# Anycast Routing for Mobile Services

**Vincent D. Park** and **Joseph P. Macker**
Naval Research Laboratory
Washington, DC 20375

## Abstract

*This paper considers the problem of locating and forwarding network traffic to any one of a set of distributed servers or service points—primarily in the context of mobile ad hoc networks. The advantages of providing such a capability through the use of anycast routing techniques at the network layer are discussed. We then illustrate how several different classes of unicast routing protocols can be extended to provide efficient construction and maintenance of anycast routes. Extensions to link-state, distance-vector and link-reversal unicast routing protocols are all conceptually realized through the representation of an anycast service as a "virtual node" in a graph based on the network topology. The initial results of a simulation study, which demonstrate how anycast routing techniques can provide a one-to-any communication capability with greater efficiency than traditional unicast based techniques, are presented and discussed. The simulation results further indicate that anycast routing can ease the configuration and management required to achieve a given level of robustness and can reduce connection setup latency and message packet delay.*

## Introduction

A communication paradigm, known as *anycasting*, has recently been introduced within the networking community—primarily for the purpose of locating a particular distributed service [1]. Anycasting essentially provides a means to locate and communicate with *any one* of a set of distributed servers or service access points within a network. This is analogous to providing an individual that needs to make a phone call with directions to a public payphone. While there are potentially many points of service, the end user only needs to find one. In a networking context, anycasting facilitates more robust distributed system design and eases network configuration and management.

Most research to date has focused on the development of anycast techniques at the application layer [2, 3]. However, we believe greater communication efficiency and robustness can be achieved through the use of anycast routing techniques at the network layer. While these gains are realizable in quasi-static hardwired networks, they are of more critical importance in mobile wireless networks—which have more dynamics (e.g., rapid and unpredictably changing interconnectivity between routers) and are more bandwidth constrained than traditional hardwired networks. We will focus our discussion in the area of mobile networks but we feel the results are also valuable in more static network environments.

While renewed research interest and progress is being made in the area of unicast (one-to-one) and multicast (one-to-many or many-to-many) routing for mobile ad hoc networks [4, 5], locating and managing mobile services for end users in such networks remains a largely unexplored topic. Anycast (one-to-any) routing helps support and manage this required functionality. Within static networks, critical networking services are often centralized or distributed with preconfigured lists creating a fundamental adaptation, robustness, and location problem. Without robust mobile support for such networking services, end systems are severely handicapped in functionality and performance regardless of the available network connectivity or bandwidth. The prevalence of performance degradation, global scale, and denial of service in today's static network infrastructures has spawned a flurry of recent developments in distributed network databases and services. Even when such distributed services are available the problem of service location, transaction, and data collection is exacerbated by the full or partial inclusion of mobile network architectures. In these scenarios, the concept and use of anycast routing technology provides an important service enhancement by efficiently supporting robust distributed location and collection services for the end users and easing network configuration burdens.

## Anycast Routing

Rather than designing completely separate anycast routing mechanisms, we illustrate how several different classes of unicast routing protocols can be extended to provide efficient construction and maintenance of anycast routes. Thus, the techniques are readily adaptable to many existing networking technologies. Extensions to link-state, distance-vector and link-reversal unicast routing protocols are all conceptually realized through the representation of an anycast service as a "virtual node" in a graph based on the network topology. The approaches are presented in detail and the advantages, disadvantages, limitations and potential tradeoffs are discussed. The techniques provide an elegant solution for anycast routing that is complementary to existing approaches for both unicast and multicast routing.

### *Anycast Extensions to Link-State Routing*

In link-state routing [6, 7], each network node typically maintains a database representation of the entire network topology. Additionally, each node must disseminate information regarding the state of its adjacent links to all other nodes in the network. This link-state information is flooded throughout the network in a manner that ensures consistency of the separate link-state databases maintained by the individual nodes. Using the link-state database as input, each node computes routes for forwarding traffic through the network. Commonly, the routing computation is based on Dijkstra's algorithm [8], which produces a set of shortest

This work was supported by the Office of Naval Research under contract number N0001499WR20017.

