SHARP: A Hybrid Adaptive Routing Protocol for Mobile Ad Hoc Networks*

Venugopalan Ramasubramanian Zygmunt J. Haas Emin Gün Sirer

Cornell University,
Ithaca, NY 14853
{ramasv@cs.cornell.edu, haas@ece.cornell.edu, egs@cs.cornell.edu}

Abstract

A central challenge in ad hoc networks is the design of routing protocols that can adapt their behavior to frequent and rapid changes in the network. The performance of proactive and reactive routing protocols varies with network characteristics, and one protocol may outperform the other in different network conditions. The optimal routing strategy depends on the underlying network topology, rate of change, and traffic pattern, and varies dynamically. This paper introduces the Sharp Hybrid Adaptive Routing Protocol (SHARP), which automatically finds the balance point between proactive and reactive routing by adjusting the degree to which route information is propagated proactively versus the degree to which it needs to be discovered reactively. SHARP enables each node to use a different application-specific performance metric to control the adaptation of the routing layer. This paper describes application-specific protocols built on top of SHARP for minimizing packet overhead, bounding loss rate, and controlling jitter. Simulation studies show that the resulting protocols outperform the purely proactive and purely reactive protocols across a wide range of network characteristics.

1 Introduction

Ad hoc networks are characterized by frequent change. Many of the diverse application areas for ad hoc networks, including emergency relief operations, battlefield applications and environmental data collection, exhibit a high degree of temporal and spatial variation. Nodes may join the network at any time, get disconnected as they run out of power, or alter the physical network topology by moving to a new location. Link characteristics, such as bit error rates and bandwidth, change frequently due to external factors like interference and radio propagation fading. Traffic patterns in the network can shift drastically as applications modify their behavior and redistribute load within the network. Consequently, a primary challenge in ad hoc networks is the design of routing protocols that can adapt their behavior to rapid and frequent changes at the network level.

Ad hoc routing protocols proposed to date fall between two extremes based on their mode of operation. *Proactive protocols*, such as DSDV [23], TBRPF [20], and OLSR [9], exchange routing information periodically between hosts, and constantly maintain a set of available routes for all nodes in the network. In contrast, *reactive protocols*, such as AODV [25], DSR [11], and TORA [21], delay route discovery until a particular route is required, and propagate routing information only on demand. There are also a few hybrid protocols, such as ZRP [7], HARP [19], and ZHLS [10], that combine proactive and reactive routing strategies.

There is a fundamental trade-off between proactive dissemination and reactive discovery of routing information. While proactive protocols can provide good reliability and low latency through frequent dissemination of routing information, they entail high overhead and scale poorly with increasing numbers of participating nodes. In contrast, reactive protocols, can achieve low routing overhead, but may suffer from increased latency due to on-demand route discovery and route maintenance. Since the characteristics of a practical network vary dynamically with time, choosing an appropriate routing protocol is an important design and implementation decision. A protocol suited for a given network size, density, and mobility may behave inefficiently as the network characteristics and application behavior change.

In this paper, we present the *Sharp Hybrid Adaptive*

*This work was supported in part by ONR grant N00014-01-0968, DARPA under AFRL grant RADC F30602-99-1-0532 and the MURI programs administered by ONR grant N00014-00-1-0564, by AFOSR grant F49620-02-1-0233, and by NSF grant ANI-0081357. The views and conclusions contained herein are those of authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of these organizations or of the Government.

Routing Protocol (SHARP), which utilizes this fundamental trade-off between proactive versus reactive routing to find a good balance between route information propagated proactively and route information that is left up to on demand discovery. SHARP utilizes both a proactive and a reactive protocol to perform routing. Each SHARP node determines the network neighborhood, called *proactive zone*, in which routing information pertaining to itself is disseminated proactively. SHARP relies on a novel proactive routing algorithm that is both efficient and analytically tractable. However, SHARP can use any reactive routing algorithm whose costs can be characterized analytically; our current implementation uses off-the-shelf AODV. SHARP finds the "sweet spot" between the two routing regimes by dynamically adjusting the extent of proactive and reactive routing. This boundary is determined by an analytical model derived herein and guided by dynamically-performed empirical measurements from the physical network.

Requirements for network performance vary among applications. Multi-media applications can tolerate high loss rates, but are sensitive to variations in delay. TCP traffic is sensitive to loss in the network, while devices running on battery power are concerned with the routing overhead. However, applications have no control over the performance of traditional routing protocols. In contrast, SHARP enables each application to pursue different quantitative metrics for guiding the inherent trade-off between increased overhead for proactive information dissemination versus reduced latency and loss rate. Each SHARP node can separately pursue different application-specific performance guarantees. For instance, one node may direct SHARP to adjust its route dissemination to reduce delay jitter, while another node concurrently uses SHARP to minimize packet overhead. This paper presents three SHARP-based protocols for pursuing application-specific goals, namely, minimizing packet overhead, bounding loss rate, and controlling delay jitter.

An adaptive hybrid routing protocol requires the following three properties for successful deployment.

- **Adaptive:** The protocol should be applicable to a wide range of network characteristics. It should automatically adjust its behavior to achieve target goals in the face of changes in traffic patterns, node mobility and other network characteristics.

- **Flexible:** The protocol should enable applications to optimize for different application-specific metrics at the routing layer. These optimization goals should not be set by the network designer, but be placed under the control of the network participants.

- **Efficient and Practical:** The protocol should achieve better performance than pure, non-hybrid, strategies without invoking costly low-level primitives such as those for distributed agreement or reliable broadcast.

Through a combination of protocol design, model, and mechanisms, SHARP's hybridization approach provides all of these properties.

SHARP achieves its performance through low cost mechanisms to determine zone sizes and control the extent of proactive routing. SHARP nodes monitor traffic pattern and local network characteristics such as link failure rate and node degree without incurring excessive overhead. The zone sizes are then determined by each node in isolation, solely based on local information. An efficient control mechanism allows SHARP to shrink or grow the region of proactive routing without excessive control and synchronization overhead. Where adjacent zones overlap, SHARP takes advantage of the overlap to disseminate proactive route information more efficiently. Finally, SHARP does not require distributed coordination to determine zone boundaries, and does not necessitate a costly reliable broadcast mechanism.

Overall, this paper makes the following contributions. First, it presents a novel hybrid and adaptive protocol that provides application-level control over its performance by integrating a reactive and a proactive routing algorithm over temporal and spatial domains. The protocol embodies a low-overhead mechanism for adaptation and an analytical model to guide the trade-off between the proactive and reactive routing regimes. Second, it describes the application of this protocol to achieve three separate goals, namely, minimizing packet overhead, controlling delay jitter, and bounding loss rate. SHARP enables multiple nodes in the network to pursue disparate goals at the routing layer. Finally, it motivates the case for hybrid adaptive routing protocols by showing that the ideal point of operation for achieving minimal packet overhead, bounded loss rate, and controlled delay jitter lies between purely reactive and purely proactive routing regimes. An extensive, 600-node simulation study demonstrates that the SHARP hybrid protocol performs better than its pure components across a wide range of network conditions.

The rest of this paper is organized as follows. In the next section, we discuss related work on hybrid and adaptive routing protocols. Section 3 provides an overview of SHARP's operation. Section 4 contains a detailed description of the routing protocols employed by SHARP and Section 5 outlines the analytical model that drives our adaptation strategy. Mechanisms and strategies for adaptation employed by SHARP are described in Section 6. Section 7 demonstrates that SHARP performs well under a wide range of network conditions. We summarize our contributions and observations in Section 8.

2 Related Work

While most of the routing protocols proposed to date for ad hoc networks are either purely reactive or purely proactive, some hybrid protocols have also been proposed. However, only a few hybrid protocols embody fine-grained control over the extent of hybridization, while none allow the participating node to pursue disparate application-specific goals. In this Section, we provide a brief overview of hybrid and adaptive routing protocols and summarize how they differ from SHARP.

The *Zone Routing Protocol* [7] (ZRP) was the first hybrid routing protocol with both a proactive and a reactive routing component. ZRP defines a *zone* around each node consisting of its k-neighborhood. Routing within a zone is performed using a proactive routing protocol and routing between nodes in different zones is performed by an on-demand routing protocol. ZRP performs efficient route discovery through *bordercasting*; route requests are spread by multicasting them directly to the nodes on the border of its zone. The size of the zone is dynamically determined based on network load [22, 26].

