

Quality of Service Routing in DSR

V. Aarthi and Arul Siromoney

School of Computer Science and Engineering

Anna University, Chennai 600 025, India

Email: asiro@vsnl.com (Contact email), aarthi_kalyan@hotmail.com

ABSTRACT

Ad hoc wireless networks consist of mobile nodes interconnected by multihop communication paths. Unlike conventional wireless networks, ad hoc networks have no fixed network infrastructure or administrative support. The topology of the network changes dynamically as mobile nodes join or depart the network or radio links between nodes become unusable. This article addresses some of the quality of service (QoS) issues for ad hoc networks that have recently started to receive increasing attention in the literature. We present the basic concepts and discuss the implementation of QoS routing in ad hoc networks, and in particular, dynamic source routing (DSR). We then provide the following enhancements: (a) introduction of ICMP QoS_LOST message to the source and (b) the caching of bandwidth information.

Index Terms: Mobile Ad hoc NETWORKS (MANETs), TDMA, DSR and Quality of Service (QoS).

1. INTRODUCTION

Conventional wireless networks require as prerequisites some form of fixed network infrastructure and centralized administration for their operation. In contrast, the so-called *ad hoc* wireless networks, consisting of a collection of wireless nodes, all of which may be mobile, dynamically create a wireless network among themselves without using any such infrastructure or administrative support. Ad hoc wireless networks are *self-creating*, *self-organizing*, and *self-administering* [4]. They come into being solely by interactions among their constituent wireless mobile nodes, and only such interactions are used to provide the necessary control and administration functions supporting such networks. The ad hoc wireless networks offer unique benefits and versatility for certain environments and certain applications. No preexisting fixed infrastructure, including base stations, being prerequisite, they can be created and used “any time, anywhere.” Such networks could be intrinsically fault-resilient, for they do not operate under the limitations of a

fixed topology. Indeed, since all nodes are allowed to be mobile, the composition of such networks is necessarily time varying. Addition and deletion of nodes occur only by interactions with other nodes; no other agency is involved. Such perceived advantages elicited immediate interest in the early days among military, police, and rescue agencies in the use of such networks, especially under disorganized or hostile environments, including isolated scenes of natural disaster and armed conflict. Numerous challenges must be overcome to realize the practical benefits of ad hoc networking. These include effective routing, medium (or channel) access, mobility management, power management, security, and, of principal interest here, *quality of service* (QoS) issues, mainly pertaining to delay and bandwidth management. Cost-effective resolution of these issues at appropriate levels is essential for widespread general use of ad hoc networking. The absence of fixed infrastructure means that the nodes of an ad hoc network communicate directly with one another in a peer-to-peer fashion. Therefore each node must be able to function as a router as well. The mobility of these nodes imposes limitations on their power capacity, and hence on their transmission range; indeed, these nodes often must satisfy stringent weight limitations for portability. As the nodes move in and out of range with respect to other nodes, including those operating as routers, the instantaneous topology changes must somehow be communicated to all other nodes as appropriate. In accommodating the communication needs of the user applications, the limited bandwidth of wireless channels and their generally hostile transmission characteristics impose additional constraints on how much administrative and control information may be exchanged, and how often. Ensuring effective routing is one of the great challenges for ad hoc networking.

All the challenges enumerated above are potential sources of service impairment in ad hoc networks, and hence may degrade the QoS seen by users of the network. As of now, the Internet has only supported best effort service; best effort in the sense that it will do its best to transport the user

packets to their intended destination, although without any guarantee. With the Internet as the basic model, the same has also been true for ad hoc networks, especially given their peculiar challenges compared to traditional wired or even conventional wireless networks. In recent years, however, QoS in ad hoc networks as a research topic has started to receive attention from a growing number of researchers.