Presented at the 33rd Annual Conference on Information Sciences and Systems (CISS '99), Baltimore, MD, March 1999.

# Report Documentation Page

*Form Approved*
*OMB No. 0704-0188*

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

| | | |
|---|---|---|
| 1. REPORT DATE<br>**MAR 1999** | 2. REPORT TYPE | 3. DATES COVERED<br>**00-00-1999 to 00-00-1999** |

| | |
|---|---|
| 4. TITLE AND SUBTITLE<br>**Anycast Routing for Mobile Services** | 5a. CONTRACT NUMBER |
| | 5b. GRANT NUMBER |
| | 5c. PROGRAM ELEMENT NUMBER |
| 6. AUTHOR(S) | 5d. PROJECT NUMBER |
| | 5e. TASK NUMBER |
| | 5f. WORK UNIT NUMBER |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>**Naval Research Laboratory,4555 Overlook Avenue, SW,Washington,DC,20375** | 8. PERFORMING ORGANIZATION REPORT NUMBER |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | 10. SPONSOR/MONITOR’S ACRONYM(S) |
| | 11. SPONSOR/MONITOR’S REPORT NUMBER(S) |

12. DISTRIBUTION/AVAILABILITY STATEMENT
**Approved for public release; distribution unlimited**

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT<br>**unclassified** | b. ABSTRACT<br>**unclassified** | c. THIS PAGE<br>**unclassified** | | **6** | |

**Standard Form 298 (Rev. 8-98)**
Prescribed by ANSI Std Z39-18

paths from a given node (typically, the calculating node) to all other nodes in the network.

Given that some subset of nodes in a network are providing essentially equivalent service, we extend the link-state routing methodology to provide routing to the nearest point of that service. As is typically done for network routing discussions, we model the network as a graph, where the vertices represent the network nodes and the edges represent the direct communication links between network nodes. We associate a cost, $d_{ij}$, for routing over the direct communication link between a node *i* and its neighboring node *j*.

A "virtual node" is added to the graph to represent the anycast service. At this point a distinction must be made as to whether the network is represented as a directed or undirected graph. If the network is represented by a *directed* graph (i.e., $d_{ij}$ need not be equal to $d_{ji}$), then a "virtual link" can be added *from* each node providing the anycast service *to* the new virtual node representing the service, Figure 1. Assuming all link costs are non-negative, Dijkstra's algorithm can be used to produce a set of shortest paths from any network node to all other nodes (including the virtual node representing the anycast service). The shortest-path spanning tree rooted at node A (for the case where all links have unit cost) is depicted in Figure 1. On any path to the virtual node, the network node preceding the virtual node along that path will be one of the nodes providing the anycast service. Thus, link-state routing implementations that represent the network topology as a directed graph can be extended to provide anycast routes with only minor modifications. Specifically, in addition to disseminating information about adjacent physical links, network nodes must also disseminate information about virtual links to anycast services that they provide.

In cases where the network is represented by an *undirected* graph, then $d_{ij} = d_{ji}$ and typically only a single value need be maintained. Again, a virtual link can be added between each node providing the anycast service and the new virtual node representing the service, Figure 2; however, a small modification to Dijkstra's algorithm is required to ensure the validity of computed routes. When applying Dijkstra's algorithm, the virtual node must be treated differently than the other nodes to prevent paths from traversing the virtual node. For example, assuming all links in Figure 2 have unit cost, then the shortest path from node A to node B based on application of Dijkstra's algorithm (unmodified) would traverse the virtual node. Since the virtual node and links are not part of the physical topology, this is not a valid path for forwarding from node A to node B. Given the ability to distinguish virtual nodes from physical nodes, the formation of such invalid paths through the virtual node can be prevented.

Before describing the needed modifications, we first briefly describe Dijkstra's algorithm. Each node *i* is labeled with a distance estimate, $D_i$, from the root node. When the estimate becomes certain, the node is added to a set *P* of permanently labeled nodes. Initially, the root node is labeled with a distance estimate of zero, and all other nodes are labeled with a distance estimate of infinity. Next, we add the root node to the set *P*, and then iterate as follows.

