more power in order to process beacon messages, and so does DBF to transmit/process updates. Thus, DSR is more attractive when power resource conservation is of paramount concern. However, this deficiency in ABR can be offset by current power conservation techniques in devices, protocols, and operating systems.

Problem with Shortest Path on Power Consumption — In routing protocols that use shortest hop or delay as a route selection metric, some nodes need to support many routes (i.e., have high route relaying load). These nodes continuously consume energy, which will eventually be exhausted, resulting in node failures. Route selection should also consider energy reserves as one factor [28]. ABR uses *route relaying load* as one of its metrics and prevents node failures of this kind. However, this is not the case for DSR.

[3] *DBF stores destination, distance, and next-hop node for each route, thus making it 3*N *for each node.*

[4] *Using advanced radio modulation techniques, beacons may appear as noise for other radio detection systems.*

## Conclusions

Many routing protocols for ad hoc mobile wireless networks have been proposed in recent years. In this article we have reviewed and studied key properties of three distinctive routing protocols. Performance evaluation of these protocols has been conducted via simulation in a common network environment. We have compared the performance of Associativity-Based Routing with Distributed Bellman-Ford and Dynamic Source Routing. Simulation results reveal that DBF incurs extensive bandwidth and computation overhead in the presence of mobility, yielding inferior performance to on-demand routing protocols (ABR and DSR) in ad hoc networks. We also report that ABR has better throughput, smaller delay, and lower control overhead than DSR. Chiefly, this is due to the use of an innovative associativity criterion, multiple route selection metrics, and local route recovery. On the negative side, ABR exhibits a slightly higher storage overhead than DSR. It is also more prone to detection and interception (by the enemy).

In summary, ABR is a strong candidate for the multihop mobile wireless environment along with DSR. The final selection of the on-demand routing scheme should take into account other considerations in addition to the measures provided by simulation.


### Acknowledgments

The authors thank Ken Tang, Russell Hayashida, and Lokesh Bajaj of UCLA for assisting in the simulation work.


### References


[1] R. E. Bellman, *Dynamic Programming*, Princeton: Princeton Univ. Press, 1957.
[2] L. R. Ford and D. R. Fulkerson, *Flows in Networks*, Princeton: Princeton Univ. Press, 1962.
[3] J. M. McQuillan, I. Richer, and E. C. Rosen, "The New Routing Algorithm for the ARPANET," *IEEE Trans. Commun.*, vol. COM-28, no. 5, May 1980, pp. 711–19.
[4] IETF MANET Working Group Charter, http://www.ietf.org/html.charters/manet-charter.html
[5] C.-K. Toh, "A Novel Distributed Routing Protocol to Support Ad Hoc Mobile Computing," *Proc. IEEE IPCCC '96*, Scottsdale, AZ, Mar. 1996, pp. 480–86.
[6] C.-K. Toh, "Associativity-Based Routing For Ad Hoc Mobile Networks," *Wireless Pers. Commun. J.*, Special Issue on Mobile Networking and Computing Systems, Kluwer Academic, vol. 4, no. 2, Mar. 1997, pp. 103–39.
[7] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," *Mobile Computing*, T. Imielinski and H. Korth, Eds., Ch. 5, Kluwer Academic, 1996, pp. 153–81.
[8] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance Vector Routing (DSDV) for Mobile Computers," *Proc. ACM SIGCOMM '94*, London, U.K., Sep. 1994, pp. 234–44.
[9] M. S. Corson and A. Ephremides, "A Distributed Routing Algorithm for Mobile Wireless Networks," *ACM/Baltzer Wireless Networks*, vol. 1, no. 1, Feb. 1995, pp. 61–81.