RFC 2386 [5] characterizes QoS as a set of service requirements to be met by the network while transporting a packet stream from source to destination. Intrinsic to the notion of QoS is an agreement or a guarantee by the network to provide a set of measurable prespecified service attributes to the user in terms of trans-network delay, delay variance (jitter), available bandwidth, probability of packet loss, and so on. The Internet of today operates in a connectionless and stateless mode. The network of routers is not aware of any association between the source and destination except on a per-packet basis. Each packet is routed individually without any information about the state of the flow of packets between the source and destination. On the other hand, QoS is meaningful only for a flow of packets between the source and destination, and thus depends on the notion of a logical association, or logical *connection*, between them for the duration of the flow. Second, to attain and preserve the service attributes for such a logical connection, the network must guarantee the availability of a set of resources associated with the flow. Consequently, the routers must remain aware of the logical connection and state of the flow to ensure that adequate network resources (e.g., link bandwidth, nodal buffers, processing power) are available for the duration of the logical connection, and their underlying routes. QoS guarantees can be attained only with appropriate resource reservation techniques. The most important element among them is *QoS routing*, that is, the process of choosing the routes to be used by the flow of packets of a logical connection in attaining the associated QoS guarantee. QoS routing has been recognized as an important part in the evolution of QoS service offerings in the network. The goal of routing solutions is twofold: (1) satisfying the QoS requirements for every admitted connection, and (2) achieving global efficiency in resource utilization.

QoS routing can be divided into three broad categories: source routing, distributed routing, and hierarchical routing. In source routing, each node

maintains an image of the global network state, which is based on a routing path that is centrally computed at the source node. The global network state is typically updated periodically by a link-state algorithm. In distributed routing, the path is computed by a distributed computation during which control messages are exchanged among the nodes, and the state information kept at each node is collectively used in order to find a path. In hierarchical routing, nodes are clustered into groups, which are recursively clustered into higher-level groups, creating a multilevel hierarchy. In every level of the hierarchy, source or distributed routing algorithms are used. The QoS routing algorithms for wired networks cannot be applied directly to ad hoc networks. First, the performance of most wired routing algorithms relies on the availability of precise state information. However, the dynamic nature of an ad hoc network makes the available state information inherently imprecise. Second, nodes may join, leave, and rejoin an ad hoc network at any time and any location: existing links may disappear, and new links may be formed as the nodes move. Hence, the established paths can be broken at any time, which raises new problems of maintaining and dynamically reestablishing the routing paths in the course of data transmission. This paper describes some of QoS-based routing issues and implementation of QoS routing in DSR. We present an overview of the On-Demand QoS routing protocol proposed by Lin [2]. We also present the following enhancements: (a) the generation of ICMP QoS_LOST message and (b) the caching of bandwidth information.

The remainder of the paper is organized as follows. Section 2 gives an overview of DSR. Section 3 gives the bandwidth calculation algorithm. Section 4 describes on-demand QoS routing and finally in section 5, we present the enhancements to the on-demand QoS routing.

2. OVERVIEW OF DSR

Dynamic source routing [4] is based on source routing, where the source specifies the complete path to the destination in the packet header, and each node along this path simply forwards the packet to the next hop indicated in the path. It utilizes a route cache where source routes it has learned so far are cached. Therefore a source first checks its route cache to determine the route to the destination. If a route is found, the source uses this route. Otherwise the source uses a route discovery protocol to discover a route. A route is sought only when desired by a source. In route discovery, the

source floods a query packet through out the ad hoc network, and the reply is returned by either the destination or another host, which can complete the query from its route cache. Each query packet has a unique ID and an initially empty list. When receiving a query packet, if a node has already seen this ID (i.e. duplicate) or it finds its own address already recorded in the list, it discards the copy and stops flooding; otherwise, it appends its own address in the list and broadcasts the query to its neighbors. If a node can complete the query from its route cache, it may send a reply packet back to the source without propagating the query packet further. Furthermore, any node participating in route discovery can learn routes from passing data packets and gather this routing information into its route cache. This route discovery protocol is similar to the Internet's Address Resolution Protocol (ARP), except that ARP requests do not propagate beyond a router. It is also similar to the route discovery protocol used in source routing bridges in IEEE 802 LAN's.

A route failure can occur since mobile hosts move from place to place. A route failure can be detected by the link-level protocol (i.e. hop-by-hop acknowledgment), or it may be inferred when no broadcasts have been received for a while from a former neighbor. When a route failure is detected, the node detecting the failure sends an error packet to the source, which then uses route discovery protocol again to discover a new route. Note that in DSR, no periodic control messages are used for route maintenance. The major advantage of DSR is that there is little or no routing overhead when a single or few sources communicate with infrequently accessed destinations. In such situation, it does not make sense to maintain routes from all sources to such destinations. In DSR, only the sources, which desire to have communication with such destinations, need to discover routes. Furthermore, since communication is assumed to be infrequent, a lot of topological changes may occur without triggering new route discoveries (i.e. with little or no communication overhead).

