A Fault-Tolerant Multicast Routing Protocol for Mobile Ad–Hoc Networks

Bidyut Gupta*, Ziping Liu** and Xianling Dong*

*Department of Computer Science, Southern Illinois University, Carbondale, USA
**Computer Science Department, Southern Missouri State University, Cape Girardeau, USA

In this work, we propose an efficient multicast routing protocol for mobile ad hoc networks. To achieve high efficiency with low channel and storage overhead, the proposed protocol employs the following mechanism: (1) on-demand invocation of route setup and route maintenance process to avoid periodical control packet transmissions, thus reducing channel overhead, (2) creation of "forwarding group" to forward multicast packets, thus reducing storage overhead, (3) exploration of multiple possible routes from a single flooded query to reduce the frequency of route discovery, thus further reducing channel overhead, (4) a new route setup mechanism that allows a newly joining node to find the nearest forwarding node to minimize the number of added forwarding nodes, thus further reducing storage overhead. To provide the capability of fault tolerance, we introduce the alternate route together with the primary route. We observe that for multicasting the channel and storage overheads of the presented approach are less than those of the DVMRP approach. Also, the channel overhead is less than that in the FGMP approach for multicasting in low mobility scenario, while the storage overheads are the same in the presented approach and in the FGMP approach.

Keywords: mobile ad-hoc network, multicasting, channel and storage overhead, fault-tolerance.

1. Introduction

Multicasting has emerged as one of the most focused areas in the field of networking. It is a technique allowing a single message to be passed to a set of destinations [1]–[4]. The explosive growth of wireless communication has led to the integration of wireless networks with the Internet. Such an integrated environment, called a mobile computing environment, allows hosts to roam around freely while retaining accesses to the Internet over a wireless medium. In mobile environments, a typical multicasting application is teleconferencing [2].

A recent interesting development has been the emergence of mobile ad hoc networks to interconnect mobile users. A mobile ad hoc network is an autonomous system of mobile hosts connected by wireless links. There is no fixed infrastructure such as base station [5]. If two hosts are not within transmission range, all message communication between them must pass through one or more intermediate hosts called "multiple hops". Such networks are very useful in military and other tactical applications such as emergency rescue or exploration missions, where fixed base infrastructure is unavailable. Commercial applications are also common where there is a need for ubiquitous communication services without the presence of a base station. Examples include on-the-fly conferencing applications, networking intelligent devices or sensors, communication between mobile robots, etc. The main characteristics of mobile ad-hoc networks are: (1) Unlike the 'single hop' (i.e. cellular) networks [9], ad-hoc mobile networks have neither fixed base stations nor wired backbone. So multi hopping over several mobile hosts may be required for communication [6]; (2) resources such as storage capacity and battery power of any mobile host as well as channel bandwidth are limited; (3) network topology changes frequently and unpredictably. All these constraints make routing and multicasting extremely challenging.

In a typical ad hoc environment, network hosts work in groups to carry out a given task [6], [7]. It is frequently necessary for one host to send a message to all the other members of the group. If the group is small, it can just send each member a point-to-point message. If the group is large, this strategy is quite expensive. Sometimes broadcasting can be used, but if most of the receivers are not interested in the message, this method is inefficient. Thus we need a way to send messages to well-defined groups that are large in size, but small compared to the network as a whole. Sending a message to such a group is called multicasting, and its routing algorithm is called multicast routing. A group of hosts that might wish to communicate with each other is called a multicast group (MG). To do multicasting, group management is required. Some way is needed to create and destroy groups, and for hosts to join and leave groups. How these tasks are accomplished, this is not of concern to the routing algorithm (therefore, it has not been considered in this paper). What is of concern is that when a host joins a group, it informs the group members of this fact. In the present work, we assume that multicast communication is source-specific. Note that in source-specific multicast communication, only specified host(s) (called sender(s)) in the multicast group send(s) data, while all the other hosts (called receivers) receive data. For instance, in a video-on-demand application, a single server provides a one-to-many transmission for customers who join the same movie at approximately the same time.

Ad hoc multicast protocols have recently attracted a lot of attention of the research community [4]–[8]. There are two general approaches: soft-state scheme, such as FGMP [4, 13], and hard-state scheme such as AODV[7, 14, 15, 16]. Soft-state scheme is claimed as robust in data delivery ratio in high mobility situation, but it has the broadcast storm problem in both low and high mobility situations. Hard-state scheme sets up and maintains multicasting route on-demand and hence gives better network performance in low mobility environment.

In this work, a link means a logical link via which two hosts can exchange messages directly, i.e. both hosts are within the transmission range of each other. So, in mobile ad hoc networks, even though no physical channel exists between two hosts, say, A and B, we still can define a 'link between A and B'. However this link can also be modeled as a broken link, for example, if host B moves outside the transmission range of host A. But B can still get messages from A via one or more intermediate hosts (i.e. via multiple hops).

2. Problem Statement

In wired networks, one of the most widely used multicasting protocols is DVMRP (Distance Vector Multicast Routing Protocol) [3, 12]. Its efficiency for multicast routing has been stated in [3, 12]. However, we cannot achieve the same efficiency in mobile networks because of two major problems, viz., high channel overhead and high storage overhead. Recently, an efficient multicast protocol known as FGMP (Forwarding Group Multicast Protocol) [4] has been proposed for mobile ad-hoc networks. FGMP is a source-specific protocol. This method has appropriately modified some of the ideas used in DVMRP to design a multicasting protocol suitable for mobile ad-hoc networks. It reduces the storage overhead to a good extent by using the concept of 'forwarding group'. However, the high channel overhead problem still exists in FGMP.

In this work, we propose an efficient multicast routing protocol based on DVMRP and FGMP. In this protocol, both the channel overhead and storage overhead are shown to be less than those in DVMRP and FGMP. A preliminary version of the present work has been reported in [8].

The paper is organized as follows. In Section 3 we give a brief idea about the DVMRP and the FGMP protocols which form the basis of our work. The proposed protocol is stated in Section 4. We present an analytical model and the associated numerical results in Section 5. In Section 6 we have presented the simulation results, and, finally, Section 7 draws the conclusion.

3. DVMRP and FGMP Protocols

3.1. Distance Vector Multicast Routing Protocol (DVMRP)

DVMRP is based on Distance Vector Routing (DVR) algorithm [9]. In DVMRP, each sender uses flooding to direct the multicast packets to all hosts, and multicast packets are selectively forwarded according to the Reverse Path Forwarding (RPF) scheme [9]. In DVMRP, each node periodically exchanges routing table update information with all its neighbors even if all the information is still the same (no demand for updating). Based on the updates from its neighbors, a node builds its multicast routing tables. Obviously, periodic flooding causes very large channel overhead for the low bandwidth wireless channel. Besides, for each sender source, each node needs to store information of its parent and all its children links. Therefore, the size of the routing tables increases linearly with the number of nodes resulting in very high storage overhead.

3.2. Forwarding Group Multicast Protocol (FGMP)

Forwarding group is a set of nodes responsible for forwarding multicast packets along the shortest paths to the receiver members in a multicast group MG. Each multicast group is associated with a forwarding group (FG). A multicast receiver can also be a forwarding node (a node in FG) if it is on the path between a multicast source and another receiver. It may be noted that FGMP keeps track not of the links but of the forwarding group. The size of the FG should be as small as possible to reduce channel overhead. In order to select the FG, it requires each source to periodically transmit control packets to all member destinations. In the process, all nodes along the shortest path from source to destination are inducted into the FG. This procedure presumes that each source knows the member destinations. This is obtained via a receiver advertising a message, called JOIN packet. JOIN packet is used by a receiver to broadcast its existence and hope that the source responds to it.

