

# Fault Tolerant Routing in Mobile Ad Hoc Networks

Yuan Xue and Klara Nahrstedt

**Abstract**—The performance of ad hoc routing protocols will significantly degrade, if there are malfunctioned nodes in the network. Fault tolerant routing protocols address this problem by exploring the network redundancy through multipath routing. Designing an effective and efficient fault tolerant routing protocol is inherently hard, because the problem is NP-complete and the precise path information is unavailable. This paper solves this problem by presenting an end-to-end estimation-based fault tolerant routing algorithm  $E^2FT$ .  $E^2FT$  deploys two complementary processes: route estimation and route selection. Through end-to-end performance measurement, the route estimation process gives improving estimation results via iterations. Based on these estimation results, the route selection process decides a multipath route for packet delivery. The route selection is refined progressively with the increasingly accurate estimation result using “confirmation” and “dropping” procedures. Through theoretical analysis and simulation, we show  $E^2FT$  can achieve a high packet delivery rate with acceptable overhead.

## I. INTRODUCTION

MOBILE ad hoc networks are dynamically formed by mobile nodes with no pre-existing and fixed infrastructures. To provide end-to-end communication throughout the network, peer hosts cooperate with each other to handle network functions, such as packet routing.

Most existing ad hoc network routing protocols [1] assume that every node in the network functions well during packet delivery. However, such assumption usually does not hold in realistic environments. First, mobile hosts are severely resource-constrained devices. Overloaded hosts may lack the CPU cycles, buffer space or available bandwidth to forward packets. Second, mobile hosts may suffer software or hardware failures. The lack of centralized monitoring and management point in an ad hoc network makes it a challenging job to detect these failures. Such difficulty results in the long term existence of faulty nodes in the network. Faulty nodes can significantly affect the performance of routing in an ad hoc network. For example, if a faulty node participating in the routing operation drops data packets, then a large number of packets will be lost. Simulation result shows that the average packet delivery rate of DSR [2] degrades by about 30%, when 20% nodes are faulty.

We approach this problem by studying routing protocols that can tolerate these faulty nodes. Particularly, we seek to design a fault tolerant routing algorithm, which is able to provide certain packet delivery guarantee in the presence of

malfunctioned nodes. In fact, ad hoc networks are highly redundant networks. There are typically multiple paths between the source and the destination. Such network redundancy allows ad hoc networks to tolerate faulty nodes. To enable such capability, a fault-tolerant routing algorithm needs to explore the network redundancy through multipath routing. Without the knowledge of malfunctioned nodes, employing multipath routing [3] [4] blindly can only improve the packet delivery rate at the cost of extremely high packet duplication. Actually, designing an effective and efficient fault tolerant routing algorithm, where packet delivery rate can be guaranteed with low overhead, is an inherently hard problem. First, even with the perfect knowledge of faulty nodes, the fault tolerant routing problem can be formulated as a packet-delivery-rate-constrained, overhead-optimization problem. We prove this problem to be NP-complete. Second, the knowledge about faulty nodes is difficult to acquire in practice. Even if such knowledge can be acquired, it is typically inaccurate. This makes the fault tolerant routing problem even more challenging.

To address the above challenges, we present the end-to-end estimation-based fault tolerant routing algorithm ( $E^2FT$ ).  $E^2FT$  is a source routing algorithm. It employs two complementary processes: route estimation and route selection. Through end-to-end performance measurement, the route estimation process gives improving estimation results via iterations. Based on these estimation results, the route selection process decides a multipath route for packet delivery. The route selection is refined progressively with the increasingly accurate estimation result using “confirmation” and “dropping” procedures. The features of  $E^2FT$  include: (1) Comparing with “blind” multipath routing,  $E^2FT$  “learns” packet delivery properties of available paths through route estimation and use them in route selection. Such “smart” route selections can achieve high packet delivery rate and low overhead simultaneously. (2) Using end-to-end performance measurement, the route estimation in  $E^2FT$  requires no additional support from intermediate nodes. Thus, estimation results are not affected by the intermediate nodes, which may exhibit various faulty behaviors.

The main contributions of this paper are: (1) To the best of our knowledge, this is the first work that proposes, formulates and addresses the problem of fault tolerant routing in mobile ad hoc networks; (2) This paper presents a fault tolerant routing algorithm  $E^2FT$  as a solution to the fault tolerant routing problem. Through theoretical analysis and simulation studies,  $E^2FT$  is shown to be effective and efficient.