3. QOS DEFINITION AND BANDWIDTH CALCULATION

Lin and Liu [1] proposed a new bandwidth routing scheme that contains bandwidth calculation and reservation for mobile ad hoc networks. In this protocol, the bandwidth information is embedded in the routing table. By exchanging the routing table, the end-to-end bandwidth of the shortest path for a given source-destination pair can be

calculated. If there is not enough bandwidth over the shortest path, the call request will be rejected. However, not enough bandwidth over the shortest path does not mean that there does not exist any bandwidth route in the network. Therefore, this protocol may miss some feasible bandwidth routes. In the protocol proposed by Lin [2], the focus is on finding a feasible bandwidth route. The routing optimality (e.g., shortest) is of secondary importance. That is, the bandwidth route obtained from this protocol may not be the shortest one.

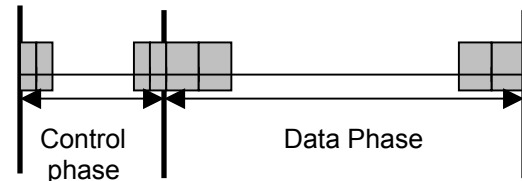


Figure 1 TDMA time frame structure.

The transmission time scale is organized in frames, each containing a fixed number of time slots. The entire network is synchronized on a frame and slot basis. Namely, time is divided into slots, which are grouped into frames. The frame/slot synchronization mechanism is not described here, but can be implemented in mobile ad hoc networks with techniques similar to those employed in the wired networks. Each time frame is divided into two phases: control phase and data phase. The size of each slot in the control phase is much smaller than the one in the data phase. The TDMA time frame structure is shown in Figure 1. The control phase is used to perform all the control functions, such as slot and frame synchronization, power measurement, code assignment, Virtual Circuit (VC) setup, slots request, etc. The amount of data slots per frame assigned to a VC is determined according to bandwidth requirement.

As depicted in Figure 1, the control phase uses pure TDMA with full power transmission in a common code. That is, each node takes turns to broadcast its information to all of its neighbors in a predefined slot, such that the network control functions can be performed distributedly. The information can be heard by all of its adjacent nodes. In a noisy environment in which the information may not always be heard perfectly at the adjacent nodes, an acknowledgment scheme is performed in which each node has to ACK for the last information in its control slot. By exploiting this approach, there may be one frame delay for the data transmission after issuing the data slot reservation. Ideally, at the end of the control phase, each node has learned the channel reservation status of the data phase. This information will help

one to schedule free slots, verify failure of reserved slots, and drop expired real time packets. The data phase must support both virtual circuit and datagram traffic. Since real time traffic (which is carried on a VC) needs guaranteed bandwidth during its active period, bandwidth must be preallocated to the VC in the data phase before actual data transmission. That is, some slots in the data sub frame are reserved for VCs at call setup time.

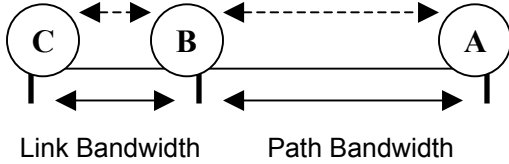


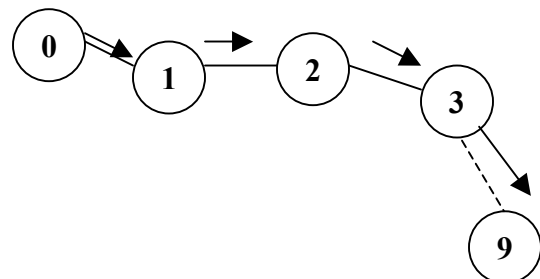
Figure 2 Bandwidth information Calculation

