Multimedia Systems (2011) 17:287–297
DOI 10.1007/s00530-010-0210-0

REGULAR PAPER

Real-time video streaming over multipath in multi-hop wireless networks

Xiaoyuan Guo · Jiangchuan Liu · Shiguo Lian

Published online: 9 November 2010

Abstract Given the limited wireless link throughput, high loss rate, and varying end-to-end delay, supporting video applications in multi-hop wireless networks becomes a challenging task. Path diversity exploits multiple routes for each session simultaneously, which achieves higher aggregated bandwidth and potentially decreases delay and packet loss. Unfortunately, for TCP-based video streaming, naive load splitting often results in inaccurate estimation of round trip time (RTT) and packet reordering. As a result, it can suffer from significant instability or even throughput reduction, which is also validated by our analysis and simulation in multi-hop wireless networks. To make real-time TCP-based streaming viable over multi-hop wireless networks, we propose a novel cross-layer design with a smart traffic split scheme, namely, multiple path retransmission (MPR). MPR differentiates the original data packets and the retransmitted packets and works with a novel QoS-aware multi-path routing protocol, QAOMDV, to distribute them separately. MPR does not suffer from the RTT underestimation and extra packet reordering, which ensures stable throughput improvement over single-path routing. Through extensive simulations, we further demonstrate that, as compared with state-of-the-art multi-path protocols, our MPR with QAOMDV noticeably enhances the TCP streaming throughput and reduces bandwidth fluctuation, with no obvious impact to fairness.

Keywords Video streaming · TCP · Path diversity · Multi-hop wireless network

X. Guo · J. Liu (✉)
School of Computing Science, Simon Fraser University, Burnaby, BC, Canada
e-mail: jcliu@cs.sfu.ca

X. Guo
e-mail: xxg1@cs.sfu.ca

S. Lian
France Telecom R&D Beijing, 2 Science Institute South Road, Haidian District, Beijing, China
e-mail: shiguo.lian@orange-ftgroup.com

1 Introduction

Multi-hop wireless networks are undergoing rapid development and commercialization, spreading Internet accesses and other network services in personal, local, and old metropolitan areas. Video streaming has been expected to be one of the most important applications over multi-hop wireless networks, e.g., for IPTV, video conference, and online games. Unfortunately, the limited bandwidth and strong interference among the wireless nodes make real-time video streaming remain a challenge task in multi-hop wireless networks.

Path diversity exploits the available paths in a network and maintains multiple routes for each session simultaneously. Through bandwidth aggregation, it enhances routing robustness and decreases delay and packet loss. A series of multipath protocols have been proposed for video streaming [2, 13, 24] and for multi-hop wireless routing [1, 28]. Yet the interaction and the best combination among them are still unclear, particularly for TCP-based streaming. We are interested in TCP for its simplicity, reliability, and built-in adaptiveness, and fairness. Despite the argument that UDP would be better customized for higher-layer applications, today's commercial streaming systems are extensively using TCP for streaming [22, 32]; examples include the RealMedia and WindowsMedia, two dominating products. Sripanidkulchai et al. [32] reported that 40–80% of live streaming used TCP. The existence of network address translators (NATs) and firewalls further makes TCP the preferred choice over UDP for many ISPs and network administrators.

Nevertheless, the backoff and retransmission mechanisms in TCP can lead to undesirable end-to-end delay, which often violates the stringent delay demand from real-time applications. Given the potentially increased bandwidth, path diversity would partly solve the problem. However, with a naive traffic split, TCP could suffer from more severe throughput degradation and fluctuation when run over multiple paths. Specifically, TCP was designed to deliver packets in order and avoid timeout within certain time interval along a single path, which is not guaranteed in multipath routing since different paths have diverse delay/bandwidth characteristics [17]. The problem is further aggravated with the out-of-order packet delivery across multi-path. In this case, TCP would spuriously retransmit packets, making its congestion window unnecessarily small, and thus resulting in significant throughput reduction and fluctuation.

As the major motivation that is to address the above challenges and make TCP a viable protocol for realtime video streaming over multi-hop wireless networks, we propose a novel cross-layer design with a smart traffic split scheme, namely, multiple path retransmission (MPR). MPR differentiates the original data packets and the retransmitted packets, taking into account the information from both the transport layer and the network layer. The key observation is that the amount of retransmitted packets is relatively smaller, but they have more stringent delay requirement. Therefore, the original packets and the retransmitted packets should follow different paths with different optimization criteria, i.e., bandwidth maximization and delay minimization. We demonstrate that this design achieves higher throughput and lower delay, and more importantly, it largely eliminates packet reordering.

To cooperate with MPR, we develop a QoS-aware routing protocol, QAOMDV, which extends the conventional Ad hoc on-demand multipath distance vector (AOMDV) routing protocol [20]. We present the detailed design of practical protocol interfaces for TCP streaming over QAOMDV. Through extensive simulations, we demonstrate that our MPR with QAOMDV noticeably enhances the TCP streaming throughput and reduces bandwidth fluctuation, with no obvious impact to the fairness.

The rest of this paper is organized as follows: Section 2 presents the background and related work. In Sect. 3, we verify the reduction of TCP streaming performance over multipath routing via *ns*-2 simulation. Section 4 proposes the MPR and analyzes its potential improvement for TCP streaming. In Sect. 5, we describe our protocol design in detail. Section 6 evaluates our proposed protocol with *ns*-2. Finally, Sect. 7 concludes the paper and provides some future directions.