The remainder of the paper is organized as follows. Section II presents the network and node behavior model that we use throughout the paper. Section III formulates the problem of fault tolerant routing. Section IV presents the  $E^2FT$  algorithm

Yuan Xue and Klara Nahrstedt are affiliated with the Department of Computer Science, University of Illinois at Urbana-Champaign. Their email addresses are {xue,klara}@cs.uiuc.edu.

This research was supported by the ONR MURI NAVY CU 37515-6281 grant, and the NSF EIA 99-72884EQ grant. Any opinions, findings, and conclusions are those of the authors and do not necessarily reflect the views of the above agencies.

and its analytical results and discusses the implementation issues. Extensive simulation study of  $E^2FT$  is presented in Section V. Section VI presents the related works. Section VII concludes the paper.

## II. MODEL

### A. Network Model

We model an ad hoc network as a graph  $G = (V, E)$ , where  $V$  is the set of nodes, and  $E$  is the set of links. A path  $p$  consisting of  $l$  nodes  $v_1, v_2, \dots, v_l \in V$  is denoted as  $p = [v_1, v_2, \dots, v_l]$ , where  $(v_i, v_{i+1}) \in E$ ,  $i \in \{1, \dots, l-1\}$ . The length of the path  $p$  is denoted as  $L(p)$ . A multipath route  $\pi$  from a source node to a destination node, which consists of  $m$  paths  $p_1, p_2, \dots, p_m$ , is denoted as  $\pi = \{p_1, p_2, \dots, p_m\}$ . The length of the multipath route  $\pi$  is defined as  $L(\pi) = \sum_{i=1}^m L(p_i)$ . Furthermore, we assume that the network links are bidirectional, i.e., if  $(v_i, v_j) \in E$ , then  $(v_j, v_i) \in E$ .

### B. Faulty Node Model

To facilitate the algorithm presentation and analysis, we present a simplified faulty node model in this section. In this model, the behavior of a node  $v$  is represented by a random variable  $X(v)$ , which follows Bernoulli distribution. The outcomes of  $X(v)$  are defined as follows:

$$X(v) = \begin{cases} 1 & \text{if } v \text{ forwards the observed data packet correctly} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

The probability that the node  $v$  forwards a packet correctly, i.e.  $Pr\{X(v) = 1\}$ , is called *the packet delivery probability* of  $v$  and denoted as  $\gamma(v)$ .

The behavior of a path  $p = [v_1, v_2, \dots, v_l]$  can also be modeled through a random variable  $Y(p)$ , where

$$Y(p) = \begin{cases} 1 & \text{if } p \text{ successfully delivers the observed data packet} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Thus,  $Y(p) = \prod_{i=1}^l X(v_i)$ . Let us assume faulty nodes on  $p$  behave independently.  $Y(p)$  also follows Bernoulli distribution. The *packet delivery probability*  $\gamma(p)$  of the path  $p$  is calculated as  $\prod_{i=1}^l \gamma(v_i)$ .

For a multipath route  $\pi = \{p_1, p_2, \dots, p_m\}$ , the *packet delivery probability*  $\gamma(\pi)$  of  $\pi$  is defined as the probability that at least one copy is received, if duplicated packets are sent along paths  $\{p_1, p_2, \dots, p_m\}$ .  $\gamma(\pi)$  can be calculated as  $1 - \prod_{i=1}^m (1 - \gamma(p_i))$

## III. FAULT TOLERANT ROUTING PROBLEM

The existence of faulty nodes in ad hoc networks can significantly affect the packet delivery rate of the routing protocols. The goal of designing a fault tolerant routing algorithm is to provide certain packet delivery rate guarantee in the presence of faulty nodes. To achieve this goal, the routing algorithms explore network redundancy through multipath routing. In multipath routing, duplicate packets are sent along the multiple paths between the source and the destination. The packet is regarded as successfully delivered, if one copy of the packets is received. Without the knowledge of faulty nodes, employing

multipath routing blindly can introduce extremely high overhead into the network. Thus, the most critical question that needs to be addressed by a fault tolerant routing algorithm is how to select the paths so that packet deliver rate can be guaranteed through minimum packet duplication.

In this paper, we confine our research to this problem within the scope of source routing algorithms, where the source node makes centralized decision on path selection. Furthermore, we assume that the source knows multiple paths between itself and the destination through route discovery procedure. We denote this set of paths as  $\Omega$ . Designing route discovery algorithms is out of the scope of this paper, readers are referred to [2] for such information.