Because only adjacent nodes can hear the reservation information, and the network is multihop, the free slots recorded at every node may be different. The set of the common free slots between two adjacent nodes is defined as the *link bandwidth*. Consider the example shown in Figure 2 in which C intends to compute the bandwidth to A. We assume the next hop is B. By using end-to-end bandwidth calculation scheme [1], if B can compute the available bandwidth to A, then C can use this information and the “link bandwidth” to B to compute the bandwidth to A. The *path bandwidth* (or called *end-to-end bandwidth*) between two nodes, which are not necessary to be adjacent, is defined as the set of available slots between them. If two nodes are adjacent, the path bandwidth is the link bandwidth. Consider the example in Figure 2 and assume that A is one hop distance from B. If C has free slots {1, 3, 4}, and B has free slots {1, 2, 3}, then the *link bandwidth* between C and B is {1, 3}. This means that we can only exploit slot 1 and slot 3 for packet transmission from C to B. Thus, if a VC session needs more than two slots in a time frame, then it will be rejected to pass through (C, B).

Observe that $link_BW(P,Q) = free_slot(P) \cap free_slot(Q)$. $free_slot(X)$ is defined to be the slots, which are not used by any adjacent host of X to receive or to send packets, from the point of view at node X. Next, we can further employ link bandwidth to compute end-to-end bandwidth. This information can provide us an indication of whether there is enough bandwidth on a given route between a source-destination pair.

An illustration is given in Figure 3 to describe how to calculate the path bandwidth. In Figure 3, the source node (node 0) delivers packets to the

destination node (node 9) through node 1 to 8. We assume there are 10 data slots in the data phase. The notation “_” means the slot has been reserved and is not available. The $free_slot(0)$, for example, is {0, 2, 4, 6, 7, 8}, and $free_slot(1)$ is {0, 1, 3, 7, 8}. Obviously, $link_BW(1, 0) = path_BW(1, 0) \cap \{0,7,8\}$. $path_BW(2, 0)$ can be calculated from $link_BW(2, 1) \cap \{1, 7, 8\}$ and $path_BW(1, 0)$ according to the bandwidth calculation algorithm illustrated in [1], and is equal to {1,8}. Recursively, $path_BW(3, 0)$ can be obtained from $link_BW(3, 2)$ and $path_BW(2, 0)$, and is equal to {2,7}. Finally, $path_BW(9, 0)$ is got from $link_BW(9, 8)$ and $path_BW(8, 0)$, and is equal to {2, 5}. The main principle of computing the $path_BW(P,Q)$ is to consider the slots not in $path_BW(P, R) \cap link_BW(R,Q)$ first, and then the common ones of both sets. For example, $path_BW(0, 7) = \{2, 6\}$ and $link_BW(7, 8) = \{2, 3, 6, 7, 8, 9\}$. Thus, node 8 must choose slots from {3, 7, 8, 9} first to forward data from node 7 to the next hop, and node 7 can only choose slots from {2, 6} to transmit data to node 8. But the total number of slots chosen by each node has to be equal. In this case, node 8 chooses {3, 7} and node 7 chooses {2, 6} (actually, node 8 can choose any pair from {3, 7, 8, 9}, not necessary {3, 7}). That is, $path_BW(0, 8) = \{3, 7\}$. Finally, $path_BW(0, 9) = \{2, 5\}$. This means that node 9 can only reserve either data slot 2 or 5 or both for any new call from source node 0, even though node 9 is recording the slot {2, 4, 5, 6, 8} to be free currently. Thus, if the bandwidth requirement of a new call from node 0 to node 9 through the above path is more than two data slots per frame (say $QoS > 2$), then the admission control will reject this call. After calculating the end-to-end bandwidth, we need to reserve the data slots from the destination (node 9) hop-by hop backward to the source (node 0). If $QoS = 2$, node 9 reserves slot 2 and 5; node 8 reserves slot 3 and 7; node 7 reserves slot 2 and 6, etc. This reservation is not released until the end of the session. On completing the reservation, node 0 begins transmitting datagrams.



```

No 0 : 0 _ 2 _ 4 _ 6 7 8 _ -----> {}
No 1 : 0 1 _ 3 _ _ 7 8 _ -----> {0 7 8}
No 2 : _ 1 2 _ _ 6 7 8 _ -----> {1 8}
No 3 : 0 _ 2 3 _ 5 _ 7 8 9 -----> {2 7}
No 4 : 0 1 2 _ 4 5 6 7 8 _ -----> {0 5}
No 5 : 0 1 2 3 4 5 _ 7 8 9 -----> {1 2}
No 6 : 0 _ 2 _ _ 5 6 7 8 _ -----> {0 5}
No 7 : 0 _ 2 3 4 _ 6 7 8 9 -----> {2 6}
No 8 : 0 _ 1 2 3 _ 5 6 7 8 9 -----> {3 7}
No 9 : _ _ 2 _ 4 5 6 _ 8 _ -----> {2 5}