In FGMP, only one flag and a timer are needed for each forwarding node. The decision to forward multicast packets by a forwarding node depends on this flag, also known as forwarding flag. The forwarding flag is associated with a timer. When a node in FG learns of a receiver member, it resets its forwarding timer. A node with enabled forwarding flag (i.e., timer has not expired) is responsible for forwarding the multicast packets. The timer is refreshed by the forwarding group updating protocol. When a forwarding node receives a multicast packet and its forwarding flag is enabled, it just broadcasts it to its neighbors. All the neighbors can hear it; but only its neighbors in the FG with enabled forwarding flag will first determine if it is a duplicate and if not, then broadcast it in turn. This process continues until all the receivers get the packet. However, in FGMP each receiver periodically floods its member information to advertise the membership. So, the channel overhead is high, even though its storage overhead is much less than that of DVMRP.

4. The Proposed Protocol

In this section, we propose an efficient multicast routing protocol with on-demand mechanism, i.e., route set-up and route recovery are done only when they are required. In our protocol we shall use the idea of the selective flooding technique (as in DVMRP) and the forwarding group (as in FGMP). The proposed protocol consists of three parts: the route set-up process, the quit process, and the route recovery process.

4.1. Route Set-up Process

A route set-up process is invoked when a new node wants to join a multicast group. A route set-up process finds the nearest forwarding node or receiver node in the multicast group and sets up a path between this node and the newly joining node. Let us use the example of Figure 1 to illustrate this process. As shown in Figure 1(a), the source node for multicasting is S1 and the receiver nodes are R1, R2, and R3. The forwarding group is FG= {F1, F2, F3} and the multicast group MG consists of the source and the three receiver nodes. Suppose that A is a new node which wants to join the MG.

At first, as in Figure 1(b), A broadcasts a JOIN packet. The packet structure is shown in Table 1. In this table, "Multicast Group ID" identifies the specific multicast group, "Receiver Member ID" identifies receiver members for this group, "Source Sequence #" identifies the sender(s) for this group, "TTL" (time to live) limits the scope of flooding [4] , "Sending Node ID" identifies the node which is currently sending the JOIN packet and "Hops" is the hop count traversed by the JOIN packet. JOIN packets are flooded until they reach the forwarding nodes (F2 in Figure 1(b)) or receiver nodes (R2 in Figure 1(b)) of multicast group MG. When a node forwards a JOIN packet, it adds its node ID to the

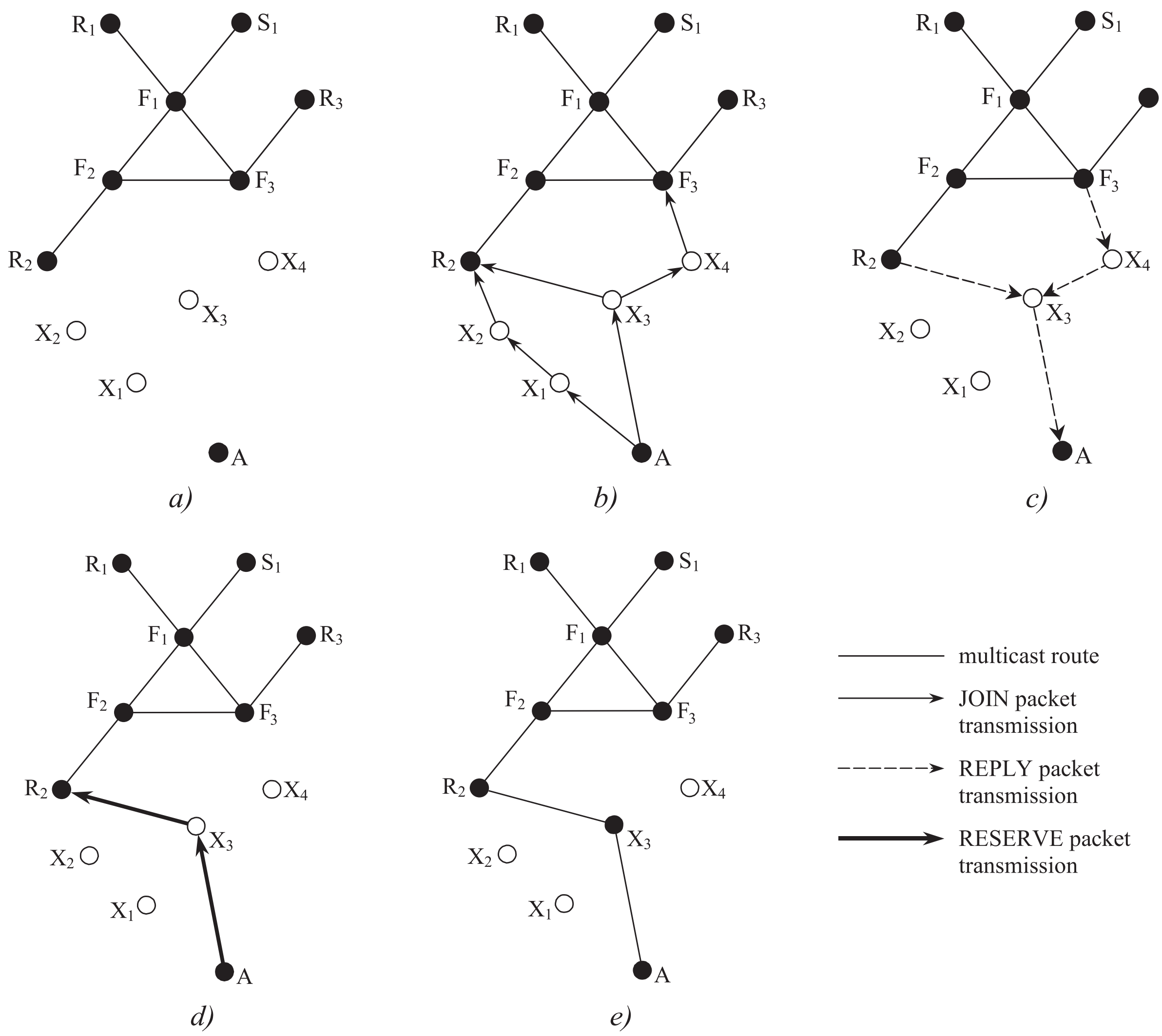

Fig. 1. An example of the route set-up process.

Multicast Group ID	Receiver Member ID	Source Sequence #	TTL	Sending Node ID	Hops

Table 1. Format of on-demand JOIN packet.

JOIN packet and increments the hop count by one contained in the JOIN packet. Therefore, a JOIN packet contains a list of nodes traversed by the JOIN packet and the hop count.

Note that the forwarding nodes or the receiver nodes in a multicast group MG may receive more than one JOIN packet and the first received JOIN packet does not necessarily have the smallest hop count. Therefore, in our protocol, forwarding nodes or receiver nodes wait until they receive a certain number of JOIN packets, and then choose the JOIN packet with the smallest hop count. Forwarding nodes or receiver nodes will send REPLY packets back to node A, following the reverse path that the selected JOIN packet has traversed. As shown in Figure 1(c), receiver node R2 sends REPLY packet via (R2-X3-A) and forwarding node F2 sends REPLY packet via (F2-X4-X3-A).

