RAPPORT DE RECHERCHE

L R I

IMPROVE TCP PERFORMANCE IN AD HOC NETWORKS, WITH DELAY ROUTING AND PATH RESERVATION

TRIANTAFYLLIDOU D / AL AGHA K

Unité Mixte de Recherche 8623
CNRS-Université Paris Sud –LRI

03/2009

Rapport de Recherche N° 1514

CNRS – Université de Paris Sud
Centre d'Orsay
LABORATOIRE DE RECHERCHE EN INFORMATIQUE
Bâtiment 490
91405 ORSAY Cedex (France)

Improve TCP Performance in Ad Hoc Networks, with Delay Routing and Path Reservation

Despina Triantafyllidou and Khaldoun Al Agha
HIPERCOM@LRI – University of Paris XI
Orsay Cedex, 91405, France
{dtriant, alagha}@lri.fr

Technical Report

March 6, 2009

Abstract

In the present work we examine the impact of average path-delay routing (APDR) to TCP, in ad hoc networks. The route discovery algorithm is based on the lowest average-delay path, and is OLSR based. Although such an approach might seem evident for the performance of TCP, its simulation study on realistic MANET scenarios allows to draw interesting conclusions.

First, the route selected by OLSR is not always the best one from the viewpoint of TCP round-trip time (RTT), so the shortest path does not necessarily guarantee the lowest delay. Therefore, a decrease on the end-to-end average is often achieved at the expense of a longer path. Even though the collision probability does not increase as a result of the increased hop count, APDR results in reducing the overall throughput. As shown, events of regular route oscillations cause packet reordering, and in the context of TCP, evoke duplicate acknowledgements (ACKs) and retransmissions. The route oscillations happen because a node cannot tell apart the delay it induces to the channel, from the total delay.

Second, only the delay as a routing metric is not sufficient, without the invocation of a path reservation scheme, to avoid the route oscillations. In ad hoc networks with no mobility, we apply APDR using source routing, without changing the path during the TCP flow. With the path reservation, APDR outperforms standard OLSR and improves the throughput. In networks incorporating mobility, because the path changes due to route disconnections, packet reordering cannot be avoided. The APDR study helps to deeper comprehend the TCP behavior with respect to density and mobility, and yields for source routing, to avoid the multiple paths.

1 Introduction

TCP remains the standard transport protocol for reliable communications between client/server pairs over the Internet. Over the years, serious efforts have been made aiming at adapting TCP to the highly unreliable wireless environment. This fact itself poses the following contradiction. Normally, TCP adapts its rate based on the capacity dynamics of a network which is assumed to be error free. In the wireless context though, the channel is highly lossy, signal attenuation and interference are very common, and with the emerging mobility, route failures happen often. TCP is unable to detect such causes of packet loss, because its basic flow control relies exclusively on round trip delay measurements and sequence number tracking. On the other hand, the proliferation of personal computing devices has increased the interest in the MANET area, demanding connectivity regardless of physical location. A self-evident solution in this context, would be to locate the good quality paths, and indicate those to TCP, to ensure connectivity and low delay for its flows. Therefore, we are referring to TCP-oriented routing.

Since the early years of ad hoc networking, it is argued that efficient routing is a major performance factor for TCP [1–3]. The routing protocol characteristics have a significant impact on its performance, and yet, the impact of route failures to TCP are much more significant than the delay variability introduced by the channel contention at the MAC layer, originating by the DCF operation. The reason is that, due to the way packets are handled at the network layer, upon a route failure everything is discarded. TCP ends up into timeout retransmissions, i.e., its fast recovery mechanisms do not apply. It is essential, however, to make TCP work seamlessly over MANETs, and this can possibly be achieved by providing smooth underlying routing and channel access services.

In the present work, we investigate whether TCP can be enhanced, when its connections are forwarded via low delay paths. These paths are discovered using a link-state protocol, e.g. OLSR [4], using a hop-delay metric. Average hop-delays are locally estimated for any two neighboring nodes, and also take into account the queueing delay. These local estimations are conveyed to the entire network via the link state advertisements (TC messages), and are used to discover the lowest delay paths for all destinations. The approach is based on this single metric, and its complexity is the same as that of the classic Dijkstra algorithm.

The rest of the article is organized as follows. Section 2 presents the bibliographic work related to i) QoS routing for MANETs, and ii) TCP improvements for ad hoc networks. Section 3 provides a detailed description of the APDR and its implementation issues, based on the asymptotic delay analysis of [5]. Section 5 describes the route oscillation problem, which is highlighted by APDR, and its effect upon TCP. Section 6 presents the findings from the implementation of APDR in mobile scenarios. Finally, Section 7 shows how to overcome the route oscillations using path reservation. Section 8 concludes the main findings and indicates future work.

2 Related work

Delay-based routing is not innovative, although its evaluation in the context of TCP is first done here. Bandwidth and delay routing metrics for ad hoc networks have been proposed by [6–8], for the benefit of real time multimedia applications. The work in [7] shows that finding a path that satisfies two of the following metrics, delay, loss probability, number of hops and jitter, is NP-complete. Actually, the paper proves that finding a path, subjecting to any two metrics with additive composition rule, is equivalent to the *partition* problem, which is NP-complete[1] [9]. So, the only combination left for multi metric routing is bandwidth, which is not additive, and one of the other four. By the same authors it is argued, however, that "a path with both maximum bandwidth and minimum delay may not necessarily exist", while [6] proves that the shortest hop path does not offer the optimal bandwidth and delay. The same argument is expressed in [8], where cross-layer routing is performed based on three parameters measured at the PHY layer, to provide the routing layer with an overall view of the MAC and PHY state.

OLSR has already been used to support multiple metric routing constraints, by including link quality information in its control messages [10]. The route oscillation problem is formally stated in [11], in the framework of OLSR QoS routing for multimedia applications. Using bandwidth requirements, the authors propose a monitor scheme over the traversing flows, to estimate the amount of bandwidth that each one consumes. For this purpose, each node must remember all its subsequent hops, apart from its next.