While SHARP shares with ZRP the insight that hybrid routing offers a flexible way to adapt between reactive and proactive routing strategies, the two approaches differ in several fundamental ways. First, ZRP is optimized for the single goal of reducing routing overhead, whereas SHARP enables application-specific adaptation strategies to bound loss rate and control jitter, in addition to controlling the overhead of the routing protocols. Second, ZRP defines a proactive zone around every node in the network irrespective of its participation in carrying data packets. In contrast, SHARP maintains proactive routing zones only around those nodes that have significant incoming data. Finally, the proactive routing component of ZRP is more expensive than SHARP's proactive routing component. ZRP's use of bordercasting in order to efficiently propagate route requests involves maintaining a multicast tree in each zone with the center node as the root of the tree and the border nodes as the multicast destinations. In order to maintain this tree, the intra-zone routing protocol needs to provide routes between every pair of nodes in the proactive zone. In SHARP, the proactive routing protocol only maintains routes to the center node. This leads to a fundamental difference in the determination of the optimal zone size. ZRP decreases the size of the proactive zone as mobility and, correspondingly, the frequency of link-failures increase, whereas SHARP increases the size of the proactive zone in response to link failures.

ZHLS [10], Zone-based Hierarchical Link State, is another zone-based hybrid routing protocol, which partitions the network into non-overlapping zones based on physical location information. It performs zone assignments once and zone sizes do not vary dynamically. HARP [19], Hybrid Ad-hoc Routing Protocol, is a hybrid protocol that combines proactive and reactive approaches. It relies on the Distributed Dynamic Routing [18] protocol for decomposing the network into zones. A set of forwarding nodes in each zone is responsible for communicating with nodes in other zones. HARP uses its own custom protocol for inter-zone routing, whose main goal is to reduce delays through early path maintenance. While HARP creates zones of variable sizes, it has no direct control over the zones and does not dynamically adjust their sizes.

In addition, there are also several routing protocols that adapt based on network characteristics to enhance performance. FSR [6], Fisheye State Routing, is a link-state protocol that exchanges periodic link-state information. The period of link state propagation is determined by the distance to the destination. ADV [1] is Adaptive Distance Vector algorithm that exhibits on-demand characteristics by varying the frequency and size of routing updates. Some researchers have examined supplanting reactive protocols with timer-directed route discoveries to produce backup routes prior to losing the primary link [15]. Their protocol uses a fixed timer value across all nodes, which is determined offline from a past history of link-failure statistics.

Several researchers have examined the behavior of reactive and proactive routing protocols through simulations [2, 5]. Our primary goal in this paper is not to perform protocol comparisons, but to identify and evaluate the trade-off between the two routing regimes. Our measurements show that the choice of the optimal routing algorithm is highly dependent on network and data traffic characteristics, and that no single point on the continuum that spans reactive and proactive protocols is well suited for dynamic networks.

3 Overview of SHARP

The Sharp Hybrid Adaptive Routing Protocol adapts efficiently and seamlessly between proactive and reactive routing strategies. This adaptation is driven by the measured characteristics of the network and can be directed to optimize for user-defined performance metrics, such as loss rate, routing overhead, or delay jitter.

SHARP adapts between reactive and proactive routing by dynamically varying the amount of routing information shared proactively. It does so by defining a *proactive zone* around some nodes. A node-specific *zone radius* determines the number of nodes within a given proactive zone. Each node at a distance less than or equal to the zone radius is a member of the proactive zone for that node. All nodes not in the proactive zone of a given des-

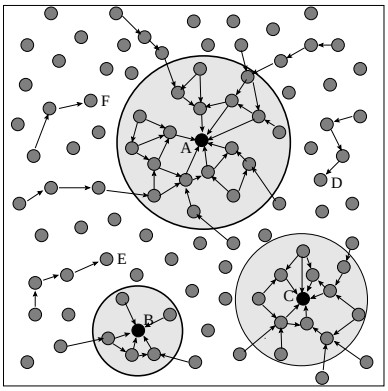

Figure 1: The SHARP proactive zones constructed around destinations A, B, and C. The vertices correspond to nodes, directed edges represent route table entries. The size of the proactive zone is adjusted independently by each destination.

tination use reactive routing protocols to establish routes to that node. Node-specific proactive routing is employed within a proactive zone. Nodes within the proactive zone maintain routes proactively only to the central node.

SHARP creates proactive zones automatically around hot destinations that receive data from many sources. The proactive zones act as collectors of packets, forwarding the packets efficiently to the destination, once the packets reach any node at the zone periphery. SHARP amortizes the cost of maintaining routes to a given destination in a proactive zone among all the sources that communicate with that destination node. Hence, it is well suited for applications that exhibit spatial locality in their network communications. For example, in sensor networks, selected nodes performing data aggregation act as hot destinations for data generated by nearby sensors. In office networks, gateways or service providers are contacted by several nodes in the network. In many general-purpose applications, including messaging, and wireless audio, the popularity of nodes follows a Zipf-distribution [4], making some nodes more popular than others.

Figure 1 shows the SHARP topology in a typical deployment. SHARP maintains proactive routing zones around popular destinations A, B, and C. It achieves this by dynamically adapting the zone radius at each destination based on incoming data traffic and mobility in the network. Consequently, it will create relatively large zones around popular destinations, and relatively small proactive zones around nodes that get little traffic. For instance, nodes with little or no data traffic, such as D, E, and F, will have no proactive routing zone and will rely purely on reactive routing.

The zone radius acts as a virtual-knob to control the mix of proactive and reactive routing for each destination. By increasing the radius, SHARP can decrease the loss rate and the variance in delay, but will pay more in packet overhead to maintain routes in a larger zone. By decreasing the radius, SHARP can reduce routing overhead, as fewer nodes need to be proactively updated; however, it may pay more in delay jitter and experience higher loss rates. Using this trade-off, SHARP can act as a completely reactive protocol by setting the zone radius of all the nodes to zero. Conversely, SHARP can emulate a completely proactive protocol by setting the radii to equal the network diameter.

The primary challenge in the design of a hybrid protocol is how to determine the optimal trade-off between the components of the hybrid. Ideally, a hybrid protocol would achieve fine-grained control over this trade-off, incur low overhead for adaptation and exploit information locality for maximum efficiency. In the next sections, we describe the proactive and the reactive components of SHARP and the analytical model by which it controls this trade-off efficiently.

4 SHARP Routing Protocols

SHARP is composed of a proactive routing component and a reactive routing component. This section describes the two routing components of SHARP in detail.

4.1 Proactive Routing Component

The SHARP Proactive Routing protocol (SPR) is an efficient protocol engineered with techniques borrowed from different routing algorithms such as Destination Sequenced Distance Vector (DSDV) [23] and Temporally Ordered Routing Algorithm (TORA) [21]. SPR maintains routes to a single destination in each zone. This enables SPR to employ low overhead mechanisms to build and maintain routes and to offer several good properties, including predictable routing overhead, low loss of data packets, and low variance in packet inter-arrival time.

SPR performs proactive routing by building and maintaining a directed acyclic graph (DAG) rooted at the destination. The DAG is built by employing a *construction protocol* initiated periodically by the destination once every *reconstruction_interval*. The DAG is restricted to nodes within the proactive-zone of the destination. In between reconstruction, it is maintained by an *update protocol* that relies on periodic exchange of updates and local recovery mechanisms to handle link-failures.

The construction protocol is used to generate a DAG periodically. The destination node initiates the construction process by generating a DAG *construction packet*. This packet carries the zone radius and a sequence number to distinguish the new DAG from the old DAG. Its time-

to-live (TTL) field is set to the zone radius and is propagated within the proactive zone by a broadcast. It builds a DAG by assigning a *height* to each node. The height is used for routing and corresponds initially to the distance of the node from the destination. A node B that receives the construction packet from a node A locally adds the $A \rightarrow B$ link to the DAG and rebroadcasts the construction packet after incrementing its height by one. A node may receive the construction packet from multiple neighbors along different links. In this case, the node stores the height information of all neighbors, sets its own height to the minimum of the height values it has received.

A naive scheme that uses flooding to disseminate construction packets suffers from packet loss due to collisions [16]. To avoid broadcast floods, each node waits for a random interval of time, bounded by T, before retransmitting the construction packet. However, this might cause imbalance in the construction of the DAG, since a node might receive the construction packet from nodes farther away from the destination before receiving the packet from nodes closer to the destination. To compensate, each node waits for W seconds after receiving the first construction packet. It collects all construction packets received during this time, sets its height to the minimum of all height values it has observed, and forwards a single construction packet, if the TTL is positive. Construction packets arriving after W seconds do not impact the height chosen by the node. While this algorithm does not guarantee that all the nodes will detect the shortest distance to the destination, a conservative choice for W ensures that the DAG will not deviate significantly from the optimal. Most importantly, each node forwards a packet at most once, thereby bounding the number of construction packets within a zone by the number of nodes in the zone.