2 Related work

Existing study on TCP-based streaming has mainly focused on how to cope with network bandwidth fluctuation introduced by congestion control and to reduce retransmission delay [5, 34], for TCP streaming has largely eliminated the error recovery in the application layer and offered flow friendliness inherently. Solutions include client-side buffering [16], receiver-based delay control (RDC) with router feedback [10], and TCP-RTM that allows 'stepping over' missing packets to mitigate the negative impact of TCP retransmission [29].

Path diversity has long been recognized as an important approach toward delay reduction, load balance, and fault tolerance. One critical concern of path diversity is that the multiple paths may share joint links, which will be a bottleneck. Various algorithms have been proposed to compute node- and link-disjoint paths [36, 38]. A typical example is AOMDV [20], which extends the ad hoc on-demand distance vector (AODV) single path routing protocol to compute multiple disjoint paths using controlled flooding. Recently, there have also been efforts toward multipath routing in multi-hop wireless networks [1, 28].

The multipath TCP (MPTCP) working group aims to support regular TCP sessions over multiple paths without significant modifications to the existing Internet infrastructure. Current drafts include proposals to extend the traditional TCP for multipath operation with multiple addresses [6, 9], to specify a new address family in the socket API for the multipath interface [30], and to provide interfaces accessing to multipath information for applications [31]. Most of them remain in draft status, and have not addressed real-time video applications in multi-hop wireless networks.

Nguyen et al. [23] argued multiple paths should be explored toward bandwidth aggregated TCP streaming, too. Wang et al. [35] analyzed the relationship between TCP throughput and video bitrate and claimed that TCP streaming could have satisfactory performance if the aggregated TCP throughput is 1.6 times the video bitrate. Customized traffic distribution and scheduling algorithms were presented in [13, 24] for maximizing video quality. Multiple description coding has also been suggested [2], albeit with a UDP configuration.

Unfortunately, it is known that, with multipath routing, TCP could suffer throughput fluctuation or even reduction due to inaccurate RTT estimation and packet reordering [8, 14, 19, 26, 27]. Proactive reordering algorithms were proposed in [26] and [27] to mitigate out-of-order delivery. Kandula et al. [14] proposed Flare to divide and cache traffic at the granularity of packet bursts, with the observation that if the time between two successive packets is larger than the maximum delay difference between multiple paths, the subsequent packets can be scheduled on any path with no risk of reordering. SPRIC [37] sorts reordered packets in the same block with interrupt coalescing to eliminate or reduce packet reordering at the destination. Other solutions include dynamically adjusting the TCP reordering threshold to avoid improper retransfer or quickly recovering from spurious congestion window shrinks [3, 41]. All these solutions are relatively complex, while our MPR is simpler with built-in mechanism to avoid reordering. Our solution further explores cross-layer design with QoS optimization for real-time streaming [4, 21, 42].

3 Understanding TCP performance with multipath

Before presenting MPR, we first conduct a series of *ns*-2 experiments to understand the TCP streaming performance with multipath routing in multi-hop wireless networks. We ran TCP-Reno, the most popular TCP version, over both AODV and AOMDV routing protocols. AODV is a widely implemented single-path routing protocol for multi-hop wireless networks, while AOMDV transmits data through two paths simultaneously with even split. We established a multi-hop wireless network with 40 nodes placed randomly in a 1,000 m $\times$ 1,000 m square field. Each node has a radio propagation range of 250 m and its carrier sensing distance is 550 m. The IEEE 802.11 MAC protocol was adopted in RTS/ CTS/Data/ACK mode with a channel data rate of 11 Mbps. The average video bitrate is 800 Kbps and the size of each packet is 1,000 bytes. Most of these settings are adapted from [23, 40].

We simulated three scenarios with different traffic patterns, including one TCP streaming session, one TCP streaming session and seven UDP flows, and eight TCP streaming sessions. For each scenario, 30 runs with different random seeds were executed, and the results are then averaged to mitigate randomness.

Figure 1 shows the average TCP throughput in the three scenarios. Surprisingly, in all the scenarios, TCP streaming performs poorer with multiple paths than with a single path, which would result in worse video quality in the receiver side given the more frequent frame drops and pauses. This challenges the conventional belief that bandwidth aggregation from multiple path routing is beneficial [1]. A closer look suggests that these existing multipath protocols are mainly designed for UDP, with few explicit optimizations for TCP streaming. As a reliable transport protocol, TCP was designed to deliver packets in order