The authors of [5] perform an asymptotic analysis of the path delay distribution in multihop networks. After the obtainment of the distribution, the authors seek the shortest path that satisfies an over-delay ratio ϵ, such that the probability that the average delay is greater than a certain threshold is less than ϵ. The delay threshold varies according to the multimedia application requirements. The main finding of the delay analysis is that for a large threshold, this probability is a power law. Finding the optimal route that minimizes an over-delay ratio, rather than the average delay is NP-hard, according to [12].

2.1 TCP Oriented Approaches

The performance of TCP in the context of mobility and the effect of the dynamic topology to the network performance have been initially studied in [1], where the authors consider dynamic source routing (DSR). Follow-up work has evaluated TCP over different routing protocols and techniques in ad hoc networks. Since in today's mobile networks routing is primarily concerned with connectivity, the effect upon the TCP performance of three different protocols' buffering strategy during a route break, is demonstrated in [2], via simulations. The results yield for packet caching and fast route restoration techniques, in order to prevent TCP from severe retransmissions. TCP tuning in the MANETs context remains, however, quite complicated. Long-term MANET scenarios have been studied in [13], where the network connectivity and MAC success are also evaluated in the context of TCP and OLSR. The most interesting finding is that, with OLSR

[1]The *partition* problem can be formalized as follows: given a set of integers $\{a_1, ..., a_n\}$, determine whether there is a partition of the integers into two subsets, such as the sum of the elements in one subset is equal to the sum of the elements in the other.

maintaining the routes, the network connectivity and the TCP throughput are improved, when mobility increases.

According to [14], the routing metric to optimize is the estimated TCP throughput. The TCP source measures loss and delay for a path, by probing packets periodically. The node changes its selected path dynamically, so that the throughput of the used path is the highest. This routing was added to the dynamic source routing (DSR) protocol, and evaluated in an ad hoc -without mobility- testbed, achieving a considerable improvement in TCP.

An attempt to upgrade the TCP performance by exploiting a link state protocol was first attempted in [15]. The OLSR HELLO messages were used to estimate the local round-trip times at the MAC layer, and those RTTs were aggregated during the TC forwarding procedure, to set the TCP retransmission timeout timer accordingly. The timer adaptation algorithm proved beneficial for the dense MANETs. What is more important, the TCP end-to-end operation and semantics remained unchanged.

Recently, avoiding interference caused by a TCP flow itself is one of the problems studied in the context of ad hoc networks. The work in [16] introduces a more conservative policy for increasing the TCP window during the congestion avoidance phase, in a static chain topology. In [17], the same authors evaluate the previous mechanism in the presence of mobility, when multiple TCP flows originate by the same sender. The goal is to equally share the goodput among the different flows, preventing a greedy TCP behavior. A bit more promising though is the work in [18], where TCP deals with contention instead of congestion. A new state variable is defined for TCP, namely the *contention window* and new states, based on the contention load and the delay estimation, are introduced, namely the *light contention* and the *severe contention*. The amount of data injected into the network is adapted according to the contention level. This approach is much more closer to the instability problem exhibited by TCP, in these networks.

A rather popular approach, until a few years ago, was the use of active queue management [19], to provide feedback to TCP from the lower network layers, usually with a form of an explicit loss or congestion notification signal [20]. As a result, a number of TCP variations for the TCP rate control were proposed. As we argue, such schemes do not provide TCP with global information of the network state, since the probability that a connection will be informed depends on the probability for dropping/marking an IP packet by an intermediate hop, along the path.

On the other hand, the bandwidth-delay product concept applied to the context of TCP [21], mainly attempts to limit the number of potential outstanding segments. Actually, such a limitation has two objectives; first, it makes the consequences of a packet loss less catastrophic for TCP, because go-back-N will be performed for a small value of N, and second, it may decrease the channel contention. The latter has not been proved, though. However, the impact of the buffer size to the TCP window is ignored, in the particular study.

3 Average Path-Delay Routing (APDR)

The throughput of TCP is traditionally seen as the data receiving ratio of the TCP client, and varies inversely with the time interval needed for the data

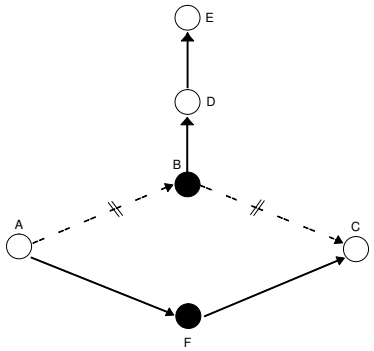


Figure 1: APDR: the {A, F, C} route is preferred, instead of {A, B, C}.

transfer. In MANETs, this interval depends on different sort of network events, the most prevalent of which are the MAC layer retransmissions and the route failures. An efficient route selection, based on current hop-delay information may reveal more efficient paths for TCP than OLSR, making it operate smoothly under the changing topology and the varying traffic conditions.

The heuristic used by standard OLSR determines the MPR nodes so as to mitigate the overhead produced by the flooding operation of the link-state algorithm. Such an approach for the MPR selection provides the shortest path in terms of number of hops, and ensures that the control overhead is as low as possible. However, it does not advertise the links with good quality characteristics, such as high bandwidth and low delay. These links might remain idle, throughout a TCP flow lifetime.

It is fairly reasonable, though, that local estimates for some quality parameter of a link might indicate that this link is preferable to serve a flow, for a specific destination. Consider, for instance, the static topology of Fig. 1, where nodes B and F are MPRs of A, because B covers D and C, and F covers some other two-hop neighbors of A. Suppose that C is also reachable via F. The default OLSR path between A and C, i.e., {A, B, C} might not ensure the lowest delay, because of the BE transmission, so the {A, F, C} route would be preferable.

In the case we examine, the parameter of interest is the average hop-delay, including also the queuing latency. The hop-delay estimation is a rather complicated procedure, which must take into account the queue waiting time, the retransmission attempts at the MAC layer -channel access delay- and the propagation delay of the MAC frames. In the next paragraph, we discuss a model for the estimation of the delay distribution in 802.11 multihop networks. In our study, we use the HELLO messages exchange, along with their sequence numbers, to compute this average value. The route discovery algorithm uses the lowest average among all possible paths, rather than the hop-count metric, to calculate the routes.