- Step 1: For each node $j \notin P$, update the distance estimate, $D_j$, to be the smaller of (a) the current distance estimate and (b) the sum of the distance, $D_k$, of the node *k* most recently added to *P* and the direct distance, $d_{kj}$, from node *k* to node *j*.
- Step 2: From the set of nodes currently not in *P*, find the node with the smallest distance estimate and add it to the set *P*.
- Step 3: If the set P contains all nodes, then the computation is complete; otherwise, return to step 1.

During this iterative process, full path or next-hop forwarding information can also easily be collected. To prevent paths from traversing the virtual node (in support of anycast routing) the only modification required is to skip step 1 following the addition of the virtual node to the set *P*. For practical application of this technique to link-state routing implementations that represent the network topology as an *undirected* graph, network nodes must be able to distinguish between physical nodes (i.e., unicast destinations) and virtual nodes (i.e., anycast destinations).

***Anycast Extensions to Distance-Vector Routing***

The distance-vector class of routing algorithms comprises the approaches based on the Bellman-Ford algorithm [8]. There has been a substantial amount of both practical and

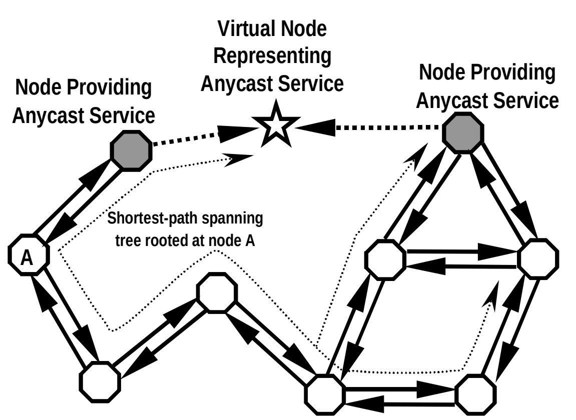



Figure 1. Network topology represented by a *directed* graph with additional "virtual" node and links representing the anycast service.

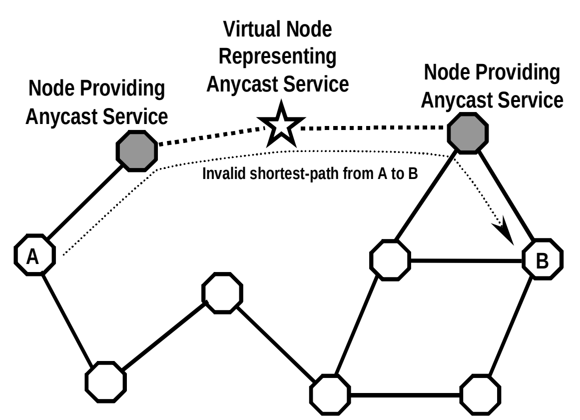



Figure 2. Network topology represented by an *undirected* graph with additional "virtual" node and links representing the anycast service.

theoretical work in the area of distance-vector routing [6, 9]. Distance-vector routing approaches are "destination-oriented"—typically allowing a separate version of the algorithm to be executed independently for each destination to which routing is required. While there are exceptions to this rule, we will limit our discussion to approaches that permit execution for a single destination.

The Bellman-Ford algorithm permits distributed computation, and can be executed either synchronously or asynchronously. The following describes a simple distributed distance-vector routing approach. Each node *i* maintains a distance estimate, $D_i$, to the destination node. The distance estimate of the destination to itself is always zero. Each node other than the destination initially sets its distance estimate to infinity. Each node periodically sends its distance estimate to its neighbors. Each node *j* other than the destination collects the distance estimates of each neighbor *k*, and updates its distance estimate according to the following equation. $D_j = \min [d_{jk} + D_k]$, where $D_k$ is the last estimate received from neighbor *k* and $d_{jk}$ is the cost of the direct link from node *j* to node *k*.