Let us assume that the source has the perfect knowledge of the delivery probability  $\gamma(p)$  of each path  $p \in \Omega$ . The fault tolerant routing problem can be formulated as a packet-delivery-rate-constrained overhead-optimization problem: Given the path set  $\Omega$ ,  $\gamma(p)$  for each path  $p \in \Omega$ , expected packet delivery rate guarantee  $\gamma^*$ , find a subset  $\pi^* \subseteq \Omega$  that satisfies:

$$\text{Constraint:} \quad \gamma(\pi^*) \geq \gamma^* \quad (3)$$

$$\text{Optimization:} \quad \forall \pi \subseteq \Omega, L(\pi^*) \leq L(\pi) \quad (4)$$

In this problem formulation, the packet delivery rate constraint shows the requirement on high packet delivery rate, which is the effectiveness of the fault tolerant routing algorithm. The goal of optimizing the length of the multipath routes is to minimize the duplicate packets at the network level, as each data packet introduces  $L(\pi)$  duplicated copies into the network.

**Theorem 1:** *The packet-delivery-rate-constrained overhead-optimization (PCOO) problem is NP-complete.*

*Proof:* We only give sketched proofs for the theorems and lemmas in this paper, due to the space limitations. The PCOO problem can be reduced to 0-1 knapsack problem. Let an arbitrary instance of 0-1 knapsack problem be given by  $u \in U$ , where  $U$  is a finite set, a size  $s(u)$  and a value  $v(u)$  of  $u$ , a size constraint  $B$  and a value goal  $K$ . An instance of PCOO can be constructed if  $L(p) = v(u)$ ,  $\gamma(p) = 1 - 2^{-s(u)}$ ,  $\gamma^* = 1 - 2^{-B}$ , optimization goal  $K' = K$ .  $\square$

In practice, this problem is even more challenging, because the source does not know the behaviors of other nodes, thus has no precise knowledge about  $\gamma(p)$ . In summary, designing an effective and efficient fault tolerant routing problem faces two main difficulties: (1) The source has no precise knowledge about the paths' packet delivery probabilities; (2) Even if such knowledge is available, this problem is still inherently hard due to its NP-completeness complexity.

## IV. $E^2FT$ : ALGORITHM AND ANALYSIS

To address the above challenges from the fault tolerant routing problem, we propose an end-to-end estimation-based routing algorithm ( $E^2FT$ ) as a solution.

### A. Algorithm

$E^2FT$  consists of two processes: route estimation and route selection. The route estimation process estimates the delivery

probability of the available paths  $\hat{\gamma}(p)$ ; The route selection process selects a set of paths  $\pi^* \subseteq \Omega$  so that  $L(\pi^*)$  can be reduced under constraint  $\gamma(\pi^*) \geq \gamma^*$ , where  $\gamma^*$  is the expected packet delivery probability guarantee.

**Route estimation.** The source node estimates the packet delivery probability  $\hat{\gamma}(p)$  of path  $p$  by sending data packets along  $p$ , and measuring its packet delivery rate. The destination helps the source with the estimation procedure by sending feedback with information how many packets were received. Formally, given the number of sent packets  $n$  and the number of received packets  $n'$  along path  $p$ , the source estimates  $\hat{\gamma}(p) = \frac{n'}{n}$ . The accuracy of the estimation result depends on the number of packets used for estimation. Intuitively, larger  $n$  will give more accurate result of  $\hat{\gamma}(p)$ . Yet, it will also cost more time and introduce more estimation overhead. To represent the estimation result and its accuracy, we define *raw estimation* as a tuple  $\langle \hat{\gamma}(p), n \rangle$ .

In practice, to balance the cost, the promptness, the stability and the accuracy of the estimation result,  $\hat{\gamma}(p)$  is acquired progressively through iterations. In each iteration, a bunch of  $b$  packets are sent. Initially  $\hat{\gamma}(p)^0 = 0$ . In the  $i$ th iteration,  $\hat{\gamma}(p)^i$  is calculated as  $\hat{\gamma}(p)^i = (1 - \frac{1}{i}) \cdot \hat{\gamma}(p)^{i-1} + \frac{1}{i} \cdot \frac{b'}{b}$ , where  $b'$  is the number of packets received in the  $i$ th iteration. Note that with this estimation method, the past behavior of the path  $p$  has the same weight as its current behavior. Thus, this estimation needs the path  $p$  behave consistently in packet delivery. We will relax this assumption in Section IV.E

The estimation results of different paths may be based on different  $n$  values. Thus they have different accuracies. To facilitate the path selection, we need a unified path delivery rate estimation so that paths can be selected fairly.

