# TCP Performance over Mobile Ad-hoc Networks - A Quantitative Study

Vaidyanathan Anantharaman and Raghupathy Sivakumar
School of Electrical and Computer Engineering
Georgia Institute of Technology
(Shorter version of this paper appeared in ACM SIGMETRICS 2002)
Email: siva@ece.gatech.edu

**Abstract**

In this paper we study the performance of TCP over mobile ad-hoc networks. We present a comprehensive set of simulation results and identify the key factors that impact TCP's performance over ad-hoc networks. We use a variety of parameters including *link failure detection latency, route computation latency, packet level route unavailability index, and flow level route unavailability index* to capture the impact of mobility. We relate the impact of mobility on the different parameters to TCP's performance by studying the *throughput*, *loss-rate*, and *retransmission timeout values* at the TCP layer. We conclude from our results that *existing approaches to improve TCP performance over mobile ad-hoc networks have identified and hence focused only on a subset of the affecting factors*. In the process we identify a comprehensive set of factors influencing TCP performance. Finally, using the insights gained through the performance evaluations, we propose a framework called *Atra* consisting of three simple and easily implementable mechanisms at the MAC and routing layers to improve TCP's performance over ad-hoc networks. We demonstrate that *Atra* improves on the throughput performance of a default protocol stack by 50-100%.

## I. INTRODUCTION

Ad-hoc networks are multi-hop wireless networks that can operate without the services of an established backbone infrastructure. The mobile-stations that form the ad-hoc network perform the additional role of routers. Since each station in the network is potentially mobile, the topology of an ad-hoc network can be highly dynamic. Such networks have traditionally been considered to have applications in the military and disaster relief environments. Recent applications in regular wireless packet data environments [1] along with capacity, energy, and range arguments for the use of such networks in tandem with the existing cellular infrastructure [2], [3], [4], [5] have increased the significance of this class of networks.

Over the past decade, a tremendous amount of research has focused on developing network protocols for ad-hoc networks [6], [7]. While the IEEE 802.11 multiple access protocol is primarily considered for the medium accesss control (MAC) layer, robust and simple protocols such as dynamic source routing (DSR) and ad-hoc on-demand distance vector (AODV) have emerged as the primary mechanisms at the routing layer.

With the research at the MAC and routing layers gaining maturity, some researchers have lately shifted focus to the transport layer performance in ad-hoc networks [8], [9], [10]. With TCP (Transmission Control Protocol) being the primary transport protocol in use in the current Internet, studying TCP's performance over ad-hoc networks is of obvious interest. Recent work in this area [8], [9], [10] has investigated the impact of ad-hoc network characteristics on TCP's performance and have proposed schemes that help TCP overcome the negative impact of such characteristics as random wireless loss and mobility. While we discuss related work in detail later in the paper, the primary mechanism in related works proposed to handle mobility involves sending an explicit link failure notification (ELFN) to the source from the link failure point. The source, upon receiving the ELFN *freezes* TCP's timers and state, re-computes a new route to the destination, and either releases the timers and state or re-starts them from their respective initial values.

In this paper we argue that while ELFN is a necessary mechanism to improve TCP performance over ad-hoc networks, it is by no means sufficient. We identify the different factors that affect TCP's performance and demonstrate that some of the factors not addressed by the ELFN mechanism in fact have a greater impact on TCP's performance. We use extensive simulations to study the impact of a variety of parameters including the *number of route failures*, *route re-computation time*, *time a packet spends without a route at the source*, and *link failure detection time*. We show the impact of the above parameters on TCP by measuring the *throughput*, *loss rate*, and *retransmission timeout values* at the TCP layer. Based on our simulation results, we identify a comprehensive set of factors that need to be addressed to enhance TCP's performance over ad-hoc networks.

Using the insights gained through the performance evaluation, we propose the *Atra* framework consisting of three simple and easily implementable mechanisms at the MAC and routing layers that significantly improve the performance of TCP. We compare the efficacy of the proposed schemes against basic TCP, TCP with ELFN, and when they act in tandem with ELFN. We also show in isolation, the performance enhancements achieved by each of the individual components of the proposed framework. The contributions of the paper are thus two-fold:

- We identify a comprehensive set of factors that affect TCP's performance over ad-hoc networks. We show that the factors addressed by mechanisms like ELFN are merely a subset of the identified set.
- Using the insights gained through the performance evaluation, we propose a framework of three simple mechanisms at the MAC and routing layers to enhance TCP's performance. We demonstrate the improvements achieved when the framework is used in isolation and in tandem with the ELFN scheme.

The rest of the paper is organized as follows: In Section II we describe our simulation environment, metrics, and protocols used. In Section III we present the simulation results and interpret the results to identify the key factors that affect TCP performance. In Section IV we outline the *Atra* framework and show how the mechanisms enhance TCP's performance. In Section V we discuss some issues left unaddressed in the paper and plausible limitations of the proposed framework. Finally in Section VI we summarize related work and conclude the paper.

## II. Simulation Model

We use the *ns2* [11] network simulator (version 2.1b7a) for all our simulations. Each data point is averaged over 50 simulation runs with different seeds. The rest of the section presents the details of the simulation environment, the algorithms and protocols used, and the metrics used in the performance evaluation, in that order.

### A. Simulation Environment and Protocols

• *Physical Layer:* The physical layer implementation consists of a combination of the free space propagation model and the two-ray ground reflection model [12]. *Reference distance* is set to a value of 100m in the free space model. Signal attenuates as $1/r^2$ in the model. Outside of the *reference distance*, the ground reflection model is used where the signal falls off as $1/r^4$. The physical layer follows IEEE 802.11's *Direct Sequence Spread Spectrum* (DSSS) specifications. We use a single channel and the data rate of the channel is set to 2 Mbps.

• *Medium Access Control:* The IEEE 802.11 medium access control protocol in the distributed coordinated function (DCF) mode is used as the MAC layer. Briefly, IEEE 802.11 is based on the CSMA/CA MAC scheme. For every unicast transmission, the sender sends a request to send (RTS) to the receiver. If the receiver is able to receive the packet, it replies with a clear to send (CTS). The sender then sends the data packet (DATA), and upon correct receipt of the data, the receiver sends an acknowledgment (ACK) back to the sender. The decision at the sender to send an RTS and at the receiver to send a CTS is taken after sensing both the local channel state and the channel state at the neighbors (through the NAV vector). Broadcast transmissions do not have a handshake and are performed solely after making sure that the channel state (both local and virtual) is idle.

• *Routing:* We use the dynamic source routing (DSR) protocol for routing. However, we believe that the same results shall apply as long as an on-demand routing protocol is used. Also, recent studies have shown that on-demand protocols (DSR, AODV) are becoming increasingly popular as the de facto routing protocols on mobile ad hoc networks. While the implementation of the mechanisms within the *Atra* framework presented in Section IV do depend on the choice of the routing protocol, the fundamental ideas behind the mechanisms can be easily incorporated in any given routing protocol. Briefly, in DSR a source S wishing to determine a route to a destination D floods the network with a route request (RREQ) packet. Each intermediate node, when forwarding the flooded RREQ, stamps its identifier onto the packet. When D receives the RREQ message, the packet contains a list of nodes that it traversed through. D sends this route back to S which then uses source routing to route packets [1].

• *Topology and Mobility Model:* We use the *setdest* tool in *ns2* to generate the topology for the simulations. All simulations are performed for a 1000m x 1000m grid consisting of 100 nodes. The nodes are distributed randomly over the grid. The *way-point* mobility model is used to generate the mobility patterns for the nodes. Briefly, nodes pick random destinations inside the grid and move toward the destinations using a speed $s$ that is randomly distributed between 0 and MAXSPEED. When the nodes reach their destination, they may remain idle at the destination for a PAUSE amount of time before they repeat the cycle all over again. MAXSPEED and PAUSE are input parameters to the mobility model. We use pedestrian (1 m/s), and vehicular (10 m/s and 20 m/s) speeds in our simulations. While the choice of the mobility model is based on its availability and popularity, we believe that the conclusions drawn in the paper and the consequent mechanisms proposed will remain valid for other mobility models too. Our current work is investigating the impact of the choice of the mobility model.

• *Traffic Generation:* We use *ftp* as the application over TCP for all the flows in the network. Rationale for choosing a backlogged application such as *ftp* was to help isolate the impact of the dynamics of the underlying network on TCP. We study the effect of loading on the network, by investigating scenarios with 1 and 25 flows respectively. We discuss why the impact of load is significant when studying TCP's performance over ad-hoc networks. Source-destination pairs for each flow are chosen randomly from the set of 100 nodes in the network. All flows last for the entire simulation run of 100 seconds. Packet sizes of 512 bytes are used. The TCP-NewReno version was used for all simulations.