[10] S. Murthy and J. J. Garcia-Luna-Aceves, "An efficient Routing Protocol for Wireless Networks," *ACM/Baltzer J. Special Topics in Mobile Networks and Apps.*, vol. 1, no. 2, Oct. 1996, pp. 183–97.
[11] C. E. Perkins and E. M. Royer, "ad hoc On Demand Distance Vector Routing," *Proc. IEEE WMCSA '99*, New Orleans, LA, Feb. 1999, pp. 90–100.
[12] R. Dube *et al.*, "Signal Stability-Based Adaptive Routing (SSA) for Ad Hoc Mobile Networks," *IEEE Pers. Commun.*, vol. 4, no. 1, Feb. 1997, pp. 36–45.
[13] V. D. Park and M. S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," *Proc. IEEE INFOCOM '97*, Kobe, Japan, Apr. 1997, pp. 1405–13.
[14] Z. J. Haas, "A New Routing Protocol for the Reconfigurable Wireless Networks," *Proc. IEEE ICUPC '97*, San Diego, CA, Oct. 1997, pp. 562–66.
[15] P. Sinha, R. Sivakumar, and V. Bharghavan, "CEDAR: a Core-Extraction Distributed Ad hoc Routing Algorithm," *Proc. IEEE INFOCOM '99*, New York, NY, Mar. 1999, pp. 202–9.
[16] S. Ramanathan and M. Streenstrup, "A Survey of Routing Techniques for Mobile Communication Networks," *ACM/Baltzer J. Special Topics in Mobile Networks and Apps.*, vol. 1, no. 2, Oct. 1996, pp. 89–104.
[17] E. M. Royer and C.-K. Toh, "A Review of Current Routing Protocols for Ad Hoc Mobile Networks," *IEEE Pers. Commun.*, vol. 6, no. 2, Apr., 1999, pp. 46–55.
[18] G. Malkin, "RIP Version 2 - Carrying Additional Information," Internet Draft, draft-ietf-ripv2-protocol-v2-05.txt, Jun. 1998, work in progress.
[19] A. S. Tanenbaum, *Computer Networks*, 3rd ed., Upper Saddle River, NJ: Prentice Hall, Mar. 1996.
[20] J. Jubin and J. D. Tornow, "The DARPA Packet Radio Network Protocols," *Proc. IEEE*, vol. 75, no. 1, Jan. 1987, pp. 21–32.
[21] A. J. Smith, "Cache Memories," *ACM Comp. Surveys*, vol. 14, no. 3, Sept. 1982, pp. 473–530.
[22] UCLA Parallel Comp. Lab. and Wireless Adaptive Mobility Lab., "GloMoSim: A Scalable Simulation Environment for Wireless and Wired Network Systems," http://pcl.cs.ucla.edu/projects/domains/glomosim.html
[23] R. Bagrodia *et al.*, "PARSEC: A Parallel Simulation Environment for Complex Systems," *IEEE Comp.*, vol. 31, no. 10, Oct. 1998, pp.77–85.
[24] IEEE Computer Society LAN MAN Standards Committee, "Wireless LAN Medium Access Protocol (MAC) and Physical Layer (PHY) Specification," IEEE Std 802.11-1997, New York, NY, 1997.
[25] T. S. Rappaport, *Wireless Communications: Principles and Practice*, Upper Saddle River, NJ: Prentice Hall, Oct. 1995.
[26] T. S. Rappaport, S. Y. Seidel, and K. Takamizawa, "Statistical Channel Impulse Response Models for Factory and Open Plan Building Radio Communication System Design," *IEEE Trans. Commun.*, vol. COM-39, no. 5, May 1991, pp. 794–807.
[27] J. Broch, D. B. Johnson, and D. A. Maltz, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks," Internet Draft, draft-ietf-manet-dsr-00.txt, Mar. 1998, work in progress.
[28] S. Singh, M. Woo, and C. S. Raghavendra, "Power-Aware Routing in Mobile Ad Hoc Networks," *Proc. ACM/IEEE MOBICOM '98*, Dallas, TX, Oct. 1998, pp. 181–90.

## *Biographies*

SUNG-JU LEE [StM] (sjlee@cs.ucla.edu) received his B.S. degree in computer science and engineering from Hanyang University, Korea, in February 1996 and his M.S. degree in computer science from the University of California, Los Angeles in June 1998. He is currently a Ph.D. student in the Computer Science Department of the University of California, Los Angeles and is also a member of the Wireless Adaptive Mobility (WAM) Laboratory. His research interests include ad hoc networks, routing, multicasting, mobile computing, and wireless networks performance evaluation and modeling. He is a student member of ACM.

MARIO GERLA [M] (gerla@cs.ucla.edu) received a graduate degree in electrical engineering from Politecnico di Milano, Italy, in 1966 and M.S. and Ph.D. degrees in computer science from the University of California, Los Angeles, in 1970 and 1973, respectively. From 1973 to 1976 he was a manager at the Network Analysis Corporation, Glen Cove, New York, where he was involved in several computer network design projects for both government and industry, including performance analysis and topological updating of the ARPANET under a contract from DoD. From 1976 to 1977 he was with Tran Telecommunication, Los Angeles, California, where he participated in the development of an integrated packet and circuit network. Since 1977 he has been on the faculty of the Department of Computer Science, UCLA. His research interests include the design, performance evaluation, and control of distributed computer communication systems and networks. His current research projects cover the following areas: topology design and bandwidth allocation in ATM networks, design and implementation of optical interconnects for supercomputer applications, design and performance evaluation of air/ground wireless communications for the aeronautical telecommunications network, and network protocols design and implementation for a mobile, integrated services wireless radio network.

CHAI-KEONG TOH [M] (cktoh@ee.gatech.edu) received his Diploma in electronics and communication engineering with a Certificate of Merit award from the Singapore Polytechnic in 1986, 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 is a member of IEE, USENIX, ACM, Sigma Xi Honor Society, New York Academy of Science, and American Association for the Advancement of Science (AAAS), and is 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, directing the Mobile Multimedia and High Speed Networking Laboratory.