An *update protocol* is used for beaconing and maintenance of the DAG during link-failures. Once every *beacon_interval*, each node in the proactive zone broadcasts its current height in an update packet, whose time-to-live is set to one. Each node records the height of its neighbors upon receiving update packets from them. The update packets also serve as 'hello' beacons to detect the formation of new links and the breakage of current links. A new link is formed whenever an update packet is received from a neighbor for the first time, while a link is considered broken if at least *beacon_loss* consecutive update packets are not received from a neighbor.

The update protocol relies on TORA's failure recovery mechanisms to restore the DAG in response to link-failures. Link-failures are detected either when consecutive update packets are lost, or when the MAC layer is unable to transmit a data packet to a neighbor. The failure recovery is invoked only when all of the downstream links at a node have failed. Following the approach described in [3], the node reverses the orientation of its upstream links by choosing a new height greater than the height of its neighbors and initiating a new update packet. Using a TORA-based failure recovery provides three properties. First, it allows SPR to recover from link-failures within the proactive zone without having to wait until the next reconstruction interval. Second, it constrains the impact of the link-failure to a small local region. Finally, as we describe later in Section 5, SPR incurs low and predictable overhead, which makes the analytical model more tractable. Note that performing local repairs distorts the optimality of the DAG, since the correct distance to the destination is no longer known. This has been shown to significantly affect the performance of TORA [2]. SHARP compensates for this problem with the construction protocol, which periodically rebuilds the DAG from scratch. Hence, SPR's DAGs contain relatively short paths to the destination most of the time.

SPR performs routing of data packets in the following manner. A data packet arriving at a node is transmitted along a downstream link to the neighbor with the lowest height. If the data packet cannot be transmitted to that neighbor because of a link-failure, it is transmitted to the next best downstream node. If all the downstream links have failed, link reversal is performed to repair the DAG and the data packet is forwarded to the best newly formed downstream node. Temporary routing loops could be created (as observed in [2]) if the data packets reach the new downstream node ahead of the update packet. To prevent this, the new height of the node is piggybacked with the data packet, so that the downstream node can update the new height before forwarding the data packet.

Multi-path routing, local link repairs and the construction protocol enable SPR to be a robust and efficient protocol, incurring low loss rate and predictable overhead.

4.2 Reactive Routing Component

SHARP's reactive routing protocol is based on the Adhoc On-demand Distance Vector (AODV) routing protocol [25]. SHARP simply uses standard AODV [24] for reactive routing, and includes several optimizations such as route caching and expanding ring search.

SHARP integrates the proactive and the reactive components seamlessly without incurring additional overhead. If the source node is within the proactive zone, routing is performed proactively as described earlier. If the source node is outside the proactive zone, route requests are broadcast by AODV. Nodes in the proactive zone of a destination generate route replies in response to route requests generated by the source node. The replying node in the proactive zone sets the distance to the destination as zero. Thus, the proactive zone effectively acts as a collective destination for the data packets from the source.

Once data packets enter the proactive zone, they are efficiently routed by the proactive protocol. Routes between the source node and the edge of the proactive zone are maintained reactively.

Overall, SHARP is a general framework for hybridization that can be used with any combination of proactive and reactive routing protocols whose costs are analytically tractable with information efficiently available to the route destination.

5 Model

An analytical model of proactive and reactive routing costs enables SHARP to make an informed trade-off between different routing regimes. This model provides an accurate estimate of the routing overhead of the proactive component and an approximate analysis of the overhead of the reactive routing component.

The notation N_r^A is used to represent the number of nodes in a region of radius r around node A. The average lifetime of a communication link is represented by λ. We assume that link lifetimes are independent of each other and are exponentially distributed. Consider a source node S that transmits data to a destination node D, which is located at a distance of h hops from S. Let a proactive zone of radius r hops exist around the destination node D. The SHARP proactive routing protocol (SPR) is employed for routing within this proactive zone, while the reactive routing protocol is used by S for discovering routes to the proactive zone. This setup is illustrated in Figure 2.

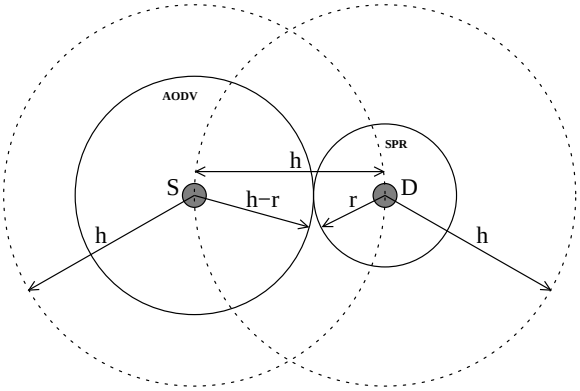

Figure 2: This figure shows a route setup between the source node S and a destination node D at a distance of h hops. A SHARP proactive zone of radius r is maintained around the destination node D, while the source node S employs AODV to find routes to the proactive zone.

Proactive Routing Overhead

The routing overhead of SPR has two components. A portion of the overhead is fixed and stems from the periodic control packets used to generate the DAG. The remaining overhead stems from event triggered control packets generated as a result of link-failures. The periodic overhead in turn consists of control packets sent during periodic DAG reconstruction, as well as update packets sent every beacon_interval. Let f_u be the frequency at which update packets are generated by each node in the proactive zone and let f_c be the frequency at which construction is started by the destination node D. The per-node overhead for sending update packets is f_u packets per second. Since during construction, each node in the proactive zone broadcasts the construction packet exactly once, the per-node overhead for construction is f_c packets per sec. Thus, the total fixed overhead of SHARP proactive routing component is $N_r^D(f_u + f_c)$ packets per second in a proactive zone of radius r.

The fixed component of the routing overhead of SPR is independent of both the mobility rate and the number of sources sending data packets to the same destination. If there are several destinations, the same node in the network could belong to the proactive zones of multiple destinations. In that case, updates for different destinations are aggregated into a single packet. Thus the number of periodic update packets sent does not increase even though the size of update packets increases with the number of destinations. If a node belongs to the proactive zone of d destinations, the fixed SHARP overhead at that node can be given by $f_u + df_c$ packets per second.

Link-breaks within the proactive zone trigger control packets carrying changes in the height of a node in the proactive zone of a destination D. SPR employs TORA's repair mechanism [3] to update the height of each node upon a link-failure. Accordingly, the height of a node in the proactive zone changes only if all downstream links in the DAG at that node fail. Assuming that different links at a node are formed and broken independently with the lifetime distributed exponentially with mean λ, the average frequency at which all the downstream links fail at a node can be estimated to be $\frac{1}{2\beta_n \lambda}$ (see Appendix), where β_n equals $\sum_{i=1}^{n} \frac{1}{i}$, and n is the number of downstream links at that node. In addition, an event triggered by link-failures at a node alters the status of several of its upstream links. This results in reduction of downstream links at other nodes. Incorporating the effect of this in our earlier estimation, we can revise the approximation of the frequency of event triggered updates to $\frac{1}{2(\beta_n - 1)\lambda}$. Each destination incurs this overhead. Hence, the overhead scales linearly with the number of destinations.

The contribution of event-triggered overhead to SHARP's proactive routing component is very low. The total routing overhead of SPR in a proactive zone of radius r around a destination node D is $N_r^D(f_u + f_c + \frac{1}{2(\beta_n - 1)\lambda})$

packets per second. In practice, the frequency of periodic updates, f_u, is chosen to be much higher than the frequency of link-failure, $\frac{1}{\lambda}$, in order to detect failures quickly. As a result, the periodic packets dominate the control overhead of SPR. This trend is also observed in our simulation results presented in Section 7. Thus, SHARP's proactive routing component presents a predictable routing overhead, which is mostly independent of the mobility rate.

The size of the proactive zone substantially determines the overhead of SPR. As the zone radius is increased, the overhead of SPR increases significantly since, the number of nodes in a proactive zone, N_r^D, often increases polynomially with the zone radius.

Reactive Routing Overhead

Accurate estimation of the routing overhead of a reactive routing protocol such as AODV is very difficult. Here, we present an approximate analysis of the routing overhead of AODV, which helps us to characterize the behavior of AODV and provide heuristics for adapting the radius of the SHARP's proactive zones. The evaluation section shows that these approximations work well in practice.