Node A also waits until it receives certain number of REPLY packets or waits for some predetermined time period, and then chooses a REPLY packet with the smallest hop count. Node A maintains a cache to store the REPLY packet, also with the second smallest hop count. We consider a control packet, named as RESERVE packet, to announce a selected route between any member pairs. A sends a RESERVE packet with flag=1 to all nodes along the path that the selected REPLY packet has traversed. This path is called *primary route*, as shown in Figure 1(d) (R2-X3-A). Then A sends another RESERVE packet with flag=0 along the path stored in cache. This path is called *alternate route*. In the process, all nodes along the primary route from source to destination are inducted into the *forwarding group*.

Upon receiving a RESERVE packet, each node updates its multicast routing table as shown in Table 2. In this table, "Receiver Member ID" is the receiver node of the multicast packet. "Neighbor Upstream Node" is the node from which the current node receives the multicast packet. "Neighbor Downstream Node" is the node to which the current node sends the multicast packet. "Source Sequence#" is used for source identification. "Hop Count" is incremented at each forwarding node, and this hop count information is recorded locally, before being forwarded. The Hop Count and the Timer information are used in the route recovery process, which is described in subsection 4.3.

After route set-up is done, multicast packets are forwarded along the primary route. As shown in Figure 1(e), R2 is a receiver node in the original multicast group and (R2-X3-A) is the new route set-up for a new node A which wants to join this group. Therefore, node R2 and node X3 are inducted into the new forwarding group. The forwarding group is in charge of forwarding multicast packets (same as in FGMP). The algorithm for this process is shown in Figure 2.

Multicast Group ID	Receiver Member ID	Neighbor Upstream Node	Neighbor Downstream Node	Hop Count	Source Sequence #	Timer

Table 2. Format of on-demand routing table.

```
{ IF A wants to join a MG
    A broadcasts a JOIN packet
    { IF x ∉ FG and x ∉ R
        x floods the JOIN packet
        add x's node ID to the JOIN packet
        increment the hop count contained in the JOIN packet
      ELSE
        x does not flood the JOIN packet
            ...
          x waits    /* for more JOIN packets to choose the best path */
            ...
        x chooses a JOIN packet with smallest hop count
        x sends a REPLY packet back to A along the traversed path
    }
        ...
      A waits        /* for more REPLY packets to choose the best path */
        ...
    A chooses a REPLY packet with smallest hop count (primary route)
    A puts the REPLY packet with second smallest hop count in the route cache
                                                    (alternate route)
    A sends a RESERVE packet with flag=1 along the primary route
    A sends a RESERVE packet with flag=0 along the alternate route
    Upon receiving a RESERVE packet, each node on the path updates its
                                                    multicast routing table
    All nodes along the primary route are inducted into FG

ENDIF
}
```

Fig. 2. Algorithm for the route set-up process.

4.2. Quit Process

This process is recursive. It presumes that after a receiver node leaves the MG, the neighbor upstream node of this receiver node also needs to leave if it has no other neighbor downstream node (since the upstream node is not a forwarding node any more). In this way we can keep the size of the forwarding group as small as possible.

When a node, say B, which is a receiver node of a multicast group MG, wants to leave the multicast group, it sends a QUIT packet to its upstream node. Upon receiving the QUIT packet, the upstream node checks if it has any downstream node other than the node B. If it has any other downstream node, it simply deletes node B from the downstream entry in its multicast routing table. Otherwise, it sends a QUIT

```
{IF B wants to leave MG
    B sends a QUIT packet to its upstream node C
    C checks its multicast routing table
    {IF number of C's downstream nodes > 1
        C deletes B from its routing table
     ELSE
        C sends a QUIT packet to its upstream node and leaves
     }
ENDIF
}
```

Fig. 3. Algorithm for the quit process.

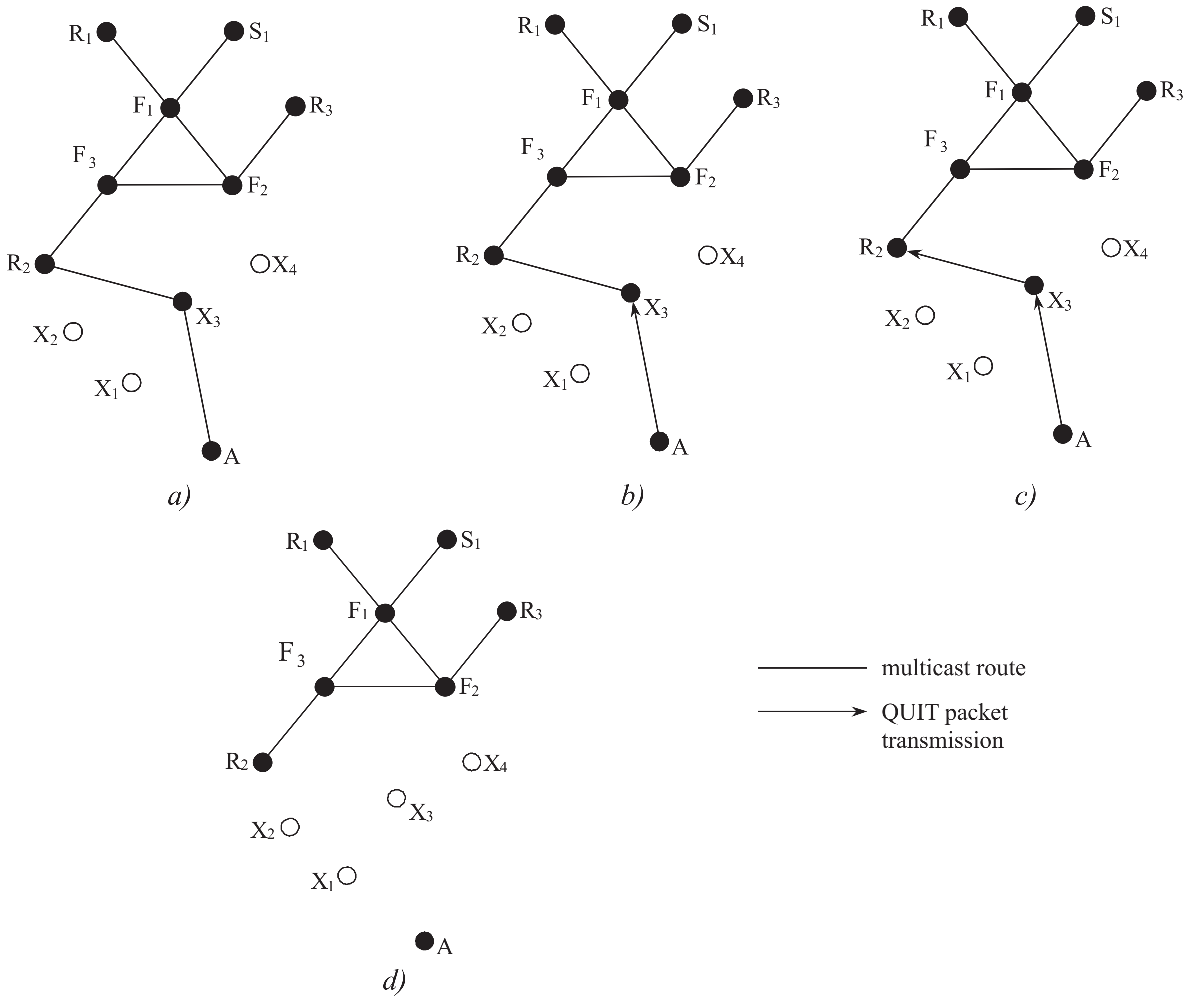

Fig. 4. An example of the quit process.

packet to its upstream node and leaves the FG. The algorithm for this process is shown in Figure 3.