• *Explicit Link Failure Notification:* While several variations of ELFN have been proposed in related work, we use the ELFN mechanism proposed by Vaidya et.al. in [8][2]. This mechanism works as follows: when a route failure occurs, an ELFN message is sent back to the source. Upon receipt of this message, the source freezes its current state in terms of window size and timers. The source sends probe packets (the first packet that is yet to be unacknowledged) to determine if a new route has been computed. When an ACK arrives (indicating the availability of a new route), the source de-freezes the state and continues as normal. The periodicity of the probe-packets is set to the minimum of the values used in [8] to elicit the best performance from ELFN.

[1] Note that the descriptions of both IEEE 802.11 and DSR are by no means complete and are provided here to give the unfamiliar reader some context.

[2] We thank the authors of [8] for making the ns2 simulation code for ELFN publicly available.

### B. Metrics

We present three categories of metrics to illustrate the impact of mobility on TCP's performance: (i) Mobility induced factors relating to TCP performance, (ii) Impact of factors (in (i)) on routing, and (iii) Impact of factors (in (i)) on TCP's performance in terms of throughput, retransmission timeout values, and loss percentages. We elaborate on each of these metrics in the rest of the section:

- *Mobility induced factors:* We measure two metrics under this category: (a) *MAC Failure Detection Latency (hereafter referred to as MAC-FD latency)*: This is the average time taken to detect a link failure at the MAC layer, and is measured as the amount of time spent in the MAC layer on an average, by the packet that encounters a link failure, (b) *Route Computation Latency (hereafter referred to as RC latency)*: This is the Average time taken to re-compute a new route when link failure is detected. This is measured as amount of time taken from when the route request is sent, upto when a valid route reply for that request, is received.
- *Impact of factors on route availability:* We study the impact of the above parameters on the *route availability for connections at both the packet level and the flow level. Packet level route unavailability index (hereafter referred to as PRU index)* is measured as a function of the average time spent by a packet at the source's buffer. The *Flow level route unavailability index (hereafter referred to as FRU index)* is measured as the absolute time during the simulation, for which a flow does not have a route to the destination.
- *Impact of factors on TCP's performance:* In order to determine the impact of the mobility induced factors on TCP's performance, we monitor three parameters at the TCP layer, namely: *(a) Throughput, (b) Maximum retransmission timeout values, and (c) Loss percentage experienced*. While throughput is a direct metric of TCP's performance, the impact of the factors on the latter two parameters provides us with an indication of how the performance degradation comes about. Since the observed throughput depends on the hop length, in order to mask the effect of hop length from the analysis, we calculate the average throughput normalized to the average hop length. We call this metric *normalized throughput*, and measure it as follows: A linear chain of nodes is simulated and throughput $T_i$ is obtained for connections traversing various hops lengths $H_i$. In our regular simulation results, if the hop count observed is $h$, and the average throughput observed is $t$, we compute the normalized throughput to be $\frac{t}{T_k}$, where $T_k : H_k \approx h$.

## III. Quantitative Analysis

In this section we present four sets of results to discuss respectively the impact of mobility on a single connection scenario and multiple connection scenarios as applicable for the default and ELFN schemes. We analyze the benefits and limitations of these schemes, summarize the results and draw conclusions that serve as motivation for the elements introduced under the *Atra* framework in the next section.

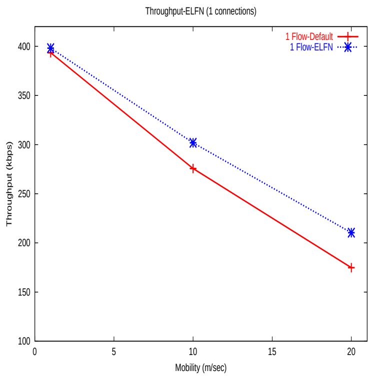
### A. Default (802.11/DSR/TCP): 1 Connection: Impact of Mobility

The study of a scenario with 1 connection, although academic in nature, is beneficial from the perspective of isolating the effects of mobility from the influence of factors such as congestion. We mention a problem associated with multiple connection scenarios, in the later sections. Related studies [8] have also adopted this approach to isolate the effects of mobility. Figure 1 presents the throughput, normalized throughput, MAC-FD latency, RC latency, PRU index, FRU index and the TCP level loss rate for the default (TCP/DSR/802.11) and the ELFN cases. Maximum value reached by the retransmission timer is also represented separately for both the cases. Due to lack of space, we present RTO values only for mobility rates of 1 m/s and 20 m/s respectively. Also, we do not present results for static scenarios because of two reasons: (i) Static scenarios have been studied extensively in related work [13], [14], [15], and (ii) the current paper focuses on factors affecting performance in scenarios with low to high mobility.

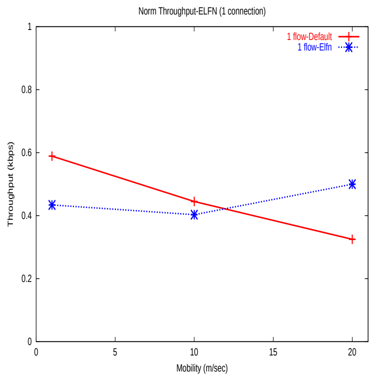
Referring to the curves for the default case in Figure 1(a), it is evident that there is a throughput degradation with increasing mobility. Specifically, the throughput degrades by more than 50% when the mobility increases from 1 m/s to 20 m/s. It is important to note that in case of such a dynamic network, the factors affecting performance, do so in varying degrees at any instance of time, thus making it difficult to pin point a single cause for any effect. This is further aggravated by the fact that the scenarios are averaged over 50 samples. Thus, it is much more tangible to consider an instance while providing explanations, since the factors showing a significant impact can be narrowed down significantly as shown in related work.[8]. However, we deliberately consider the average case so as to get a global perspective on the factors affecting performance.

Normalized throughput shown in Figure 1(b) is flatter than the throughput curve. This is because as mobility increases, average hop count decreases. Since the throughput also decreases with mobility, this results in flatter values for the normalized throughput. We believe that the phenomenon of decrease in average hop count with increase in mobility is a function of the *way point* mobility model used in our simulations.

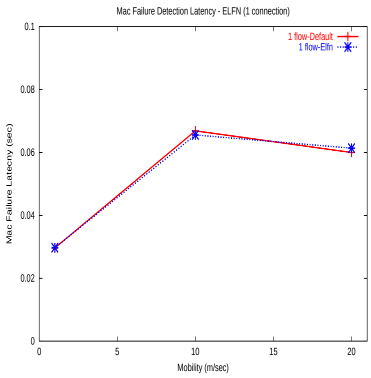
Because of the low load in the network, MAC-FD and RC latencies have relatively small values. However, the interesting observation here is that these two times are of similar order, and their values become significant when the load on the network is increased. Observation of the FRU index reveals that *mobility does not have a significant impact on these parameters*. The FRU index does not show any marked variation as mobility increases because the RC latency is insignificant. The increase in PRU index is explained as follows: there is an increase in the number of route computations as mobility increases. TCP is unaware of route calculations and hence pumps in packets even when a route calculation is in progress. Hence, packets end up spending more time on an average, in the routing layer, and the PRU index shows an increase. However, the key cause behind the throughput



