

# Distributed Bandwidth-Constrained Routing for Data Streaming Across Multiple Routes for Improving Call Blocking Rate and Channel Utilization in Wireless Ad Hoc Networks<sup>\*</sup>

CHUN-HUNG RICHARD LIN AND YI-SIANG HUANG

*Department of Computer Science and Engineering  
National Sun Yat-Sen University  
Kaohsiung, 804 Taiwan*

In wireless networks, bandwidth is likely to remain a scarce resource. We foresee scenarios wherein mobile hosts will require simultaneous data transfer across multiple links to obtain higher overall bandwidth. A call request of a connection in a wireless network is blocked if there exists no bandwidth route. This blocking does not mean that the total system bandwidth capacity is less than that requested, but that there is no path in which each link has enough residual unused bandwidth to satisfy the requirement. Like routing in a datagram network, if packets of a virtual circuit can stream across multiple paths, we can select multiple bandwidth routes such that the total bandwidth can meet the requirement of a source-destination pair. Therefore, even though there is no feasible single path for a bandwidth-constrained connection, we may still have a chance of success this one if we can find multiple bandwidth routes to meet the bandwidth constraint. In this paper, we propose a bandwidth-constrained routing algorithm to aggregate the bandwidth of multiple wireless links by splitting a data flow across multiple paths at the network layer. That is, the algorithm allows the packet flow of a source-destination pair to be delivered over multiple bandwidth routes with enough overall resources to satisfy a certain bandwidth requirement. Our algorithm considers not only the QoS requirement, but also the cost optimality of the routing paths to improve the overall network performance. Extensive simulations show that high call-admission ratios and resource utilization are achieved with modest routing overheads. This algorithm can also tolerate node moving, joining, and leaving.

**Keywords:** ad hoc network, quality-of-service, multi-path, routing, blocking rate

## 1. INTRODUCTION

In recent years, the level of interest in wireless ad hoc networks has grown along with the range of wireless communication devices, such as laptop computers, PDAs, and Bluetooth devices. A wireless ad hoc network [1, 3] can be a collection of wireless mobile hosts forming a temporary network without an infrastructure. Each mobile host acts as a router, forwarding data packets for other nodes. Because of the dynamic topology, it is difficult to design a routing protocol. The routing protocols in wireless ad hoc networks can be categorized into two types. One is table-driven (e.g., DSDV [18]) and the

---

Received January 31, 2003; accepted July 4, 2003.

Communicated by Ming-Syan Chen.

<sup>\*</sup> A preliminary version of this paper has been presented at the 2002 International Computer Symposium, 2002.

other is on-demand (e.g., DSR [10], AODV [19], TORA [17], LMR [6]). However, both types are only deal with best-effort traffic. Connections that need a quality-of-service (QoS) requirement are not supported.

In wireless ad hoc networks, due to their dynamic nature and lack of centralized admission control, it is more difficult to establish a QoS connection [12, 13, 16] than it is in cellular networks. In [15], the authors proposed a QoS routing protocol based on a table-driven routing protocol that adds bandwidth information to routing tables. Like a table-driven protocol, it periodically exchanges routing information, and when a connection request arrives, looks up the routing table to decide on the next hop. There is one difference in that it establishes a route, which satisfies the QoS requirement. Similarly, the work in [14] is based on an on-demand routing protocol that adds bandwidth information to Route\_Request (RREQ) messages and broadcasts messages initialized by source nodes to their neighboring nodes. Finally, RREQ messages reach destination nodes. Then, one satisfied QoS requirement is picked, and Route\_Response (RREP) message is sent along the picked route to the source node. Both protocols are simple and available, but they also inherit the drawback of the overhead involved in maintaining a routing table in table-driven routing protocol even though there are no connections in the network and broadcasting RREQ messages in an on-demand routing protocol.

In [5], the authors proposed a ticket-based probe (TBP) scheme to search a QoS route. A ticket represents permission to search one path. The source node issues a number of tickets based on the QoS requirement. Probes are sent from the source to the destination to search for a low cost path that satisfies the QoS requirement. At the intermediate node, a probe with more than one ticket is allowed to be split into multiple ones, each of which will search a different downstream sub-path. An intermediate node splits the tickets according to the local state and the end-to-end information, which is updated periodically by a distance-vector protocol. If all the probes arrive at the destination, it will select a single path that has the lowest cost and satisfies the QoS requirement. The goal of designing tickets is to limit the number of messages to be sent. However, the work in [5, 14], and [15] are for single path routing.

In bandwidth-constrained routing, bandwidth utilization of a single path QoS routing protocol is low, because the residual bandwidth of the links can not satisfy the new connection request. It is essential to effectively use resource in wireless ad hoc networks, since there are limited. In this paper, we propose a bandwidth-constrained routing protocol that improves bandwidth utilization by searching for a multi-path to satisfy the QoS requirement. This protocol includes a scheme for searching multi-path, splitting algorithms, and route management.

The rest of the paper is organized as follows. The system models are given in section 2. The concept of multi-path routing is explained in section 3. Simulation results are presented in section 4. Finally, section 5 concludes the paper and describes future work.

## 2. SYSTEM MODEL

In our system model, we partially use the one described in [5]. The system model is as follows:

#### a. Ad Hoc Network Model

One can represent a network using a graph  $G = (V, E)$ , where  $V$  is a set of nodes that are interconnected by a set  $E$  of full-duplex directed communication links. A link  $(a, b)$  means that node  $a$  considers node  $b$  as a valid neighbor for packet forwarding and adjusts transmitting power according to their distance, and vice versa. That is, they are in transmission range of each other. Assume that there are neighbor discovering and MAC protocol in every node. Especially in MAC protocol, it is needed to resolve the media connection, resource reservation, and reorder the out-of-order packets in our model.