AODV expends several control packets, such as route requests, route replies, and route errors during route discovery and maintenance [24]. Route requests for a destination D are generated at the source node S when a route to D is requested for the first time, or when a route is broken due to link-failures. Route replies are transmitted in response to route requests, while route errors are generated when a route to a destination fails. The number of route error packets is typically low compared to the number of route requests and route replies, since the route errors are propagated only by the nodes in the route between S and D, whereas route requests and route replies could be propagated by any node in the network.

The overhead for a source node S to reactively discover a route of length h is approximately proportional to N_h^S packets, where N_h^S is the number of nodes at a distance of at most h from S. In reality, the source node does not know the correct distance to destination D. Instead, it restricts the propagation of route requests to within a few hops of h by using the expanding ring search. AODV routes fail at an average rate of $\frac{h}{\lambda}$ per second under the assumption of independent and exponentially distributed link-failures (see Appendix). Thus, the route maintenance overhead of AODV, for active routes, is approximately proportional to $N_h^S \frac{h}{\lambda}$. Other factors such as congestion could also cause route breaks. In this analysis, we assume that the impact of congestion is insignificant compared to the effect of mobility on routing performance. AODV's routing overhead increases with the distance between the source and destination, since longer routes break more often than shorter ones. Also, destinations farther away require route requests to be propagated in a larger area. The routing overhead of AODV is proportional to the number of active routes in the network. Hence, AODV's overhead increases with the number of sources and the number of destinations in the network. If s sources are transmitting data packets to a single destination, the total routing overhead of AODV is approximately proportional to $\sum_{i=1}^{s} N_{h_i}^{S_i} \frac{h_i}{\lambda}$, where S_i represents i^{th} source and h_i its average distance from D.

While the model presented here provides a quantifiable metric that can guide how to modify the proactive radius, it is still an approximation. The values it computes may diverge from the actual behavior of the deployed routing protocols. Optimizations such as route caches, local route repair, and multiple routes would impact the actual cost observed in the network. However, we show in the evaluation section that the model captures the overheads of routing protocols adequately and leads to the construction of adaptive hybrid protocols that outperform both the purely proactive and the purely reactive routing components across a wide range of network conditions.

In the next section, we discuss the application of this analysis to the construction of specialized adaptation algorithms. Since each node makes independent decisions, further discussions in the paper only describe adaptation at a single node. However, these protocols apply equally well to multiple nodes, as the adaptation does not require any consensus or communication between the participating nodes.

6 SHARP Adaptation

SHARP controls the performance of the routing protocol by dynamically adjusting the zone radius around each destination. Each destination node varies the size of the proactive zone around itself by taking into account the network characteristics, such as the mobility rate and the node-degree, as well as the data traffic characteristics, such as the number of sources and the distance of the sources from the destination. The decision to alter the present zone radius is made by a destination locally and independent of other destinations. This enables SHARP to control different application-specific performance metrics, such as routing overhead, loss rate, and delay jitter, and to have different nodes in the network that optimize for different metrics simultaneously. This section explains the mechanisms and the strategies used to perform the dynamic adaptation in SHARP.

6.1 Adaptation Mechanisms

To enable efficient operation and frequent adaptation, we need low cost mechanisms for monitoring the network and changing the size of the zone radius accordingly. In particular, SHARP requires mechanisms for estimating low-level network characteristics, such as average link lifetime and average node-degree, as well as traffic characteristics, such as loss rate and delay jitter. It also requires an efficient mechanism to propagate the zone radius and modify the proactive zone based on these measurements and the analytical model.

SHARP nodes continually monitor network characteristics. In particular, nodes in a proactive zone measure and maintain the average link lifetime of their immediate links. In addition to average link lifetime, the nodes in a proactive zone also measure the average node-degree, that is, the number of immediate neighbors. They aggregate the average link lifetime and the average node-degree with the values received from other upstream nodes as part of the update protocol and broadcast the aggregated-values periodically along with the update packet. These values are combined throughout the update process, taking care not to double-count data, until they ultimately reach the destination at the center of the DAG. The destination then estimates λ and computes N_r^D at the end of an update cycle.

The destination node also locally maintains statistics about the data traffic that it has received. Each destination keeps track of the identity of the source, the current distance to the source, and the average performance in terms of loss rate and delay jitter over a period of time. To facilitate these measurements, each source node inserts a SHARP header as an IP option to every data packet. The SHARP header contains the number of packets transmitted by the source to this destination and a field marking the distance to the destination, which is initialized to 1. Every intermediate node transmitting this packet increments the distance field, enabling an up-to-date measurement of the hop-count between the source and the destination. The destination records the number of data packets it received and estimates the loss rate in the network from the count of the data packets sent. The destination keeps track of the arrival times of consecutive packets and directly computes the delay jitter from the variance of inter-arrival times. Note that computing the delay jitter does not require clock-synchronization between the source and the destination, since a local clock is sufficient to measure the variance of inter-arrival times. Combined with the low-level network characteristics, traffic metrics enable each destination to independently compute the optimal radius for its proactive zone.

Once a new zone radius has been computed, it needs to be efficiently disseminated to the nodes in the new proactive zone. The decision to expand or shrink the zone radius is made periodically, just before a DAG reconstruction is initiated by the destination. The destination-centered DAG is reconstructed as described in Section 4, with the new value of the radius. Thus, expanding the radius from r to s for $s > r$ is straightforward and imposes no extra routing overhead on SHARP proactive protocol.

When the radius is shrunk from r to s for $r > s$, care must be taken to purge the nodes in the older DAG that are no longer part of the newly formed DAG. This is performed by setting the *time-to-live* field of the construction packet to r instead of the new radius s. Nodes receiving this construction packet terminate all proactive activity for that destination if they do not belong to the newly built DAG. In addition, the nodes that do not receive this construction packet due to network congestion stop receiving periodic update packets from their neighbors if they do not belong to the new DAG and, thus, perceiving link-failures will start purging themselves from the DAG. Thus, this scheme exhibits graceful degradation without the need for costly reliable multicast services or distributed consensus protocols.

Decisions to alter the radius of the proactive zone are taken periodically every *reconstruction_interval*. In order to impart stability to the process of adaptation, SHARP employs hysteresis in the form of two thresholds. The *up_thresh* and the *down_thresh*, with values 1 and 1.5, respectively, determine the high and low water marks in a given metric, before a change in the zone radius is triggered.

6.2 Adaptation Strategies

SHARP uses the efficient mechanisms described in the previous section to dynamically manage the trade-off between proactive and reactive routing. It provides a general framework that can be used to achieve different application-specific goals. In this section, we describe applications of this framework to achieve three separate goals.

Minimal Packet Overhead

Routing overhead is a critical consideration when choosing a routing protocol. In mobile environments, nodes are typically limited by battery power. Routing algorithms that require excessive communication will experience greatly diminished system longevity.

Our strategy for adapting the zone radius in order to reduce the routing overhead is based on the previously discussed trade-offs between proactive and reactive routing components. The cost of the proactive component is independent of the number of sources, and it can be amortized across the multiple sources that are sending packets to the

same destination. Since SPR performs local repair, the overhead of SHARP's proactive component is largely independent of the mobility rate. The cost of reactive routing is tightly coupled to the number of sources as well as the mobility rate. As the mobility in the network increases, the reactive routing component is forced to incur higher aggregate costs for route discovery. Thus, depending on the instantaneous values of network parameters, there is an opportunity for optimization by choosing one routing regime over another.

We propose a protocol for minimizing the per-packet overhead of routing algorithms based on the SHARP adaptation framework. Called SHARP-PO, this protocol performs a dynamic adaptation between predictable, high cost SPR protocol versus the varying costs of reactive AODV protocol in order to find the optimal mix of routing overhead. The goal of the SHARP-PO protocol is to dynamically find values for proactive radii that optimize the total cost. The model introduced in Section 5 enables SHARP to quantify the trade-off between different routing regimes in terms of overhead. The overhead of routing in a proactive zone of radius r around the destination D is $N_r^D(f_u + f_c + \frac{1}{2(\beta_n - 1)\lambda})$. The overhead of AODV for each source outside the proactive zone at a distance h hops from D is approximated by $N_{h-r}^S \frac{h-r}{\lambda}$. Note that the source node only needs to find a route to some node on the edge of the proactive zone. Hence only $h - r$ nodes in the path perform reactive routing.