In Figure 4, we present an example to show how the quit process works. Let us start from the scenario shown in Figure 4(a) (same as in Figure 1(e)). Suppose A wants to leave the MG, then A sends a QUIT packet to its upstream node X3 (Figure 4(b)). The node X3 checks its routing table and finds that it does not have any downstream node other than A. So, X3 sends QUIT packet to R2 and leaves (as in Figure 4(c)). The final scenario is shown in Figure 4(d).

4.3. Route Recovery Process

Sometimes, a multicast route breaks (logical link failure) due to node movements as discussed in Section 1. When a multicast route breaks, route recovery process is invoked.

Assume that node A and node B belong to a multicast group MG, and that node B is the neighbor downstream node of node A. Also assume that the link between node A and node B is broken because node B moves out of transmission range of node A. In this scenario, there are two approaches to implement the recovery process. One approach is based on the assumption that node A has the responsibility to find a new route to node B. Another approach is based on the assumption that node B has the responsibility to find a new route to node A. Our scheme belongs to the first approach and works as follows.

When the link between A and B fails, then node A floods a special packet called BROADCAST-MULTICAST packet, which contains the original multicast packet. This flooding is done with limited TTL (i.e. as with limited hop count). When a node forwards the BROADCAST-MULTICAST packet, it adds its node ID to the packet. Therefore, when node B receives a flooded BROADCAST-MULTICAST packet, it can tell the exact route that the packet has traversed. Node B then sends a RESERVE packet to node A along the reverse path that the BROADCAST-MULTICAST packet has traversed. The RESERVE packet sets up a new route between node A and node B. The algorithm for this process is shown in Figure 5.

Figure 6 presents an example of the route recovery process. Suppose that the link between F1 and F2 is broken as shown in Figure 6(b). We can use the alternate route of (F1-F3-F2) to reach R2. Only when this alternate route is also broken as shown in Figure 6(c), the route recovery process is invoked. Node F1 floods the BROADCAST-MULTICAST packet to R1 and X5; then node X5 forwards it to F2 and adds its node ID to the packet (Figure 6(d)). After F2 receives the BROADCAST-MULTICAST packet, it will send a RESERVE packet to F1 via X5, as shown in Figure 6(e). In this way, a new route (F1-X5-F2) is a set-up between F1 and F2 to reach R2. The final scenario is shown in Figure 6(f).

Note that route recovery sometimes may fail. For example, route recovery fails when a RESERVE packet is lost because one of the links in the path that the RESERVE packet should traverse is broken. Route recovery also fails when node B is not in the flooding scope of node A. In order to recover from such a failure, a separate timer can be associated with each multicast group in the multicast routing table. The timer for the multicast group MG is refreshed when a multicast packet for the group MG is received. In the absence of refreshes, the timer will expire and the node leaves the forwarding group if it is not a receiver node for the group MG. If it is a receiver node for the group MG, it invokes the route setup process.

```
{IF primary route fails AND alternate route fails
    The route recovery process is invoked
    A floods a broadcast-multicast packet
    Any node forwarding this packet adds its node ID to the packet
    Upon receiving the packet, B sends a RESERVE packet to A along the traversed path
    The RESERVE packet sets up a new route between A and B
ELSE
              ...
    Multicasting is going on
              ...
ENDIF
}
```

Fig. 5. Algorithm for the route recovery process.

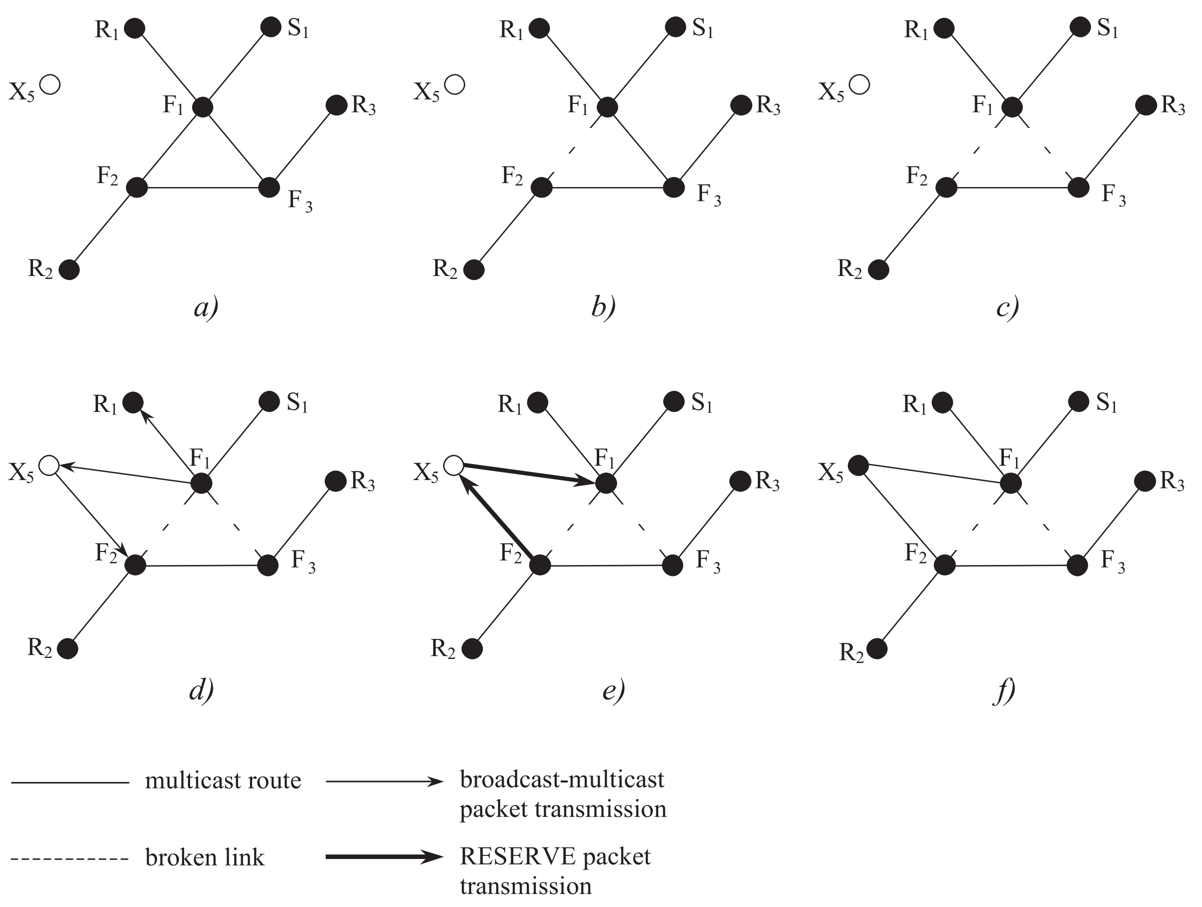
R1
S1
X5
F1
R3
F2
F3
R2
a)
b)
c)
d)
e)
f)
multicast route
broadcast-multicast packet transmission
broken link
RESERVE packet transmission

Fig. 6. An example of the route recovery process.