#### b. QoS State Metrics

A QoS connection (call) is a connection that has an end-to-end performance requirement, such as a delay or bandwidth constraint. The QoS metrics we consider here are the bandwidth only. This is because bandwidth guarantee is one of the most critical requirements for real time applications. We assume that every node has precise information about its local state. A node  $i$  keeps the up-to-date local state of all outgoing links. The state information of link  $(i, j)$  includes: 1)  $b_{ij}$ , the residual (unused) bandwidth of the link; and 2)  $c_{ij}$ , the transmission cost, which can be simply one as a hop count or a function of link utilization. The bandwidth and cost of a path  $p = i \rightarrow j \rightarrow \dots \rightarrow k \rightarrow l$  are defined as follows:

$$b(p) = \min \{b_{ij}, \dots, b_{kl}\},$$

$$c(p) = c_{ij} + \dots + c_{kl}.$$

The main objective of the routing algorithms presented here is to set up connections (i.e., virtual circuits) in a network when sessions are initiated, and to maintain them during the lives of the sessions. Like RSVP [22], the set-up process includes reservation of network resources for the connection; if these resources are not available the set-up fails, or in other words, the connection is blocked. Our aim is to minimize the blocking probability for a connection.

#### c. Routing Problem

Given a source-destination pair and a bandwidth requirement  $BW$ , the problem of bandwidth-constrained routing is to find a feasible path  $p$  between the source and the destination such that  $b(p) \geq BW$ . When multiple feasible paths exist, finding the least-cost path is an NP-complete problem. The routing decision influences the blocking probability. For example, if two routing algorithms differ in the length of the selected routes and, thus, in the consumption of network resources, usually, the probability of successfully finding sufficient resources for more connections is lower the algorithms that waste resources. Thus, the quality of the routing decision also affects the blocking probability in the network.

The bandwidth of a wireless link is quite limited and unstable. For a network to deliver QoS guarantees, it must reserve and control resources. A virtual circuit (VC) can be accepted if it not only has enough available bandwidth, but also can not disrupt the existing QoS VCs. Otherwise, it will be blocked. Sometimes, a network has enough bandwidth capacity to accept the traffic of a new VC. However, the VC may still be blocked because there is no existence of a bandwidth route. If the packets of VC traffic are allowed to travel along different paths like routing for datagram traffic, then we can con-



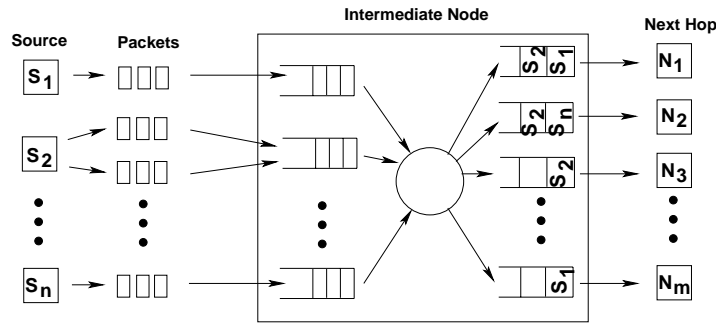


Fig. 2. Concept of splitting in the transport layer or network layer.

### 3. BANDWIDTH-CONSTRAINED ROUTING

#### 3.1 Route Discovery

We use a request/reject scheme to discover multiple routes for QoS routing in an ad hoc network. Initially, a source node sends a route request message with a split bandwidth requirement to each possible next-hop node, which has available bandwidth for routing, based on the node's local routing table and a splitting algorithm. The splitting strategy and algorithms for determining the bandwidth requirement of each candidate will be further explained in section 3.3. Assume that an intermediate node receives a request message; it will do its best to search possible next-hop nodes from its local routing table to forward the request message. If there is no route or if there are routes but insufficient bandwidth to satisfy the bandwidth requirement, the intermediate node will send a reject message back to its upstream node. Consider that an intermediate node receives a reject message; it must try to search the remaining routes, which may satisfy the bandwidth requirement because the aggregate bandwidth of all the routes is larger than the bandwidth requirement specified in the request message. In this way, the intermediate nodes endeavor to set up multiple routes for the QoS routing request until a connection with the target is successfully established. Provided that the source node receives a reject message and cannot find enough bandwidth from the remaining routes to balance the bandwidth request in the reject message, a route connection failure message will be returned to the upper application layer.

Fig. 3 shows a typical flow diagram for route discovery using the proposed request/reject scheme. Assume that node  $s$  wants to establish a QoS connection with 10 units with bandwidth to a target node  $t$ ; node  $s$  initially sends a request with 10 units of bandwidth to its next-hop neighbors. The request message is denoted as  $req_{st}(10)$ . Observe that when node  $i$  receives the  $req_{st}(10)$  message, the residual bandwidth of each link between node  $i$  and its neighbor nodes can not satisfy the QoS requirement, but the aggregation of the residual bandwidth of each link can satisfy the QoS requirement. Therefore, node  $i$  splits the bandwidth requirement into two requests, e.g.,  $req_{st}(4)$  and  $req_{st}(6)$ , and sends them to their corresponding neighbors, e.g., node  $f$  and node  $j$ . Afterwards, node  $j$  receives the message  $req_{st}(6)$  and wants to forward the message to node  $k$ . However, node  $k$  knows that there is no residual bandwidth available in each link between