```

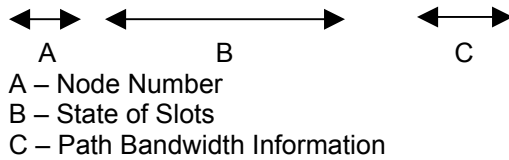


Figure 3 Bandwidth calculation and reservation example

In general, to compute the available bandwidth for a path in a time-slotted network, one not only needs to know the available bandwidth on the links along the path, but also needs to determine the scheduling of the free slots. To resolve slot scheduling at the same time as available bandwidth is searched on the entire path is equivalent to solve the Satisfiability Problem (SAT) which is known to be NP-complete. Here a heuristic approach is used to assign slots. In this bandwidth calculation algorithm, only the size of the path bandwidth is computed. Observe that the information of the end-to-end bandwidth is useful for admission control when a new VC session comes in the system. The admission control can immediately determine whether the VC traffic flow can be accepted at the beginning of connection request according to the bandwidth requirement and available path bandwidth. Actually, this QoS indication enables more efficient call admission control. It reduces the possibility of the failure of the call setup.

Lin [2] proposed a new protocol that focus on finding a feasible bandwidth route in the network. We use the same process used in [2] for performing route discovery. In Route Maintenance phase, we introduce ICMP QOS_LOST message to the source. When any node along the path detects that the requested bandwidth can no longer be maintained, then that node must originate ICMP QOS_LOST message back to the node, which had originally requested the now unavailable parameters. We also cache the bandwidth information for the corresponding destination. This

indicates the minimum bandwidth to the destination. It enables the intermediate node to respond to a later RREQ with a Minimum Bandwidth Extension by comparing the requested minimum bandwidth and the available bandwidth recorded in the cache.

4. THE ON-DEMAND ROUTING PROTOCOL WITH BANDWIDTH CONSTRAINT

We give an overview of the on-demand routing protocol proposed by Lin [2]. The design principle of the on-demand QoS routing is as follows.

Route Discovery

Like Ad-hoc On-demand Distance Vector routing (AODV) and Dynamic Source Routing (DSR) this protocol conforms to a pure on-demand rule. No routing table is maintained and hence there is no periodic exchange of information. When a source node wants to communicate with another node for which it has no routing information, it floods a route request (RREQ) packet to its neighbors. If the topology exists, a route from the source to the destination, RREQ will find (record) it. All packets contain following uniform fields:

<packet_type, source_addr, dest_addr, sequence#, route_list, slot_array_list, data, TTL>

All packet types, defined, are shown in Table 1. The <source_addr, sequence#> pair is used to uniquely identify a packet. This *sequence#* is monotonically increasing, which can be used to supersede stale cached routes. *route_list* records the routing information; *slot_array_list* records the status of slot assignment on the route. When any host receives a RREQ, it will perform the following operations:

1. If the pair <source_addr, sequence#> for this RREQ was seen recently, discard this redundant request packet and do not process it further.
2. Otherwise, if the address of this node appeared in the *route_list* in the RREQ, drop this RREQ (do not rebroadcast it) and do not process it further.
3. Otherwise,
 - (a) Calculate the bandwidth from the source to this node following the algorithm given by [2] and record the status of the available data slots to the *slot_array_list*. This RREQ is dropped if the result does not satisfy the QoS requirement, and do not process it further. It is worth noting that the state of the data slots is not modified at this time.