The destination node estimates the incremental difference in the overhead of the two components, when the radius is increased by 1, using the above expressions. If the reduction in the overhead of the reactive component is more than *up_thresh* times the increase in the overhead of the proactive component, the radius is incremented by 1. Similarly, the difference in overhead is computed for the case of radius decrement and the radius is decremented by 1, if the reduction in the overhead of the proactive component is more than *down_thresh* times the increase in overhead of the reactive component. Otherwise, the radius is kept unchanged. This algorithm is executed periodically, once every *reconstruction_interval*. During these estimations, locally observed values of mean link lifetime and node-degree are substituted for both SPR as well as AODV.

Bounded Loss Rate

Loss rate is a critical parameter for a network-layer routing protocol. Higher layer protocols such as TCP are quite sensitive to packet loss in the underlying layers. A routing protocol that results in a high loss rate will experience greatly diminished TCP throughput [8, 17]. Other applications such as compressed video also exhibit relatively low tolerance to loss rate.

There is a fundamental trade-off in terms of overhead and reliability between the proactive and reactive routing components. SPR constantly maintains destination-directed DAG with multiple redundant paths. Consequently, it incurs low loss rates, as it can quickly find alternative routes in response to link-failures. The reactive protocol detects route-failures lazily, that is, only when attempting to send data packets, and hence can suffer from packet loss when links fail. However at low mobility, the reactive routing component offers small loss rate at very low cost. The proactive component, though effective, incurs fixed overhead in regions of low mobility. This motivates an adaptive strategy to pick the minimal sufficient proactive radii to guarantee a targeted loss rate without incurring excessive control overhead.

We propose an application-specific protocol, named SHARP-LR, for achieving a target loss rate specified by the user, while simultaneously minimizing packet overhead. This protocol operates by adjusting the proactive zone in response to perceived loss at the destination, so that the protocol does not experience loss greater than the targeted rate. SHARP-LR uses the packet loss measured at the destination as the driving metric for adaptation. The heuristic used here is to increase the zone radius by 1, if the perceived loss rate is *up_thresh* times more than the targeted loss rate and to decrease the zone radius by 1, if the perceived loss rate is less than *down_thresh* times the target. The proactive zone is then expanded or shrunk according to the chosen radius.

Controlled Delay Jitter

Delay jitter, measured as the variance of the inter-arrival time between packets [13], is a critical performance metric for many real-time applications. For instance, applications involving periodic transmission of sensor data, such as object tracking, as well as multi-media applications for transmitting uncompressed voice streams are highly sensitive to the variance in packet arrival times [13].

There is a trade-off between the overhead entailed by the two routing components and the delay performance they provide at different network conditions. SHARP's proactive component exhibits lower variance in latency in highly mobile networks. The reactive component reconstructs routes upon route failures resulting in a temporary halt of data flow. Consequently, frequent route-failures increase jitter. In low mobility scenarios, the reactive component offers the desired jitter performance at a very low overhead compared to the proactive component. This motivates an adaptation strategy to utilize the trade-off and provide good jitter performance at low overhead as the network conditions vary. Note that, in addition to route failures, jitter is also affected by congestion in the network. Currently, our protocol does not take into account

the effect of congestion on delay jitter.

Our third protocol, called SHARP-DJ, aims to achieve an application-specified delay jitter with minimal overhead. SHARP-DJ aims to pick the minimal sufficient proactive radii to guarantee a targeted jitter without incurring excessive overhead. SHARP-DJ employs a heuristic very similar to SHARP-LR. Packet inter-arrival times are tracked by the destination as packets are received. If the perceived jitter is more than *up_thresh* times the target value, the zone radius is increased by 1. Similarly, if the perceived jitter is less than *down_thresh* time the target value, the zone radius is decreased by 1.

7 Evaluation

In this section, we evaluate the three application-specific protocols that we introduced in the previous section. We examine the performance of SHARP-PO, SHARP-LR, and SHARP-DJ, and show that these protocols perform well in a variety of settings. We also provide the intuition justifying why they work well by examining a non-adaptive version of SHARP with statically selected, fixed values for the zone radius.

SHARP was implemented in GloMoSim [27], a scalable packet-level simulator with an accurate radio model. The many parameters necessary for a realistic simulation were derived from real-world applications. The MAC protocol and radio characteristics were set to approximately correspond to the Lucent WaveLAN^{TM} card with a data rate of 11 Mbps at 2.4 GHz radio frequency operating at 250m nominal transmission range. IEEE 802.11b was used as the MAC protocol. The internal parameters of the AODV protocol were set as suggested in [24]. For SPR, we used the value of 1 second for *beacon_interval* and the value of 2 for *beacon_loss*, which correspond to standard values suggested in hello protocols [24]. We used 5 seconds for *reconstruction_interval*, and 10 milliseconds for T and W in the construction protocol. The values of *up_thresh* and *down_thresh* were set to 1 and 1.5, respectively.

We performed simulations with several topologies. Here, we present simulations performed by distributing 200 nodes uniformly at random in an area of $1700m \times 1700m$ and 600 nodes in an area of $3000m \times 3000m$. Each simulation was run for a duration of 360 simulated seconds. The mobility in the environment was simulated using a random-waypoint mobility model, wherein each node randomly chooses a point in the field and moves towards it at a randomly chosen velocity. The node then pauses for a specified period at the destination before continuing the same pattern of motion. In our simulations, velocities ranged between 0 m/s and 20 m/s, while the pause time was set to 0 seconds, which corresponds to constant motion. We controlled the mobility rate by varying the number of mobile nodes in the network. A *mobility fraction*, F, of 0 corresponds to all stationary nodes, while a mobility fraction of 1 corresponds to all nodes in continuous motion.

We show data from two scenarios. In the *multi-destination* topology, we examined a network where 4 destinations are sent data packets from 1, 4, 7, and 10 sources respectively. The source and destination pairs were chosen randomly, and the relative popularity of the destinations approximated hot spots in the network. In the *single-destination* topology, we examined a single destination in isolation to provide the intuition behind the performance of the multi-destination case. A constant bit rate (CBR) generator drove the data traffic in our simulation. We set the data traffic rate to correspond to audio traffic at 16 kbps as described in [14]. Accordingly, each source sent packets of 256 bytes at a rate of 8 packets per second in the 200-nodes topology. However, in the 600-nodes case, we had to restrict the data rate to 2 packets per second due to scalability limitations of the simulator. The sources started transmitting from a time randomly chosen between 50 seconds and 100 seconds of the simulation, and stopped data transmission after 250 seconds.

Each data point corresponds to a mean of 30 repeated measurements with different seeds of the random number generator. We plot the 95% confidence interval as error bars on the graphs.

Minimal Packet Overhead

We first demonstrate the need for a hybrid algorithm by examining routing overhead for different zone radii, and show that the optimal point lies between purely proactive and purely reactive routing regimes. In the first experiment, we set the zone radius to a fixed value for the duration of each simulation. We vary the zone radius from 0 (corresponding to AODV) to 13 gradually and then increase it to a very high value to cover the entire network (marked as SPR in the figure). Figure 3(a) shows the total packet overhead when 5 sources send packets to a single destination in a network of 600 nodes. Figure 3(b) shows the total packet overhead for the multi-destination case, where 1, 4, 7, and 10 sources respectively send packets to the 4 different destinations in a 600-nodes network. The graphs also show the packet overhead for different values of mobility fraction between 0 (static network) to 1 (all nodes in motion).

These graphs validate the trade-off between reactive and proactive routing components inferred from the model presented in Section 5. The overhead of the reactive component gradually increases as the network becomes more mobile. The local-repair mechanism employed by the proactive routing component entails very

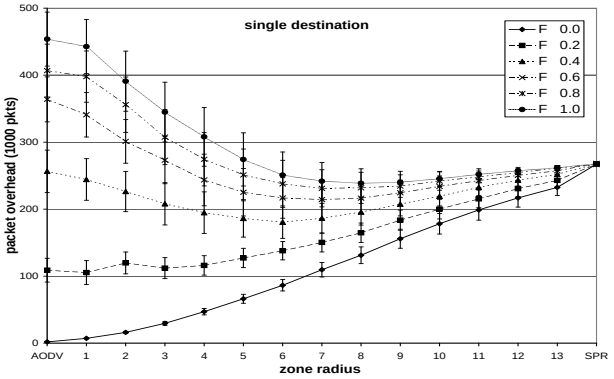

Figure 3(a): Total packet overhead vs. zone radius for a single destination with 600 nodes. This graph illustrates that the optimal routing strategy varies with mobility in the network.