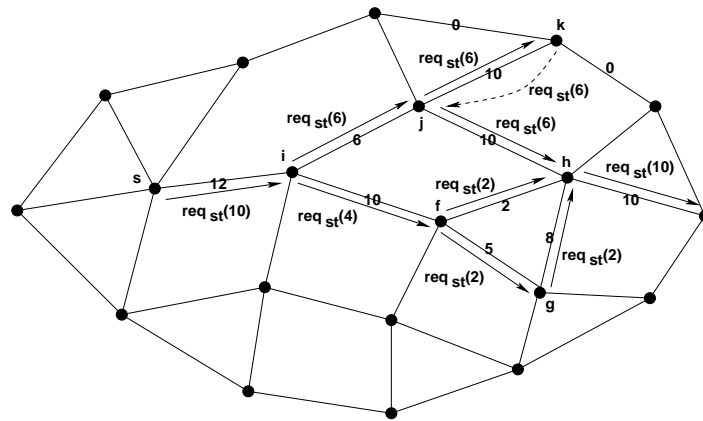


Fig. 3. Topology of an ad hoc network.

node  $k$  and its neighbors. It seems impossible to forward the request message to its neighbor nodes, so node  $k$  returns a reject message  $rej_{st}(6)$  back to node  $j$ . Therefore, node  $j$  tries to re-forward the  $req_{st}(6)$  message to node  $h$ . Since node  $h$  has 12 units of residual bandwidth, node  $h$  can forward the requests from nodes  $f$ ,  $g$ , and  $j$  to node  $t$ . The QoS connection between node  $s$  and node  $t$  is eventually established and maintained until all the request messages successfully arrive at node  $t$  within a predetermined timeout value.

To prevent a cycle from forming during the route-discovery stage, all traversed nodes are recorded in the route request message. The set of traversed nodes is specified in the path field of the message format shown in Table 1. A node must discard a received message provided that the node's ID already exists in the traversed node list. Hence, the route discovery scheme can guarantee loop-free message forwarding.

Table 1. Data structure.

Parameters	Description
id	a unique system identification for the connection request
type	request or reject message
s	source node ID
t	destination node ID
B	initial bandwidth requirement
B'	current bandwidth requirement
path	a list of nodes that have been traversed so far
rejected	a set of nodes that have been rejected for the request
cost	accumulated cost of the path traversed so far

### 3.2 Message Format

The format of a request/reject message is shown in detail in Table 1. The reject field in the message format records the set of nodes that have been rejected for this connection

request. To avoid repeatedly forwarding the request message to a node that has previously been rejected, a node must discard a received message along with its node's ID in the 'rejected' field. The cost field represents the accumulated cost of the routing path so far. The cost can be replaced by other QoS constraints [8], such as delay time, jitter level, or hop count.

### 3.3 Splitting Algorithms

Assume that a QoS connection is required to be established between a source node  $s$  and a target node  $t$ . The connection request message  $m$  is initially sent from  $s$ . The request message that arrives at an intermediate node  $i$  is denoted as  $m_i$ .  $N_i$  denotes the set of neighbors of node  $i$ . The bandwidth requirement of the received message is denoted as  $BW_i^m$ .  $T_m$  is the list of nodes that  $m_i$  has traversed so far.  $R_m$  is the set of nodes that  $m_i$  has rejected so far.  $F_i^m$  denotes a set that contains all the forwarding candidates of node  $i$  for request message  $m_i$ . Thus,  $F_i^m = N_i - T_m - R_m$ .  $l_{ij}$  denotes the link between node  $i$  and  $j$ . The residual bandwidth and cost of  $l_{ij}$  are represented by  $b_{ij}$  and  $c_{ij}$  respectively.

Fig. 4 shows an example in which node  $E$  forwards a request message with a bandwidth requirement of 10 units to node  $A$ . After receiving the request, node  $A$  has to determine which nodes in its set of neighbors should be selected as forwarding candidates and how many units of bandwidth need to be allocated to each selected link. In this example, there are three nodes in the set of neighbors of node  $A$ .  $(b, c)$  along with each edge in the figure means the residual bandwidth and the cost of the link respectively. With respect to the example,  $F_i^m = \{B, C, D\}$ .

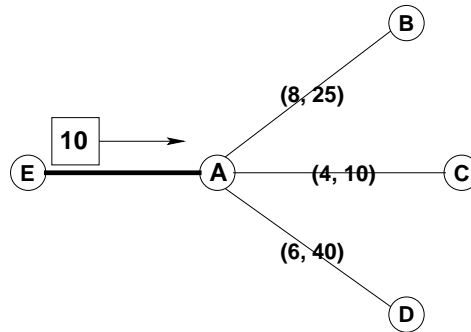


Fig. 4. An example of forwarding request message.

#### 3.3.1 Algorithm A

Before splitting the request message, algorithm A sorts all the possible forwarding nodes in descending order based on the residual bandwidth  $b_{ij}$  for each node  $j, j \in F_i^m$ . Hence, an ordered set  $F_i^m$  can be obtained. Let  $B_i$  be the corresponding residual bandwidth of each node in  $F_i^m$  and  $B_i = \{b_{i1}, b_{i2}, \dots, b_{in}\}$ , where  $n = |F_i^m|$ , and  $b_{i1} \geq b_{i2} \geq \dots \geq b_{in}$ . In order to limit the number of splitting routes, the nodes from the beginning of

$F_i^m$  are selected as forwarding nodes such that  $BW_i^m = \sum_{j=1}^k b_{ij}$ , where  $1 \leq k \leq n$ .

```

/* Algorithm A */
bandwidth_requirement =  $BW_i^m$ 
descending_sort_by_b( $F_i^m$ );
j = first node ID in  $F_i^m$ ;
while (bandwidth_requirement > 0)
{
  split_bandwidth =  $b_{ij}$ ;
  if (split_bandwidth > bandwidth_requirement)
    split_bandwidth = bandwidth_requirement;
  bandwidth_requirement -= split_bandwidth;
   $BW_j^m$  = split_bandwidth;
  j = next node ID of j in  $F_i^m$ ;
}

```

Regarding the example shown in Fig. 4, the possible forwarding neighbors of node  $A$  are sorted according to the residual bandwidth of each node. Thus, we can obtain  $F_i^m = \{B, D, C\}$  and  $B_A = \{8, 6, 4\}$ . According to the algorithm, the connection request message with  $BW_A^m = 10$  that arrives at node  $A$  is split into two messages by firstly choosing node  $B$  with  $b_{AB} = 8$  and secondly choosing node  $D$  with  $b_{AD} = 6$ . The bandwidth requirement assigned to  $l_{AB}$  is 8 units; thus, the bandwidth requirement assigned to  $l_{AD}$  is simply  $10 - 8 = 2$  units. Therefore, the first split request with bandwidth requirement of 8 is forwarded to node  $B$ . The second split request with a bandwidth requirement of 2 is forwarded to node  $D$ . The results of request splitting and forwarding are shown as small squares with arrows in Fig. 4 (a).