The basic distance-vector routing approach lends itself to anycast routing in a very straightforward manner. Consider a distance-vector algorithm, such as the one described above, executing with the virtual node depicted in Figure 2 as the destination. The two nodes providing the anycast service can simply execute the algorithm as if a distance estimate of zero has been received from the virtual node.

Alternatively, we can consider a graph based on the network topology, but with the set of nodes providing the anycast service consolidated into a single virtual node, Figure 3. In this case, the set of nodes consolidated into the single virtual node collectively form the destination for which the algorithm is running. Thus, each of the nodes providing the anycast service sets its distance estimate to the anycast destination to zero. Since the Bellman-Ford algorithm supports asynchronous operation, no coordination of the distance estimates sent by the nodes providing the anycast service to their respective neighbors is required.

While these techniques are applicable to most of the derivative work on distance-vector routing, they are not applicable to all distance-vector approaches. For example, in Destination-Sequenced Distance-Vector (DSDV) routing [10], periodic routing messages sent by the destination are identified with a monotonically increasing sequence number. This may necessitate some coordination between the nodes providing the anycast service or require other modifications to the protocol to support anycast routing.

***The Cost of the Virtual Link***

In the anycast routing approaches based on link-state routing and the first approach presented based on distance-vector routing, a cost can be associated with each virtual link between a node providing the anycast service and the virtual node. The relative cost of these virtual links has the potential to effect the formation of the anycast routes and consequently the loading on the nodes providing the anycast service. However, the ability to effect the formation of anycast routes based on the cost of the virtual links is also dependent on the physical topology of the network. Thus, the effectiveness and utility of load balancing based on this notion may be limited.

***Anycast Extensions to Link-Reversal Routing***

As with distance-vector routing algorithms, link-reversal routing algorithms are destination-oriented and maintain protocol state on a per destination basis. These routing approaches are based on distributed algorithms that build and maintain a directed acyclic graph (DAG) rooted only at the destination [11, 12]. Links between nodes are directed (to form the DAG) based on a metric, maintained by the nodes, that can conceptually be viewed as a "height" (i.e., a link is directed from the "higher" node to the "lower" node). The DAG serves as a multipath routing structure and, by design, ensures that all directed paths are loop-free and lead to the destination. Immediately following a topological change in the network (e.g., failure of some node's last downstream link), some directed paths may no longer lead to the destination (i.e., the DAG may no longer be rooted only at the destination). This triggers an algorithmic reaction—reversing the direction of one or more links—to re-orient the DAG such that all paths again lead to the destination.

Link-reversal routing algorithms can be extended to provide anycast routes using similar approaches to those used for distance-vector routing. The similarity stems from the destination-oriented nature of the routing algorithms. In the basic distance-vector and link-reversal routing algorithms, the destination (to which the algorithm is computing routes) provides little or no contribution to the distributed computation. As with the distance-vector algorithms, the contribution of the destination in a link-reversal algorithm can be inferred by the neighbors of the destination. Therefore, if we consider a link-reversal algorithm, executing with the virtual node depicted in Figure 2 as the destination. The two nodes providing the anycast service can simply execute the algorithm as if they are neighbors of the anycast destination (i.e., the virtual node), which has an assumed height of zero.

Link-reversal algorithms also support the approach depicted in Figure 3. In this case, the set of nodes providing the anycast service—which have been consolidated into the single virtual node for representation—collectively form the destination for which the algorithm is running. Thus, for a link-reversal algorithm, each of the nodes providing the anycast service simply represents itself as the destination having a height of zero.

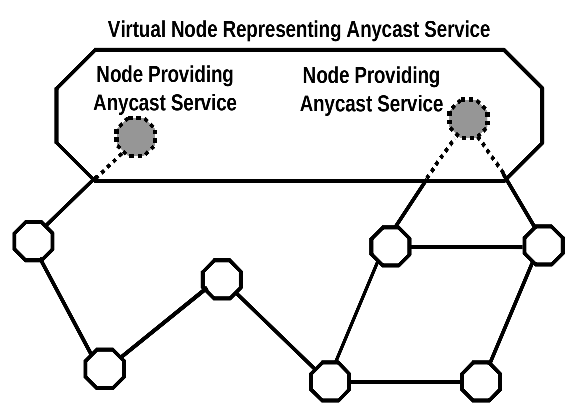



Figure 3. Network topology represented by a graph in which the nodes providing the anycast service are consolidated into a single "virtual" node.