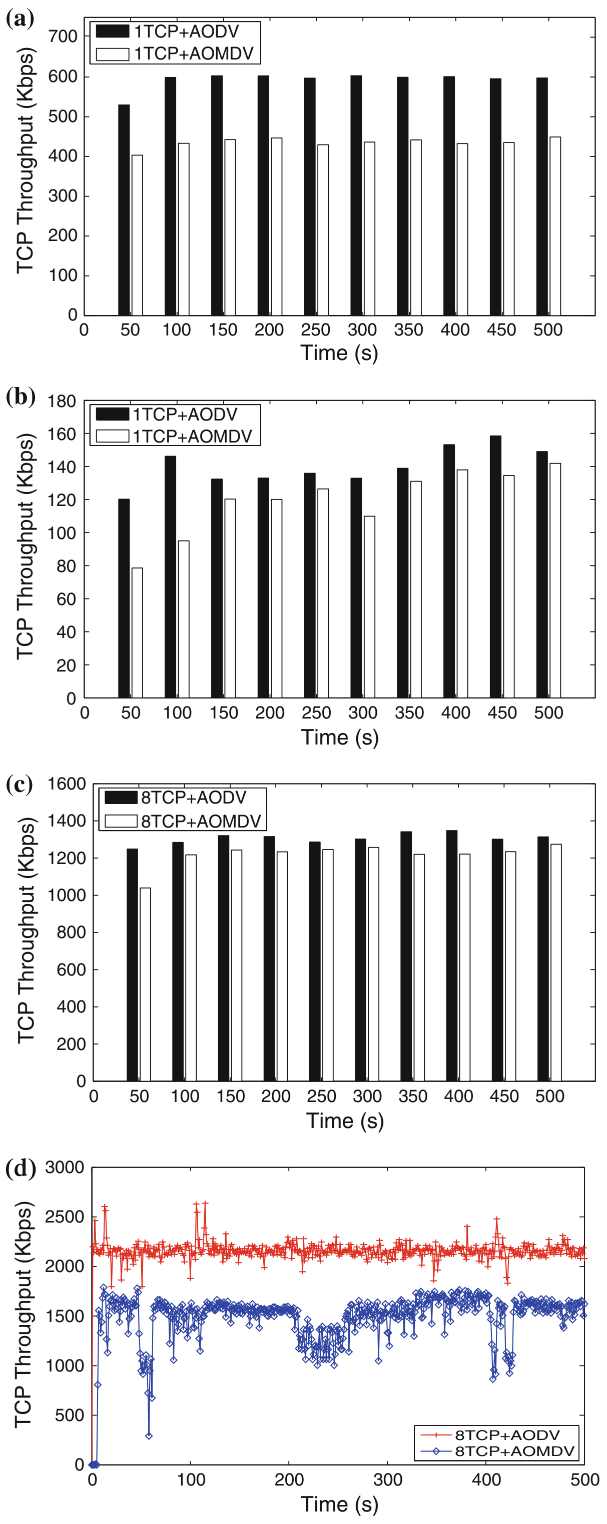

Fig. 1 Comparisons of TCP throughput. **a** TCP throughput (1 TCP session). The average TCP throughputs for AODV and AOMDV are 592.8 and 435.1 Kbps, respectively. **b** TCP throughput (1 TCP session and 7UDP flows). The average TCP throughputs for AODV and AOMDV are 139.9 and 119.6 Kbps, respectively. **c** TCP throughput (8 TCP sessions). The average TCP throughputs for AODV and AOMDV are 1,306.4 and 1,218.8 Kbps, respectively. **d** TCP throughput (8 TCP sessions in one run)

through a single path. To ensure reliable transmission, it uses sequence numbers to identify the order of bytes sent from the source. The destination expects in-order segments and if it receives a packet with an unexpected sequence number, it buffers the out-of-order packet and returns a duplicate ACK to the source. Retransmission is triggered in the sender side with 3 duplicate ACKs or a timeout. Multiple path routing, however, would introduce extra timeout and packet reordering events, resulting in TCP spuriously retransmitting segments and keeping its congestion window unnecessarily small [17]. Consistent with the previous studies [26, 39], we have the following observations:

1. Packets on slow paths may suffer from timeout persistently because of RTT underestimation. For illustration, assume there are n paths between a source and a destination, r_j is the RTT of path j, f_j is the fraction of packets on path j. The retransmission timeout (RTO) period for the i-th packet is RTO_i. According to [26], the current estimated RTT, R_i, can be calculated approximately as

$$R_i = \sum_{j=1}^{n} f_j r_j.$$

   The packet on path j will experience timeout permanently if $r_j > \mathrm{RTO}_i$. Once a timeout event is triggered, the TCP congestion window will be set to one with slow start, which underutilizes the available network resource seriously and brings extra bandwidth fluctuation.
2. Packets may arrive at destination out of order, i.e., the packet receiving sequence in a flow is different from its sending sequence. As shown in Fig. 2, in the 2-path scenario, if packets 2, 3, and 4 arrive earlier than packets 1, 3 duplicate ACKs will be received by the sender, which has to retransmit packet 1. This unnecessary retransmission not only wastes network bandwidth, but also makes the utilization of available network resource low and introduce additional bandwidth variability as well.
3. The interference among multiple paths in multi-hop wireless networks further aggravates the degradation and fluctuation. Different from wired networks, in a multi-hop wireless network, even if different paths do not share common links or nodes, simultaneous transmissions along these paths would still suffer from serious interferences given the broadcast nature of wireless video.

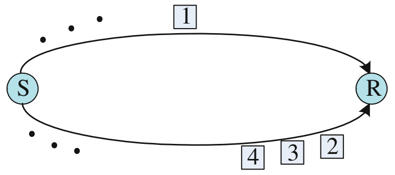

Fig. 2 Packet reordering in multiple path routing

4 Multiple path retransmission (MPR)

In this section, we present the multiple path retransmission scheme (MPR), and analyze how it eliminates the persistent timeout as well as packet reordering problems and thus improves TCP streaming performance with multiple path routing.

Different from the conventional even traffic split, our MPR differentiates the original data packets and the retransmitted packets, and delivers them over two separated paths with different optimization criteria. Specifically, as illustrated in Fig. 3, the number of the original packets is in general dominating, and thus they should be scheduled over the maximum-bandwidth path. On the other hand, the retransmitted packets are of relatively smaller amount, but expected to be delivered more quickly, particularly for real-time streaming. Hence, they will be scheduled over the path with the minimum end-to-end delay. Here we assume that these two paths are disjoint, and we will discuss the impact of correlations between them in Sect. 5.3

MPR can reduce interference among concurrent multi-path transmissions. On one hand, the missing packets are much less than the original packets. On the other hand, the arrival pattern of the retransmitted packets is not as continuous as that of the original packets. Thus, the interference between the retransmitted packets and the original packets along the two paths is much less severe.