- (b) Decrement TTL by one. If TTL counts down to zero, drop this RREQ and do not process it further. TTL can limit the length of the delivery path. There may exist a very long path that satisfies the bandwidth requirement. However, this path will be difficult to be maintained within a dynamic environment. In addition, unlimited packet flooding will deteriorate the network performance. The use of TTL can control the flooding traffic.
- (c) Append the address of this node to the *route_list* to track the route, which the packet has traversed, and re-broadcast this request if this node is not the destination. The packet is cached before re-broadcast

Packet Type	Function
ROUTE_REQUEST (RREQ)	Send to discover route
ROUTE_REPLY (RREP)	Send to reserve route
RESERVE_FAIL	Nack for unsuccessful reservation
ROUTE_BROKEN	Nack for route broken
CLEAN_RREQ	Clean surplus RREQs
NO_ROUTE	Nack for finding no route
DATA	Use to transport datagram

TABLE 1

Each component in the *slot_array_list* contains the host ID and the set of available time slots. As is to be expected, a destination node will receive more than one RREQ. Every RREQ packet indicates a unique feasible QoS path from the source to the destination. Thus, the destination node can keep more than one path. Multiple connectivity between a source–destination pair can provide a more robust packet delivery. This is especially important in a multihop mobile network. In order to reduce the overhead of flooding, the destination node can broadcast a *CLEAN_RREQ* packet to clean RREQ packets that are still roaming around the network after receiving enough paths.

Route Reservation

When the destination node receives one RREQ packet from the source node, it returns a route reply (RREP) packet by unicasting back to the source following the route recorded in the

route_list. As a RREQ travels from the source to the destination, it automatically sets up the reverse path from the destination back to the source. To set up a reverse path, a node records the address of the neighbor from which it received the copy of the RREQ. From the RREQ packets, the state of the data slots can be obtained. According to the information recorded within the RREQ, the destination can set up a bandwidth route and reserves resources (slots) hop-by-hop backward to the source. Using the source routing algorithm, copy the fields *<route_list, slot_array_list>* from RREQ to RREP. As the RREP traverses back to the source, each node along the path reserves those free slots that were calculated in advance. When the source receives a RREP, the end-to-end bandwidth reservation is successful, and the VC is established. Then, the source node can begin transmitting datagrams. It is notable that this establishment protocol for a VC connection from the source to the destination is two-way handshaking. When a new call request arrives, the call admission control drives this establishment protocol. This new call will not be accepted until the reservation process is successfully completed.

Unsuccessful Reservation

When the RREP travels back to the source, the reservation operation may not be successful. This may result from the fact that the slots that need to be reserved were occupied a little earlier by another VC or the route breaks. The interrupted node sends a NACK (i.e., *RESERVE_FAIL*) back to the destination, and the destination re-starts the reservation process again along the next feasible path (note that in the route discovery process, each RREQ which arrives at the destination piggybacks a feasible bandwidth route). All nodes on the route from the interrupted node to the destination must free the reserved data slots when receiving *RESERVE_FAIL*. If there is no VC that can be setup along all feasible bandwidth routes, the destination broadcasts another NACK (i.e., *NO_ROUTE*) to notify the source. Upon receiving *NO_ROUTE*, the source can either re-start the discovery process if it still requests a route to the destination, or reject the new call. In addition to *NO_ROUTE* arrival, if there is no response back to the source before the timeout occurs, the source can also re-perform the route discover operation. Once a VC is established, the source can begin sending datagrams in the data phase. At the end of the session, all reserved slots must be released. These free slots will be contended by all new connections.

Connection Breakage

During the active period of a connection, a topological change may destroy a VC. The connection control must reroute or re-establish the VC over a new path. When a route is broken, the breakpoints send a special NACK (i.e., *ROUTE_BROKEN*) to the source and the destination. That is, once the next hop becomes unreachable, the breakpoint, which is near the source, sends an unsolicited NACK to the source, and the other breakpoint does to the destination. Each node along the path relays this *ROUTE_BROKEN* to its active neighbors and so on. Furthermore, they release all reserved slots for this connection and drop all data packets of this connection, which are still waiting for sending in the queue. Upon receiving the *ROUTE_BROKEN*, the source re-start the discovery process to re-establish a VC over a new path, and the destination only drops the *ROUTE_BROKEN* (the main purpose of the travel of the *ROUTE_BROKEN* from the breakpoint to the destination is to release the reserved slots). This procedure is repeated until either the completion of data delivery or timeout. If timeout occurs, the source stops any data delivery for this session. If a link on the VC is broken before the completion of a session, the last data packet may be still on the way to the destination. This packet, thus, cannot reach the destination, and is suspended within an intermediate node. In this situation, some resources are still occupied by this connection and cannot be used by the others. In order to solve this problem, the timeout scheme is used for each reserved slot. If a reserved slot is not used to deliver data packets for a couple of data frames and timeout occurs, this slot is freed automatically. Such free slots will be fully utilized by the other new sessions.