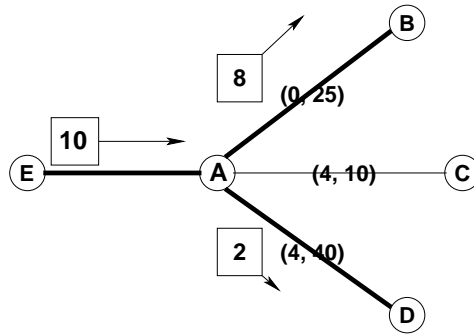


Fig. 4 (a). A example of forwarding a request message using splitting algorithm A.

### 3.3.2 Algorithm B

Algorithm B splits the connection request message by choosing the least-cost links between a splitting node and its neighbors. Assume that the cost of each link is represented by  $c_{ij}$  for all  $j \in F_i^m$ . First of all, the set of forwarding nodes  $F_i^m$  is sorted in ascending order based on the corresponding costs of each node. The set of cost for nodes in  $F_i^m$  is denoted by  $C_i$ . Let  $C_i = \{c_{i1}, c_{i2}, \dots, c_{in}\}$ , where  $n = |F_i^m|$  and  $c_{i1} \leq c_{i2} \leq \dots \leq c_{in}$ . In

order to obtain the minimal least-cost routes, the algorithm chooses forwarding nodes from the beginning of  $F_i^m$  such that bandwidth requirement is satisfied by  $BW_i^m = \sum_{j=1}^k b_{ij}$ , where  $1 \leq k \leq n$ .

```

/* Algorithm B */
bandwidth_requirement =  $BW_i^m$ 
descending_sort_by_c ( $F_i^m$ );
j = first node ID in  $F_i^m$ ;
while (bandwidth_requirement > 0)
{
    split_bandwidth =  $b_{ij}$ ;
    if (split_bandwidth > bandwidth_requirement)
        split_bandwidth = bandwidth_requirement;
    bandwidth_requirement -= split_bandwidth;
     $BW_j^m$  = split_bandwidth;
    j = next node ID of j in  $F_i^m$ ;
}
    
```

From Fig. 4, the sorted set of all possible forwarding neighbors is  $F_i^m = \{C, B, D\}$  and the cost set corresponding to  $F_i^m$  is  $C_A = \{10, 25, 40\}$ . Based on algorithm B, a connection request message with  $BW_A^m = 10$  that arrives at node A is split into two messages by firstly choosing node C with  $b_{AC} = 4$  and then choosing node B with  $b_{AB} = 6$ . The bandwidth requirement allocated to  $l_{AC}$  is 4, and the bandwidth requirement allocated to  $l_{AB}$  is  $10 - 4 = 6$  units. Therefore, one split request with a bandwidth requirement of 4 is forwarded to node C. Another split request with a bandwidth requirement of 6 is forwarded to node B. The results of request splitting and forwarding are shown in Fig. 4 (b).

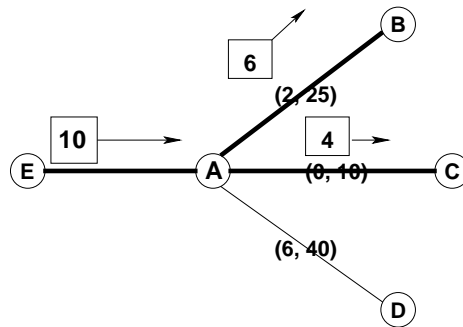


Fig. 4 (b). An example of forwarding a request message using splitting algorithm B.

### 3.3.3 Algorithm C

Let  $q_{ij}$  be the bandwidth-cost ratio, defined as a new QoS metric for route-splitting algorithm C. Let  $q_{ij} = \frac{b_{ij}}{c_{ij}}$ , for all  $j \in F_i^m$ . The algorithm sorts all the forwarding nodes

in descending order based on the  $q_{ij}$  metric of each node  $j$ . Let  $Q_i$  represent the set of the bandwidth-cost ratios corresponding to all the nodes in  $F_i^m$ , and  $Q_i = \{q_{i1}, q_{i2}, \dots, q_{in}\}$ , where  $n = |F_i^m|$  and  $q_{i1} \geq q_{i2} \geq \dots \geq q_{in}$ . In order to limit the number of splits and to obtain a better QoS connection, the algorithm selects forwarding nodes from the beginning of set  $F_i^m$  such that  $BW_i^m = \sum_{j=1}^k b_{ij}$ , where  $1 \leq k \leq n$ .

```

/* Algorithm C */
bandwidth_requirement =  $BW_i^m$ 
descending_sort_by_q ( $F_i^m$ );
j = first node ID in  $F_i^m$ ;
while (bandwidth_requirement > 0)
{
  split_bandwidth =  $b_{ij}$ ;
  if (split_bandwidth > bandwidth_requirement)
    split_bandwidth = bandwidth_requirement;
  bandwidth_requirement -= split_bandwidth;
   $BW_j^m$  = split_bandwidth;
  j = next node ID of j in  $F_i^m$ ;
}

```

In Fig. 4 (c), the 3-tuple  $(b, c, q)$  represents the residual bandwidth, cost, and bandwidth-cost ratio for a link. In the example,  $F_A^m = \{C, B, D\}$  and  $Q_A = \{0.4, 0.32, 0.15\}$ . According to algorithm C, a connection request message with  $BW_A^m = 10$  arrives at node A is split into two messages by firstly choosing node C with  $b_{AC} = 4$  and then choosing node B with  $b_{AB} = 8$ . The bandwidth requirement split to  $l_{AC}$  is 4 and the bandwidth requirement split to  $l_{AB}$  is  $10 - 4 = 6$  units. Therefore, the first split request with a bandwidth requirement 4 is forwarded to node C, and the second split request with bandwidth requirement of 6 is forwarded to node B. The results of request splitting and forwarding are shown in Fig. 4 (c).