Our intention is to reveal the delay efficient paths to the TCP flows, and to examine whether TCP can benefit from this selection. Intuitively this seems reasonable, but as the present study shows, the impact upon TCP is not quite predictable. Before presenting the study results, we discuss some practical im-

plementation issues on the delay estimation over an 802.11 ad hoc network.

3.1 Average Hop-Delay Estimation

The work in [5] obtains both one hop and multihop analytical delay estimates for an ad hoc network using probability generating functions. The main finding is that the asymptotic delay distribution can be expressed as a power law. Based on the latter result, a cross-layer delay estimation protocol and a delay-based routing algorithm are derived. The routing scheme is well adapted to the QoS requirements of real time multimedia applications, as argued in the article. The model has characteristics as following described.

- The single hop delay distribution, β, is based on the knowledge of the collision probability p, and the average of the channel occupancy distribution $C(z)$ [2], as follows,

$$\beta(z, L, p, k) = \frac{C(z)^{k+1} - C(z)}{C(z) - 1} \frac{z^L}{k} \times (1 - p + p\beta(z, L, p, 2k)),$$

  where L is the packet length, and k the maximum contention window of the 802.11 MAC protocol. The above are MAC layer parameters, therefore the extended protocol needs to interact with the MAC layer.

- Additionally, the multihop delay distribution is based on the knowledge of single hop characteristics along the given route. This clearly concerns the functionality of the routing protocol, so the protocol does not need inter-level interactions on this part.

This is a well defined model for the delay estimation, and its authors also propose a protocol framework for an implementation in a real environment.

For the needs of our present evaluation, the average of the hop delay is taken by giving a certain *weight* to the most recent delay measurement, taking also into account the previous average values,

$$avg\_delay = weight * delay + (1 - weight) * avg\_delay.$$

This allows to provide the network with some stability. Apparently, a large *weight* makes the routing more impulsive to topology alterations, causing frequent path changes. On the contrary, a small *weight* value yields for a more conservative routing, where the system keeps a memory of the past path conditions. The role of the *weight* is explored in the simulations.

[2] The collision probability can be estimated by OLSR, by detecting the missing HELLOs' sequence numbers. On the other hand, information on channel occupancy is not known; that concerns the internal functions of the wireless card. However, the card sends special interrupts to the driver acknowledging successful frame transmissions. This allows to measure the service time of transmitted packets, and to deduce the access time in case of broadcast packets, such as OLSR HELLOs, since they are not retransmitted upon a collision.

3.2 Path Selection

The average delay value has to be conveyed to the rest of the network for estimating the fastest route. This is possible to do with the link state advertisements of OLSR (TC messages) which are forwarded by the routing protocol. Each advertisement can carry the average delay value, along with the topology map advertized by each link. For this reason, the TC messages must carry the whole neighbor set information, instead of the MPR set, in order to supply a global view of the network topology.

The goal is then, to choose a certain route that minimizes the sum of the average delays. A Dijkstra-based route discovery algorithm selects the path that minimizes these costs. Note that the shortest hop constraint is not of interest, therefore the calculated path can be longer, in terms of number of hops. In contrast with standard OLSR, routes are not only re-estimated when the network topology changes, but also upon a new average delay estimate. As will be shown, due to the delay based re-calculation of routes, a TCP flow might follow more than one paths in its lifetime, which eventually harms TCP.

4 Evaluation Model

For the performance evaluation of APDR, OPNET [22] is used applying the optimized simulation model of [13]. The TCP New Reno version [23] runs over an IEEE 802.11 MAC layer operating at 11 Mbps. OLSR maintains the routing layer, and the interarrival times for the HELLO and TC messages are kept with the default values. The rest of the parameters values for TCP and MAC 802.11b are as discussed in [13].

For the scenarios incorporating mobility, the nodes are randomly distributed in a region of 1000m $\times$ 1000m. Their movement consists of a sequence of random length intervals of average 30 seconds, during which each node moves towards a constant direction, at a constant velocity. The nodes mobility is uncorrelated, and their velocity is uniformly distributed over $(0, V_{max})$. To balance the arrivals and the departures in the area, as soon as a node leaves the zone of periphery, it is again reflected inside.

Results for the mobile scenarios are gathered after 8 hours of simulation time, which gives the network the ability to generate lots of connections between randomly selected TCP pairs. So, enough samples are taken in order to estimate the throughput at the end of the simulation, which corresponds to the network's steady state. For further analysis regarding the point of stability of the system, we redirect to the evaluation of the simulation model [13].

Unless otherwise stated, during the experiments FTP connections transfer files whose sizes follow a pareto distribution with a *location* parameter equal to 1 and *shape* equal to 2. The mean file size is 2 MB. The FTP flows begin after the first 50 seconds of the simulation, in order to have the routes already established by the link state protocol. The flows interarrival time is uniformly distributed in the [0,5) seconds interval. The sender/receiver pairs are randomly chosen. Upon a connection termination, a new FTP application is initiated, after a 2$\times$MSL[3] period.

[3]Maximum Segment Lifetime

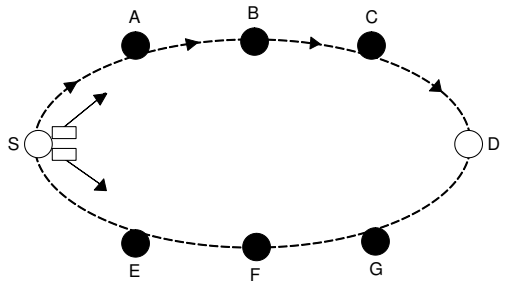


Figure 2: The route oscillations scenario.

A channel with 10% packet loss ratio due to errors is assumed. The buffer size is set to 50 segments. Using this model, we next investigate how TCP responds to APDR. We initially consider a static multihop topology, to exclude the effect of mobility and route disconnections from the study.