5. ENHANCEMENTS TO ON-DEMAND QOS ROUTING PROTOCOL

We propose the following enhancements to the protocol proposed by Lin [2].

1. Caching the bandwidth for future use.
2. Generating ICMP QoS_LOST message.

As a part of the RREQ we include the Quality of Service parameter that is Bandwidth that a route to destination must satisfy. The RREQ and RREP packet contains the following fields.

<packet_type, source_addr, dest_addr, sequence#, route_list, slot_array_list, bandwidth, data, TTL>

Minimum Bandwidth Extension

In a RREQ message this field indicates the minimum bandwidth (in Kbits/sec) that must be available along an acceptable path from source to destination. In a RREP message this indicates the minimum bandwidth available in the route from the node forwarding the RREP to the destination.

The Minimum Bandwidth Extension can be appended to a RREQ by a requesting node in order to specify the minimum amount of bandwidth that must be made available along an acceptable path from the source to the destination.

Before forwarding the RREQ, an intermediate node must compare its available link capacity to the Bandwidth field. If the requested amount of bandwidth is not available, the node must discard the RREQ and not process it any further. Otherwise, the node continues processing the RREQ.

When the destination generates a RREP in response to a RREQ, the Bandwidth field in the RREP is initially infinity (a very large number). Each node forwarding the RREP compares the Bandwidth field in the RREP and its own link capacity and maintains the minimum of the two in the Bandwidth field of the RREP before forwarding the RREP.

Full use of the Route Cache

The RREP packet is cached before forwarding to the neighboring nodes. The data in a host's route cache can be stored in any format, but the active routes in its cache in effect form a tree of routes, routed at this host, to the other hosts in the ad hoc network. Caching is done for the following reasons:

1. For example, Figure 4 shows an ad hoc network with four mobile hosts, in which mobile host A has earlier completed a route discovery for the mobile host D through B and C. Since mobile hosts B and C are on the routes to D, host A also learns the route to both of these hosts from the discovery for D.
2. A host can add entries to its route cache any time it learns a new route. In particular, when a host forwards a data packet as an intermediate hop on the route in that packet, the following host is able to observe the entire route in that packet. If a host forwards a route reply packet, it can also add the route information from the route record being returned in that route reply, to its own route cache.
3. A host may use its route cache to avoid propagating a route request packet

received from another host. In particular, suppose a host receives a route request packet for which it is not the target and is not already in its list of recently seen requests; if the host has a route cache entry for the target of the request, it may append its cached route to the accumulated route record in the packet, and may return this route in a route reply packet to the initiator without propagating (re-broadcasting) the route request. This enables the intermediate node to respond to a later RREQ with a Minimum Bandwidth Extension by comparing the requested minimum bandwidth and the available bandwidth recorded in the cache.

4. A host may also notify the source nodes with an ICMP QoS_LOST message in case there is a change in link capacity at this node by comparing the bandwidth field entered in the cache with the available bandwidth.

B-C-D

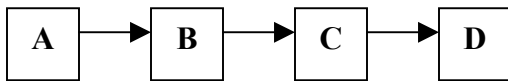


Figure 4 An example of ad hoc network illustrating the route cache.

ICMP QoS LOST Message

ICMP is Internet Control Message Protocol that is used by hosts, routers, and gateways to communicate network layer information to each other. ICMP is specified in RFC 792 [6]. The most typical use of ICMP is Error Reporting. For example, when running a Telnet, FTP or HTTP session, we may have encountered an error message such as “ Destination Network Unreachable”. This message had its origin in ICMP.