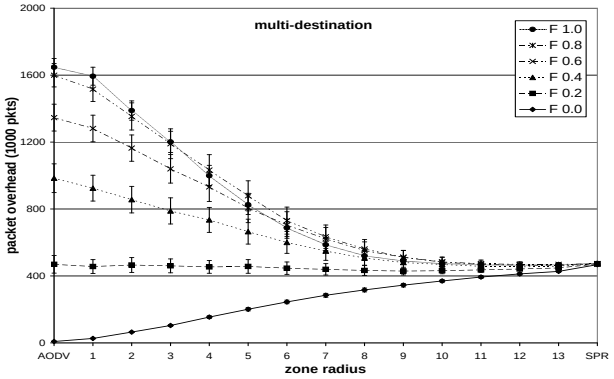

Figure 3(b): Total packet overhead vs. zone radius for multiple destinations with 600 nodes. This graph shows the trade-off between reactive and proactive components at different mobility rates.

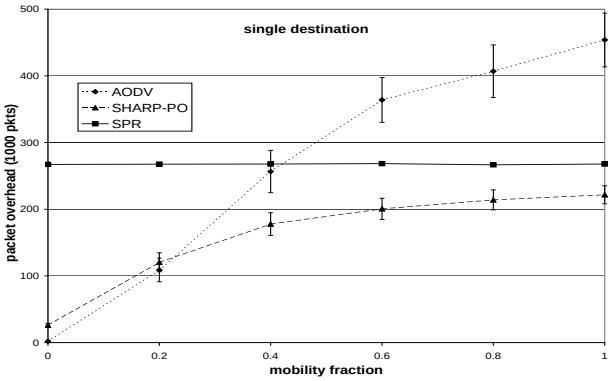

Figure 4(a): The routing protocol overhead vs. mobility for a single destination with 600 total nodes. This graph illustrates that SHARP-PO achieves comparable or better overhead than the proactive and the reactive components.

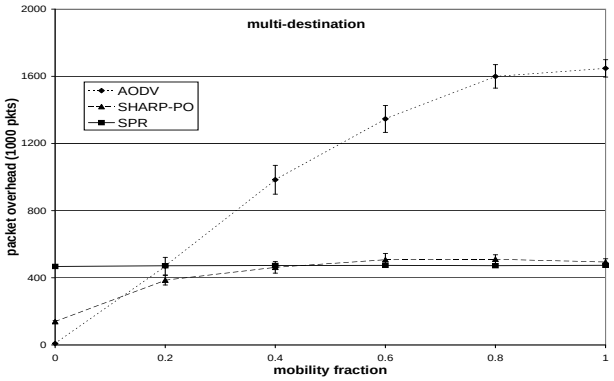

Figure 4(b): The routing protocol overhead vs. mobility for multiple destinations with 600 total nodes. This graph illustrates that SHARP-PO achieves comparable or better overhead than the proactive and the reactive components.

little communication, and therefore, does not contribute much to the total overhead. Consequently, the reactive component achieves low overhead when the mobility is low, while the proactive component incurs lower overhead when the mobility is high. Figure 3(a) shows that for high mobility, there are intermediate values of the zone radius where the packet overhead is less than both, the purely reactive and the purely proactive routing components. Thus, no single value of zone radius is the best choice for all levels of mobility. A protocol that dynamically adapts the zone radius with changing network conditions is necessary to achieve optimal performance.

Figures 4(a) and 4(b) show the performance of SHARP-PO, which adapts the zone radius dynamically to minimize packet overhead, in the single-destination and the multi-destination simulations with 600 nodes. These graphs illustrate how SHARP-PO adapts as the mobility in the network changes. In low mobility conditions, SHARP-PO follows the overhead of the reactive component, whereas in the high mobility conditions, SHARP-PO follows the overhead of the proactive routing component. By finding intermediate values of zone radii, SHARP-PO entails overhead lower than both the purely reactive and the purely proactive components (see Figure 4(a)). When the mobility is very low, the overhead of SHARP-PO is a little higher than AODV's overhead, because our current implementation of SHARP-PO always operates with a minimum zone radius of 1 in order to measure network characteristics.

Bounded Loss Rate

We next analyze the rate of packet loss in SHARP, as the zone radius and mobility are varied. Figure 5(a) shows the packet loss experienced by SHARP in a network of 600 nodes with multiple destinations receiving packets.

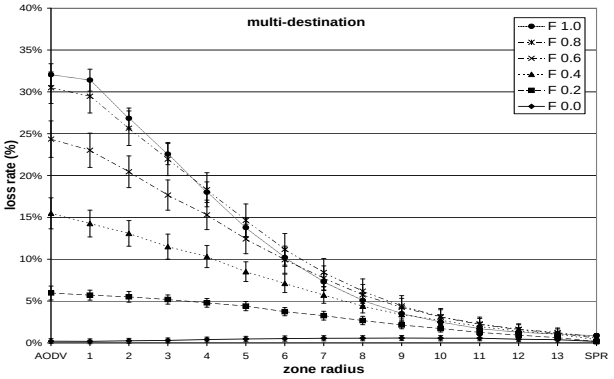

Figure 5(a): Packet loss rate vs. zone radius in the multi-destination topology with 600 nodes. This graph shows that proactive dissemination of routing information can reduce loss rate at the expense of overhead.

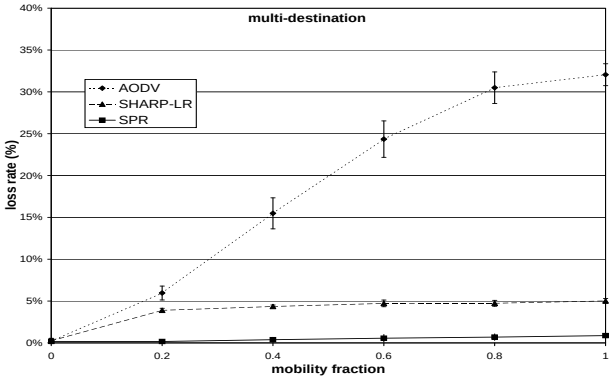

Figure 5(b): Packet loss rate vs. mobility for the multi-destination topology, with 600 nodes and a target loss rate of 5%. This graph illustrates that SHARP-LR bounds the loss rate by dynamically varying the zone radius.

At low mobility, the packet loss experienced is overall quite low. At high mobility, the loss rate incurred by the reactive component increases significantly, while the proactive component continues to provide good reliability. On the basis of loss rate alone, SPR appears to be a good choice for all scenarios. However, the overhead incurred by the routing components introduces a trade-off to be considered while picking the optimal point of operation. Figure 3(b) shows that the reactive routing component entails very low overhead and provides good loss rate during low mobility. The proactive routing component incurs low loss at all levels of mobility, but entails very high overhead at low mobility. These graphs validate the trade-off in terms of loss rate and packet overhead, which we observed earlier. A dynamically adapting protocol is necessary to capture this variation with network conditions.

Figure 5(b) shows the performance of SHARP-LR, the SHARP scheme to achieve a target loss rate, compared to the purely reactive and the purely proactive component in a network of 600 nodes with multiple destinations. The target loss rate was set to 5%. The graph shows that SHARP-LR is successful in attaining the set target for all levels of mobility considered. Figure 5(c) shows the packet overhead incurred by SHARP-LR during its operation. At low mobility, SHARP-LR chooses less proactivity and achieves the target loss rate with very low overhead. As the mobility increases, SHARP-LR increases the amount of proactivity to achieve the target loss rate. The overall cost of this scheme is comparable to the minimum of the purely proactive and the purely reactive components at all times. SHARP-LO experiences slightly more overhead than the purely proactive SPR, since it takes some time to choose the appropriate low overhead zone radius. SHARP-LR efficiently adapts the zone radius to guarantee the set target loss rate of 5%.

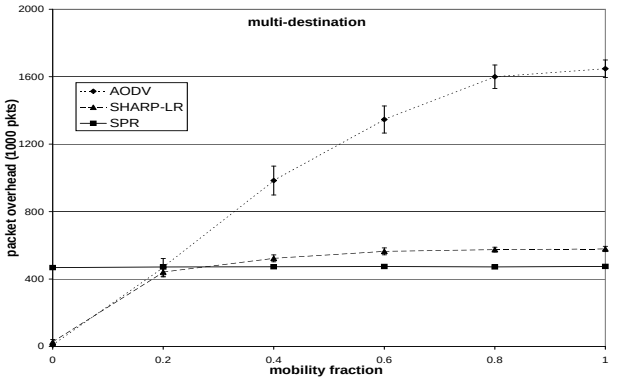

Figure 5(c): Packet overhead vs. mobility for the multi-destination topology, with 600 nodes and target loss rate of 5%. This graph shows that through dynamic adaptation, SHARP-LR utilizes the trade-off between loss rate and packet overhead to achieve its goal efficiently.