Intuitively, the MPR will not introduce extra packet reordering as other multipath protocols do. It will not underestimate the RTT, either, given that the Karn's algorithm [15] in TCP does not count retransmitted segments in RTT estimation. In other words, the RTT estimation in MPR is the same as that in a single-path case. Moreover, since the path delay for the retransmitted packets is smaller than that for the original packets, the retransmitted packets will seldom suffer from further timeout. These together improve the responsiveness and throughput of TCP, and better streaming quality can therefore be expected. Want et al. [35] suggested the number of early packets as the metric to measure

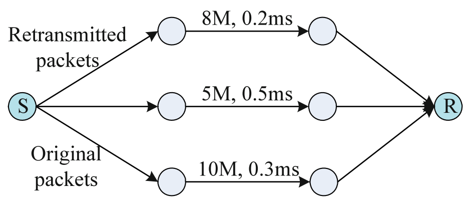

Fig. 3 A network with 8 nodes. The available bandwidth and the end-to-end delay of each path are shown along each path

streaming quality and quantified the number of early packets to be proportional to TCP throughput. Here, the early packets refer to as the packets arriving earlier than their playout times. To validate the effectiveness of MPR for TCP-based streaming, we next analyze the improvement of TCP throughput with MPR.

Consider an example in Fig. 4. At first (time t_1), packet $k+1$ is lost and congestion window size is 4. The receiver buffers the packets with higher sequence numbers and returns duplicate ACKs. At time t_3, three duplicate ACKs are received at the sender side, and packet $k+1$ is therefore retransmitted. Meanwhile, according to the TCP congestion avoidance algorithm [33], the threshold *ssthresh* is set to 2 and the congestion window size is changed to 5, which is one-half of the current congestion window size, 5, plus 3. Then the congestion window increases linearly until a new ACK is received at time t_5 and the congestion window size is set to *ssthresh*, which is 2. The evolution of the congestion windows size is depicted Fig. 5. We define the *loss recovery time* as the time between retransmitting the missing packet and receiving a new ACK. MPR delivers the retransmitted packets along the minimum-delay path, and hence its recovery time is smaller than that of the single path case (see dashdotted lines in Fig. 5). Meanwhile, MPR benefits from bandwidth aggregation in multipath routing, and therefore enhances TCP throughput, too.

We now formally evaluate the performance gain of MPR. For simplicity, we assume that the wireless link losses are masked by link-layer forward error correction (FEC) and therefore focus on the transport-layer throughput improvement. Our analysis extends the classical TCP throughput model [25] to the multipath scenario. We adapt the notations in [25], which are summarized as follows:

- TDP_i: interval between two ‘triple-duplicate’ ACK (TD) loss indications;
- Ad_i: duration of TDP_i;
- A_i: duration of TDP_i without loss recovery time;
- d_i: loss recovery time of TDP_i;
- Y_i: number of packets sent in TDP_i;
- r: RTT;
- p: packet loss rate;
- B: TCP throughput;
- b: the number of packets acknowledged by a received ACK;
- Q: $\frac{1}{E[n]}$, where $\{n_i\}_i$ is an independent identical distribution (i.i.d) sequence of random variables;
- Z^{TO}: duration of a sequence of time-outs.

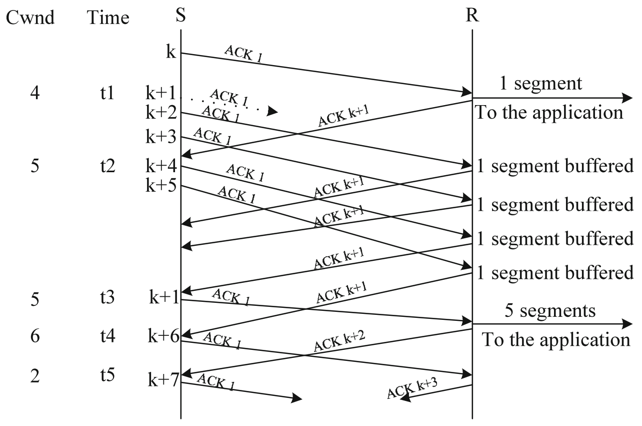

Fig. 4 TCP loss recovery process in single path routing

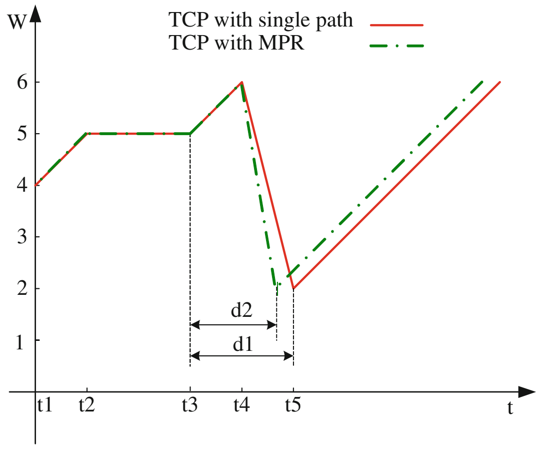

Fig. 5 Evolution of congestion window size during loss recovery in Fig. 4. d_1 and d_2 are the loss recovery times of TCP in a single path and that of MPR, respectively. If the retransmitted packets are sent on the path with minimum delay, we have $d_2 < d_1$