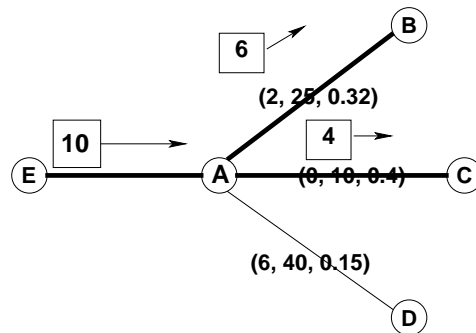


Fig. 4 (c). An example of forwarding a request message using splitting algorithm C.

### 3.3.4 Algorithm D

In this algorithm, a bandwidth requirement is split onto all possible links by means of a weighted distribution. Thus, there will be  $n$ -splitting routes, where  $n = |F_i^m|$ . Like algorithm A,  $F_i^m$  denotes the set of all the possible forwarding neighbors of node  $i$ , and  $B_i$  represents the set of residual bandwidth of each neighbors of node  $i$ . Algorithm D initially sorts all the possible forwarding nodes in descending order based on the residual bandwidth  $b_{ij}$  of each node  $j$ . Then, the algorithm chooses every node in  $F_i^m$  one by one and calculates the splitting bandwidth for each node  $j$ . The maximum amount of bandwidth split onto each node  $j$  is denoted as

$$BW_j^m = \left[ \frac{b_{ij}}{\sum_{k \in F_i^m} b_{ik}} \cdot BW_i^m \right]$$

```

/* Algorithm C */
bandwidth_requirement = BW_i^m
descending_sort_by_b ( F_i^m );
j = first node ID in F_i^m ;
while (bandwidth_requirement > 0)
{
    split_bandwidth =  $\left[ \frac{b_{ij}}{\sum_{k \in F_i^m} b_{ik}} \cdot BW_i^m \right]$ ;
    if (split_bandwidth > bandwidth_requirement)
        split_bandwidth = bandwidth_requirement;
    bandwidth_requirement -= split_bandwidth;
    BW_j^m = split_bandwidth;
    j = next node ID of j in F_i^m ;
}

```

From Fig. 4,  $B_i = \{8, 6, 4\}$  and  $F_i^m = \{B, D, C\}$  can be obtained. According to the algorithm, a connection request message with  $BW_A^m = 10$  that arrives at node  $A$  will be split into three messages. The sequence to choose nodes from  $F_i^m$  is node  $B, D, C$  one by one. The split bandwidth of each node is calculated as 5, 4, 1, respectively. Therefore, the first split request with a bandwidth requirement of 5 is forwarded to node  $B$ . The second split request with a bandwidth requirement of 4 is forwarded to node  $D$ . Finally, the third split request with a bandwidth requirement of 1 is forwarded to node  $C$ . The results of request splitting and forwarding are shown in Fig. 4 (d).

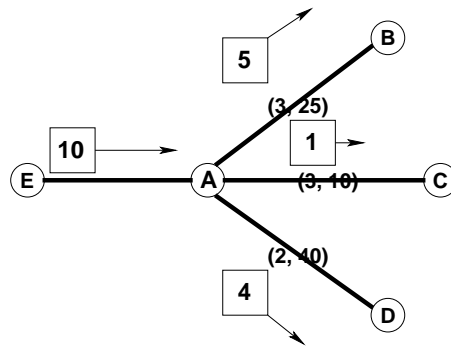


Fig. 4 (d). An example of forwarding a request using splitting algorithm D.

### 3.4 Rerouting

Every node in ad hoc network is mobile, and a link between two mobile nodes is liable to be interfered with by noise or to be blocked by barriers. Thus, a link may be disconnected due to a long period of disturbance or obstacle. Once the link becomes disconnected, the path has to be reconstructed as soon as possible. During the reconstructing period, the QoS requirement can hardly be guaranteed. Past researches [5, 14, 15] revealed that three approaches can be utilized to reconstruct the broken path: path rerouting, path redundancy, and path repair. Path rerouting is the most common approach among these three for dealing with the problem of path breaking and reconstruction in an ad hoc network.

Assume that a path is established, and that a link of the path is then broken abruptly during real-time communication. The upstream intermediate node will detect the broken link and buffer the forwarding packets in its queue. Then, it will initiate a path rerouting function. The scheme for path rerouting is similar to that for normal routing in that it uses request and reject messages. First, the node sends a request message to each forwarding neighbor chosen from  $F_i$  of all possible forwarding nodes by the splitting algorithms described in the previous section. Provided that some other routes exist besides the broken one, newer connections will be reestablished. Then, the blocked node can resume forwarding the buffered data packets along the new links. Thus, the rerouting scheme is started at the node that detects the broken link, not at the source node. Note that the buffered packets may be dropped when the buffer is full. This problem can be solved by designing an adequate buffer length. If the time for path rerouting is too long to satisfy the QoS constraint, then the connection will be abandoned based on a pre-determined time-out value.