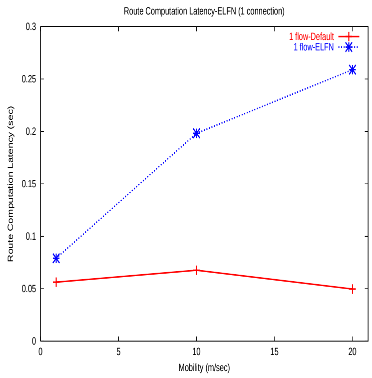

(a) Throughput



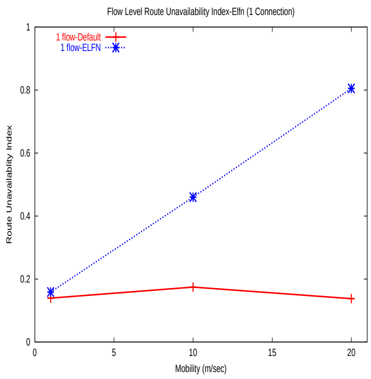

(b) Normalized Throughput



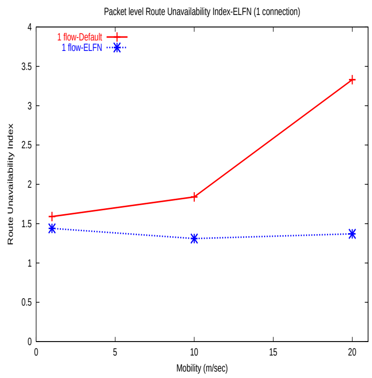

(c) MAC failure Detection Latency



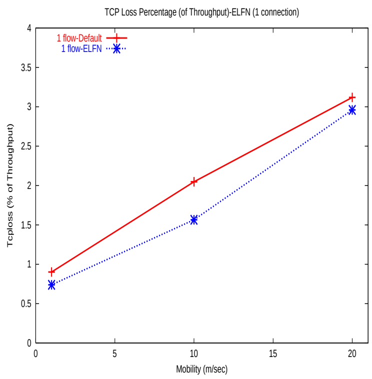

(d) Route computation latency




(e) Flow level Route Unavailability Index

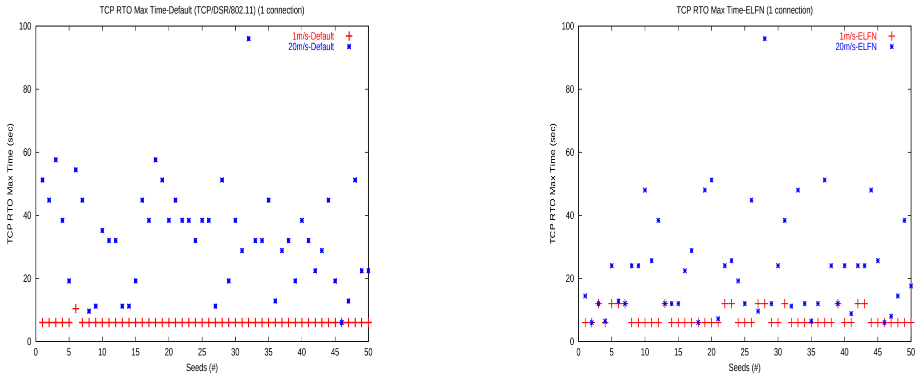



(f) Packet level Route Unavailability Index




(g) TCP Loss Percentage

Fig. 1. RTO vs. Mobility: Default (TCP/DSR/802.11) and ELFN : 1 Connection




(a) TCP RTO max time (b) TCP RTO max time

Fig. 2. RTO vs. Mobility: Default (TCP/DSR/802.11) and ELFN 1 Connection

degradation can be identified in the TCP loss percentage graph in Figure 1(g). Each route failure induces upto a congestion window worth of losses. Losses, as discussed earlier, have an absolute influence on the observed performance, in addition to having a negative impact on TCP's reaction. Observing the retransmission timeout values at the source captures the negative impact of the losses on the behavior of the TCP source. Figure 2(a) shows the maximum retransmission timeout values for each simulation run for mobility rates of 1 m/s and 20 m/s respectively. It can be seen that for the higher mobility rate, there is a distinct increase in the average maximum retransmission timeout value indicating the firing of timers at the TCP source.

Thus, based on the results shown in Figures 1 and 2, we can conclude that the impact of mobility on a 1 connection scenario is primarily because of TCP losses and the consequent negative reaction at the TCP source. In addition, there is a negligible contribution to the performance degradation by the MAC-FD and RC latencies.

### B. *Explicit Link Failure Notification: 1 Connection: Benefits and Limitations*

In this section we present the results when ELFN is used in the scenario described in the previous section. Figure 1 presents the results for the single connection scenario for ELFN, along with that of the default case as well.

From Figure 1(a) it is clear that ELFN does have a positive impact on the performance observed by TCP as mobility increases. At a mobility rate of 20 m/s, the improvement due to the use of ELFN is around 20%[3]. An important reason for the improvement is the freezing of TCP state and transmissions when a route is unavailable and melting the state and restarting transmissions only upon the discovery of a new route. The throughput result is further corroborated by the normalized throughput graph shown in Figure 1(b). The normalized throughput for ELFN shows an increase at higher mobility, over that of the the default case.

Observation of the latency graphs shows that the MAC-FD latency in Figure 1(c) for ELFN exactly follows the behavior of the default case. This shows that even though this factor is as important as the route computation latency, ELFN does not impact it in any way. RC latency for ELFN shown in Figure 1(d), shows a significant increase, when compared to the default case. The FRU index also shows a similar increase in magnitude. This can be attributed to ELFN's aggressive route probing and increased number of route computations that increase the load in then network. This factor has also been identified in related work [10], [14]. It is interesting to note that the PRU index shows a significant decrease in magnitude. This is because TCP freezes its state and stops sending packets every time there is a route computation. This reduces the number of packets spending time in the routing layer buffer because of the unavailability of a route, thus reducing this factor. ELFN mechanism reduces the number of packets sent out when there is a route failure, and hence reduces the number of packets lost. This is evident from Figure 1 (g) where the loss percentage has come down, as much as, by a factor of 33%. The maximum retransmission timeout values for ELFN are shown in Figure 2(b). The interesting point to note here is that for a higher mobility, the retransmission timer values are not as scattered as they are, as in the default case. The confidence interval values for 1 and 20 connection scenarios for both Default and ELFN schemes is shown in the table 3. As seen, they are approximately within 5 to 20 percent of the mean.

### C. *Default (802.11/DSR/TCP): 25 Connections: Impact of Mobility*

In Figure 4, we present the same set of results as above, but for a scenario with 25 flows. The throughput degradation is evident as before from Figure 4(a). However, the percentage degradation is about 36% when mobility rate increases from 1 m/s to 20 m/s as opposed to about 56% in the single connection scenario. This can be explained as follows. When a flow, say $f_k$, experiences performance degradation due of mobility, it is possible for one of the other 24 flows, say $f_j$, that is not experiencing mobility related degradation, to consume the bandwidth given up by $f_k$. However, during the transient phase when $f_j$ is catching up to occupy the bandwidth given up by $f_k$, the network is in a state of under-utilization when compared to an identical but static scenario. Given a

[3] Although in isolated scenarios ELFN produced significantly higher improvements to the tune of 50-60%, over the 50 samples 20% improvement was consistently observed.

| Mob m/sec | 1 Connection Default | | | | 1 Connection ELFN | | | |
|---|---|---|---|---|---|---|---|---|
| | Avg(kbps) | Min(kbps) | Max(kbps) | % | Avg(kbps) | Min(kbps) | Max(kbps) | % |
| 1 | 393.465 | 318.07 | 468.86 | 19.16 | 398.1598 | 319.35 | 476.97 | 19.79 |
| 10 | 275.7008 | 224.28 | 327.12 | 18.65 | 301.8654 | 231.14 | 372.59 | 23.4 |
| 20 | 174.7985 | 139.52 | 210.08 | 20.18 | 210.4042 | 170.39 | 250.42 | 19.0 |
| | 25 Connections Default | | | | 25 Connections ELFN | | | |
| 1 | 33.4146 | 30.82 | 36.01 | 7.76 | 34.276 | 31.53 | 37.02 | 8.0 |
| 10 | 26.511 | 24.14 | 28.89 | 8.95 | 17.486 | 15.69 | 19.28 | 10.26 |
| 20 | 20.762 | 19.32 | 22.2 | 6.93 | 8.33 | 7.22 | 9.45 | 13.3 |

Fig. 3. Confidence Interval values for Throughput: Default/ELFN

scenario in which the network is consistently dynamic, it is highly likely that flow $f_j$ itself experiences mobility related degradation shortly before it manages to take up the entire bandwidth relinquished by $f_k$.

While TCP loss rate and the consequent impact on the retransmission timers follows the same pattern as in the single connection scenario, the MAC-FD and the RC latencies have undergone an order of magnitude increase in this scenario, when compared to the single connection case. This is because of a higher load in the network. The FRU and the PRU indices also increase, when compared to the values in a single connection scenario. Observing the MAC-FD latency in Figure 4(b), shows that it decreases with the increase in mobility. This decrease can be attributed to the fact that when there is a route failure and a subsequent route computation, TCP encounters congestion and backs off. When the new route is computed, TCP invariably is in the slow start phase, thus reducing the number of packets in the network. Hence the packets do not oberve much latency when they come into the MAC layer. RC latency, shown in Figure 4(c), is of a similar order, when compared to the MAC-FD latency. The FRU and PRU indices do not show much of a variation with respect to mobility.The percentage of TCP loss is higher when compared that of the 1 connection case. This is obvious since the per flow throughput decreases, packet losses as a percentage of achieved throughput would increase.

While we draw summarized conclusions at the end of the section, one of the critical observations that can be made based on the results presented is the significant contribution (both in the absolute sense and in terms of the impact on TCP) of the MAC-FD and RC latencies, when the load in the network is heavier.

### *D. Explicit Link Failure Notification: 25 Connections: Benefits and Limitations*

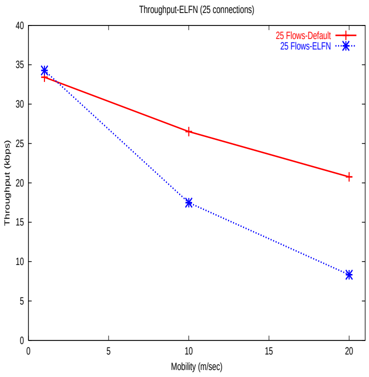
The performance of ELFN in a loaded network is however quite in contrast to the observations made thus far. The throughput, as evident from Figure 4(a), in fact degrades as mobility rate increases. While other related work have also identified a similar drawback of ELFN [10], the key reason behind the degradation is the aggressive probing by ELFN to recover from route failures. When such flooding[4] is being performed by all 25 connections, the broadcast storm problem identified in [16] exacerbates. However, it should be noted that even if the routing protocol were to be an idealistic one, ELFN would still not improve on the performance of the default mechanisms because it does not explicitly address either the absolute influence of the large MAC-FD and RC latencies or the impact of the former on TCP's behavior. The degradation in throughput could also be attributed to the behavior of MAC-FD latency. This factor shows a significant increase for ELFN, when compared to that of the default case. This could be attributed to two reasons: (i) The network load increases because of aggressive route probing, as pointed out earlier. (ii) The ELFN mechanism causes TCP to freeze its state upon the occurance of a route failure. When a new route is calculated, TCP restarts sending traffic at the older rate. Hence, the congestion window estimate can potentially be incorrect. If the congestion window happens to be an overestimate, this leads to a buffer build at intermediate nodes, thus increasing the MAC-FD latency.