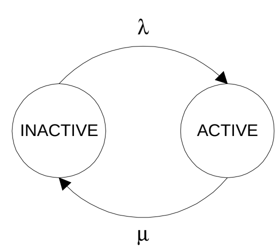
In some cases the destination may provide a more significant contribution to the routing computation. For example, a periodic optimization/refresh process is described as a possible enhancement to the Temporally-Ordered Routing Algorithm (TORA) [12]. This process would require that optimization packets be periodically propagated outwards from the destination. As with DSDV, supporting such a mechanism for an anycast destination would likely require some coordination between the nodes providing the anycast service or additional modifications to the protocol.

## Anycast Routing Performance

While some of the primary advantages of anycast routing in mobile ad hoc networks are the potential to ease network configuration and facilitate more robust distributed system design, there are potential performance gains as well. We conducted a limited simulation study using the Optimized Network Engineering Tools (OPNET) to demonstrate how anycast routing techniques can provide a one-to-any communication capability with greater efficiency and robustness than traditional unicast based techniques. In order to provide sufficient control of the networking environmental characteristics (e.g., rate of topological change, average network connectivity) and to permit simulation of relatively large networks in a reasonable time, we modeled the network using a *fixed* topology with the ability to control the failure/recovery of individual links. Thus, operational links in the fixed topology essentially indicate radio connectivity between node pairs.

We implemented the anycast extension to link-state routing for the case where the network is represented as an undirected graph. Thus, it was necessary that the anycast destination address be distinguishable from the set of unicast destination addresses. Each node in a given network was assigned a unique unicast address from the set of integers in the interval [0, *n*-1], where *n* was the number of nodes in the network. Since the number of anycast services in the initial model was limited to one, the anycast address was simply set to *n*. Nodes that were designated as providing the anycast service would receive packets sent to either their unique unicast address or the anycast address. When a node generated a packet destined for the anycast service, the packet was forwarded using the anycast address.

We compared this anycast routing technique to the use of unicast routing with the destination selected based on a prioritized list of the nodes providing the anycast service. Thus, each node maintained a prioritized list with the unicast addresses of the nodes designated as providing the anycast service. When a node generated a packet destined for the anycast service, it queried the list to determine the unicast address of the highest priority server to which a valid route was available and forwarded the packet using that unicast address.




P[ACTIVE] = $f$

P[INACTIVE] = 1 - $f$

$$\frac{1}{\lambda} = \frac{(1-f)}{f} * \frac{1}{\mu}$$

Figure 4. Link, state transition diagram

### *Simulation Design*

For a given baseline network topology, each link in a given network continuously cycled between two states (ACTIVE and INACTIVE) independently of all other links. Once ACTIVE, the time a link remained ACTIVE was determined randomly based on an exponential distribution. The mean of the distribution ("mean-time-to-failure," $1/\mu$) was an input parameter of the simulation. Essentially, a lower link mean-time-to-failure corresponded to a higher rate of topological change. The long-term average fraction of time each link would be operational, $f$, was also a simulation input parameter. Variation of this parameter affected the average overall network connectivity (i.e., when $f = 0.2$, on average 20% of the links in the baseline topology are operational at any given time). The parameter $f$ was also used to determine the initial state of each link at the beginning of each simulation execution. Once INACTIVE, the time a link remained INACTIVE was also determined randomly by an exponential distribution. However, the mean of the distribution ("mean-time-to-repair," $1/\lambda$), was computed from $1/\mu$ and $f$. The state transition diagram for this continuous-time Markov process, and the equation by which $1/\lambda$ is computed, are presented in Figure 4.