### 3.5 Soft State

Once a node accepts a request message, it also reserves the required resource for the request. Normally, the reserved resource corresponding to the request should be released by the node when it receives a reject message. However, every node is essentially considered to be mobile, the topology of an ad hoc network is highly dynamic, an established routing path may be broken, or the network may be partitioned. Therefore, releasing the reserved resource becomes a problem.

Usually, there are two ways to solve this problem. In one approach, during a connection, an intermediate node sends a release message to each of its downstream node to release all the reserved resources along the path toward the destination when it detects that the link between it and its upstream node is broken. This approach will add an extra load to the ad hoc network by generating a large number of release packets. The other approach is to use the soft state to automatically release a reserved resource after the resource has been unused for a designated period of time. In our simulation, we simply adopt the second method to avoid forwarding too many packets.

In [5], in order to manage paths in soft state, every node in the ad hoc network maintains a connection table. In our paper, we need to modify the table slightly and there are two differences: the definitions of incoming and outgoing entries. Because of multi-path route, there are many sub-paths pass a node, and the number of incoming and outgoing links is different. Therefore, we record them in two sets respectively. An entry in a connection table is described in Table 2. The TTL value of an entry will be refreshed while the data packet of the connection arrives at the node. When the TTL counts down to zero, the entry will be deleted from the table and the reserved resource will be released, too.

**Table 2. Connection table.**

<b>Entry</b>	<b>Description</b>
id	a unique system identification for the connection request
s	the source node
t	the destination node
incoming	the set of links from which the node receives packets
outgoing	the set of links to which the node forwards packets
resource	the reserved resource for this connection
TTL	Time-to-Live

#### 4. SIMULATION AND RESULTS

The network topology used in our simulation is randomly generated. Thirty nodes are randomly placed within a  $10 \times 10$  square area. The radius of the service range of each node is 3.5 normalized units. If the distance between two nodes is smaller than the service range, a link is added between them. The source-destination pairs with a bandwidth requirement are randomly generated. The bandwidth requirement is uniformly distributed over the range [5, 15]. The cost of each link is uniformly distributed over [1, 200].

Five algorithms were applied in the simulations. The first one was a single-path routing algorithm, and the other four were the multi-path routing algorithms described in the previous section. The single-path routing algorithm is similar to the request/reject scheme that multi-path routing algorithms use, except that there is not request splitting in the single-path routing algorithm. If there is no link satisfying the bandwidth requirement of a connection request, the connection will be rejected and declared to be a failure by the single-path routing algorithm. The detailed operations of the four multi-path routing algorithms can be found in the section covering the splitting algorithms.

#### 4.1 Bandwidth Utilization

A connection will last for a certain period of time, which is randomly generated by means of an exponential distribution with a mean equal to 3 units of time. Once the connection is established by the administrator module, the source node continuously sends data packets through the connection to the destination node during the connection period. As time passes after the simulation starts, the number of connections increase and the network reaches a saturated state, in which case no more connections are allowed to be created. In order to observe the performance of each routing algorithm, we define bandwidth utilization as follows:

$$\text{bandwidth utilization} = \frac{\text{total bandwidth of links in use}}{\text{total bandwidth of all links}}.$$

Fig. 5 shows the bandwidth utilization of each routing algorithm. As expected, the bandwidth utilization of the multi-path routing algorithms is better than that of the single-path routing algorithm. This is because the multi-path routing algorithms can allow multiple forwarding routes for the connection, provided that the total bandwidth of the multiple routes can satisfy the bandwidth requirement of the request, which may be larger than the residual bandwidth of any single link, in which case the single-path routing algorithm rejects the connection request. The bandwidth utilization shown by the curves in the figure reaches the maximum while the network is in a saturated state.

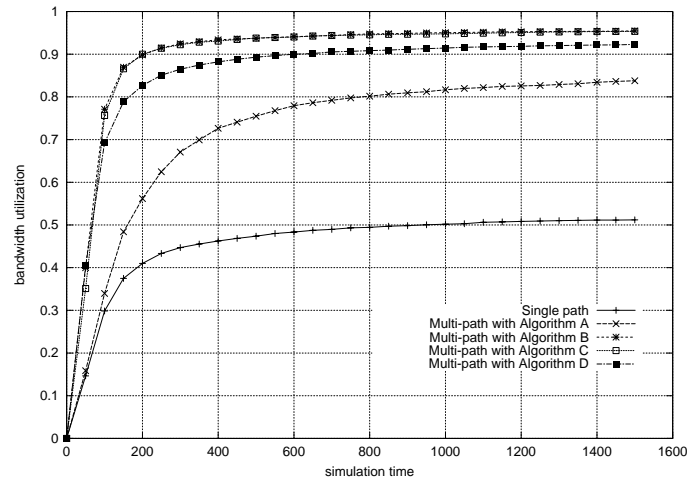


Fig. 5. Bandwidth utilization.

#### 4.2 Splitting Observations

Fig. 6 compares the splitting percentages for all nodes which have more than one split child. In Fig. 6, the percentages increase as the average bandwidth requirement increases, except for the single path routing algorithm, which does not split any request.

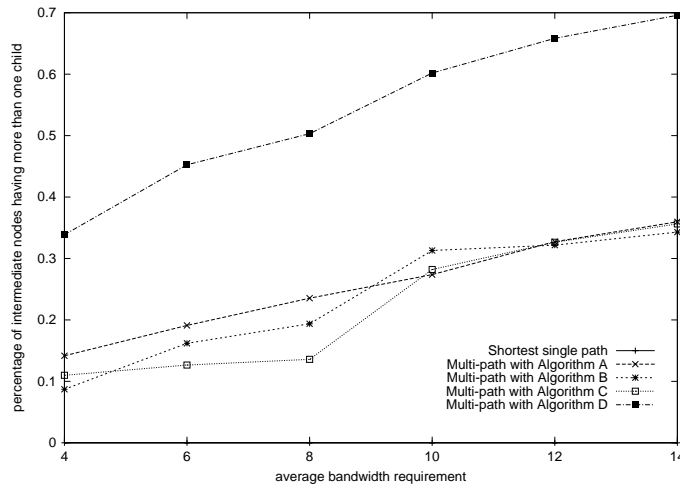


Fig. 6. The number of nodes that have more than one child.