Controlled Delay Jitter

Figure 6(a) shows the average delay jitter of SHARP as the zone radius and the mobility fraction are varied for a network of 200 nodes with the 4 destinations receiving data at a rate of 8 packets per second from 1, 4, 7 and 10 sources respectively. The values of the delay jitter have been multiplied by the square of the data rate in order to normalize them. The figure shows that as the mobility increases, the packet inter-arrival times are less predictable for the reactive component. This behavior is caused by route maintenance performed to handle link-failures. By utilizing multiple available paths, the proactive component is able to provide very low values of delay jitter. Considering the performance in terms of delay jitter alone, SPR appears to be the best choice for routing in all scenarios. However, the overhead incurred by SPR in providing

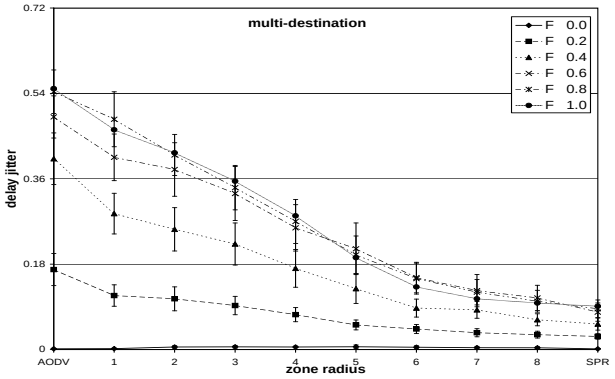

Figure 6(a): Delay jitter vs. zone radius for the multi-destination topology, with 200 nodes. This graph shows the trade-off in jitter entailed by the reactive and proactive routing components as the mobility in the network varies.

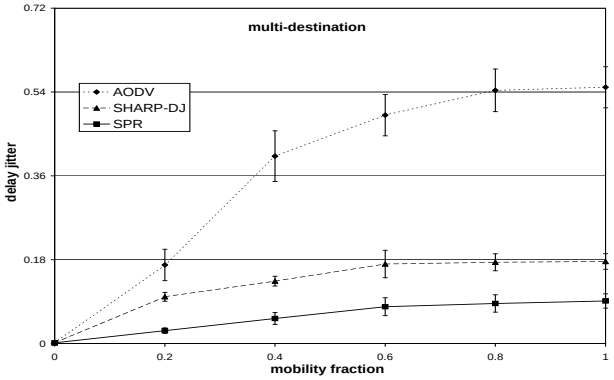

Figure 6(b): Delay jitter vs. mobility for the multi-destination topology, with 200 nodes and a target jitter of 0.18. This graph shows that SHARP-DJ adapts to network conditions to achieve the target delay jitter efficiently.

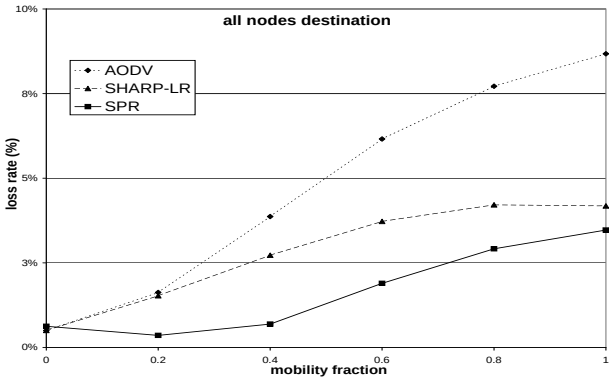

Figure 7(a): Loss rate vs. mobility when all nodes in a network of 80 nodes are destinations for a target loss rate of 5%. This graph shows that SHARP-LR adapts dynamically to network conditions to achieve the target loss rate.

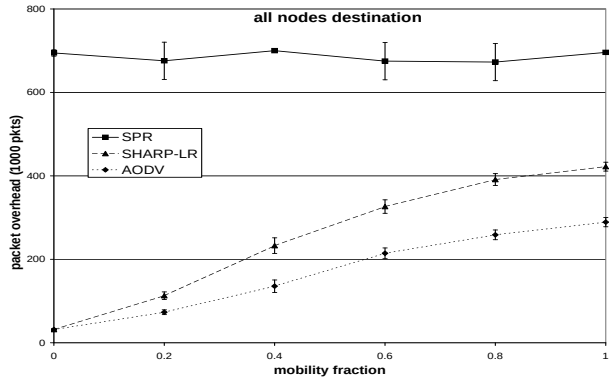

Figure 7(b): Packet overhead vs. mobility when all nodes in a network of 80 nodes are destinations for a target loss rate of 5%. This graph shows that SHARP-LR achieves its target without incurring excessive overhead.

low delay jitter introduces a trade-off that needs to be considered while picking the optimal point of operation. The reactive component provides good performance in terms of delay jitter at a much lower cost than the proactive component when the mobility rate is low. This validates the earlier observed trade-offs between delay jitter and overhead of reactive and proactive components and justifies the demand for a dynamically adapting protocol to control delay jitter by operating at an intermediate point.

SHARP-DJ provides good jitter performance at low cost for all simulated levels of mobility. Figure 6(b) shows the performance of SHARP-DJ for the same network topology, as it tries to achieve a target delay jitter of 0.18 (normalized). This graph shows that SHARP-DJ closely follows the performance of the reactive component when the mobility is low, but slowly increases the zone radius in order to satisfy the goal at high mobility. SHARP-DJ performs dynamic adaptation successfully to guarantee a user-defined target of delay jitter for different network conditions.

All Nodes Destinations

The previous sections showed that SHARP is efficient for networks in which the traffic is Zipf-like, that is, a few destinations have several sources sending data to them. Yet SHARP could also be invoked in the case where the traffic distribution is not Zipf-like. We studied the performance of SHARP in cases where all nodes serve as destinations. We simulated networks of 80 nodes with data traffic destined to all nodes from randomly chosen sources at a rate of 2 packets per second. The limited scalability of GloMoSim prevented us from simulating larger networks with increased traffic.

Figure 7(a) shows the loss rates achieved by AODV, SPR, and SHARP-LR for a target loss rate of 5%, when

all nodes are equally popular as route destinations. The reactive component incurs low loss rates at low mobility, but greater than 5% loss rates at high mobility. The proactive routing component achieves smaller than 5% loss rates at all levels of mobility. However, Figure 7(b) shows that it incurs a very high overhead. SHARP-LR balances this trade-off between loss rate and overhead and meets the target loss rate at all levels of mobility, incurring significantly lower overhead than the purely proactive component. These simulations demonstrate that SHARP can work well even when the traffic distribution follows a non-Zipf pattern.

Summary of Results

In this section, we have demonstrated the performance of three different adaptation schemes under varying network conditions. Our simulations show that there is a fundamental trade-off between reactive versus proactive dissemination of routing info, and that no fixed strategy, reactive or proactive, is suitable for a wide range of network conditions. This trade-off mandates the need for protocols to dynamically adapt based on network conditions. SHARP protocols dynamically find a good balance between these two routing strategies and outperform each one under varying network conditions. Further, they enable multiple nodes in the network to pursue different application-specific metrics at the routing layer.

8 Conclusions

This paper presents SHARP, a hybrid routing protocol that dynamically adapts to changing network characteristics and traffic behavior. SHARP is driven by the fundamental trade-off between proactive dissemination and reactive discovery of routing information. Our quantitative measurements demonstrate that no pure strategy is suited for all network conditions, and that the optimal routing strategy often lies somewhere in between the two extremes. SHARP is a hybrid protocol that can automatically find the balance point between these two strategies through an analytical model for making an informed trade-off and dynamic network measurements. It can act as a purely reactive protocol in a quiescent network, or use purely proactive routing for hosts to which routes are in wide demand. SHARP uses efficient mechanisms for dynamically manipulating the zone size and simultaneously performs fine-grained adaptation with low overhead. Further, SHARP enables applications with different demands to control the performance of the routing layer. We described how SHARP could be used to minimize packet overhead, to bound loss rate, and to control delay jitter. Our evaluation demonstrates that SHARP achieves performance that is better than each one of its concomitant purely reactive and purely proactive protocols across a wide range of network conditions. Overall, there is a large spectrum of design points between purely proactive and reactive protocols. SHARP enables applications to explore this space in networks, whose patterns of data traffic, node-degree, and mobility rates are subject to dynamic change.

References

[1] R. Boppana and S. Konduru. An Adaptive Distance Vector Routing Algorithm for Mobile, Ad Hoc Networks. In *IEEE Infocom 2001*, Mar 2001.