ICMP is often considered as a part of IP, but architecturally lies just above IP, as ICMP messages are carried inside IP packets. That is, ICMP messages are carried as IP payload just as any TCP or UDP segments are carried as IP payload. The format of ICMP message is

<Type, Destination Address>

Type refers to the Type of message and the Destination Address, which is the address of the destination node using the link for which there has been a change in bandwidth.

ICMP QoS_LOST message has been proposed earlier for QoS in AODV [3]. We extend this in this

paper to DSR and caching of bandwidth in DSR. An ICMP QoS_LOST message is generated when an intermediate node experiences a decrease in link capacity. This may affect the QoS guarantees previously provided to some sources. The Minimum Bandwidth Extension is used when there is a drop in link capacity and the change in bandwidth is indicated in the Bandwidth field.

The QoS_LOST message is forwarded to all sources potentially affected by the change in the bandwidth. These are those sources to which a RREP with a QoS extension has been forwarded before. When any intermediate node finds that there is no available bandwidth along the existing path then, it will perform the following operations:

1. The interrupted node sends an ICMP QoS_LOST message to the original source and the source re-starts the reservation process again along the next feasible path.
2. The interrupted node also sends a RESERVE_FAIL packet to all the nodes on the route from the interrupted node to destination. On seeing the RESERVE_FAIL packet all the nodes along the route that is from the interrupted node to the destination should free the reserved data slots so that other nodes could utilize the data slots.

If there is no VC that can be set up along all feasible bandwidth routes, the destination broadcasts another NACK (i.e., NO_ROUTE) to notify the source. Upon receiving NO_ROUTE, the source can either re-start the discovery process if it still requires a route to the destination or reject the new call.

6. CONCLUSIONS

In summary, we have presented the following enhancements (a) introduction of ICMP QoS_LOST message to the source and (b) the caching of bandwidth information, to the on-demand routing protocol which is suitable for use with multihop mobile networks. During the route discovery process, the route request (RREQ) packets are used not only to find paths between the source-destination pair, but also to calculate bandwidth hop by hop. If there is not enough bandwidth to satisfy the bandwidth requirement at any intermediate node, the route is dropped. Thus, when a RREP packet arrives at the destination, the route piggybacked on the RREP packet must have satisfied the end-to-end bandwidth requirement. However, the route may not be the shortest in hop length. In the route reply process, the route

reservation is made hop-by-hop backward from the destination to the source. Caching of Bandwidth enables the intermediate node to respond to a later RREQ with a Minimum Bandwidth Extension by comparing the requested minimum bandwidth and the available bandwidth recorded in the cache. An ICMP QoS_LOST message is generated when an intermediate node experiences a decrease in link capacity. This helps the source to maintain the QoS guarantee that is previously provided.

References

1. C. R. Lin and J. -S. Liu "QoS Routing in Ad hoc Networks", IEEE Journal in Selected Areas in Communications. Vol. 17 No. 8 Aug 1999, pp. 1426 – 1438.
2. Chunhung Richard Lin "On-Demand QoS Routing in Multihop Mobile Networks", Proceedings of IEEE Infocom 2001 Vol.3 April 2001, pp. 1735 – 1744.
3. Charles E. Perkins, Elizabeth M. Royer, Santa Barbara, Samir R. Das, "Quality of Service for Ad hoc On-Demand Distance Vector Routing" Mobile Ad Hoc Networking Working Group, IETF Internet Draft (draft-ietf-manet-aodvqos-00.txt).
4. D. Johnson and D. Maltz, "Dynamic source routing in ad hoc wireless networks," Mobile Computing, E. Imielinski and H. Korth, Eds. Norwell, MA: Kluwer, 1996.
5. E. Crawley, R. Nair, B. Rajagopalan, H. Sandick, "A Framework for QoS-based Routing in the Internet", Networking Group, Request for Comments : 2386 August 1998.
6. J. Postel, RFC 792 " Internet Control Message Protocol", Network Working group.
7. J. Broch, D.Maltz, D. Johnson, Y. -C. Hu and J. Jetcheva " A performance comparison of multi-hop wireless ad hoc network routing protocols", In proceedings of IEEE/ACM MOBICOM '98 pp 85-97, October 1998.