It can thus be concluded that while ELFN does have its benefits in terms of masking the impact of mobility in lightly loaded scenarios. But in heavily loaded scenarios, ELFN can have a negative impact, when aggressive route probing is performed [14], [10]. On the other hand, if aggressive route probing is not performed, the route recovery process will be delayed. In both cases, ELFN will still not improve on the performance of the default mechanisms because it does not explicitly address the issues of large failure detection and route computation latencies.

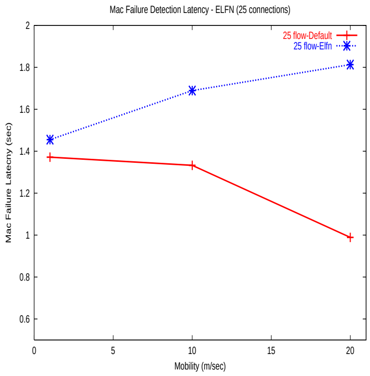
### *E. Discussion and Conclusions*

Based on the results presented thus far, we can identify the different components that contribute to performance degradation in the presence of mobility to be:

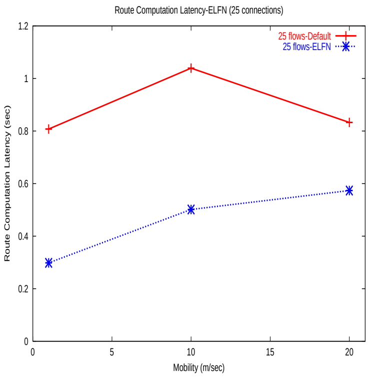
[4]Flooding is an artifact of the routing protocol used. For a good discussion on the negative effects of flooding in ad-hoc networks, see [16].



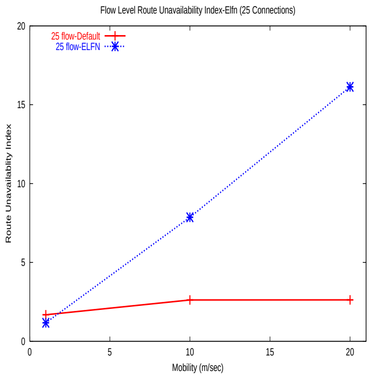

(a) Throughput



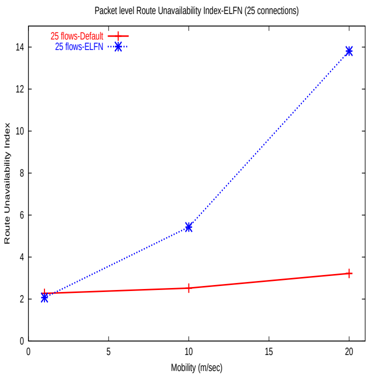

(b) MAC failure Detection Latency



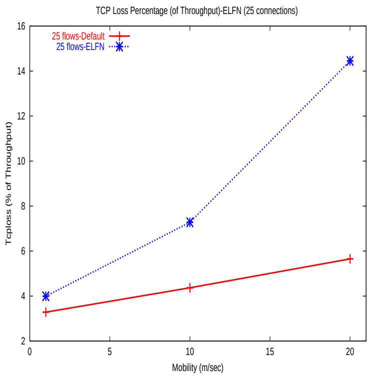

(c) Route computation latency




(d) Flow level Route Unavailability Index




(e) Packet level Route Unavailability Index




(f) TCP Loss Percentage

Fig. 4. Overall vs. Mobility: ELFN 25 Connections

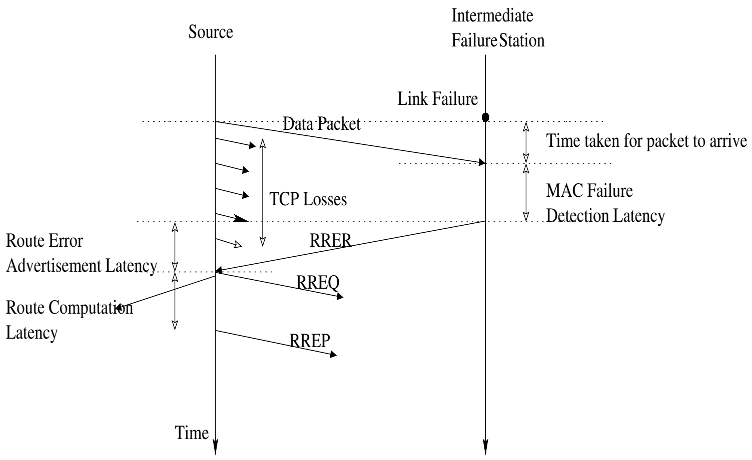



Fig. 5. Illustration of Components Affecting TCP Performance

• *TCP Losses:* Every route failure induces upto a TCP-window worth of packet loss. While the losses have a direct and absolute impact on the performance degradation, the TCP source will also react to the losses by performing congestion control (TCP NewReno will adapt only once for any number of losses within a window).
• *MAC Failure Detection Time:* Since the MAC layer (802.11) has to go through the cycle of multiple retransmissions before concluding link failure, there is a distinct component associated with the time taken to actually detect link failure from when the failure occurs. Importantly, the detection time increases with increasing load in the network. A high MAC detection time will result in a higher likelihood of the TCP source pumping in more packets (upto a window 's worth) into the broken path, and an eventual timeout.
• *MAC Packet Arrival:* When a failure is detected as is mentioned above, the link failure indication is sent only to the source of the packet that triggered the detection. If another source is using the same link in the path to its destination, the node upstream of the link failure will have to wait till it receives a packet from the other source before that source is informed of the link failure. This also contributes to the delay after which a source realizes that a path is broken.
• *Route Computation Time:* Once a source is informed of a path failure, the time taken to recompute the route also increases with increasing load. For a load of 25 connections in the ELFN case, the per-flow average of the aggregate time spent in route computation during a 100 second simulation went as high as 15 seconds resulting in an immediate degradation of performance by 15%. In addition to the absolute impact, TCP is also likely to experience timeouts during each route computation time, especially in the higher load scenario where route computation time is around a couple of seconds.
In the next section, we present the *Atra* framework, that derives on the learnings in the previous sections, and thus develops three mechanisms to address the problems detailed earlier.

## IV. The Atra Framework

In this section we propose a framework called *Atra* that consists of a set of simple and easily implementable mechanisms that are motivated by the insights gained from the performance evaluation in Section III. We first describe the Atra framework and evaluate its performance in isolation and in tandem with the ELFN approach. In order to understand the benefits of the individual mechanisms within the Atra framework, we also present results observed when different subsets of the mechanisms are in use.

### *A. The Atra Framework*

The mechanisms in the Atra framework are based on the following three goals: (i) To minimize the probability of route failures, (ii) To predict route failures in advance and thus enable the source to recompute an alternate route before the existing route fails, and (iii) To minimize the latency in conveying route failure information to the source, for route failures that are not successfully predicted. The Atra framework consists of three mechanisms targeted toward each of the above goals respectively:
• *Symmetric Route Pinning:* The DSR routing protocol does not explicitly use symmetric routes between a source and a destination, i.e. the route taken from the source to the destination can be different from the route taken from the destination to the source. In fact, for most runs of our simulations we observed asymmetric routes being used for the TCP data path and the corresponding ACK path. While the use of asymmetric routes is not an issue in a static network[5], in a dynamic network where nodes are mobile using an asymmetric path increases the probability of route failure for a connection.
A TCP connection will stall irrespective of whether the forward or the reverse path is broken. Consider a simple scenario using two edge-disjoint routes with hop lengths of $h1$ and $h2$, for the DATA and ACK paths. Assuming an uniform probability of link failure $p$ for all links in the network (which is not unrealistic given the use of way-point mobility model in the simulations), the probability of a path failure is $1 - (1 - p)^{(h1+h2)}$.
The first mechanism in the Atra framework is called *symmetric route pinning* (SRP). In this mechanism. the ACK path of a TCP connection is always kept the same as the data path. The mechanism implemented at the DSR layer does the route pinning only for non-piggybacked ACKs. Note that the last design decision is immaterial for unidirectional connections, but in case of bidirectional connections, the forward path progression can be asynchronous to the reverse path progression. If the paths are assymetric, it leads to an implict load balancing on both the paths, but performing route pinning for piggybacked ACKs can severely increase the congestion along the path.
• *Route Failure Prediction:* While the symmetric route pinning mechanism merely reduces the probability of route failures for a connection, the second mechanism in Atra attempts to predict the occurrence of a link failure based on received signal strengths. Specifically, a node predicts the occurance of a link failure based on the progression of signal strengths of packet receptions from a particular neighbor. Maintaining a history of the progression enables nodes to dynamically profile the speed at which the the two nodes are moving away from each other by observing the slope of the progression[6]. The threshold to trigger a prediction is a tunable parameter that would determine the *look-ahead time* for the link failure. Since the objective is to enable the completion of an alternate route computation before the current path fails, we empirically set the value for the look-ahead time to be 2 seconds for all simulations. This value is assumed to be the coarse upper bound for route computation time for all simulations. A critical