We generalize the window evolution process (Fig. 1 in [25]) with loss recovery time between TDPs, as shown in Fig. 6. As such, $E[Ad]$ can be expressed as

$$E[Ad] = E[A] + E[d]$$

where $E[A]$ has the same formula as that in [25]. As defined earlier, $E[d] = E[r]$. Let Δd be the difference between TCP loss-recovery time in MPR and that with a single-path

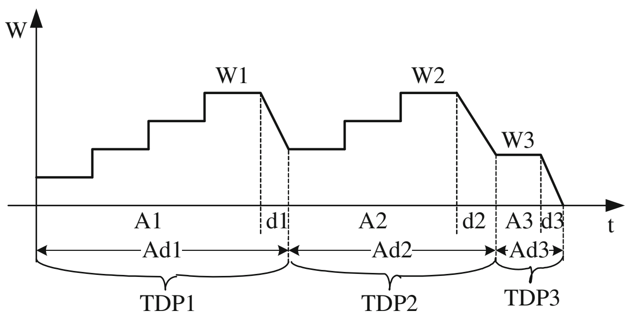

Fig. 6 Evolution of window size over time when retransmission is triggered by three-duplicate ACKs

routing. We can view $\{\Delta d_i\}_i$ as a random sequence obeying uniform i.i.d in [0, $E[r]$]. It follows that $E[\Delta d] = \frac{1}{2}E[r]$. Note that $B = \frac{E[Y]}{E[Ad]}$ is the TCP goodput instead of throughput in MPR, because the packets on the path with the maximum bandwidth are all original ones. Let B_1 be the TCP throughput in MPR and B_2 be the TCP throughput in a single path routing. The throughput improvement is therefore

$$\frac{B_1 - B_2}{B_2} = \frac{\frac{E[Y](1+p)}{E[A]+E[d_1]} - \frac{E[Y]}{E[A]+E[d2]}}{\frac{E[Y]}{E[A]+E[d2]}} > \frac{E[\Delta d]}{E[A] + E[d]}$$

where

$$\frac{E[\Delta d]}{E[A] + E[d]} = \frac{\frac{1}{2}}{\frac{2+b}{6} + \sqrt{\frac{2b(1-p)}{3p} + \left(\frac{2+b}{6}\right)^2} + 2}.$$

Similarly, given triple-duplicate ACKs and timeouts, the TCP throughput improvement can be expressed as

$$\frac{B_1 - B_2}{B_2} > \frac{E[\Delta d] + QE[\Delta d]}{E[A] + E[d] + Q(E[Z^{\text{TO}}] + E[d])}.$$

Here, $E[Y]$, $E[r]$, Q and $E[Z^{\text{TO}}]$ all have the same formulas as in [25]. We can see the TCP throughput improvement can also be interpreted as the enhancement of the loss recovery time of multipath than that of single path case. Therefore, as long as the delay for the path of the retransmitted path is shorter, the TCP throughput can be improved.

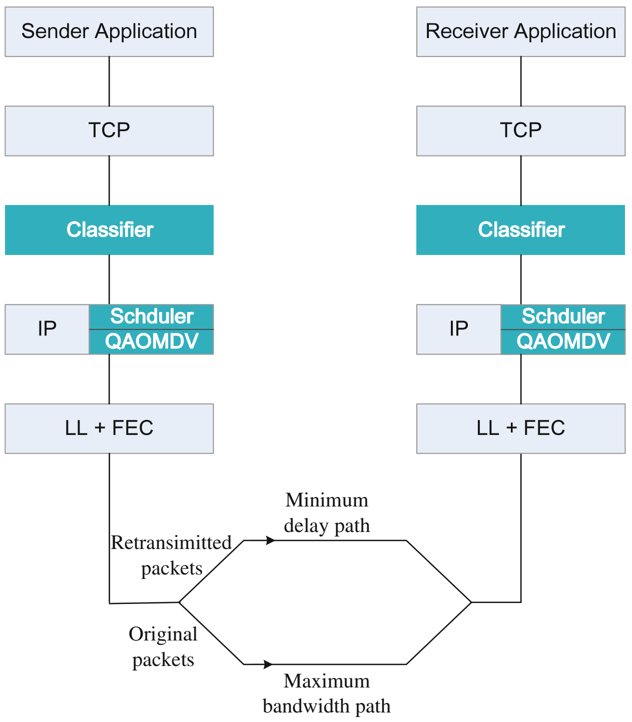

Fig. 7 Overview of the cross-layer implementation

5 System design and practical implementation

We now discuss the seamless integration between TCP and multipath routing for video streaming. Our implementation adopts a cross-layer design, as illustrated in Fig. 7. We add a classifier between the transport layer and the IP layer, which distinguishes the retransmitted data from the original packets. In the IP layer, we design QAOMDV, a QoS-aware multiple path routing protocol that extends the AOMDV protocol [20] to collaborate with the classifier. QAOMDV discovers and maintains the maximum-bandwidth path and the minimum-delay path concurrently, so as to deliver the original packets and the retransmitted packets, respectively. In the link layer, FEC is configured to mask wireless losses.

We now discuss the detailed implementation of the different modules.

5.1 Classifier and scheduler

The classifier characterizes the original packets and the retransmitted packets by checking the *sequence number* field in the TCP header of each incoming packet. Specifically, it reserves the newest TCP sequence number. When receiving a packet from the transport layer, the classifier gets TCP sequence number from the packet header and compares it with the newest sequence number it maintains. If it is an old sequence number, the classifier marks it as a retransmitted packet. Otherwise, it designates an *original* label to the packet, and the classifier updates its newest sequence number.

The scheduler receives marked packets from the classifier and schedules them according to MPR. It keeps two queues, retransmitted queue and original queue, for the two paths, respectively. The retransmitted queue has higher priority since missing packets look forward to be delivered as quickly as possible.