Each ACTIVE link permitted error-free transmission in either direction, and we assumed that channel access was handled at the link level. The transmission rate was set to 64 kbps and the propagation delay was set to zero. When a node needed to "broadcast" a packet to its neighbors, copies of the packet were forwarded over each of its ACTIVE adjacent links. For accounting purposes, when computing the number of packets or bits transmitted, each "broadcast" was counted only once—even though in the simulation, a separate copy of the packet had to be delivered to each neighbor.

Each node randomly generated message packets (with a payload size of 100 bytes) for the anycast service based on an exponential distribution with an expected packet interarrival time of ten seconds. Each node maintained two first-in-first-out (FIFO) packet queues—one for routing control packets and one for message packets. Routing control packets were given (non-preemptive) priority over message packets. When a message packet was removed from the queue for transmission, the routing table was queried to determine the "next-hop" for forwarding the packet based on packet's destination. If no next-hop forwarding information was available for the given the destination, the packet was discarded and the next packet scheduled for transmission was serviced.

Provided that there was an ACTIVE link over which to transmit, packets were transmitted consecutively without intermediate processing delays. The length of the packet and preset transmission rate determined the packet's transmission delay. If a link transitioned to INACTIVE at any time during the transmission of a packet, the packet was considered lost and was discarded by the receiving node. In summary, end-to-end packet delay was solely a function of route selection, queueing delays and transmission delays.

### *Results and Discussion*

We collected results using a baseline topology defined by a "complete" graph of 20 nodes (i.e., each node was connected to every other node by a direct link). Since this allows for the possibility that any two nodes may be able to communicate directly at some point in time (i.e., when the link between them is ACTIVE), this is perhaps the best representation of a mobile network given the limitation of using a fixed baseline topology. Three nodes were selected as anycast destinations (i.e., nodes providing an anycast service). Since any combination of three nodes was essentially equivalent in all respects, the selection was arbitrarily.

A sequence of simulations was conducted to investigate the performance tradeoffs as a function of network connectivity. The fraction of time operational, *f*, was varied from 0.02 to 0.25 for successive simulation runs in the sequence—while the link mean-time-to-failure, $1/\mu$, was kept constant at 60 seconds. The entire simulation sequence was executed first using unicast routing (as previously described) to forward message packets to the nodes providing the anycast service and then repeated using anycast routing. For each simulation run the two approaches were subjected to an identical sequence of random events (e.g., topological changes and packet arrivals).

The amount of additional routing control traffic due to the anycast extensions was measured during each simulation run. In each case, the anycast extensions increased the number of routing control packets by approximately one to two percent. As expected, this increase corresponds approximately to the percent increase in the number of links represented in the link-state database (i.e., three virtual links added to the 190 physical links in the baseline topology).

While the anycast extensions increased the bandwidth utilization for routing control traffic, there was also a reduction in the bandwidth utilization for message traffic. The reduction in bandwidth utilization for message traffic was realized because message packets forwarded based on the anycast routing technique were delivered to the destination using shorter paths on average. The mean message packet hop count for both the anycast and unicast routing techniques is plotted as a function of average network connectivity in Figure 5. The plot clearly illustrates the mean number of hops (i.e., transmissions) required for message delivery using anycast routing is less. While the anycast routing technique forwards to the nearest node providing the anycast service, the unicast technique forwards based on the server priority list and path availability to the servers. Thus, the unicast technique will forward to the primary server (if a path is available) despite the fact that the secondary or tertiary server may be available via a shorter path. Depending on the traffic load and networking environment, the reduction in bandwidth utilization due to the use of shorter paths may outweigh the increase due to the additional routing control traffic.

We also collected statistics regarding the availability of paths from traffic sources to the nodes providing the anycast service. For both the anycast and unicast techniques, upon generation of a message packet, if an available route could not be determined by the source (i.e., no next-hop forwarding information in the routing table), the packet was discarded. For the unicast routing technique, the source would check route availability in the order specified by the priority list and forward using the first valid route determined (i.e., the highest priority server for which valid next-hop forwarding information was available). In all cases when a valid route for a given destination was not available a statistic was collected.