4.4. Comparison of DVMRP, FGMP, and the Proposed Protocol

We have shown in the previous section that the proposed scheme uses on-demand invocation of the route set-up and the route recovery processes to avoid periodic transmission of the control packet. Hence the channel overhead is less compared to that either in DVMRP or in FGMP. Moreover, the proposed forwarding group structure enables the exploration of multiple possible routes from a single flooded query. Therefore, in effect, it reduces the frequency of flooding as well. This fact has been supported further in the next section by the analytical modeling and the associated numerical results. Also, reduction in the frequency of flooding ensures that the channel overhead in our scheme is reduced further.

Note that we have introduced the idea of the use of an alternate route together with the primary route to provide the capability of fault tolerance in the proposed protocol. However, such a capability is absent in both DVMRP and FGMP.

It may also be noted that, as an improvement over FGMP, the proposed protocol does not try to find the shortest path (unlike in FGMP) from a source to a receiver at any time; instead, it tries to find the nearest forwarding node when some new node wants to join a multicast group. Nodes along the path between the nearest forwarding node and the new node become new forwarding nodes. This ensures that the number of the newly added forwarding nodes is minimum. Thus, our scheme can deliver multicast packets to the receivers with smaller number of multicast transmissions compared to FGMP.

Finally, as in FGMP, in our protocol we need only a timer and a flag for each forwarding node. Thus, FGMP and our protocol have identical storage overheads. However, in DVMRP for each sender source, each node needs to store information of its parent and its all children links. Therefore, the size of the routing table increases linearly with the number of the nodes resulting in a high storage overhead for each node [3]. This shows that storage overhead in the proposed protocol is much less than that of

	DVMRP	Our Protocol
Frequency of Route Discovery (Flooding)	Periodic	On-demand
Fault-Tolerance	No	Yes
Selective Flooding	Yes	Yes
Information Stored	For each sender source, each node needs to store information of its parent and all its children links.	Only a flag and a timer are needed for each forwarding node.
Channel Overhead	High	Low
Storage Overhead	High	Low

Table 3. Comparison of DVMRP and our protocol.

	FGMP	Our Protocol
Frequency of Route Discovery (Flooding)	Periodic	On-demand
Fault-Tolerance	Yes	Yes
Selective Flooding	Yes	Yes
Size of forwarding group	Larger than that of our protocol	Smaller than FGMP
Channel Overhead	High	Low (for low mobility)
Storage Overhead	Same	Same
Route Set-up process	When a node wants to join a multicast group, the JOIN packet has to be sent all the way until it reaches the source.	When a node wants to join a multicast group, the JOIN packet stops when it reaches a forwarding node or a receiver node.

Table 4. Comparison of FGMP and our protocol.

DVMRP. In Table 3 and Table 4, we list some of the important differences among our proposed protocol, DVMRP, and FGMP (as evident from their working principles).

5. Analytical Model

5.1. Multiple Routes Extension

From Section 4 it is obvious that in our proposed protocol, from a single flooded query, we can explore multiple possible routes. When the primary route breaks, the alternate route is activated (set flag=1), and thus enables the multicast communication to continue. Only when both of them break, route recovery process is invoked. It not only provides good capability of fault tolerance, but it also reduces the frequency of route discovery (selective flooding of JOIN packet), which is recognized as a major overhead in any on-demand protocol [7].

5.2. Basic Assumptions

We represent the lifetime of a wireless link between a pair of nodes by a random variable. Consider a route from S to D that consists of a sequence of k wireless links over $k - 1$ intermediate nodes. Let L_i be the i-th link in the route. The lifetime of L_i is denoted by X_{l_1}. Assume that X_{l_1}, $i = 1, 2, \ldots, k$, are independent and exponentially distributed random variables [10], each with mean l. We will use upper-case letters to denote random variables and the corresponding lower-case letters to denote their values.

Since a route fails when *any* one of the wireless links in its path breaks, the lifetime of a route P, consisting of k wireless links, is a random variable X_p that can be expressed as

$$X_p = \min(X_l, X_{l_2}, \ldots, X_{l_k}). \qquad (1)$$

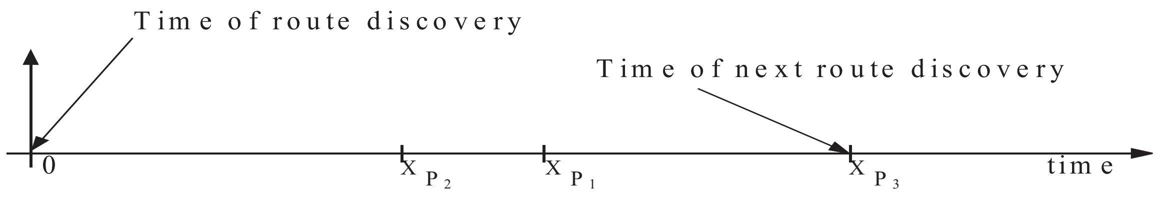

Fig. 7. An example of multiple routes.

It is well known that X_p is also an exponentially distributed random variable with a mean of $\frac{l}{k}$ [11].

Using the basic assumptions about the link failure behavior as above, we proceed to derive the statistics of the time interval between successive route discoveries for the proposed scheme.

5.3. Analytical Modeling

Without any loss of generality, assume that a source S has N routes to a destination D. The primary route is denoted by P_1 and the N−1 alternate routes are denoted by $P_2, P_3, \ldots, P_N$. The length of route P_i is k_i. In Figure 7, we give an example of multiple routes where the source has three independent routes P_1, P_2, P_3 to the destination ($N = 3$). The figure represents the lifetimes of the three routes. When the primary route P_1 breaks at time X_{p_1}, S attempts to use P_3 as P_2 is already broken. P_3 breaks at time X_{p_3}, when new route discovery is initiated.

The time after which none of the routes are useful is a random variable T, where

$$T = \max(X_{p_1}, X_{p_2}, \ldots, X_{p_n}) \tag{2}$$

T represents the time between successive route discoveries. Here, we assume that the end-to-end packet transmission latency in the network is very small compared to the interval between route changes. Thus, the time to discover routes or propagate error packets etc. can be ignored relative to T. These are very reasonable assumptions, because, otherwise, routes will fail while discovery is in progress. No routing protocol will perform well in such situations.

Claim: The probability density function (pdf) of T, the time between successive route discoveries, is given by

$$\begin{aligned} f_T(t) = \lambda_1 e^{-\lambda_1 t}(1 - e^{-\lambda_2 t}) \\ \cdot (1 - e^{-\lambda_3 t}) \ldots (1 - e^{-\lambda_n t}) \\ + \lambda_2 e^{-\lambda_2 t}(1 - e^{-\lambda_1 t}) \\ \cdot (1 - e^{-\lambda_3 t}) \ldots (1 - e^{-\lambda_n t}) \\ + \ldots + \lambda_n e^{-\lambda_n t}(1 - e^{-\lambda_1 t}) \\ \cdot (1 - e^{-\lambda_2 t}) \ldots (1 - e^{-\lambda_{n-1} t}) \end{aligned} \tag{3}$$

where $\lambda_i = k_i/l = 1/$lifetime of the i-th route.

Proof: Consider N exponentially distributed random variables, $X_{p_1}, X_{p_2}, \ldots, X_{p_n}$, where the pdf of X_{p_i} is $f_{X_{p_i}}(t) = \lambda_i e^{-\lambda_i t}$, $i = 1, 2, \ldots, N$. Note that X_{p_i}'s are independent. Therefore, the cumulative distribution function (cdf) of T, $F_T(t)$, is obtained as

$$\begin{aligned} F_T(t) &= P[T \le t] \\ &= P[\max(X_{p_1}, X_{p_2}, \ldots, X_{p_N}) \le t] \\ &= P[(X_{p_1} \le t) \cap (X_{p_2} \le t) \cap \ldots \cap (X_{p_N} \le t)] \\ &= \prod_{i=1}^{N} F_{X_{p_i}}(t) \end{aligned} \tag{4}$$

where $F_{X_{p_i}}(t) = 1 - e^{-\lambda_i t}$ is the cdf of X_{p_i}. By differentiating (4) with respect to t, we get the pdf of T, as shown in (3).