**Definition:** Given  $\alpha$ , the  $\alpha$ -estimation  $\hat{\gamma}_\alpha(p)$  of a path  $p$  is

$$\hat{\gamma}_\alpha(p) = \max\{\hat{\gamma}(p) - \frac{1}{\sqrt{4n(1-\alpha)}}, 0\} \quad (5)$$

In the  $\alpha$ -estimation,  $\alpha$  is a parameter indicating the confidence of such an estimation.

**Lemma 1:** (Property of  $\alpha$ -estimation)  $\hat{\gamma}_\alpha(p)$  satisfies:

$$\Pr\{\gamma(p) \geq \hat{\gamma}_\alpha(p)\} \geq \alpha \quad (6)$$

*Proof:* From weak law of large number, we have  $\Pr\{|\hat{\gamma}(p) - \gamma(p)| \leq \epsilon\} \geq 1 - \frac{1}{4n\epsilon^2}$ . Thus, if  $1 - \frac{1}{4n\epsilon^2} = \alpha$ , then  $\epsilon = \frac{1}{\sqrt{4n(1-\alpha)}}$ . We have  $\Pr\{\hat{\gamma}(p) - \epsilon \leq \gamma(p)\} \geq \Pr\{\hat{\gamma}(p) - \epsilon \leq \gamma(p) \leq \hat{\gamma}(p) + \epsilon\} \geq 1 - \frac{1}{4n\epsilon^2}$ , which gives (6)  $\square$

**Route selection.** Based on the unified  $\alpha$ -estimation results, the route selection process can select paths on a fair basis. Since the accuracy of the path estimations is increased through iterations, the route selection also refines its path selection progressively with the increasingly accurate estimation results. Let  $\pi$  be the current set of paths used for packet delivery.

Initially, since no estimation result is available,  $\hat{\gamma}(p_i) = 0$ ,  $\forall p_i \in \pi$ . The selected path set  $\pi = \Omega$ . That is, all paths in  $\Omega$  are selected for packet delivery. With increasingly accurate estimation results,  $E^2FT$  refines its path selection through two procedures – confirmation and dropping.

*Confirmation* is a procedure that, through “accurate enough” estimation, a single path is verified to be able to deliver the packets with the expected rate  $\gamma^*$  alone.  $E^2FT$  confirms a path  $p$ , if the confirmation condition is satisfied:

$$\hat{\gamma}_\alpha(p) \geq \gamma^* \quad (7)$$

Once a path  $p$  is confirmed, it becomes the only path in use. That is  $\pi = \{p\}$ . No further estimation is necessary on  $p$ . The destination does not need to send feedback any more. If there are multiple paths that satisfy the confirmation condition, the one with the shortest length will be used.

*Dropping* is a greedy decision procedure that removes the unnecessary paths from  $\pi$ . This procedure picks a path  $p_{min} \in \pi$  with minimum  $\alpha$ -estimation, i.e.,  $\hat{\gamma}_\alpha(p_{min}) \leq \hat{\gamma}_\alpha(p)$ ,  $\forall p \in \pi$ . If path set  $\pi' = \pi - \{p_{min}\}$  satisfies the following dropping condition (8), then  $p_{min}$  will be removed from  $\pi$ ,  $\pi'$  will be used for future packet delivery.

$$\hat{\gamma}_{\alpha \frac{1}{m}}(\pi') \geq \gamma^* \quad (8)$$

where  $\hat{\gamma}_{\alpha \frac{1}{m}}(\pi') = 1 - \prod_{p \in \pi'} (1 - \hat{\gamma}_{\alpha \frac{1}{m}}(p))$ .

## B. Analysis

We analyze the packet delivery rate and the overhead of  $E^2FT$  in this section.

**Lemma 2:** If confirmation procedure is used and path  $p$  is confirmed, then the packet delivery probability  $\gamma(\pi)$  of resulting path set  $\pi = \{p\}$  satisfies:

$$\Pr\{\gamma(\pi) \geq \gamma^*\} \geq \alpha \quad (9)$$

*Proof:* By the property of  $\alpha$ -estimation (6) and the confirmation condition (7), we have (9).  $\square$

**Lemma 3:** If dropping procedure is used and  $\pi$  is the resulting path set, then  $\pi$  satisfies:

$$\Pr\{\gamma(\pi) \geq \gamma^*\} \geq \alpha \quad (10)$$

*Proof:* Using probability operations on (6), we have  $\Pr\{\gamma(\pi) \geq \gamma_{\alpha \frac{1}{m}}(\pi)\} \geq \alpha$ . By dropping condition (8), we have (9).  $\square$