5.2 QoS-aware AOMDV (QAOMDV)

Before discussing QoS-awareness in multipath routing, we first review the basic AOMDV, a multipath extension to AODV. AOMDV computes multiple loop-free and link-disjoint paths with the employment of a customized flooding, and takes hopcount as its routing metric. A node recodes the maximum hopcount of the multiple paths for each destination, referred to as the *advertised hopcount* for that destination. The protocol only picks up alternative routes that have hopcount less than the *advertised hopcount*. Meanwhile, the *first hop* field in an AOMDV RREQ or RREP packet, which indicates the first hop taken by the packet, is used to ensure the disjunction of the multiple paths.

To incorporate QoS parameters in route discovery, we replace the header of each RREQ packet to $\langle min-$bandwidth, delay, AOMDV RREQ header$\rangle$. When an intermediate node receives an RREQ packet from the source, it first calculates its residual bandwidth and delay from the previous hop and then updates the *min-bandwidth* and *delay* fields in the RREQ packet. If the residual bandwidth is less than *min-bandwidth*, it replaces the *min-bandwidth* value. The update rule of *delay* field is similar to that of *hopcount* in the AOMDV header. When the destination receives a RREQ packet, it first updates the *min-bandwidth* and *delay* fields as other nodes. However, in a multi-hop wireless network, it cannot directly claim that the current network can offer sufficient bandwidth as indicated in the RREQ packet, as the nodes along the route would suffer from mutual interference during data transmission. To address this, following up the relationship between end-to-end throughput and hopcount [4, 18], we calculate the minimum bandwidth with Algorithm 1. That is, if the route is relatively long (hopcount >4), the available bandwidth is $\frac{1}{4}$ (an optimal parameter in pratice) of the raw channel bandwidth; otherwise, the available bandwidth is inversely proportional to hopcount.

Up to now, the reverse paths have been set up. Subsequently, the destination or the nodes that have routes to the destination send back RREP packets. Here, the header of a RREP packet is changed to $\langle$*bandwidth*, *delay*, AOMDV RREP header$\rangle$, which assists the establishment of forwarding paths. Figure 8 shows the structure of the routing table entries for AOMDV and QAOMDV, respectively. In QAOMDV, we add *bw* and *delay* fields in route_list. A route update is invoked whenever a node receives a route advertisement. The updating rule is similar as that of AOMDV. When a node receives an RREQ or RREP packet, it first checks whether the *sequence number* in the RREQ or RREP packet is new. If it is, the node will delete all existing paths to the destination or the source and insert a new path. Otherwise, it examines whether it is a better path, e.g., a path with higher bandwidth or less delay, and if it is disjoint with existing paths. If so, a new path will be inserted to the current route_list.

Algorithm 1 Minimum bandwidth calculation algorithm

```
Let $n$ be the *hopcount* in the RREQ packet
Let $bw$ be the *min-bandwidth*in the RREQ packet
**if** ($n \leq 4$)
    $bw = bw/n$
**else**
    $bw = bw/4$
**end if**
```

(a) AOMDV	**(b)** QAOMDV
destination	destination
Sequence number	Sequence number
route_list {(*nexthop1, hopcount1*) (*nexthop2, hopcount2*)}	route_list {(*nexthop1, bw1, delay1*) (*nexthop2, bw2, delay2*)}
expiration timeout	expiration timeout

Fig. 8 Structure of routing table entries for AOMDV and QAOMDV

5.3 Parameter estimation

A key concern in QAOMDV is the estimation of the available bandwidth and delay. We estimate the available bandwidth of a node through passively monitoring the channel. Given the 'busy' and 'idle' periods of a channel, the available bandwidth can be calculated as the channel capacity times the ratio of the idle time to the overall time [7]. We take the time period between two packets sent from the same node as the idle time of that node. The delay between two nodes is evaluated by calculating the time interval between the same packet sent from one node and received by the other node.

Since such values are needed for route discovery, during initialization of QAOMDV, we let each node broadcast probe packets. After establishing routes, QAOMDV will then use real data packets to estimate bandwidth and delay. The results are also periodically updated through an exponential weighted moving average (EWMA) [11].

A natural suspicion to MPR and QAOMDV is that the maximum-bandwidth path would simply be the minimum-delay path, too. We, however, find that this is not necessarily the case. The available bandwidth of a path is limited by the bottleneck bandwidth among all links on that path, while the end-to-end delay is an additive QoS parameter and has more correlation with hopcount. Besides bandwidth, one-hop delay depends on such other factors as MAC access delay, queuing delay, and packetization delay in video streaming. In our simulation, we observe that in most cases, the minimum-delay route is not the maximum-bandwidth path. Even if the two are the same, our solution will be rendered to a single-path transmission, which will not suffer from reordering and RTT underestimation.

6 Performance evaluation

In this section, we evaluate the TCP streaming performance with MPR. We also compare it with single-path transmission (AODV) and conventional multipath transmission (AOMDV), denoted as nMPRs and nMPRm, respectively. We are particularly interested in the following performance measures:

Throughput. The TCP throughput is the average number of packets all destinations receive during a time unit. It serves as a good indication of video streaming quality, as explained in Sect. 4.

Fairness index. The fairness index is to measure friendliness among the streaming sessions, and Jain's fairness index [12] has been widely adopted in the literature. All streaming sessions enjoy complete friendliness if Jain's fairness index equals 1, and have no friendliness if the fairness index equals 0.