The splitting percentage of algorithm D is the highest one. This is because algorithm D adopts a weighted distribution for all forwarding links. In contrast, algorithms A to C keep the number of their forwarding links to a minimum, and they have an aggregate bandwidth equal to the requirement. Therefore, the percentages of algorithms A to C are close to each other.

Fig. 7 compares the number of children that each splitting node has. Since the single-path routing algorithm does not split any request, it is omitted from this figure. In Fig. 7, the number of split children for multi-path routing algorithms A, B, and C is about 2.2, whereas that for algorithm D is 3.3. This is because algorithm D splits more links for the same request, by applying a weighted distribution to all possible links, in contrast to algorithms A, B, and C, which using the smallest number of routes to provide bandwidth exactly equal to the requirement.

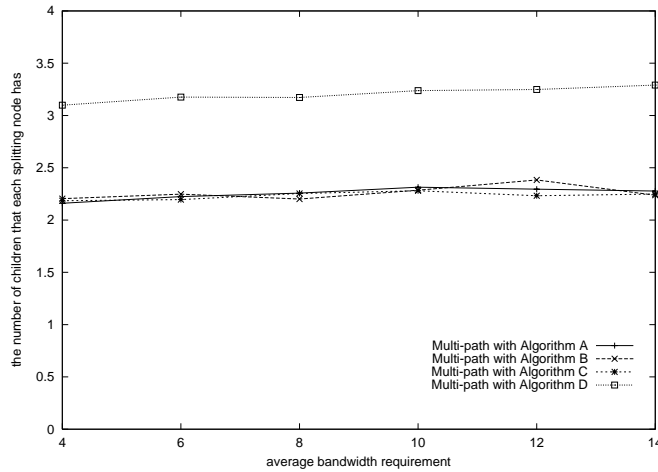


Fig. 7. Average number of children that each splitting node has.

### 4.3 Mobility Observations

The way-point model of mobility is adopted in our simulation. Each node stays at its current position for a period of time, and then it randomly selects a new position to move to. When a node detects a broken link of a connection, it blocks data packets in its queue buffer and tries to reroute the connection. The queue length of every node is set to be 100 packets. Data packets begin to be dropped when the buffer is full. Provided that the connection can not be re-established within a pre-determined time-out value, the connection is forced to termination by the administrator module. The forced terminated ratio is defined as follows:

$$\text{forced terminated ration} = \frac{\text{number of connections forced to terminate}}{\text{total number of established connections}}.$$

In the simulation, the moving rate is generated by an exponential distribution with a mean value. The pause time is uniformly distributed over a range of [1, 20] time units. The purpose is to observe the outcome of the forced termination ratio as the average moving rate is gradually increased. Here, we only compare the forced termination ratio of the single-path routing algorithm with that of multi-path routing algorithm C. Fig. 8 shows that the forced termination ratios of both algorithms increase as mobility increases. Moreover, the forced termination ratio of the multi-path routing algorithm is higher than that of the single-path routing algorithm. The reason is that the route established by the multi-path routing algorithm passes more links, which increases the probability of link breakage due to higher mobility.

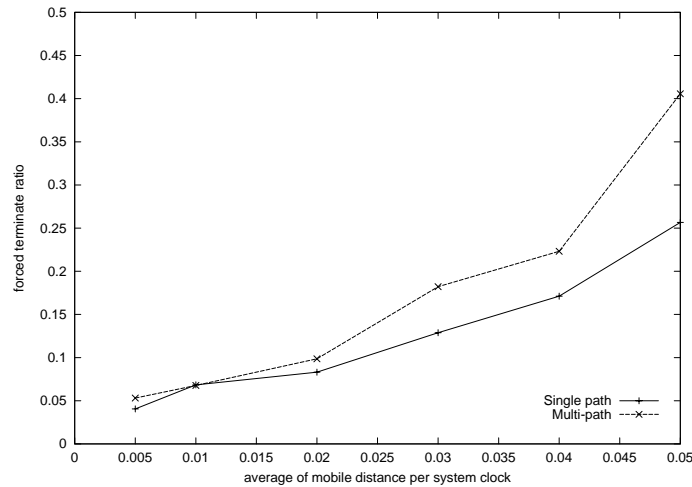


Fig. 8. Forced terminated ratio.

Note that the multi-path routing algorithms have better bandwidth utilization and can accommodate more connections in the case of a static ad hoc network, especially when the ad hoc network is in a traffic-saturated state. It is very important to fully utilize

the bandwidth of each link without wasting any piece in a bandwidth restricted wireless network. However, the performance of the algorithms will deteriorate due to a high forced termination ratio of connections caused by high mobility in a highly dynamic ad hoc network.

## 5. CONCLUSIONS AND FUTURE WORKS

In this paper, we have proposed four bandwidth-constrained routing algorithms for data streaming across multiple routes to improve the call blocking rate and channel utilization in wireless ad hoc networks. The performance of bandwidth utilization, the percentage of route splitting, the number of split links, and the forced termination ratio have been defined and observed through extensive simulations. The results reveal that a multi-path routing algorithm performs much better than a single-path routing algorithm, provided that the mobility of the network is not too high.