The expected value of T can be derived from (3), by knowing the hop-wise length s of all the routes k_i, $i = 1, 2, \ldots, N$. For example, for $N = 2$, the expected value of T, $E[T]$ is

$$E[T] = \frac{\lambda_1^2 + \lambda_2^2 + \lambda_1\lambda_2}{\lambda_1\lambda_2(\lambda_1 + \lambda_2)}. \tag{5}$$

Consider also the special case when all the routes are equal. i.e. $k_1 = k_2 = \ldots = k_N = k$.

In this case,

$$E[T] = \frac{N}{\lambda} - \frac{N}{2^2\lambda}\binom{N-1}{1} + \frac{N}{3^2\lambda}\binom{N-1}{2} + \ldots + (-1)^{N-1}\frac{1}{N\lambda} \tag{6}$$

where $\lambda = k/l$.

5.4. An Example

Let us consider an example of our proposed scheme. A source S has $N = 2$ routes to Destination D. For the case that the primary route has $k_1 = 3$ hops, and the alternate route has $k_2 = 4$ hops, we have $E[T] = \frac{9+16+12}{12 \times 7}l = \frac{37}{84}l$. If we compare this with DVMRP with only a single route of 3 hops, we find that $E[T] = \frac{l}{3}$. This represents almost 25% reduction in the frequency of route discoveries compared to the single path case.

5.5. Numerical Results

Using the analysis presented above, we present now some numerical results showing performance benefits of our scheme. The pdf of the time interval between successive route discoveries is given by Equation 3, as mentioned earlier. Without any loss of generality, we evaluate the expected value of this interval using numerical techniques for the three following cases:

Case A: assume that all the N routes from S to D are of the same length (Equation 6). This implies the "best case" scenario.

Case B: assume that all the N routes from S to D are of increasing lengths, with the primary route being the shortest. The successive route lengths increase by one.

Case C: assume that all the N routes from S to D are of increasing lengths, with the primary route being the shortest. The successive route lengths increase by two. Thus,

Case A: $k_1 = k_2 = \ldots = k_N = k$

Case B: $k_1 = k, k_2 = k+1, \ldots, k_N = k+N-1$

Case C: $k_1 = k, k_2 = k+2, \ldots,$
$k_N = k + 2(N-1)$

In Figure 8, the expected time intervals between successive route discoveries for the three cases are plotted against different values of the path length of the primary route. Three cases for two routes (one primary route, and one alternate route) are compared with the single path case. The mean lifetime of a single link (l) is assumed to be 5. The interval increases from

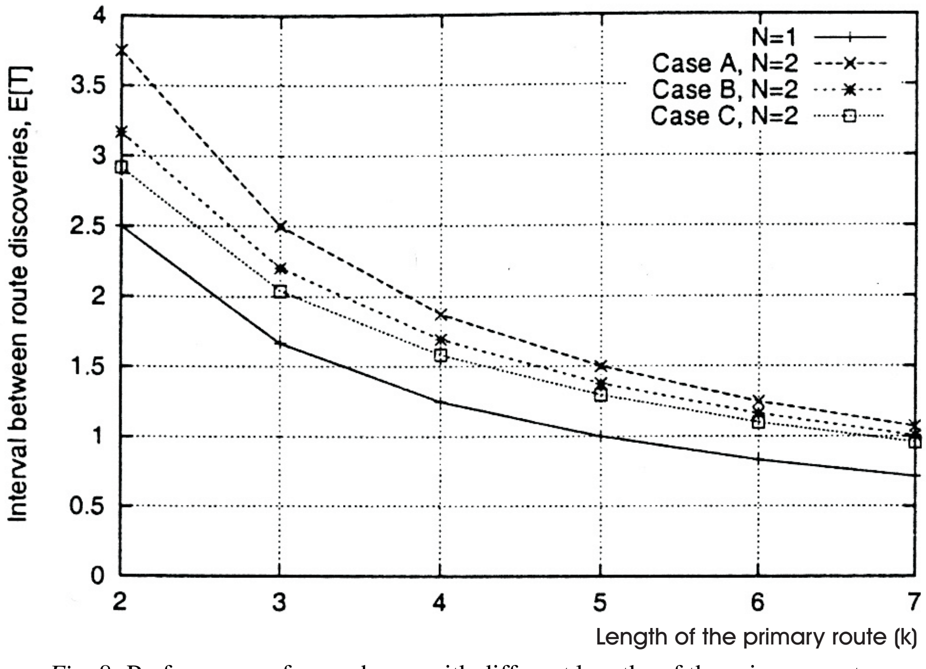

Fig. 8. Performance of our scheme with different lengths of the primary route.

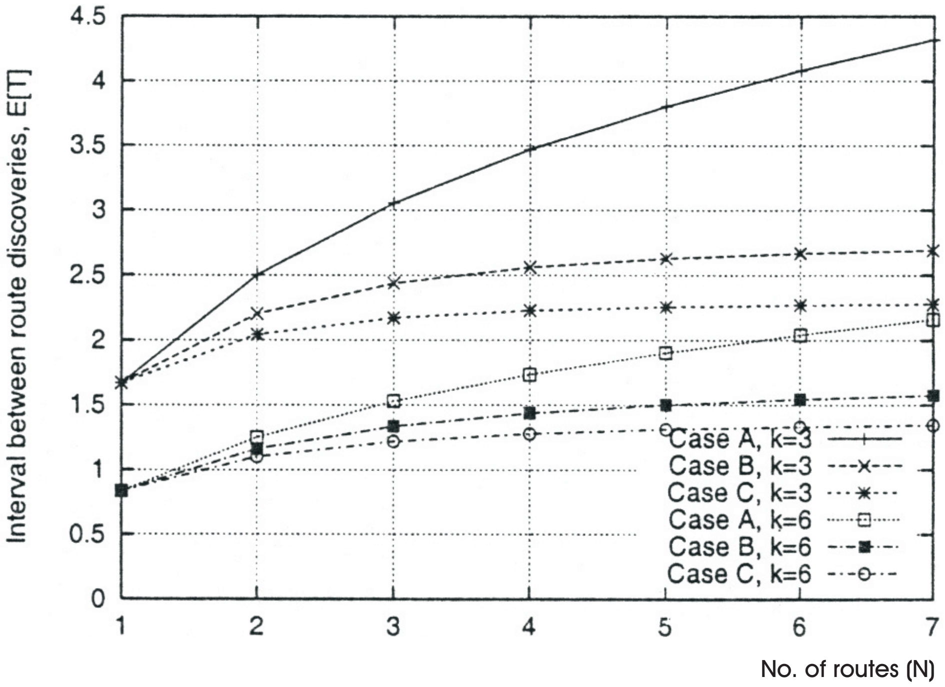

Fig. 9. Performance of our scheme with varying number of routes (N).

Case C to B to A. For all cases the interval is longer than the single path case denoting less frequent route discovery.