5 Route oscillations

Consider the static network topology depicted in Fig. 2, where a source node S has to transmit data to the destination D. There are two possible routes from S to D, namely $p_1 = S, A, B, C, D$ and $p_2 = S, E, F, G, D$. One TCP connection serves traffic from S to D. Suppose that the routing algorithm initially selects p_1 for the data transfer, i.e., the next hop from S is A, and the links of p_1 are serving the data traffic while the links of p_2 are transmission free. This does not imply that p_2 is idle; in case of a parallel transmission, its links are still subjected in setting their NAV, as required by the CSMA/CA protocol.

Now, let $d(p_1)$ and $d(p_2)$ be the end-to-end delays along the two paths, p_1 and p_2, respectively. After the p_1 selection, it becomes $d(p_1) > d(p_2)$. At that point, S should change to path p_2, selecting E as its next hop. In a similar way, after a while, p_1 is advertised as the least delay path, as it becomes $d(p_1) < d(p_2)$. In response to this change of the relation between the two delays, S will again select a different next hop.

The case described above was detected during simulations conducted with the topology of Fig. 2. All forwarding nodes are 200m away from their immediate neighbors, and the vertical -top to bottom- distance is 400m. Only one TCP flow between S and D is initiated, and the simulation duration is 1800 seconds.

In Fig. 3, the points upon the graph represent the average delay of the path at the instants of the route change, for two different values of the *weight* parameter. When a larger *weight* for the average delay estimate is used, the path changes more often, because the algorithm is more sensitive to the delay variations. On the contrary, with a smaller *weight* value, the routing algorithm is more stable, keeping memory of the previous states. In other words, taking into account the latest delay estimation with a large coefficient results in more often route oscillations, throughout the simulation. The graph also exhibits an

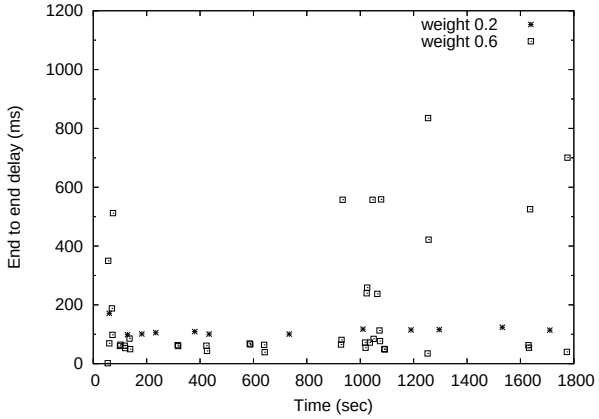


Figure 3: Average delay of path at instances of route change.

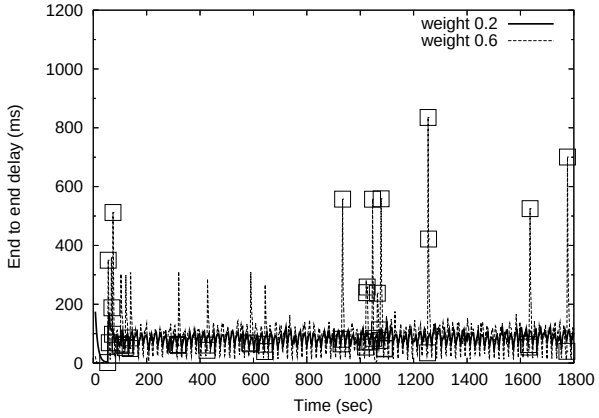


Figure 4: Average delay during simulation.

increased delay jitter, for a large *weight* value.

The higher delay variation due to the larger *weight* is also depicted in Fig. 4. The end-to-end delay is measured every 2 sec. The small *weight* offers a smoother delay, with less bursts. The marked points upon the graph with the *weight* value of 0.6 represent again the route change instants. There are however local peaks, upon which the path does not change. In all those cases, we note two successive path changes in the last four seconds before the peak, which means that, at the time of the peak, some packets are served by p_1 while some others by p_2. Therefore, the local buffers of the stations in both paths are occupied, and their packets subject to increased contention. This explains the increased delay, after the consecutive route changes.

TCP responds to the often path changes with a throughput reduction, as shown in Fig. 5. The reason is found in the same Figure, where the percentage of reordered segments increases with the route oscillations -more precisely with the delay weight. Since there are two different paths serving the same flow, the network cannot guarantee an in order delivery of packets. TCP, however, de-

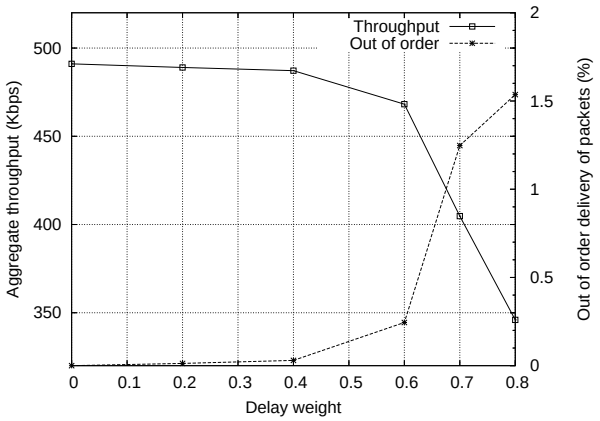


Figure 5: Throughput and out of order delivery percentage, for different values of *weight*.

Table 1: packet reordering given an increasing delay weight

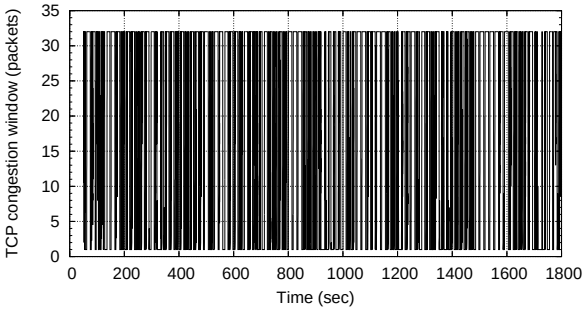
Delay Weight	**Reordered Segments**
0.2	551
0.4	1121
0.6	5052
0.8	29081

pends on the correct sequence number delivery to create its ACKs and to adapt its rate [24]. Remember that, upon a third out of order delivery, a duplicate ACK is sent as a feedback to the TCP server, which enters the fast retransmit and fast recovery phases. Table 1 shows in absolute numbers the amount of reordered packets, with respect to different delay weight values. The out of order delivery entails lots of retransmissions and congestion window reductions, as shown in Fig. 6, for $weight = 0.8$. So, the throughput degrades.