[5]Note that the effects on path asymmetry on TCP performance has been studied in related work [17]. However, the impact of asymmetry we consider here is different from those related works.
[6][18] also proposes a similar mechanism for route failure prediction (RFP), although not in the context of TCP performance enhancement. Please see the reference for a more detailed discussion of why signal strength is a reliable metric to predict link failures

aspect of the prediction process is the propagation model used. Since we assume the two ray ground reflection model for distances greater than 100m, we use the following equation to calculate the threshold receive power corresponding to a particular slope (and hence speed) and the look-ahead time. Based on the model, the received power $P_r$ can be specified as:

$$P_r = \frac{K}{d^4}$$

where K is a constant and is a function of the gains and heights of receive and transmitter antenna, and the transmit power. $d$ is the distance between the transmitter and the receiver. Thus given a look-ahead time $t$ and the observed relative speed between the two nodes $s$, the threshold power to trigger a prediction can be calculated as:

$$P_{RFP} = \frac{K}{(r - s * t)^4}$$

where $r$ is the transmission range. The speed $s$ is computed from the slope of the history of transmission powers observed. If the size of the history is $N$ (N packet reception powers and corresponding times), the speed is calculated as follows:

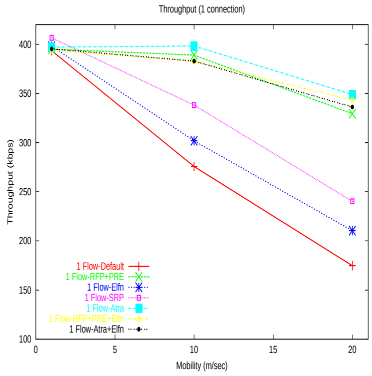
$$s = \frac{\sum_{i=1}^{N} \frac{(1/P_{i+1})^{1/4} - (1/P_i)^{1/4}}{t_{i+1} - t_i}}{N - 1}$$

where $P_i$ and $t_i$ represent the received power and time of reception for the $i^{th}$ packet in the history. We use $N = 3$ in all our simulations. When a source receives a predicted route failure message it issues a new route request and continues to use the current path. This path is used either till a new route is computed or the current route fails and a normal route error is received. Route requests are suppressed based on the same thresholds used to predict route failures. Hence, a route that is close to failure will not be chosen during any route computation process. Note that the mechanism employed to predict link failures is merely a heuristic and can fail either by incorrectly predicting a link failure or by failing to predict an actual link failure. In case of incorrect prediction, the throughput of the corresponding connection remains unaffected because the source will continue to use the current path until a new alternate path is computed or the current path fails. Since the current path will not fail, the source will switch its connection only upon the recomputation of an alternate path. Also if the current path is the best path, it will again be recomputed as the alternate path thus preventing any sub-optimality because of wrong predictions. The drawback of incorrect predictions is a route recomputation overhead would be incurred. On the other hand, if an actual route failure is not predicted successfully, the performance of the connection will be only as bad as the scenario in which there is no prediction mechanism employed. As seen from our performance evaluations presented later in the section, use of the prediction mechanism significantly improves the observed throughput.
Also note that the prediction mechanism can successfully predict only the mobility related route failures. Other possibilities like congestion based route failures will not be (and should not be) captured by the prediction mechanism. These will trigger normal route errors as usual.

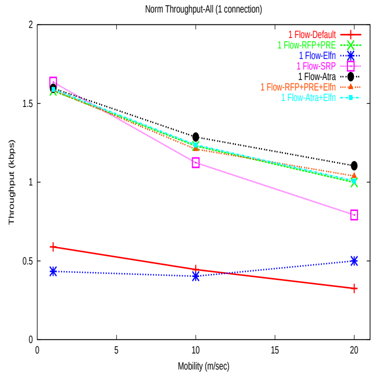
- *Proactive Route Errors (PRE):* If a link failure occurs (either due to congestion or due to mobility), but has not been successfully predicted by the failure prediction mechanism, the third mechanism in Atra tries to minimize the latency involved in the route failure information being carried to the source(s) that were using the link. In the default set-up, DSR will issue a route error only to the source of the packet that triggered the link failure detection at the MAC layer. If multiple sources are using the same link, packets will have to arrive from those source as well, before route errors are sent back to them, thus increasing the latency between the link failure detection and the time at which the sources are informed. This latency is further inflated because subsequent packets, arriving from other sources will have to go through the *MAC failure detection time* cycle before the link failure is inferred. To get around this problem, each node in Atra maintains a cache of the source ids of TCP connections, that have used a particular link in the past $T$ seconds ($T$ set to a 1 second in our simulations). When a link failure is detected, all sources that have used the link in the past $T$ seconds are informed about the link failure through normal route errors. This reduces the latency involved in the route failure information delivery which consequently reduces the number of losses and also triggers earlier alternate route computations. It is interesting to note that the proactive route error mechanism can prove to be disadvantageous when a link failure has occurred due to congestion. Consider an example where a link between nodes $A$ and $B$ is traversed by 2 TCP connections $f1$ and $f2$. In the default set-up, when a packet belonging to $f1$ experiences congestion related link failure, only $f1$ would be informed of the link failure prompting $f1$ to choose a different route and thus relieving congestion along the original path for $f2$. However, when the proactive route error mechanism is used, both $f1$ and $f2$ will be informed of the route failure making both of them to recompute their routes (although the same path might be chosen all over again). However, the characteristic of the default set-up to let route requests through in preference to data packets results in routes being chosen irrespective of the congestion along the path. Hence, in the example considered there is nothing to prevent flow $f1$ from choosing the same path again even under the default set-up.

### B. *Performance Evaluation*

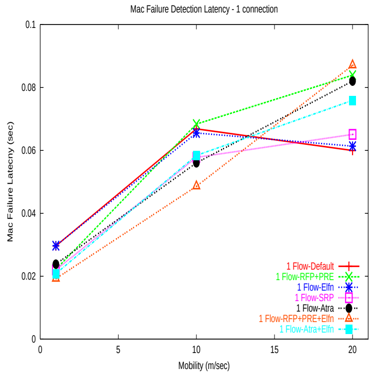
Figure 6 presents the observed performance for the Atra framework in the one connection scenario. In order to better understand the contributions made by each of the three mechanisms, we present results for the following results: RFP+PRE, SRP and Atra.