If neither confirmation nor dropping procedure is used, then  $\Omega$  is used for packet delivery all the time. In this scenario, it is possible that  $\gamma(\Omega) < \gamma^*$ . This scenario may happen due to two reasons: (1) the network does not have enough redundancy to tolerate the faulty nodes and provide the expected packet delivery rate; (2) The route discovery is inadequate so that  $\Omega$  does not include all the paths between the source node and the destination. In the first case, the property of the network puts an upper bound on the packet delivery rate; In the second case, it is the route discovery mechanism that puts the bound. In either case, no route selection algorithm can improve the packet delivery rate. To summarize, we have the following conclusion on the packet delivery probability of  $E^2FT$ :

**Theorem 2:** The packet delivery probability of  $E^2FT$  has a lower bound of  $\min\{\gamma(\Omega), \gamma^*\}$ .

*Proof:* By Lemma 1 and Lemma 2, we have this result.  $\square$

The theorem shows that  $E^2FT$  gives “soft” packet delivery guarantee. That is, if the network and the route discovery procedure provide enough redundancy,  $E^2FT$  can achieve the expected packet delivery rate; Otherwise,  $E^2FT$  will try its “best effort” to deliver packets.

The  $E^2FT$  algorithm introduces two types of overhead into the network: control packets (e.g., the feedback packets from the destination) and duplicated data packets. Here, we focus on the second type. Comparing with the optimal solution which

is derived based on perfect packet delivery probability information,  $E^2FT$  may incur additional packet duplications due to the estimation process and the sub-optimal choice. Since finding the optimal solution in a generic network environment is NP-complete, we consider a simplified scenario where the network has a good path for analysis purpose. Comprehensive study of  $E^2FT$ 's overhead will be done through simulations. Here we are interested in the number of packets used for confirming the good path, as it indicates the estimation cost of  $E^2FT$ :

**Theorem 3:** *The number of packets  $n$  required to confirm a good path is upper bounded by  $\frac{1}{4(1-\alpha)(\hat{\gamma}(p)-\gamma^*)^2}$ .*

*Proof:* The result can be derived from the definition of  $\alpha$ -estimation (5) and the confirmation condition (7).  $\square$

### C. Example

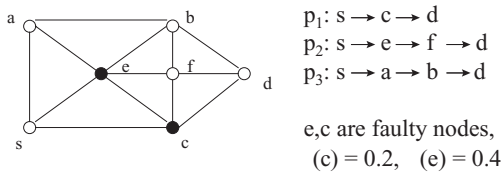


Fig. 1. Network Topology

In this section, we illustrate  $E^2FT$  algorithm and compare it with DSR and multipath routing algorithms through examples. The network topology used in the example is shown in Figure 1.

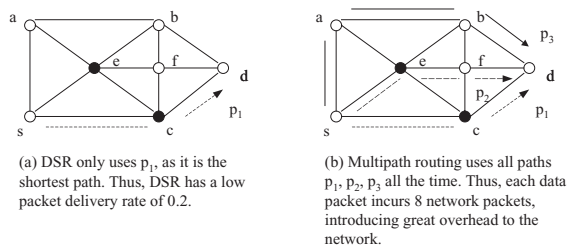


Fig. 2. DSR and multipath routing

First, we show DSR and multipath routing algorithm in Figure 2(a) and Figure 2(b). DSR uses the shortest path to route packets. When there exist faulty nodes on this path, DSR suffers low packet delivery rate. Multipath routing algorithm uses all paths to route packets all the time. Although high packet delivery rate is achieved through multipath routing, duplicated data packets also incur high overhead into the network.

Next, we illustrate  $E^2FT$  algorithm in Figure 3. The route selection of  $E^2FT$  is a dynamic procedure. Through improving route estimation results,  $E^2FT$  can use fewer paths for packet delivery via iterations (by using dropping procedure, 2nd iteration in the example) or directly “jump” to one path (by using confirmation procedure, 3rd iteration in the example). In this way,  $E^2FT$  gracefully achieves high packet delivery rates with low overhead.