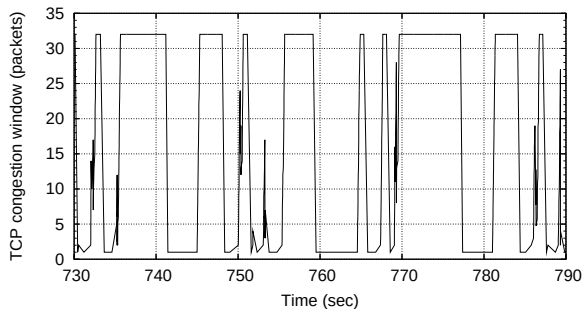
More specifically, what is shown in Fig. 6(a) is the evolution of the congestion window during the connection. Taken in a smaller period of 60 seconds, in Fig. 6(b), it seems that the window often reaches its maximum value, which keeps for a relatively long time, i.e., a few seconds. A closer examination showed that during the time the window is kept that high, the TCP timeout has a magnitude of seconds. According to [25], the maximum timeout value is at least 60 seconds. We find that, during the time the congestion window appears with its maximum value, TCP waits only one packet to be acknowledged, which consequently costs in throughput.

The same figure demonstrates the congestion window decreasing either to one, due to congestion, or in half, due to out of order packet delivery. We verify that there are no dropped data at the MAC layer, and although the virtual carrier sensing is not enabled for this scenario, the data collisions percentage does not exceed 0.55%. This percentage is falling down to 0.4%, as the *weight* increases up to 0.8.

Again, regarding the examined scenario, if the TCP source was aware of the load that its originating traffic adds to the selected path, it would take into



(a) During whole simulation



(b) During one minute

Figure 6: Congestion window progress, for $weight = 0.8$.

account its own resource occupation before changing between the two paths. In other words, it would know that itself is responsible for a specific delay amount added to its path. Speaking with bandwidth metrics, for example, it would be feasible to estimate the bandwidth that each node utilizes for each traffic stream. By subtracting this bandwidth from the total available, a node can have an estimation of the remaining available, and can compare it with the respective available bandwidth of all the alternative paths, in order to select the one with the highest bandwidth. A similar procedure, given a delay metric, is by far more complex, since in a mobile ad hoc topology with a varying number of nodes and flows per time, the delay contribution cannot be estimated. Moreover, constant route changes are a priori expected due to mobility, therefore one cannot distinguish between new beneficial routes and oscillating routes.

6 Evaluation in Dynamic Environment

Since APDR is primarily intended to be evaluated in MANETs, we next investigate its effectiveness upon realistic mobile scenarios. The nodes move according to the random walk scheme described in Sec. 4, and their number remains constant throughout the simulation. Unless otherwise stated, the number of concurrent TCP connections is 10, and $V_{max} = 2$m/sec. The mobility pattern is the same in all the experiments, because the same seed is used for the initial node placement and for the selection of the velocity and the direction.

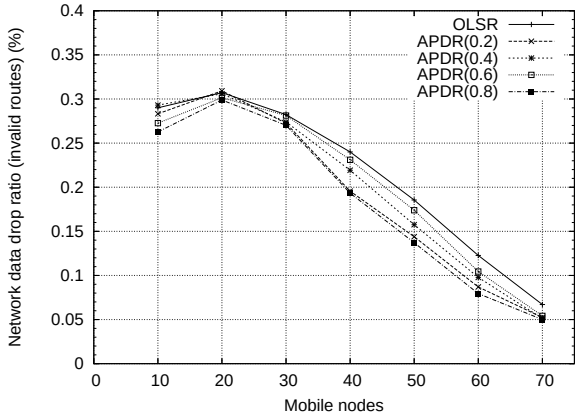


Figure 7: Comparison of routing failures of standard OLSR and APDR in mobile topology.

Therefore, we first compare the connectivity level achieved by the two routing schemes. In Fig. 7, the invalid routes percentage is depicted for an increasing number of mobile nodes and for different *weight* values. Note that the overall percentage does not exceed 3‰, depending on the network density. Although standard OLSR achieves slightly better connectivity, the difference between the two routing schemes is of the magnitude of 1‰. The two routing schemes can be considered equivalent, due to the negligible value of the maximum drop percentage.

Regarding TCP, Fig. 8 depicts the achieved throughput of the APDR scheme compared to standard OLSR, for different *weight* values. In all cases, standard OLSR prevails on the APDR. The difference is very small for 10 mobile nodes, and increases with the network density. This throughput reduction, when applying APDR, motivates further investigation of these scenarios. However, we already have a clue that this might be attributed to the packet reordering induced by route oscillations. The throughput reduction is more intense in dense networks, as shown by the graphs, in which case the range of paths that can be selected is greater.

First, we explore the RTT that TCP encounters, whose mean values are shown in Fig. 9(a). Using APDR leads to a smaller RTTs, and while the network density increases, the RTT drops up to 22% for *weight* 0.2, and up to 37% for *weight* 0.8, compared to standard OLSR. Therefore we have the following contradiction; given a reduced round-trip delay, the throughput decreases. Mind, however, that the RTT estimates conducted by TCP are based on successful round-trips of packets carrying the expected -by the receiver- sequence number, and of their corresponding ACKs. We call this a "healthy" round-trip. TCP, however, never conducts measurements based on out of order packets, or duplicate ACKs. In cases like those, the RTT estimations are disregarded. Therefore, what is shown in Fig. 9(a) is that a healthy round-trip is completed faster, due to the low delay path that APDR has selected. So the previous contradiction stands.