In Figure 9, the expected time interval between successive route discoveries is plotted against the varying number of routes (N), all for the same primary route length. Two primary route lengths are used (k=3 and k=6). The mean lifetime of a single link (l) is assumed to be 5. It may be obvious that the expected performance always improves with increasing number of alternate routes. However, the incremental improvement is very small for N>3, except when the paths are of the same length (Case A). Note that this case is very unlikely to occur in practice. This indicates that usually only one or two alternate routes will be sufficient to get reasonably efficient performance (this is why our protocol can work very well with only one alternate route).

6. Simulations

The simulations were performed using the QualNet Simulator developed at Scalable Network Technologies [17]. IEEE standard 802.11 [18] is the MAC protocol used in the simulations. We simulated networks of three different sizes: 100 nodes over a square (2000×2000m) space, 50 nodes over a square (1200×1200m) space and 25 nodes over a square (900×900m) space. Nodes move according to the "random waypoint" model [17] with pause time of 5 seconds and the maximum speed is 35 meters/sec. All data packets are 512 bytes long, and the interarrival time is one second. The simulated application for data packets is a constant bit rate (CBR). Two performance metrics of our protocol and FGMP are evaluated: (i) data packet delivery ratio-measured as the percentage of data packets received by multicast group members; (ii) number of route discovery control packets.

Figures 10, 11, and 12 show the packet delivery ratio for the aforementioned three networks. We have noticed that when the nodes are static or move slowly, there is no significant difference

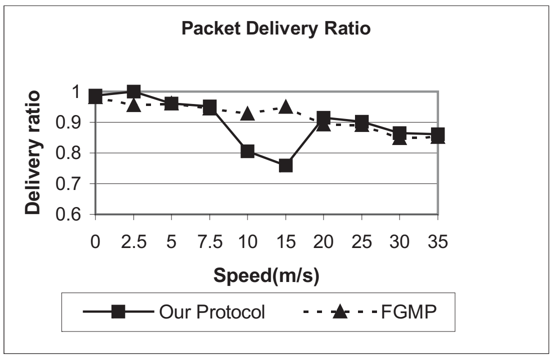

Fig. 10. Packet Delivery Ratio of 100-node Network.

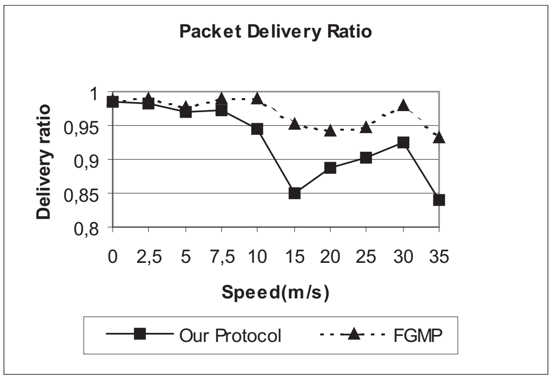

Fig. 11. Packet Delivery Ratio of 50-node Network.

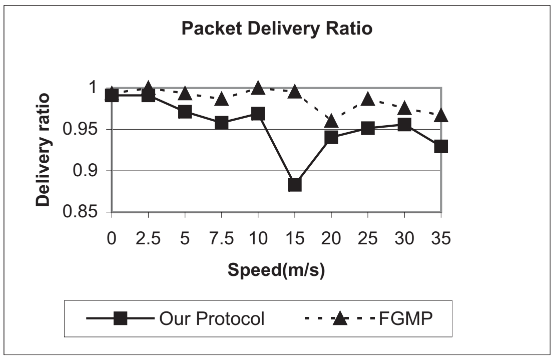

Fig. 12. Packet Delivery Ratio of 25-node Network.

of packet delivery ratio between our protocol and FGMP. When the nodes begin to move fast, the packet delivery ratio of our protocol drops more than that of FGMP and the difference becomes significant. FGMP uses periodic route discovery messages to maintain an up-to-date routing path, therefore its packet delivery ratio does not decrease much, even in high mobility situation. For our protocol, the result also shows that an increase in speed yields a decrease in the packet delivery ratio. This is due to the fact that the routing topology of our protocol encounters more link breaks and repairs as nodes move faster. Frequent change of network topology often results in multiple attempts per repair to re-establish the connections. During the repair, some of the group members thus may not receive data packets.

As to the appearance of local minima in delivery ratio of our proposed protocol for speeds between 7.5 and 20 m/s in Figures 10–12, it is caused by the lack of (enough) nodes to form a delivery route from the sender to certain receivers for a relatively longer period of time. Even though more attempts are made to repair the broken routes, it still takes more time for some nodes to move into their corresponding radio communication ranges to re-establish the connections.

We next investigate route discovery overhead via the number of route discovery control packets. The simulation results of route discovery overhead for our protocol and FGMP are given in Figures 13, 14, and 15. We observe that our protocol has significantly fewer number of route discovery packets than FGMP, as nodes move slowly. This is due to the reason that our protocol uses existing routing topology, as well as alternate routing paths, in case of broken link, while FGMP uses periodic network wide route discovery packets to find the path and deliver data packets. As nodes move faster, the route discovery overhead of our protocol increases abruptly, while the overhead increase of FGMP is not big. This is because the frequent change

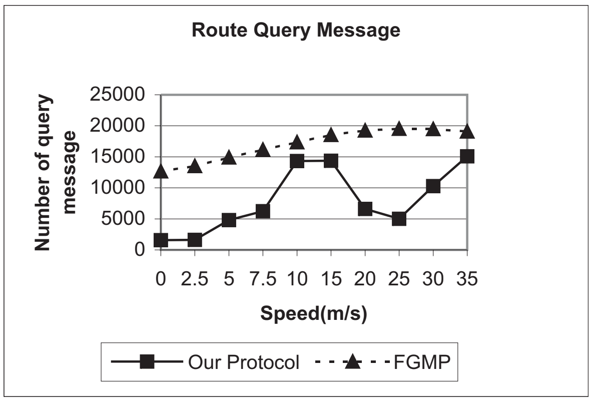

Fig. 13. Route Discovery Packet Numbers of 100-node Network.

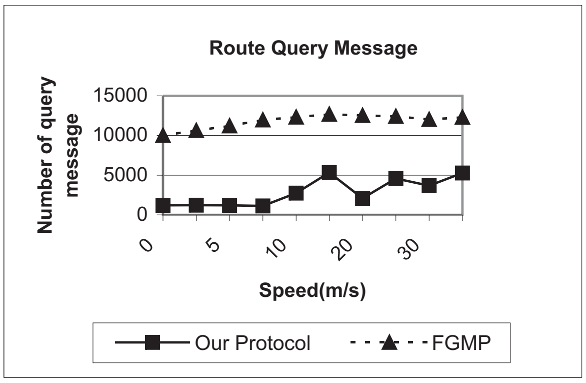

Fig. 14. Route Discovery Packet Numbers of 50-node Network.

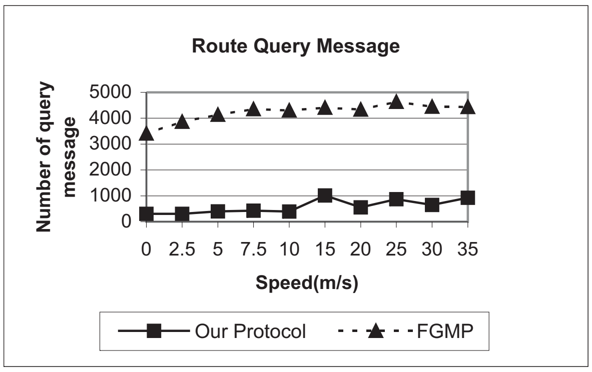

Fig. 15. Route Discovery Packet Numbers of 25-node Network.

of network topology causes more link repairs and hence generates more route repair packets for our approach. However, the faster the nodes move, the more the overhead increases in both protocols, and the difference of route discovery overhead between the two protocols becomes less significant.