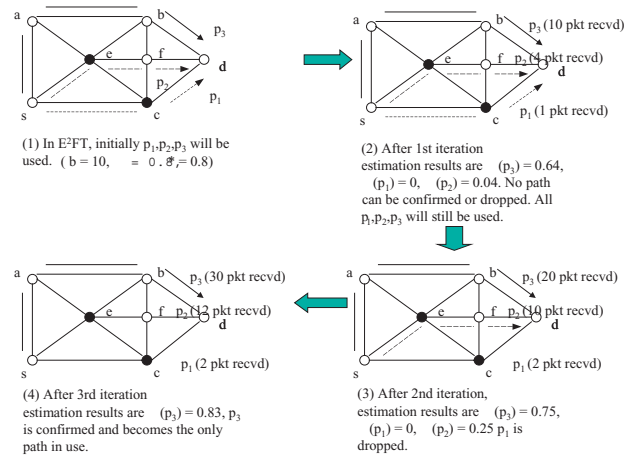


Fig. 3.  $E^2FT$  routing

### D. Mobility

Above discussions on  $E^2FT$  only consider static network environments, where the path set  $\Omega$  between the source and the destination is fixed. When node mobility is taken into consideration, current  $E^2FT$  algorithm can be less efficient. The reasons are: (1) If a path in use is broken, the packet delivery probability  $\gamma(\pi)$  of current path set  $\pi$  is changed, and may no longer satisfy the expect rate  $\gamma^*$ . Thus, the route needs to be re-selected; (2)  $E^2FT$  evaluates the packet delivery probability  $\hat{\gamma}(p)$  of a path  $p$  through its end-to-end performance. It does not explicitly evaluate the packet delivery probability  $\hat{\gamma}(v)$  of each individual node  $v \in p$ . Thus, when the path  $p$  is broken, the estimation result  $\hat{\gamma}(p)$  is wasted. Without historical estimation results, the re-selection needs a new estimation procedure, which may be expensive.

To address this problem, we present optimization for  $E^2FT$  to accommodate node mobility. The essential idea is to calculate the packet delivery probability  $\hat{\gamma}(v)$  of each node  $v$  on the path  $p$  based on  $\hat{\gamma}(p)$ .  $\hat{\gamma}(v)$  will be kept when  $p$  is broken and will be used to assist future route selection. In particular,  $\forall v \in p, \hat{\gamma}(v) = \hat{\gamma}(p)$ ; The number of sent packets  $n_v$  on  $v$  inherits  $n$  from  $p$ 's raw estimation, i.e.,  $n_v = n$ . This estimation method is a conservative one, as  $\gamma(v) \geq \gamma(p)$  always holds. If  $v$  appears in multiple paths  $p_1, p_2, \dots, p_k$ , and the according estimation results derived from  $\hat{\gamma}(p_1), \hat{\gamma}(p_2), \dots, \hat{\gamma}(p_k)$  are  $\hat{\gamma}(v)^1, \hat{\gamma}(v)^2, \dots, \hat{\gamma}(v)^k$ , then  $\hat{\gamma}(v) = \hat{\gamma}(v)^{max}$ , where  $\hat{\gamma}_\alpha(v)^{max}$  is the maximum value of  $\hat{\gamma}_\alpha(v)^i, i \in \{1, \dots, k\}$ . With the calculated  $\hat{\gamma}(v)$ , the raw estimation of a new path  $p$  does not start with  $\langle \hat{\gamma}(p), n \rangle = \langle 0, 0 \rangle$ . Instead, if  $p = [v_1, v_2, \dots, v_l]$ , its raw estimation is  $\langle \prod_{i=1}^l \hat{\gamma}(v_i), \min_{i=1}^l \{n_{v_i}\} \rangle$ . Note that if there exists a node on path  $p$  whose estimation is  $\langle 0, 0 \rangle$ , then  $p$ 's raw estimation is still  $\langle 0, 0 \rangle$ .

### E. Discussion

Now we discuss the implementation issues of  $E^2FT$ .

1) On-demand routing. We implement  $E^2FT$  as an on-demand routing protocol, because it has been shown to

be much more efficient than proactive routing protocols. In the on-demand routing protocol, the source only selects a route when it has a packet to send. The route selection is based on the path estimation result at the current time, which will be updated each time the feedback from the destination is received. Due to the round trip time difference, the estimation may not be updated at the same time. As a result,  $E^2FT$  is in favor of the path with short RTT time, which is a desirable property.