We summarize our main observations as follows:

- MPR constantly outperforms nMPRs and nMPRm in terms of TCP throughput;
- MPR does not noticeably affect TCP friendliness, and its fairness index is similar to that without multipath transmission.

6.1 Simulation environment

Besides the same scenarios used in Sect. 3, we also simulated a much larger network with 100 nodes, distributed in a 10×10 grid topology. The distance between adjacent nodes is 200 m. Other configurations are the same as that of the 40-node random network. That is, each node has a radio propagation range of 250 m and its carrier-sensing distance is 550 m. The IEEE 802.11 MAC protocol was adopted in RTS/ CTS/Data/ACK mode with a channel data rate of 11 Mbps. The average video bitrate is 800 Kbps and the size of each packet is 1,000 bytes. We again performed 30 runs, with 10 streaming sessions between randomly chosen sources and destinations in each run.

6.1.1 Throughput results

In Fig. 9, we show the TCP throughput under different traffic and network configurations. The first three subfigures are for the 40-node random network. It can be seen that MPR generally outperforms nMPRs, and the improvement ranges from about 10–25%. Figure 9d compares the TCP throughputs under MPR, nMPRs and mMPRm in the 100-node grid network. It again confirms that MPR enhances the TCP throughput. Through analyzing the data traces, we find that three factors contribute to such improvement: (1) Path diversity, which inherently amplifies the path bandwidth through aggregation; (2) faster retransmission, which is achieved through selecting the low-delay path for retransmission; and (3) no extra packet reordering and timeout events are introduced.

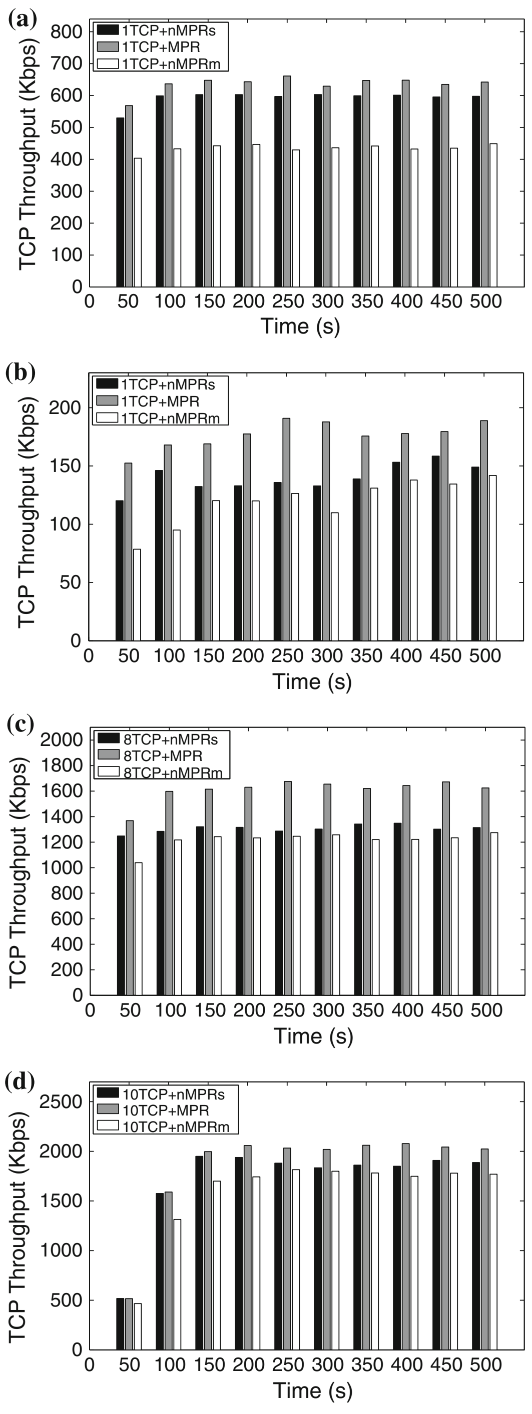

Fig. 9 Comparisons of TCP throughput. **a** TCP throughput (1 TCP session in a 40-node random network). The average TCP throughputs for nMPRs, MPR, and nMPRm are 592.8, 635.8, and 435.1 Kbps, respectively. **b** TCP throughput (1 TCP session and 7 UDP flows in a 40-node random network). The average TCP throughputs for nMPRs, MPR, and nMPRm are 139.9, 176.7, and 119.6Kbps, respectively. **c** TCP throughput (8 TCP sessions in a 40-node random network). The average TCP throughputs for nMPRs, MPR, and nMPRm are 1,306.4, 1,610.2, and 1,218.8 Kbps, respectively. **d** TCP throughput (10 TCP sessions in a 100-node grid network). The average TCP throughputs for nMPRs, MPR, and nMPRm are 1,720.2, 1,842.1, and 1,591.7 Kbps, respectively

Note that the throughput improvement in the 100-node grid network is not as much as that in the 40-node random network. We find that this is mainly due to the special grid layout of the network. In this layout, if the sender and the receiver are in the same line, it is probable that the paths for the original packets and the retransmitted packets in MPR are the same, which thus brings no benefits with MPR.

Also note that TCP throughput at 50s is noticeably lower than that in other time instances in Fig. 9. This is because TCP takes time to reach an equilibrium. In addition, the more flows compete in networks, the slower TCP will attain its stability, which can be observed by comparing the first three subfigures with the last one. The TCP throughput also fluctuates more often when UDP flows exist.