The above simulation results demonstrate that in a low mobility scenario our protocol not only has comparable packet delivery ratio as FGMP, but also has lower channel overhead that can save ad hoc network's power, energy, and processing space. But in the presence of high mobility, the packet delivery ratio of our protocol is lower than FGMP while the difference of channel overhead is not significant. This shows that only in high mobility scenario the periodic route set-up used by FGMP provides a better fault resilience with respect to our protocol.

7. Conclusion

In this work, a multicast routing protocol for mobile ad hoc networks is proposed. Unlike some important existing protocols, such as DVMRP and FGMP, our protocol requires low channel overhead, since the route set-up and route maintenance processes are invoked only on-demand. It does not require periodical transmission of control packets to maintain multicast group membership and multicast routes, thereby saving a lot of bandwidth. Also, we employ a new route set-up mechanism that allows a newly joining node to find the nearest forwarding node or receiver node, in order to minimize the number of added forwarding nodes. So, compared to FGMP, the proposed scheme can deliver multicast packets to receivers with smaller number of multicast transmissions.

The design of the alternate route provides fault tolerance. When the primary route breaks, the alternate route is activated; thus it enables the multicast communication to continue. Only when both of them break, the route recovery process is invoked. We thus reduce the frequency of route discovery, which is recognized as a major overhead in any on-demand protocol. Note that some of the important differences among the three protocols considered in this work (as listed in Tables 3 and 4) become evident from their respective working principles. We have provided a framework for modeling the time interval between successive route discoveries for on-demand protocols based on simple assumptions on the lifetime of a single wireless link. We have performed simulations to compare our protocol, in particular with FGMP. The simulation results demonstrate that, in low mobility scenario, our protocol and FGMP have comparable packet delivery ratio, and our protocol offers lower channel overhead. In high mobility scenario, FGMP offers better fault resilience.

Acknowledgment

We are indebted to the referees for their valuable suggestions that have helped immensely in preparing the revised manuscript.

References

[1] K. P. Birman, A. Schiper, and P. Stephenson, *Lightweight causal and atomic group multicast*. ACM Trans. Comp. Sys., 9 (1991), pp. 272–314.

[2] J. Mysore and V. Bharghavan, A new multicasting-based architecture for Internet host mobility. *Proceedings of the 3rd Ann. ACM/IEEE Conf. Mobile Comp. And Networks*, 26–30 September, (1997), pp. 161–172, New York, USA.

[3] S. E. Deering and D. R. Cheriton, *Multicast Routing in Datagram Internetworks and Extended LANs*. ACM Transactions on Computer Systems, 8(2), (1990), pp. 85–110.

[4] CHING-CHUAN CHIANG, MARIO GERLA, AND LIXIA ZHANG, *Forwarding Group Multicast Protocol (FGMP) for Multihop. Mobile Wireless Networks*. ACM-Baltzer Journal of Cluster Computing: Special Issue on Mobile Computing, 1(2), (1998), pp. 187–196.

[5] D. JOHNSON AND D. MALTZ, *Dynamic source routing in ad hoc wireless networks. Mobile Computing*. (T. Imielinski and H. Korth, Ed), Kluwer Academic, 1996.

[6] J. MACKER AND S. CORSON, Mobile ad hoc networking (MANET). `http://www.ietf.org/html.charters/manet-charter.html`, (1999), IETF Working Group Charter.

[7] C. PERKINS AND E. ROYER, *Ad hoc on-demand distance vector (AODV) routing*. IETF, Internet Draft: draft-ietf-manet-aodv-00.txt, (1997).

[8] B. GUPTA, Z. LIU, AND X. DONG, An Efficient Multicast Routing Protocol For Mobile Ad-hoc Networks. *Proceedings of the 2002 Int'l Conf. Parallel and Distributed Processing Techniques and Applications*, June 24–27, (2002), pp. 1810–1816, Las Vegas, USA.

[9] A. S. TANENBAUM, *Computer Networks (Fourth edition)*. Prentice Hall, 1996.

[10] M. SCHWARTZ, *Telecommunication Networks Protocols, Modeling and Analysis*. Addison-Wesley Publishing Company, 1987.

[11] S. ROSS, *Introduction to Probability Models*. Academic Press, 1998.

[12] T. PUSATERI, *Distance Vector Multicast Routing Protocol*. IETF RFC 1075, August (2000).

[13] J. J. GARCIA-LUNA-ACEVES AND E. L. MADRUGA, *The Core-assisted Mesh Protocol*. IEEE Journal Selected Area Communication. (Special Issue on Ad-Hoc Networks) 17(8), (1999), pp. 1380–1394.

[14] C. W. WU, Y. C. TAY, AND C.-K. TOH, *Ad Hoc Multicast Routing Protocol Utilizing Increasing Id-numbers (AMRIS) Functional Specification*. Internet draft, IETF, August (1998).

[15] E. BOMMAIAH, M. LUI, A. MCAULEY, AND R. TALPADE, *AMRoute: Ad-hoc Multicast Routing Protocol*. Internet draft, IETF, August (1998).

[16] L. JI AND M. S. CORSON, *A Lightweight Adaptive Multicast Algorithm*. Proceedings of the IEEE GLOBECOM, pp. 1036–1042, December (1998).

[17] *QualNet Simulator Version 3.0 User's Manual*. Scalable Network Technologies, Inc., Los Angeles, (2001).

[18] *I.S. Department, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. IEEE Standard 802.11 — 1997, 1994

Received: May, 2002
Revised: December, 2003
Accepted: March, 2004

Contact address:

Bidyut Gupta,
Xianling Dong
Department of Computer Science
Southern Illinois University
Carbondale, IL 62901-4511
USA
e-mail: `bidyut@cs.siu.edu`

Ziping Liu
Computer Science Department
Southern Missouri State University
Cape Girardeau, MO 63701
USA
e-mail: `zliu@semovm.semo.edu`

BIDYUT GUPTA received M.Tech. degree in Electronics Engineering and Ph.D. degree in Computer Science from the Department of Radio Physics and Electronics and Department of Computer Science, University of Calcutta, India in 1978 and 1986 respectively. Currently, he is a professor at the Department of Computer Scence, Southern Illinois University at Carbondale, USA. His research interests include design and analysis of algorithms for fault-tolerant computing in distributed systems and design of communication protocols for wireless networks.

ZIPING LIU received B.S. and M.S. degrees in Engineering from Hefei University of Technology, China, in 1990 and 1993, and M.S. degree in Computing Science and Ph.D. degree in Engineering Science from the Southern Illinois University at Carbondale, Illinois, USA, in 1999 and 2000 respectively. She is an assistant professor of Computer Science at Southeast Missouri State University, USA. Her research interests include wireless networks, especially QoS issues, modeling and performance evaluation of the mobile ad hoc network, design and analysis of algorithms for distributed system, finite element analysis and boundary element analysis. She was a software engineer at PSC Motorola from January to August 2001.

XIANLING DONG received Bachelor of Engineering degree in 1996 and Master of Economics Science degree from Wuhan University, China. She received M.S. degree in Computer Science in 2001 from Southern Illinois University at Carbondale, USA. At present, she is an associate analyst in Southwestern Bell Communication, St. Louis, USA.