Another point that helps justify the reduced throughput with respect to the lower packet delay is the large value of the timeout timer. Although this value

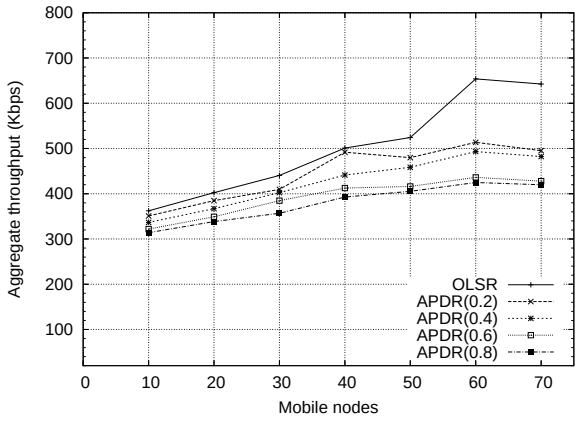


Figure 8: Throughput comparison of standard OLSR and APDR in mobile topology.

Table 2: Average path length in mobile scenarios

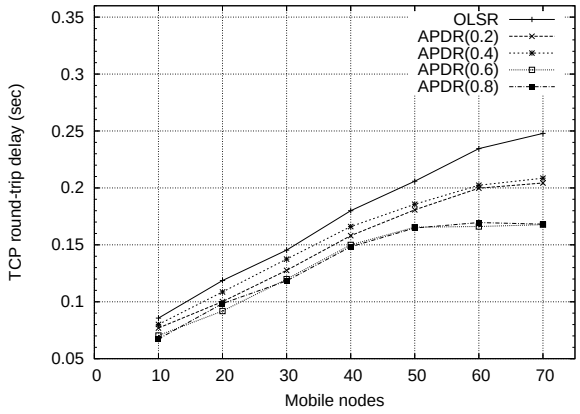
Routing scheme	**Mobile nodes**						
	10	20	30	40	50	60	70
standard OLSR	2.2	2.7	3.2	3.5	3.7	3.8	3.8
APDR	**2.5**	**3**	**3.5**	**3.8**	**4.1**	**4.2**	**4.3**

is set according to RTT measurements, tracking down the RTT values does not reveal at what point the timer is doubled. We noticed this in the scenario of Fig. 2.

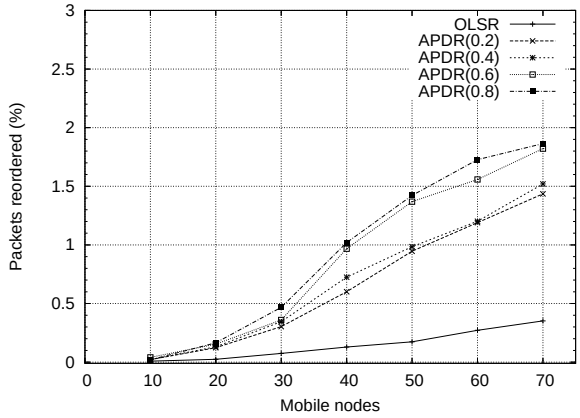
Nevertheless, TCP is again harmed by the packet reordering, as shown in Fig. 9(b). The percentage of packets arriving with a sequence number greater than the expected increases with the *weight* value, i.e., with the frequency of the path change. The connection wastes too many resources in retransmiting packets that -may- have not been dropped at all, while at the same time, the healthy packets experience quick round-trips. As noted in Sec. 5, the larger the *weight* value, the more severe the out of order delivery of the TCP segments. TCP, therefore, loses the race in the retransmissions, despite the quick paths that serve its flows. The out of order delivery percentage is not increasing in the case of 10 nodes; the difference is negligible. This is attributed to the fact that when the network is sparse, there are not many different paths to select.

Table 2 summarizes the average path length, for the two routing schemes. We have found that the route length is independent of the *weight* value. APDR provides longer paths in average, which means that each path consists of shorter hops. An unexpected consequence of the above is that the decrease on the TCP packet round-trip delay, as it was seen in Fig. 9(a), is done in the detriment of a longer path. So, the shortest hop metric used by OLSR is not the most efficient from the standpoint of the round-trip minimization. Moreover, due to the shorter hops, the collision probability decreases, as it is presented in Fig. 10. This is a quite positive aspect of APDR, since it implies that the traffic is spread along different paths, and hot-spots are avoided.

At last, it is worth noting that similar results are also drawn for higher



(a) TCP estimated round-trip delay.



(b) Percentage of packet reordering.

Figure 9: Performance metrics of TCP. Comparison for standard OLSR and APDR in mobile topology.

mobility scenarios, i.e., V_{max} = 5m/sec. Different number of concurrent FTP flows (2 and 5) were also considered, but we find that in that case the network is under-utilized, and the results are less stable.

7 Path reservation for TCP flows

Route oscillations have to be avoided in the presense of TCP traffic. This is achieved by preserving the path of a TCP flow, ensuring that all its segments follow the same hops. APDR with path reservation (APDR-PR) is next investigated. We find it wise to exclude mobility and any effects of topological changes, in order to understand whether the route stability can determine the performance of TCP over APDR.

APDR-PR is implemented as follows. Link state advertisements are exchanged during the whole simulation time, as usual, so the nodes are up to

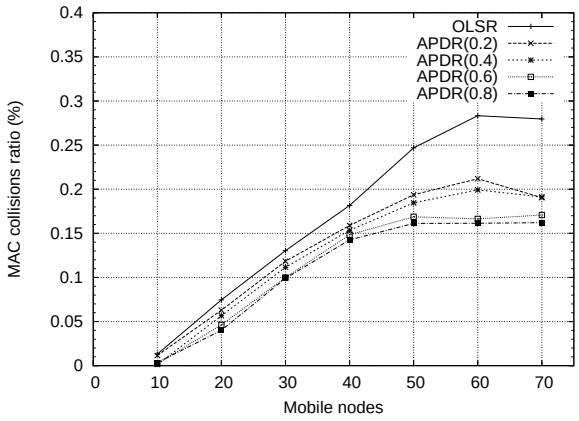


Figure 10: Lower collisions percentage of APDR, compared to standard OLSR.

date, with respect to average delays. New TCP connections take into account the most recent information, to select the route that, at that time, offers the minimum average delay. During the TCP flow lifetime, however, the path which is initially selected remains unchanged.