- 2) Feedback. The route estimation process needs the destination node to send feedback with the information of how many packets were received along the path  $p$ . The feedback can be implemented either through network acknowledgement or piggybacking onto TCP acknowledgement packets. Such packets need to be sent along  $p$  in the reverse direction, which requires the wireless links in the network to be bi-directional (our network assumption).
- 3) Overhead control. Although  $E^2FT$  is much more efficient than blind multipath routing algorithm, it can still incur large packet overhead due to its conservative estimation method, which is designed to achieve a strict theoretical packet delivery rate guarantee. This problem becomes severe, when  $\alpha$  is very close to 1. In practice, a more relaxed estimation method can still achieve a similar packet delivery rate, yet with a lower overhead. To do that, we set an upper bound  $n^*$  to the number of estimation packets. We show how a small value of  $n^*$  can achieve high packet delivery rate in simulation study.
- 4) Soft state. The behavior of mobile nodes may be dynamically changing in the real network environment: On one hand, a functional node at current time may become a faulty node at a later time due to a software/hardware failure; On the other hand, a faulty node may become a functional node after being repaired. To accommodate such node behavior dynamics, we have two approaches: (a) For long term node dynamics, we use soft state for path/node estimation; (b) For node dynamics during the estimation process, we use  $\hat{\gamma}(p)^i = (1 - \beta) \cdot \hat{\gamma}(p)^{i-1} + \beta \cdot \frac{b'}{b}$  to estimation  $\hat{\gamma}(p)$ , where  $\beta$  can be tuned to reflect the importance of recent node behavior.

## V. SIMULATION RESULTS

In order to evaluate the performance of the  $E^2FT$ , extensive simulations have been conducted in ns-2 (network simulator) to compare the performance of  $E^2FT$  with DSR and “blind” multipath routing.

The simulated network is deployed in a flat square with 700 meters on each side. The network has 50 nodes, which include both well-behaved nodes and faulty nodes. The number of faulty nodes  $M$  is a simulation parameter. By varying  $M$ , we show how  $E^2FT$  provides packet delivery rate guarantee under different network faulty levels. Both well-behaved nodes and malfunctioned nodes use the IEEE 802.11 radio and MAC model. Faulty nodes drop all data packets after participating

in the routing operation. Each node moves using “random waypoint” model. In our simulation, we choose the maximum speed of mobile nodes as 20m/s. The pause time is a simulation parameter to show how  $E^2FT$  accommodates to node mobility. Each simulation runs 500 seconds. During the simulation period, CBR traffic will be generated randomly between a pair of nodes with rate 16K byte/second. Default values of the design parameters used in the simulation are listed as follows:  $n^* = 15$ ,  $\alpha = 0.8$ ,  $\gamma^* = 0.8$ ,  $b = 5$ ,  $M = 10$ , pause time = 100 second.

### A. Packet Delivery Rate and Overhead

First, we show the effectiveness of  $E^2FT$  by comparing its packet delivery rate with DSR and the multipath routing. As shown in Figure 4, the packet delivery rate of  $E^2FT$  is comparable with the multipath routing algorithm and is constantly higher than DSR. The packet delivery rates of both  $E^2FT$  and the multipath routing algorithm drop slightly with the increasing number of malfunctioned nodes. This is mainly because the network redundancy is not high enough to tolerate that many faulty nodes.

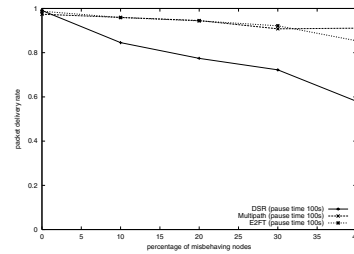


Fig. 4. Packet delivery rate of  $E^2FT$  in comparison with DSR and multipath routing

Next, we show the efficiency of  $E^2FT$ . Figure 5 plots the overhead of  $E^2FT$  in comparison with DSR and multipath routing. From the figure, we can observe that, although incurring more overhead than DSR,  $E^2FT$  has much lower overhead than the “blind” multipath routing.

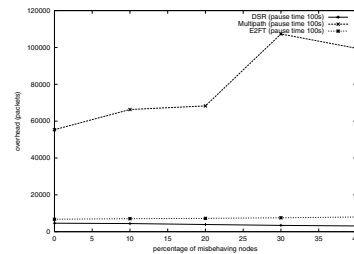


Fig. 5. Overall overhead of  $E^2FT$  in comparison with DSR and multipath routing

### B. Overhead Control

Now we study the impact of overhead control parameter  $n^*$  on the packet delivery rate. From Figure 6, we can see the packet delivery rate increases with the increment of  $n^*$ , when

$n^* \leq 15$ . This result is intuitive: large  $n^*$  allows the route estimation process to give more accurate estimation result, thus avoiding false route selection decision. Yet, from Figure 6, we also observe that when  $n^* > 15$ , larger  $n^*$  can decrease the packet delivery rate. This is mainly because, large  $n^*$  will lengthen the route estimation time, thus incurring a large number of duplicated packets into the network, these packets contend with each other, decreasing the packet delivery rate.