Figure 6 illustrates the route availability statistics for the unicast routing technique. This plot can be interpreted as follows. The lowest curve approximates the probability that a route is available to the primary server. The middle curve approximates the probability that a route is available to the secondary server, given that a route is not available to the primary server. Finally, the highest curve approximates the probability that a route is available to the tertiary server, given that a route is not available to either the primary or secondary server. Although not included on the plot, the route availability for anycast routing was essentially equivalent to the highest curve depicted. This illustrates the improvement in robustness achieved by increasing the number of nodes providing the anycast service. It also illustrates the difference in robustness that would be seen if the unicast routing technique were to only maintain a partial list of the nodes

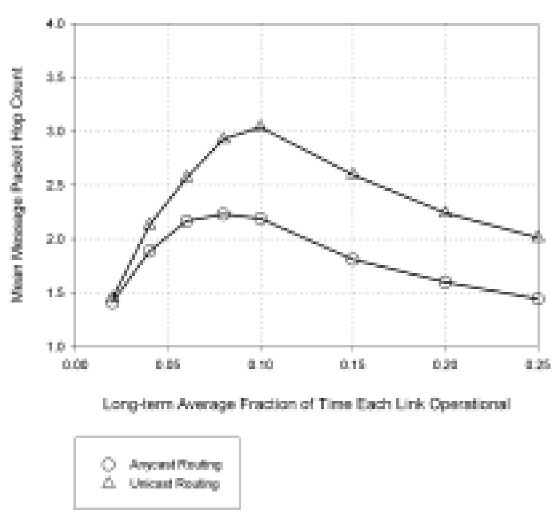



Figure 5. Mean message packet hop count as a function of network connectivity.

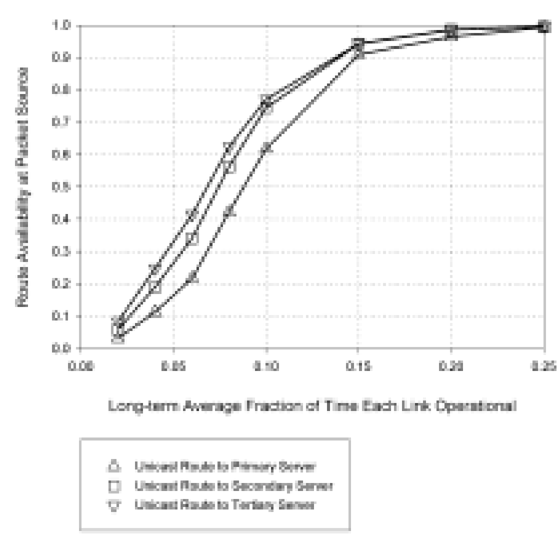



Figure 6. Route availability as a function of network connectivity.

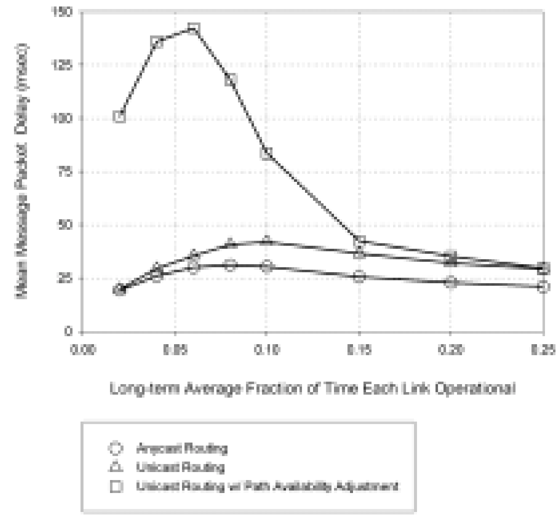



Figure 7. Mean message packet delay as a function of network connectivity.

providing the anycast service. If the number of nodes providing the anycast service is large or the set of nodes is dynamic, maintaining a complete list of anycast servers at all nodes will be complex and potentially impractical. The anycast routing technique provides a mechanism to maximize the robustness with minimal configuration and management.