The scheme for a connection established by using one of the proposed splitting algorithms is similar to that of a depth-first search algorithm. Hence, it takes a long time to establish a connection. In the future, we plan to develop a better scheme with less overhead to construct multiple routes that can satisfy the QoS requirement of the connection request. Furthermore, the splitting algorithms will be extended to bring power consumption into consideration. Battery lifetime is also a critical issue in a wireless ad hoc network [4]. To extend the lifetime of the whole ad hoc network, the power conservation problem [9] will also be addressed.

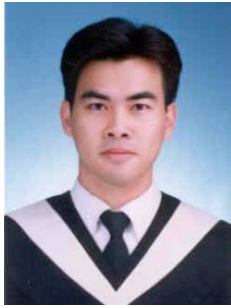
## REFERENCES

1. "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," *ANSI/IEEE Standard 802.11*, Part 11, 1999 Edition.
2. D. Bertsekas and R. Gallager, *Data Networks*, Prentice-Hall, Inc., 1987.
3. V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: a media access protocol for wireless LAN's," in *Proceedings of ACM SIGCOMM '94*, Vol. 24, 1994, pp. 212-225.
4. J. H. Chang and L. Tassiulas, "Energy conserving routing in wireless ad-hoc networks," in *Proceedings of IEEE INFOCOM 2000*, Vol. 1, 2000, pp. 26-30.
5. S. Chen and K. Nahrstedt, "Distributed quality-of-service routing in ad hoc network," *IEEE Journal of Selected Areas on Communications*, Vol. 17, 1999, pp. 1488-1505.
6. M. S. Corson and A. Ephremides, "A distributed routing algorithm for mobile wireless networks," *ACM-Baltzer Journal of Wireless Networks*, Vol. 1, 1995, pp. 61-81.
7. B. P. Crow, I. Widjaja, J. G. Kim, and P. T. Sakai, "IEEE 802.11 wireless local area networks," *IEEE Communication Magazine*, Vol. 35, 1997, pp. 116-126.
8. Z. Jiang, L. F. Chang, and N. K. Shankaranarayanan, "Providing multiple service classes for bursty data traffic in cellular networks," in *Proceedings of IEEE INFOCOM 2000*, Vol. 3, 2000, pp. 1087-1096.
9. C. E. Jones et. al., "A survey of energy efficient network protocols for wireless networks," *Kluwer Academic Wireless Networks*, Vol. 7, 2001, pp. 343-358.

10. D. Johnson and D. Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing*, MA: Kluwer, Vol. 353, 1996, pp. 153-181.
11. E. S. Jung and N. H. Vaidya, "A power control MAC protocol for ad hoc networks," in *Proceedings of Mobicom 2002*, 2002, pp. 36-47.
12. S. J. Lee and M. Gerla, "Split multipath routing with maximally disjoint paths in ad hoc networks," in *Proceedings of IEEE ICC 2001*, 2001, pp. 3201-3205.
13. C. R. Lin, "Admission control in time-slotted multihop mobile networks," *IEEE Journal on Selected Areas in Communications*, Vol. 19, 2001, pp. 1974-1983.
14. C. R. Lin, "On-demand QoS routing in multihop mobile networks," in *Proceedings of INFOCOM 2001*, Vol. 3, 2001, pp. 1735-1744.
15. C. R. Lin and J. S. Liu, "QoS routing in ad hoc wireless networks," *IEEE Journal on Selected Areas in Communications*, Vol. 17, 1999, pp. 1426-1438.
16. G. Lin and C. K. Toh, "Performance evaluation of a mobile QoS adaptation strategy for wireless ATM networks," in *Proceedings of QoS Mini Conference Conjunction IEEE ICC '99*, Vol. 2, 1999, pp. 744-748.
17. V. D. Park and M. S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," *IEEE INFOCOM '97*, Vol. 3, 1997, pp. 1405-1413.
18. C. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance vector routing (DSDV) for mobile computers," in *Proceedings of ACM SIGCOMM '94*, Vol. 24, 1994, pp. 234-244.
19. C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proceedings of IEEE Workshop on Mobile Computing Systems & Applications (WMCSA '99)*, Vol. 3, 1999, pp. 90-100.
20. W. L. Stutzman and G. A. Thiele, *Antenna Theory and Design*, John Wiley & Sons, 1998.
21. C. K. Toh, "Associativity-based routing for ad-hoc mobile networks," *Wireless Personal Communications*, Vol. 4, 1997, pp. 103-139.
22. L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, "RSVP: a new resource reservation protocol," *IEEE Network*, Vol. 7, 1993, pp. 8-18.



**Chun-Hung Richard Lin (林俊宏)** was born in Kaohsiung, Taiwan. He received the B.S. and M.S. degrees from the Department of Computer Science and Information Engineering, National Taiwan University, in 1987 and 1989, respectively, and the Ph.D. degree from Computer Science Department, University of California, Los Angeles (UCLA), in 1996. Dr. Lin joined National Chung Cheng University in Taiwan in 1996. Since August 2000, he has been with the Department of Computer Science and Engineering, National Sun Yat-Sen University, Kaohsiung, Taiwan. His research interests include the design and control of personal communication networks, protocol design and implementation for differentiated/integrated services in mobile wireless networks, mobile internet, distributed simulation, and embedded operating system design and implementation. Dr. Lin is an ACM member. He received the 2001 Junior Professor Research Award from National Sun Yat-Sen University and the 2000 Investigative Research Award from the Pan Wen Yuan Foundation, Taiwan, R.O.C.



**Yi-Siang Huang (黃億祥)** was born in Kaohsiung, Taiwan, in 1977. He received the B.S. degree from the Department of Information Science, Tunghai University, in 2000 and M.S. degree from the Department of Computer Science and Engineering, National Sun Yat-Sen University (CSE NSYSU), Kaohsiung, Taiwan in 2002. Currently, he is pursuing the Ph.D. degree in CSE NSYSU. His research interests include the design and implementation of wireless mobile networks and the real-time embedded operating system design.