Note that APDR-PR cannot guarantee that the packets always follow the least delay path. It is possible that delay estimations during the file transfer indicate lower delay routes. However, the TCP flow has to adhere to the one initially selected. What is achieved however, is that (i) a delay efficient route is initially chosen, (ii) all segments of a certain TCP flow follow the same hops, and (iii) new TCP connections select their path based on current link state information, i.e., the link state principle is preserved. The results of APDR-PR are as described next.

For a static network, the system performance strongly relies on the topology, i.e., the initial nodes' placement, which is random, in the examined scenario. The experiments are run for several hours, during which hundreds of TCP flows are initiated -concurrently there always exist 10. Note in Fig. 11 that the throughput improves notably, as the network becomes more dense.

As before, the RTT encountered by TCP is found considerably smaller; the RTT and other performance metrics are presented in Table 3. Only now, the out of order delivery of TCP segments is not increased, as shown in Fig. 12. In fact it decreases, contrary to what happened in Fig. 9(b). Because in the particular scenario the nodes do not move, one would have expected that -since the network is not partitioned- the paths should never change, and this percentage should be zero. However, OLSR incorporates timeout procedures, for renewing its topology entries. Therefore, the routing table invalidates its old entries once in a while.

The same remark is highlighted by the non-zero -although small- percentage of invalid routes. As we confirm, this is attributed to the timeout intervals OLSR uses to refresh its topology map and its routes. This percentage is lower when applying APDR-PR. This is expected, since the reserved path gets never invalidated. The only chance that a route is not available is upon an arrival of a new TCP flow, in case OLSR has not yet established the required path. The estimated collisions percentage at the MAC layer is quite small, but also appears

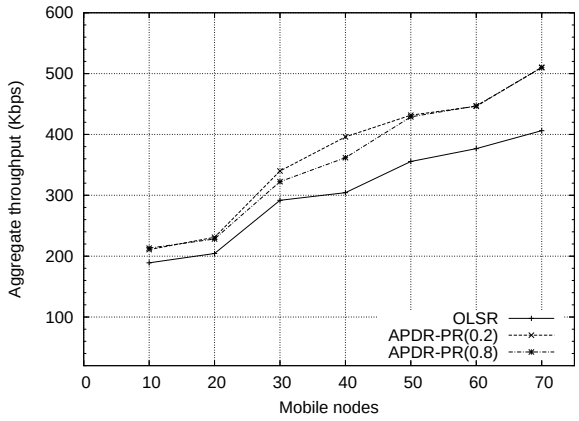


Figure 11: Throughput comparison of standard OLSR and APDR-PR in static topology.

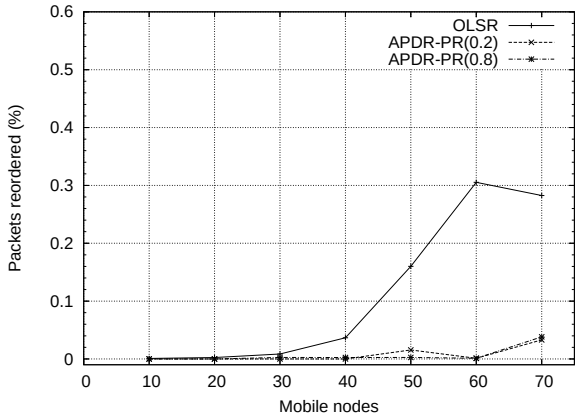


Figure 12: Out of order delivery percentage with standard OLSR and with APDR-PR in static topology.

Table 3: Comparison of performance parameters from different network layers

Performance parameter	**Routing scheme**	
	Standard OLSR min – max*	**APDR-PR min – max***
TCP RTT (msec)	60 – 420	60 – 150
Invalid routes (%)	0.34 – 0.78	0 – 0.34
MAC collisions (%)	0 – 0.26	0 – 0.13

* with respect to network density

reduced when APDP-PR applies. Neither the collisions nor the invalid routes are dominant performance factors, since their percentage is small; however we find improvements in those also.

At last, note that with the path reservation the value of the *weight* parameter

is not significant. Since the path does not change, this is reasonable. The *weight* value would be influential in mobile scenarios, where the topology changes, and the sugested route depends on the average delay estimation.

8 Conclusions

In this paper, we investigate whether TCP can improve its performance, when selecting low delay paths for its connections. For the verification of this hypothesis, we apply an average-delay based routing scheme, by modifying the metric of the standard OLSR route discovery algorithm. The evaluation is done in both static and mobile ad hoc networks, by simulations. The conclusions drawn help understand aspects of the TCP functioning over ad hoc networks.

Among the most interesting findings is that the lowest delay path is not always the shortest one. Hence, routing with respect to average delay conflicts with the minimization of the hop count. Moreover, both in static and in mobile scenarios, APDR exhibits route oscillations. The more impulsive the network is to the delay variations, the more often the route decisions change, leading a TCP connection to split into more than one paths. In scenarios involving mobility, a delay-based selection of the path improves the inter-flow interference as well as the RTT estimated by TCP, but the latter cannot take advantage of this improvement, because the out of order delivery prevails on the reduced delay.

Selecting a route based on its delay, causes the entered traffic to anew increase the delay of the route. However, it is difficult for a wireless station to gain accurate estimates of the amount of delay it introduces to the links. Typically, there are two possible solutions to this problem. First, multipath or load-balancing algorithms have been introduced, to alleviate the relaying load in the congested areas, but this can totally harm TCP. Yet, depending on the particular traffic pattern, load-balancing may not help alleviate the concentrated load [26]. On the other hand, a resource reservation scheme where a node can select the best path given a delay constraint from the application, could overcome this problem. However, it cannot be expected that the optimum path will always the reserved one. At least, it is ensured that a maximum delay threshold is not exceeded. By selecting a path that preserves some delay boundaries, the routing operation is relaxed, and provides more stable paths.