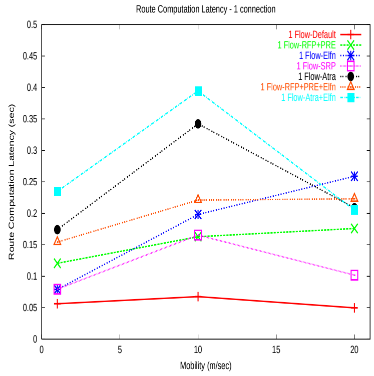

(a) Throughput



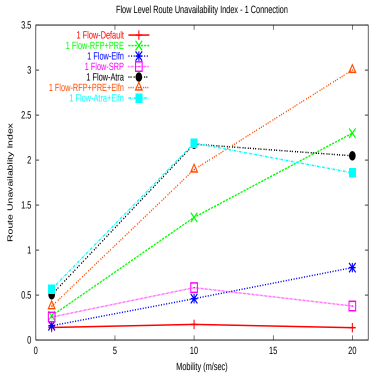

(b) Normalized Throughput



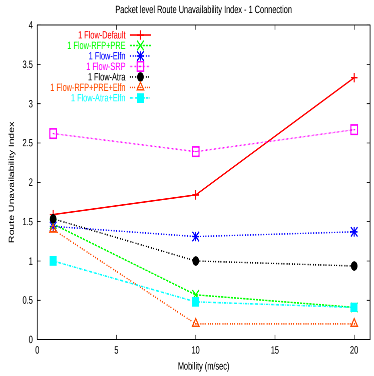

(c) MAC failure Detection Latency



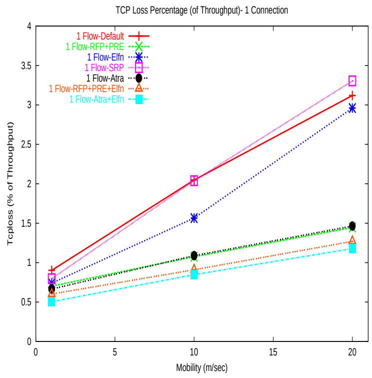

(d) Route computation latency



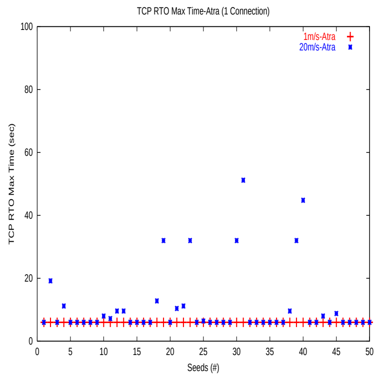

(e) Flow level Route Unavailability Index




(f) Packet level Route Unavailability Index Rate




(g) TCP Loss Percentage




(h) TCP RTO max time

Fig. 6. Overall vs. Mobility: All 1 Connection

| Mobility | 1 Connection ATRA | | | |
|---|---|---|---|---|
| m/sec | Avg (kbps) | Min (kbps) | Max (kbps) | % |
| 1 | 395.0592 | 321.76 | 468.36 | 18.5 |
| 10 | 389.0298 | 327.08 | 450.98 | 15.9 |
| 20 | 329.3454 | 284.63 | 374.06 | 13.5 |
| | 25 Connections ATRA | | | |
| 1 | 43.4905 | 39.91 | 47.07 | 8.10 |
| 10 | 37.8511 | 35.24 | 40.46 | 6.60 |
| 20 | 31.3688 | 29.59 | 33.15 | 6.04 |

Fig. 7. Confidence Interval values for Throughput: Atra

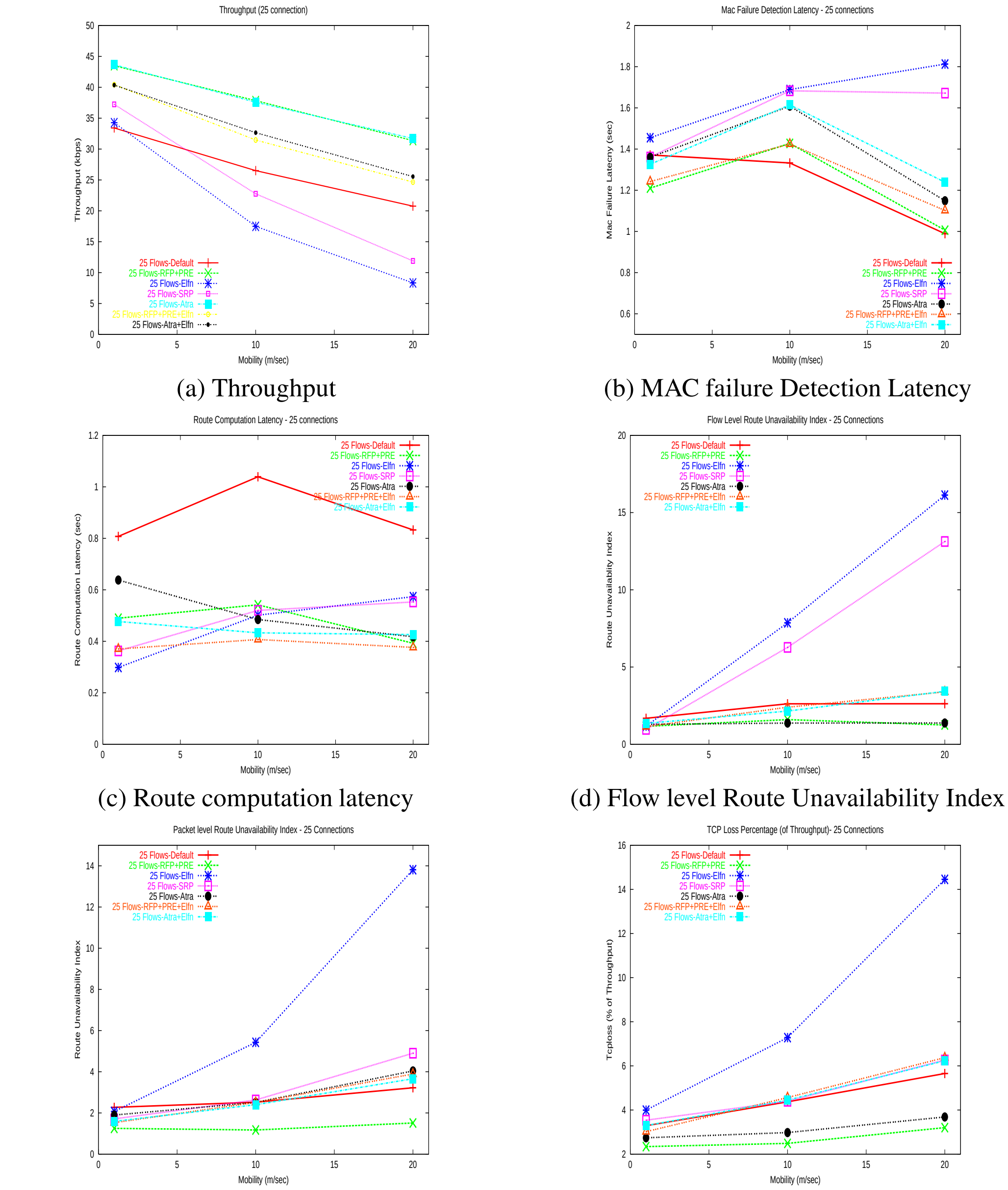
Also, in order to study the impact of using ELFN with the proposed mechanism we also present the results of RFP+PRE+ELFN and Atra+ELFN. The default and only ELFN cases are also included for comparison. We present results for the same set of metrics, as presented in the previous section.

From Figure 6(a), *it can be observed that Atra and its various variations outperform the default protocol stack and ELFN by a large margin*. Specifically,Atra exhibits throughput improvement of around 100% over the default case at a mobility rate of 20 m/s. The improvement is around 70% when compared to the ELFN approach. Since Atra does not have any explicit mechanisms to minimize MAC failure detection latency, there is no perceivable difference in the values for that metric. Although Atra's mechanisms do not inherently perform anything to increase the route computation time, we do observe an increase in the route computation latency. We attribute this increase to the significant increase in the utilization of the network. A true indication of the causes for the performance enhancement can be observed from the TCP loss rate and RTO values. The loss rate experienced by the Atra framework is significantly lower than that of either ELFN or the default scenario. The improvement in loss rate is a direct result of alternate routes being computed even before the current path fails preventing any disruptions of traffic. The prevention of disruptions is also evident from the TCP RTO values result shown in Figure 6(h). While the RTO values for the default and ELFN scenarios are highly dispersed with large values, the values when Atra is used predominantly remains around the default value for the TCP retransmission timer even as mobility rate increases.

Normalized throughput for *Atra* and its related mechanisms, has increased, when compared to the default and ELFN cases. This is attributed to the better throughput, that is achieved with the use of *Atra*. Observation of the MAC-FD latency in Figure 6(c) shows that the progression of curves for all the cases are comparable. This is logical because none of the mechanisms under consideration impact the MAC-FD latency. RC latency shown in Figure 6(d), seems to increase steadily for all the newer mechanisms, and it is the lowest for the default case. The increase in case of ELFN has been explained earlier. For Atra, the increase is primarily because of the RFP and PRE mechanisms, which cause an increase in the network load. Note that this is not the case with SRP, and the individual simulation with SRP is of a lower magnitude than that of ELFN and *Atra*. The interesting fact to note is that the RC latency for ELFN is still higher than that of Atra, for higher mobility. As expected, Atra+ELFN generates the highest network load, which is reflected in the graph for this case, as the curve with the highest magnitude. The FRU index shown in Figure 6(e) for *Atra* is also coupled with how the RC latency behaves, and it shows an increase for *Atra* and its related mechanisms. However the PRU index shown in Figure 6(f) for *Atra* is much lower than that of the default and ELFN cases. This is obvious because the mechanisms in *Atra* aim to proactively calculate routes, in case of imminent route failures. Thus the node has a valid route to the destination, most of the times. As a result, the packets spend much less time at the node, and hence PRU index decreases. Note that for the curve for SRP is even higher than that of the default case. Also note that *Atra*+ELFN performs the best in this case, thus proving beyond doubt that ELFN works well with *Atra*. All these factors help in reducing the TCP Losses shown in Figure 6(g) to a large extent, which leads to a better throughput. The retransmission loss graphs shown in Figure 6(h) are self explanatory. The values for *Atra* are kept to their minimum values for all the samples, for both 1 and 20 m/s.

The confidence interval values for 1 and 20 connection scenarios for *Atra* is shown in the table 7. As seen, they are approximately within 5 to 15 percent of mean.