[2] J. Broch, D. Maltz, D. Johnson, Y. C. Hu, and J Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. *The Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, Oct 1998.

[3] S. Corson and V. Park. Temporally-Ordered Routing Algorithm (TORA) Version 1 Functional Specification. *IETF MANET Internet Draft*, Jul 2001 (work in progress).

[4] M. Crovella and A. Bestavros. Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes. In *Proceedings of SIGMETRICS 1996*, May 1996.

[5] Samir R. Das, Charles E. Perkins, Elizabeth M. Royer and Mahesh K. Marina. Performance Comparison of Two On-demand Routing Protocols for Ad hoc Networks. *in IEEE Personal Communications Magazine special issue on Ad hoc Networking*, Feb 2001, p. 16-28.

[6] Mario Gerla. Fisheye State Routing Protocol (FSR) for Ad Hoc Networks. *IETF MANET Internet Draft*, Jun 2002 (work in progress).

[7] Zygmunt J. Haas and Marc R. Pearlman. The Performance of Query Control Schemes for the Zone Routing Protocol. *IEEE/ACM Transactions on Networking*, Aug 2001.

[8] G. D. Holland and N. H. Vaidya. Analysis of TCP Performance over Mobile Ad Hoc Networks. In *Fifth Annual ACM/IEEE Conference on Mobile Computing and Networking (MOBICOM)*, Aug 1999.

[9] P. Jacquet, P. Muhlethaler and A. Qayyam. Optimized Link-State Routing Protocol. *IETF MANET Internet Draft*, Mar 2002 (work in progress).

[10] M. Joa-Ng and I-Tai Lu. Peer-to-peer Zone-based Two-level Link State Routing for Mobile Ad Hoc Networks. *IEEE Journal on Selected Areas in Communications*, 17(8):1415–1425, Aug 1999.

[11] D. Johnson, D. Maltz, Y Hu and J Jetcheva. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks. *IETF MANET Internet Draft*, Feb 2002.

[12] Leonard Kleinrock. Queuing Systems Volume 1: Theory, A Wiley-Interscience Publication, pp. 53-86, 1975.

[13] P. Mehta and S. Udani. VoIP: Sounding Good on the Internet. *IEEE Potentials Magazine*, pp. 36-40, Oct/Nov 2001.

[14] Art Mena and John Heidemann. An Empirical Study of Real Audio Traffic. In *IEEE INFOCOM*, Israel, Mar 2000.

[15] A. Nasipuri, R. Burleson, B. Hughes, and J. Roberts. Performance of a Hybrid Routing Protocol for Mobile Ad Hoc Networks. In *IEEE International Conference on Computer Communication and Networks (ICCCN2001)*, Phoenix, AZ, Oct 2001.

[16] S. Y. Ni, Y. C. Tseng, Y. S. Chen and J. P. Sheu. The Broadcast Storm Problem in a Mobile Ad Hoc Network. In *Proceedings of ACM/IEEE MobiCom*, Aug 1999.

[17] Mathis, Semke, Mahdawi and Ott. Macroscopic Behavior of the TCP Congestion Avoidance Algorithm. In *Computer Communications Review*, 27(3), 1997.

[18] Nikaein Navid, Labiod Houda, and Christian Bonnet. DDR: Distributed Dynamic Routing Algorithm for Mobile Ad hoc Networks. *First Annual Workshop on Mobile Ad Hoc Networking and Computing,2000*, Boston, 2000.

[19] Nikaein Navid, Wu Shiyi, and Christian Bonnet. HARP: Hybrid Ad hoc Routing Protocol. *International Symposium on Telecommunications, IST 2001*, 2001.

[20] M. Lewis, F. Templin, B. Bellur and R. Ogier. Topology Broadcast based on Reverse-Path Forwarding. *IETF MANET Internet Draft*, Nov 2002 (work in progress).

[21] V. D. Park and M. S. Corson. A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks. In *IEEE INFOCOM*, Japan, Apr 1997.

[22] Marc R. Pearlman and Zygmunt J. Haas. Determining the Optimal Configuration of the ZRP. *IEEE JSAC*, 17(6), Aug 1999.

[23] C. Perkins and P. Bhagwat. Highly dynamic destination sequenced distance vector routing for mobile computers. In *Proceedings of ACM SIGCOMM*, 24(4), Oct 1994.

[24] C. Perkins, E. M. Royer, and S. Das. Ad-Hoc On Demand Distance Vector (AODV) Routing. *IETF MANET Internet Draft*, Nov 2002 (work in progress).

[25] Elizabeth M. Royer and Charles E. Perkins. Ad-hoc On-Demand Distance Vector Routing. In *2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, Feb 1999.

[26] P. Samar, M. R. Pearlman, and Z. J. Haas. Hybrid Routing: The Pursuit of an Adaptable and Scalable Routing Framework for Ad Hoc Networks. The Handbook of Ad hoc Wireless Networks, CRC Press, 2002.

[27] Xiang Zeng, Rajive Bagrodia, and Mario Gerla. GloMoSim: A Library for Parallel Simulation of Large-scale Wireless Networks. In *12th workshop on Parallel and distributed simulation*, pages 154–161, Banff, Canada, May 1998.

Appendix

In this Section, we analyze the probability distribution of link-failure events and derive expressions to compute the following: the average frequency of route-failures in AODV and the average frequency of event-triggered updates in SPR. Section 5 employs these expressions to characterize the overhead of SHARP's reactive and proactive routing components.

Let the link formation and expiry be independent Poisson events with an exponential distribution of link lifetime. This exponential distribution has a probability density function of $\frac{1}{\lambda}e^{-t/\lambda}$, where $\frac{1}{\lambda}$ is the rate of link formation or expiry, and λ represents the mean link lifetime. Accordingly, the probability that no link-failures occur in a period of T seconds is given by $e^{-T/\lambda}$.

First, we shall derive the average frequency of route-failures in AODV. Consider a route between a source and a destination formed by h links. This route breaks if any of the h links fail, that is, the route lifetime is the minimum of the lifetimes of all its links. Assuming that the link-failures are independent Poisson events with a rate $\frac{1}{\lambda}$, and that they are identically distributed (iid), a route-failure can be considered to be a poisson event that is a merger of h independent link-failure events. Then, the route lifetime follows an exponential distribution with mean route lifetime of $\frac{\lambda}{h}$ [12]. The average rate of route-failure is given by $\frac{h}{\lambda}$.

Next, consider a node running SPR, which generates event-triggered updates upon the failure of all its downstream links. Let the node have n links connected to it, and let each link fail at a rate of $\frac{1}{\lambda}$. A link fails in T seconds with probability $1 - e^{-T/\lambda}$. Assuming that the link-failures are iid, all links fail in T seconds with probability $(1 - e^{-T/\lambda})^n$ and hence the probability that no repair is performed in T seconds is given by $1 - (1 - e^{-T/\lambda})^n$. Analyzing this probability distribution provides the mean time to repair activity of $\lambda \sum_1^n \frac{1}{i}$. Each node running SPR has 50% downstream links on an average. Since a breaking link could be upstream or downstream, the downstream links can be assumed to fail at a rate of $\frac{1}{2\lambda}$. Thus the average rate at which event triggered updates are generated by a SHARP node can be given by $\frac{1}{2\beta_n \lambda}$, where β_n is $\sum_1^n \frac{1}{n}$ and n is the number of downstream links.

A SHARP node running SPR may loose a downstream link for two reasons, namely, failure of the downstream link due to mobility, and reversal of the downstream link due to an event-triggered update at the downstream node. Let $A \rightarrow B$ be a downstream link at A whose downstream node is B. Let the downstream link fail at a rate of $\frac{1}{2\lambda}$ because of mobility, and let the downstream link be reversed by B at a rate of x_B. Thus the combined failure rate of the downstream link at node A is $x_B + \frac{1}{2\lambda}$. Note that, x_B is nothing but the rate at which event-triggered updates are generated by node B. Assuming that all nodes generate event-triggered updates at the same rate x on an average, the total failure rate of downstream links at node A is given by $x + \frac{1}{2\lambda}$. Thus the rate at which node A generates event-triggered updates is given by $\frac{x+\frac{1}{2\lambda}}{\beta_n}$.

However by assumption, the rate of generation of triggered updates at A is x. Thus, we can set up the following equation: $x = \frac{x+\frac{1}{2\lambda}}{\beta_n}$. Solving for x, we arrive at a combined average rate of $\frac{1}{2(\beta_n - 1)\lambda}$ at which a node looses all its downstream links due to link breaks and link reversals. This expression represents the rate at which event triggered updates are generated by a SHARP node running SPR.