TCP, however, does not have specific delay requirements, and such a threshold cannot easily be defined. Although its performance partially depends upon the retransmission timer setting, because this is adaptive, its value is not unambiguously defined. So, an a priori delay threshold value cannot be fixed. What seems interesting to study, is how TCP throughput is affected by different delay threshold values. For this, the asymptotic delay model of [5] might be interesting to evaluate with TCP traffic, since routing is achieved based on an over-delay ratio.

The alternative solution of our work was to preserve the first best path, during the whole TCP flow lifetime. This avoided the route oscillations and improved the throughput, in random static topologies. What remains to explore is how a path reservation scheme can work effectively in a mobile environment. The mobility effects provide unstable results for TCP. APDR-PR cannot guarantee a throughput improvement; this rather depends on the combination of the topology and the mobility patterns. Future work consists of adapting TCP

rate to a given path, satisfying an end to end delay parameter, and achieving the optimum window for this path.

References

[1] G. Holland and N. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," in *MobiCom '99: Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*. ACM, 1999, pp. 219–230.

[2] S. Papanastasiou, L. M. Mackenzie, M. Ould-Khaoua, and V. Charissis, "On the interaction of TCP and routing protocols in MANETs," in *AICT-ICIW '06*. IEEE Computer Society, 2006.

[3] S. Rajagopalan and C.-C. Shen, "What does using tcp as an evaluation tool reveal about manet routing protocols?" in *IWCMC '06: Proceedings of the 2006 International Conference on Wireless Communications and Mobile Computing*. New York, NY, USA: ACM, 2006, pp. 521–526.

[4] *RFC 3626: Optimized Link State Routing Protocol (OLSR)*, IETF Std. 3626, 2003.

[5] P. Jacquet, A. M. Naimi, and G. Rodolakis, "Asymptotic delay analysis for cross-layer delay-based routing in ad hoc networks," *Advances in Multimedia*, pp. 11, vol. 2007, Article ID 90 879, 2007.

[6] H. Badis, H. Munareto, K. A. Agha, and G. Pujolle, "Optimal path selection in a link state QoS routing protocol," in *IEEE VTC 2004-Spring.*, 2004, pp. 2570–2574.

[7] Z. Wang and J. Crowcroft, "Bandwidth-delay based routing algorithms," in *IEEE Global Telecommunications Conference, 1995. GLOBECOM '95*, 1995, pp. 2129–2133, vol.3.

[8] L. Iannone, R. Khalili, K. Salamatian, and S. Fdida, "Cross-layer routing in wireless mesh networks," in *1st International Symposium on Wireless Communication Systems, 2004*, 2004, pp. 319–323.

[9] M. R. Garey and D. S. Johnson, *Computer and Intractability - A Guide to the Theory of NP-Completeness*. Freeman, California, 1979.

[10] A. Munaretto, H. Badis, K. A. Agha, and G. Pujolle, "A link-state QoS routing protocol for ad hoc networks," in *4th International Workshop on Mobile and Wireless Communications Networks*, 2002.

[11] H. Badis, I. Gawedzki, and K. A. Agha, "QoS routing in ad hoc networks using QOLSR with no need of explicit reservation," in *IEEE VTC 2004-Spring.*, 2004, pp. 2654–2658.

[12] R. A. Guerin and A. Orda, "QoS routing in networks with inaccurate information: theory and algorithms," *IEEE/ACM Transactions on Networking*, vol. 7, no. 3, pp. 350–364, 1999.

[13] D. Triantafyllidou and K. AlAgha, "Evaluation of TCP performance in MANETs using an optimized scalable simulation model," *IEEE MASCOTS*, Istanbul, Turkey, October 2007.

[14] H. Takahasi, M. Saito, H. Aida, Y. Tobe, and H. Tokuda, "Estimated-tcp-throughput maximization based routing," in *LCN '03: Proceedings of the 28th Annual IEEE International Conference on Local Computer Networks*. IEEE Computer Society, 2003, p. 120.

[15] D. Triantafyllidou and K. A. Agha, "Efficient adaptation of TCP's RTO timer to avoid spurious timeouts in MANETs," in *10th International Symposium on Wireless Personal and Multimedia Communications (WPMC)*, Jaipur, India, December 2007.

[16] S. Papanastasiou and M. Ould-Khaoua, "TCP congestion window evolution and spatial reuse in MANETs," *Wirelless Communications and Mobile Computing*, vol. 4, no. 6, pp. 669–682, 2004.

[17] S. Papanastasiou, M. Ould-Khaoua, and L. M. Mackenzie, "Exploring the effect of inter-flow interference on TCP performance in MANETs," in *HETNETs'04*, 2004, pp. 41/1–41/10.

[18] E. Hamadani and V. Rakocevic, "A cross layer solution to address tcp intra-flow performance degradation in multihop ad hoc networks," *Journal of Internet Engineering*, vol. 2, no. 1, 2008.

[19] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking.*, vol. 1, no. 4, pp. 397–413, 1993.

[20] *The Addition of Explicit Congestion Notification (ECN) to IP*, IETF Std. 3168, 2001.

[21] K. Chen, Y. Xue, S. H. Shah, and K. Nahrstedt, "Understanding bandwidth delay product in mobile ad hoc networks," *Special issue on protocol engineering for wired and wireless networks, Elsevier Computer Communications.*, vol. 27, pp. 923–934, 2004.

[22] Opnet simulator. [Online]. Available: http://www.opnet.com/

[23] *RFC 2582: The NewReno Modification to TCP's Fast Recovery Algorithm*, IETF Std. 2582, 1999.

[24] *RFC 2581: TCP Congestion Control*, IETF Std. 2581, 1999.

[25] *RFC 2988: Computing TCP's Retrans. Timer*, IETF Std. 2988, 2000.

[26] S. Kwon and N. B. Shroff, "Paradox of shortest path routing for large multi-hop wireless networks," in *INFOCOM*, 2007, pp. 1001–1009.