Figure 8 compares the Atra framework with the other schemes when the load in the network is increased to 25 connections. The improvement in throughput shown in Figure 8(a) over that of the default scenario is around 50%. The reduction in performance improvement, when compared to the single connection case, can be attributed to the fact that other flows can take up bandwidth given up by a suffering flow. However, note that the transient phase involved will still contribute to an overall performance degradation when compared to a static scenario. Note that the performance of ELFN goes down below that of the default protocol stack, the reason for which, has been explained in section III. The effective improvement exhibited by *Atra* over ELFN amounts to




(a) Throughput

(b) MAC failure Detection Latency

(c) Route computation latency

(d) Flow level Route Unavailability Index

(e) Packet level Route Unavailability Index Rate

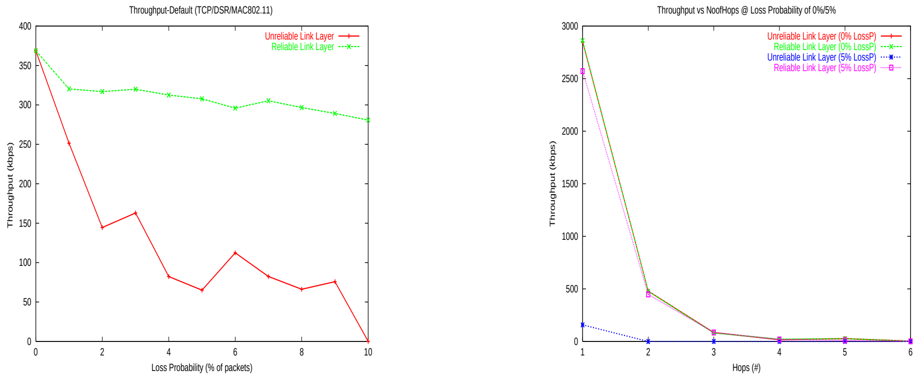
(f) TCP Loss Percentage

Fig. 8. Overall vs. Mobility: All 25 Connections

around 300%.

It is interesting to note that the MAC-FD latency curves shown in Figure 8(b), for *Atra* and other mechanisms, all lie within the curve for the ELFN and default case. Also note that the MAC-FD curve for *Atra* is the closest to that of the default case, which is the minimum obtained here. The reason for such a behavior is that the mechanisms in *Atra* do cause an increase in network load because of all the additional route calculation traffic, thus increasing magnitude of MAC-FD latency. The RC latency shown in Figure 8(c) is considerably lowered when compared to the default case. Note that the FRU and PRU indices for *Atra* are also insignificant, when compared to the ELFN and SRP case. The explanation of ELFN exhibiting such a behavior has been explained earlier in section III. The FRU index for SRU increases because of contention on the reverse path because of a symmetric path characteristic, which increases the number of route calculations, thus increasing the load on the network. This fact is corroborated by the MAC-FC curve for SRP, which is comparable to that of ELFN. Note that the FRU and PRU index values for *Atra* is lower than the default and ELFN cases. Specifically, the PRU index value reduces by a factor of 70% when compared to ELFN. Figure 8(f) shows the TCP loss percentage for all the schemes. The loss percentage for RFP+PRE is the lowest, which proves that these two mechanisms are sufficient in reducing the losses. Note that the loss percentage for *Atra* is higher than that of RFP+PRE. This can be attributed to the SRP mechanism, which results in packet losses, due to contention on the reverese path. The loss percentage for *Atra* is lowered by a factor of 75% when compared to ELFN.

ELFN's aggressive route probing, as mentioned earlier, can severely degrade performance due to the broadcast storm problem




(a) Random Loss Probability (% of packets) (b) No of Hops

Fig. 9. Throughput vs Random Loss Probability and No of Hops

[16]. A heuristic tuning of ELFN's probing rate can potentially alleviate the throughput degradation observed. However, note that ELFN does not address the issues of TCP losses (and consequently RTO backoffs), MAC-FD latency and the consequent impact on TCP, and the impact of the absolute value for FRU index. The SRP mechanism on the other hand suffers performance degradation at higher loads because of the additional load imposed by the ACK path on the route taken by the data path. This results in an increase in the number of congestion based route failures which triggers further route requests that in turn increases the load in the network. As mentioned earlier, this is seen as an increase in the flow level route unavailability index for SRP Figure 8(d). However, an interesting observation is that when SRP is used in tandem with the other mechanisms in Atra, it does not exhibit the same negative effect on the throughput performance.

In conclusion, the mechanisms in the Atra contribute to (i) reducing the number of route failures, (ii) predicting route failures before they occur and (iii) minimizing the latency for route error information delivery to sources, and thus, in the process, significantly improves throughput performance both when compared to the default protocol stack and an ELFN enabled protocol stack.

## V. Issues and Related Work

In this section we discuss some issues with the proposed Atra framework along with some problems unaddressed in this paper.

- *Random Wireless Errors:* In order to keep within the focus of the paper we have not discussed the impact of random wireless errors on the performance of TCP. However, we have performed extensive simulations to study such impact and have observed that a reliable link layer mechanism[7] like the one provided by the IEEE 802.11 protocol (by virtue of its ACK handshake) is sufficient to handle even non-significant amounts of wireless random losses. Figure 9(a) shows the observed throughput in a one connection scenario where the per-link wireless random loss probability is varied from 1% to 10%. Results for both unreliable and reliable link layers are shown. It can be observed that the use of a reliable link layer mechanism effectively hides the lossy nature of the underlying wireless network.

In Figure 9(b) we study the performance of connections with different hop lengths with and without a reliable link layer for a scenario with a random packet loss probability of 5%. The goal was to study the effect of the hop length of a connection on the error-recovery capabilities of the reliable link layer. The motivation behind the study is to investigate the possibility of the link layer retransmissions interfering with TCP layer retransmissions for short hop length connections as reported in [19]. However, for the assumed network environment with a 2Mbps data rate, the granularity of recovery at the link-layer was significantly lower (few tens of milliseconds) than the TCP error recovery time (which is at-least one second as specified in [20]). The result for a loss probability of 0% is also shown as the reference curve.

- *Transport Layer Mechanisms:* Since the goal of this work was to study the impact of mobility on TCP's performance, no changes have been proposed to TCP per se. While an argument can be made against TCP changes based on its extensive usage in the backbone Internet and consequent compatibility issues, recent trends have been toward designing tailored transport protocols that are TCP friendly but suited to the target environments [21], [19], [22]. A non-comprehensive set of changes that could be incorporated at the transport layer given the characteristics of an ad-hoc network environment include: (i) *Rate based transmissions.* Ad-hoc networks have the unique property wherein the transmission of packets belonging to the same flow on the multiple hops along the flow's path contend with each other for access to the shared channel. Hence, from a purely utilization perspective, significant gains can be achieved by making the transmission at the source rate based rather than the window-based burst transmissions employed by TCP. (ii) *TCP-SACK Extensions.* The Selective ACK scheme introduced in [23] is particularly addressed toward lossy environments where the receiver buffer can have several holes, enabling the sender to retransmit only the holes and thus save on precious bandwidth. However, the current specification for TCP-SACK supports the advertisement of only three blocks of non-contiguous data. While the limit of three is suitable for the wireline Internet environment, it would be a severe limitation in an ad-hoc network

[7] Pseudo-reliable since IEEE 802.11 infers link failure after 7 retransmission attempts.

environment. (iii) Since typical ad-hoc networks are expected to consist of a few hundred nodes at a maximum, and considerable inter-layer interactions exist even in the current set-up of an ad-hoc network protocol stack, feedback from the intermediate routers in the network can significantly help the transport layer's congestion control scheme. For instance, while it can be extremely difficult to distinguish between random wireless losses, congestion based losses, and mobility related losses based on purely end to end mechanisms, appropriate feedback from the routers can help in distinguishing between the loss types and reacting accordingly.

• *Prediction Accuracy:* The route failure prediction scheme proposed in this paper is a heuristic mechanism and hence can fail to either predict route failures successfully or wrongly predict route failures. As explained in Section IV both scenarios are not catastrophic to the operation of the network. While in the first case, the performance will be as bad as a scheme without any prediction capabilities, the second case would lead to an additional overhead caused by the alternate route computation process. However, we briefly discuss the possibility of wrong route failures and the robustness of the proposed scheme to overcome possible causes of misprediction. There are two possible scenarios in which the proposed scheme can mispredict a route failure: (i) When two nodes $A$ and $B$ are relatively stationary, but the distance between them is marginally smaller than the transmission range; and (ii) When the received signal strength varies due to causes other than distance, say for example channel fading. However, recall that the proposed approach maintains a history of the received signal strengths and makes its link-failure prediction based on both the configured look-ahead time and the slope of the curve representing the progression of received signal strengths on that particular link. This simple mechanism of maintaining the history will prevent mispredictions from occurring due to both the above causes. A history as small as three packets was sufficient to maintain a high degree of accuracy in our simulations.