Finally, we combine these results in consideration of message packet delay and the effect on higher-layer protocols. The mean message packet delay for both the anycast and unicast routing techniques is plotted as a function of average network connectivity in Figure 7. The difference in delay corresponds to the difference in hop count for the two approaches. That is, message packets forwarded based on the anycast routing technique experience less delay because they were delivered using shorter paths on average.

The third curve illustrates the potential effect that route availability may have on higher-layer protocols. This curve was generated by adjusting the mean message packet delay for the unicast routing technique based on an approximation of a retransmission timer and the route availability statistics in Figure 6. The retransmission timer was approximated as $2(\eta+2\sigma)$, where $\eta$ is the mean message packet delay and $\sigma$ is the standard deviation of the message packet delay. This illustrates the additional delay that may be experienced for connection setup or reliable packet delivery when route availability is not known or is not signaled to higher-layer protocols. The retransmission timer approximation is quite conservative; thus, in many applications the retransmission timers may be much larger—resulting in much larger delays.

## Conclusions

The anycast communication paradigm functionally provides the capability to locate and forward network traffic to any one of a set of distributed servers or service points that provide *equivalent* service. Such a mechanism facilitates more robust distributed system design, which will likely be critical in mobile ad hoc networks. While there are many possible approaches to providing an anycasting capability, the use of anycast routing algorithms is perhaps the best-suited approach for the mobile wireless networking environment. It is more communication efficient and requires less configuration and management of end systems than most application-layer approaches.

We have illustrated how several different classes of unicast routing protocols can be extended to provide efficient construction and maintenance of anycast routes. The techniques are readily adaptable to many existing networking technologies and provide an elegant solution for anycast routing that is complementary to existing approaches for both unicast and multicast routing.

The performance aspects of anycast routing have been compared to traditional unicast routing based techniques. We have shown that, depending on the traffic load and networking environment, the use of anycast routing can reduce the overall bandwidth utilization by forwarding message traffic over shorter paths. The simulation results also indicate that anycast routing can ease the configuration and management required to achieve a given level of robustness and can reduce connection setup latency and message packet delay.

While anycast routing has benefits even in quasi-static hardwired networks, the realizable gains are of critical importance for more dynamic networking environments such as a mobile ad hoc network. Although, open issues remain regarding the use of anycast routing in Internet Protocol (IP) based internetworks [1]; the technology is readily applicable and should be further developed.

## References

[1] C. Partridge, T. Mendez and W. Milliken, Host Anycasting Service, *Internet RFC 1546* (November 1993).

[2] J. Guyton and M. Schwartz, Locating Nearby Copies of Replicated Internet Servers, *Proc. SIGCOMM '95* (August 1995).

[3] S. Bhattacharjee, M. Ammar, E. Zegura, V. Shah and Z. Fei, Application-Layer Anycasting, *Proc. INFOCOM '97* (April 1997).

[4] S. Corson and J. Macker, Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations, *Internet RFC 2501* (January 1999).

[5] Working documents of the Internet Engineering Task Force (IETF) Mobile Ad Hoc Networks working group, http://www.ietf.org/html.charters/manet-charter.html.

[6] S. Keshav, *An Engineering Approach to Computer Networking* (Addison-Wesley, 1997).

[7] J. Moy, OSPF Version 2, *Internet RFC 2328* (April 1998).

[8] D. Bertsekas and R. Gallager, *Data Networks* (Prentice-Hall, 1992).

[9] C. Hedrick, Routing Information Protocol, *Internet RFC 1058* (June 1988).

[10] C. Perkins and P. Bhagwat, Highly dynamic destination-sequenced distance vector routing (DSDV) for mobile computers, *Proc. SIGCOMM '94* (August 1994).

[11] E. Gafni and D. Bertsekas, Distributed algorithms for generating loop-free routes in networks with frequently changing topology, *IEEE Trans. Commun.* (January 1981).

[12] V. Park and M.S. Corson, A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks, *Proc. IEEE INFOCOM '97* (April 1997).