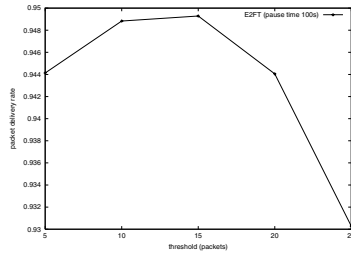


Fig. 6. Packet delivery rate of  $E^2FT$  with different  $n^*$

### C. Accommodation to Node Mobility

Now we study how  $E^2FT$  accommodates to node mobility. Figure 7 shows that node mobility does not greatly affect the packet delivery performance of  $E^2FT$ . Figure 8 plots the overhead of  $E^2FT$  and its node-mobility-optimization version  $E^2FT-O$  and compares it with DSR. The figure shows that  $E^2FT$  without optimization has twice overhead as DSR when the node moves aggressively, while  $E^2FT-O$  can reduce the overhead to about 60%.

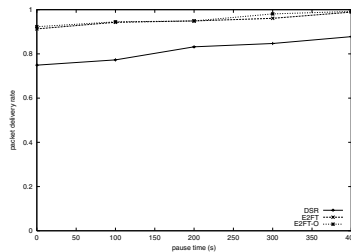


Fig. 7. Packet delivery rate of  $E^2FT$  and  $E^2FT-O$  in comparison with DSR

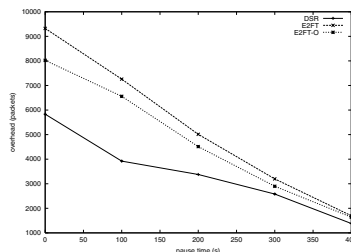


Fig. 8. Overall overhead of  $E^2FT$  and  $E^2FT-O$  in comparison with DSR

In summary, the results of  $E^2FT$  show that, by using route estimation and “smart” route selection,  $E^2FT$  can achieve high packet delivery rate which is comparable to multipath routing, but reduce the overhead to an acceptable level.

## VI. RELATED WORK

There exist related works that address the problem of malicious nodes in ad hoc networks. Security-Aware Routing (SAR) by S. Yi et al. [5] separates nodes into trusted and non-trusted and forward packets only through nodes that share a priori trust relationship. [6] by Hu and Johnson presents a secure on-demand ad hoc network routing protocol. Our work differs from these approaches as follows: (1) We make a stronger assumption on the node behavior. Instead of providing a solution for malicious nodes, which may exhibit arbitrary Byzantine behavior, we only consider benign failure nodes, which only drop packets. (2) Under such a stronger node behavior model, our solution does not require any heavy-weighted security support from the network as related work does.

Zhou and Haas [7] first point out that routing protocols can take advantage of the inherent redundancy in ad hoc networks, which provides multiple routes between nodes, and defend against route disruption attacks. One main limitation of this approach is the high overhead introduced by the multipath routing.  $E^2FT$  is shown to outperform the multipath routing with lower overhead but similar packet delivery behavior.

## VII. CONCLUSION

In this paper, we propose and formulate the problem of fault tolerant routing in mobile ad hoc networks. This problem is inherently hard due to its NP-completeness complexity and the unavailability of precise faulty node information. To address these challenges, we present an end-to-end estimation-based fault tolerant routing algorithm  $E^2FT$ . Through theoretical analysis and simulation study,  $E^2FT$  is shown to obtain high and stable packet delivery rate with acceptable additional overhead under various network environments.

## REFERENCES

- [1] E. Royer and C. Toh, “A review of current routing protocols for ad-hoc mobile wireless networks,” *IEEE Personal Communications Magazine*, pp. 46–55, 1999.
- [2] D. B. Johnson and D. A. Maltz, “Dynamic source routing in ad hoc wireless networks,” *Mobile Computing*, pp. 153–181, 1996.
- [3] S. J. Lee and M. Gerla, “Split Multipath Routing with Maximally Disjoint Paths in Ad hoc Networks,” in *ICC*, 2001, pp. 3201–3205.
- [4] E. P. A. C. R. Leung, J. Liu and B. Li, “On-Demand Multipath Routing for Mobile Ad Hoc Networks,” in *Proceedings of the 26th International Conference on Local Computer Network*, 2001.
- [5] P. N. S. Yi and R. Kravets, “Security-Aware Ad-Hoc Routing for Wireless Networks,” in *Report No. UIUCDCS-R-2001-2241, UILU-ENG-2001-1748*, 2001.
- [6] A. P. Yih-Chun Hu and D. B. Johnson, “Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks,” in *To appear Mobicom 2002*, 2002.
- [7] L. Zhou and Z. Haas, “Securing ad hoc networks,” *IEEE Network Magazine*, vol. 13, 1999.