### A. Related Work

Several research works have focused on developing better routing protocols ([6], [7]), while some have attempted to identify factors affecting the performance in static multihop wireless network scenarios ([13], [14], [15]). [13] studies the performance of TCP on static wireless networks and evaluates explicit feedback techniques for improving throughput. It also focuses on burst errors in wireless networks and discussed the effect of packet size variations on throughput. [14] studies the effect of routing and link layer mechanisms on TCP performance. The focus is on route cache management strategies for a mobile ad hoc network and also briefly discuss the effect of link layer retransmissions on TCP throughput, in a fixed wireless network. [15] concentrates on fixed wireless multihop networks and evaluates TCP performance on various MAC schemes. It does a good job of outlining the problems that might occur in a fixed multihop network (linear and grid toplogies) because of TCP and MAC layer interactions. Recent work ([8], [10], [9], [24]) discusses effect of mobility on TCP performance and suggests various transport layer mechanisms to solve the problems caused due to mobility. [8] evaluates an explicit link failure notification technique in the context of improving TCP performance over multihop mobile ad hoc networks. It discusses in detail, the effect of link failures due to mobility on throughput of the connections and shows that the throughput improves by a significant amount when ELFN is used in the network. However, the focus is on a single connection scenario, and they mention that they intend to study the effect on multiple connections. [10] also studies the performance of ELFN on static and dynamic networks and corroborates the results obtained in [8]. However, this paper also focuses only on a single connection scenario, possibly because of ease and clarity of explanations. It claims that the throughput increases by around 5% on static networks and also cites best case values to show around 7 times improvement. Note that these values are not averaged but are truly best case. [9] discusses a mechanism called TCP-Feedback, which uses route failure and re-establishment notifications to provide feedback to TCP, and thus reduce the number of packet retransmissions and TCP back offs during route calculation, to improve throughput. However, this mechanism is not evaluated for ad hoc networks. [24] studies the performance of TCP on three different routing protocols and proposes a heuristic called fixed RTO, which essentially freezes the TCP RTO value whenever there is a route loss. They also evaluate the effectiveness of TCP's selective and delayed acknowledgements in improving the performance. [25] examines the capacity of wireless ad hoc networks by studying the interactions of 802.11 and ad hoc forwarding, and its effect on network capacity. This does a very good job of outlining the raw capacity that one can obtain, while using 802.11 on an ad hoc network. [26] evaluates an alternate path routing scheme on a mobile ad hoc network and claims that it provides a 20% reduction in the end to end delay for bursty data streams. It also highlights a route coupling problem that occurs because of overlap of different route paths. Most of the related work investigates only a select few factors affecting performance namely the route computation time, delay, effect of path lengths and mobility. We investigate a comprehensive list of parameters affecting performance at the MAC, routing and transport layers and thus come up with solutions at the MAC and routing layers to solve the problems.

## VI. SUMMARY

Multi-hop wireless ad-hoc networks have gained considerable attention over the last decade by virtue of their applications to military and disaster relief applications, and more recently even in conventional wireless packet data networks. In this paper we investigate the impact of the mobility of nodes in an ad-hoc network on TCP's performance. We identify the key factors that contribute to TCP's performance degradation as (i) TCP losses, (ii) MAC link failure detection latency, (iii) Link failure notification latency, and (iv) Route computation time. We show that the above factors contribute both in absolute terms and in terms of their impact on TCP's behavior. We argue that existing solutions to improve TCP performance that fall under the broad category of ELFN schemes address only a subset of the factors. Finally, we propose a framework called *Atra* consisting of three easily implementable mechanisms at the medium access and routing layers that alleviates the impact of mobility on TCP's performance. We demonstrate through simulations that the proposed framework improves TCP's performance by about 100% over the default TCP/DSR/802.11 protocol stack, and by about 70% by an ELFN enabled protocol stack in a lightly loaded (one connection) dynamic (speed of 20 m/s) scenario. The performance enhancement over the default and ELFN protocol stacks in a heavily loaded (25 connections) mobile (20 m/s) scenario is observed to be around 50% and 300% respectively.

## REFERENCES


[1] Rooftop Inc., "http://www.rooftop.com," .
[2] Y-D. Lin, Y-C. Hsu, and C. Tung, "Multihop Cellular: A New Architecture for Wireless Communications," in *Proceedings of IEEE INFOCOM*, Mar. 2000.
[3] T. Rouse, S. McLaughlin, and H. Hass, "Coverage-Capacity Analysis of ODMA in UTRA TDD," in *3G Mobile Communication Technology Conference*, Mar. 2001, pp. 26–28.
[4] G. Aggelou and R. Tafazolli, "On the Relaying Capacity of Next-Generation GSM Cellular Networks," in *IEEE Personal Communications Magazine*, Feb. 2001, pp. 40–47.
[5] H-Y. Hsieh and R. Sivakumar, "Performance Comparison of Cellular and Multi-hop Wireless Networks: A Quantitative Study," in *Proceedings of ACM SIGMETRICS*, June 2001.
[6] D.A.Maltz D.Johnson and J. Broch, "The Dynamic Source Routing Protocol for Mobilke Ad Hoc Networks (Internet Draft)," Mar 1998.
[7] C.E.Perkins and Elizabeth M Royer, "Ad Hoc On Demand Distance Vector (AODV) routing (Internet Draft)," Aug 1998.
[8] G. Holland and N. H. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," in *Proceedings of ACM MOBICOM*, 1999, pp. 219–230.
[9] K. Chandran, S. Raghunathan, S. Venkatesan, and R. Prakash, "A feedback based scheme for improving TCP performance in ad-hoc wireless networks," in *Proceedings of International Conference on Distributed Computing Systems*, 1998, pp. 472–479.
[10] Vaduvur Bhargavan Jeffrey P. Monks, Prasun Sinha, "Limitations of TCP-ELFN for Ad hoc Networks," in *Workshop on Mobile and Multimedia Communication*, Oct. 2000.
[11] K. Fall and K. Vardhan, "*ns* notes and documentation," available from http://www-mash.cs.berkeley.edu/ns/, 1999.
[12] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," in *Proceedings of ACM MOBICOM*, Dallas, TX, Oct. 1998.
[13] B. Bakshi, P. Krishna, N. H. Vaidya, and D. K. Pradhan, "Improving Performance of TCP over Wireless Networks," in *Proc. 17th International Conf. on Distributed Computing Systems (ICDCS)*, Baltimore, May 1997.
[14] Gavin Holland and Nitin Vaidya, "Impact of Routing and Link layers on TCP performance in Mobile Ad-hoc Networks," in *Proceedings of IEEE WCNC*, September 1999.
[15] Ken Tang Mario Gerla and Rajive Bagrodia, "Tcp performance in wireless multi hop networks," 1999.
[16] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, and Jang-Ping Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proceedings of ACM MOBICOM*, 1999, pp. 151–162.
[17] U. Madhow T.V. Lakshman and B. Suter, "Window-based Error Recovery and Flow Control with a Slow Acknowledgement Channel: A Study of TCP/IP Performance," in *Proceedings of IEEE INFOCOM*, April 1997.
[18] D. S. Phatak T. Goff, N.N. A-Ghazaleh and R. Kahvecioglu, "Preemptive Routing in Ad-hoc Networks," in *Proceedings of MOBICOM*, 2001.
[19] P. Sinha, N. Venkitaraman, R. Sivakumar, and V. Bharghavan, "WTCP: A Reliable Transport Protocol for Wireless Wide-Area Networks," in *ACM MOBICOM*, Aug. 1999.
[20] V. Paxson and M. Allman, "Computing TCP's Retransmission Timer," in *IETF Request for Comments 2988*, November 2000.
[21] G. Morabito I. F. Akyildiz and S. Palazzo, "TCP-Peach: A New Congestion Control Scheme for Satellite IP Networks," in *IEEE/ACM Transactions on Networking*, June 2001.
[22] D. Bansal and H. Balakrishnan, "Binomial Congestion Control Algorithms," in *Proceedings of IEEE INFOCOM*, April 2001.
[23] S. Floyd M. Mathis, J. Mahdavi and A. Romanow, "TCP Selective Acknowledgement Options," in *IETF Request for Comments 2018*, October 1996.
[24] Thomas D. Dyer and Rajendra Bopanna, "A Comparison of TCP Performance over Three Routing Protocols for Mobile Ad Hoc Networks ," Proceedings of MobiHoc 2001, Oct 2001.
[25] Douglas S. J. Decouto Hu Imm Lee Jinyang Li, Charles Blake and Robert Morris, "The Capacity of Ad Hoc Wireless Networks," in *Proceedings of Mobicom*, 2001.
[26] P.Sholander Marc R.Pearlman, Z.J.Haas and S.S.Tabrizi, "On Impact of Alternate Path Routing for Load Balancing in Mobile Ad Hoc Networks," Proceedings of MobiHoc 2000, Aug 2000.