To better illustrate the TCP throughput fluctuation, we show the detailed TCP throughput over time in Fig. 10. We can see that the variability of TCP throughput with nMPRm is the worst. This is again due to the frequent spurious retransmissions caused by packet reordering and timeout. The more false congestion controls are triggered, the more frequent TCP throughput fluctuates. Our MPR however is much more stable.

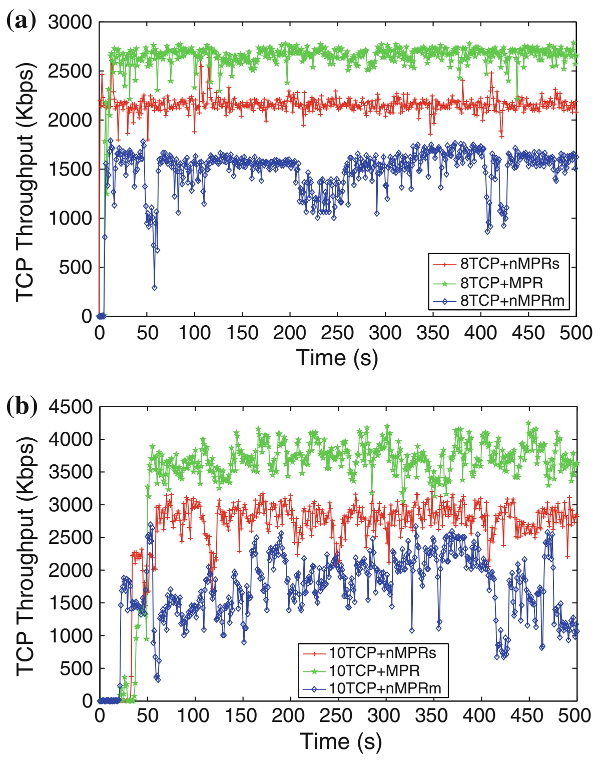

Fig. 10 Comparisons of TCP throughput fluctuation. **a** TCP throughput (8 TCP sessions in a 40-node random network in one run). **b** TCP throughput (10 TCP sessions in a 100-node grid network in one run)

6.1.2 *Fairness results*

We next show the fairness index in Fig. 11. We observe MPR has similar average fairness index as that of nMPRs, which implies that our solution will not affect TCP fairness under path diversity. On the other hand, nMPRm has lower fairness index, and we believe that it is due to the excessive reordering of packets and the timeout events. Consider an extreme scenario where the re-ordered packets are all from the same TCP flow; the throughput of this flow will be degraded significantly while the throughput of other flows are not affected at all, resulting in serious unfairness among TCP streaming sessions. On the other hand, MPR does not have such problem since every streaming session has retransmitted packets and the possibility of retransmission is equal across the streaming sessions.

The insight reason is that improper load splitting may result in large RTT differences among TCP flows and therefore brings more unfairness. TCP ensures fairness through its additive increase and multiplicative decrease (AIMD) congestion control, which is tightly coupled with RTT estimation. As a result, TCP throughput changes with RTT variations. The larger differences of RTTs exist among

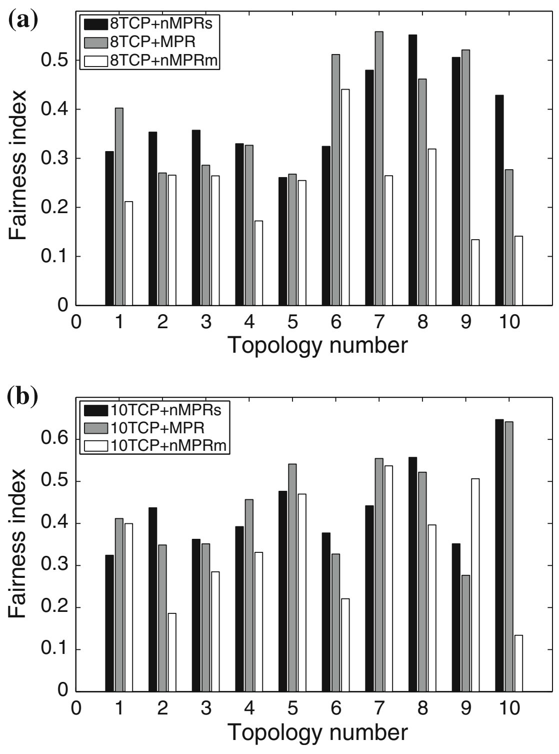

Fig. 11 Comparisons of TCP fairness index. **a** Fairness index (8 TCP sessions in a 40-node random network and only one run for each topology). The average fairness indexes for nMPRs, MPR and nMPRm are 0.391, 0.388, and 0.247, respectively. **b** Fairness index (10 TCP sessions in a 100-node grid network and only one run for each topology). The average fairness indexes for nMPRs, MPR and nMPRm are 0.437, 0.44, and 0.347, respectively

the flows, the less fairness TCP offers. The variations of fairness indexes in different topologies are also likely due to the RTT differences, because the average distance between sender and receiver pairs differs with topology.

7 Conclusion and future work

In this paper, we have presented MPR, a retransmission scheme employing QoS-aware multi-path routing for real-time video applications in multi-hop wireless networks. MPR maps the original packets and the retransmitted packets to different paths according to their distinctive QoS requirements. To cooperate with this scheme, we have designed QAOMDV routing protocol, which extends the conventional AOMDV by establishing distinct paths with the maximum bandwidth and the minimum delay. Our MPR does not trigger spurious retransmissions caused by reordering of packets and RTT underestimation. Therefore, different from conventional multipath transmission, it can enjoy the benefits from path diversity, and meanwhile it does not introduce additional bandwidth degradation and fluctuation. This has been validated by both theoretical analysis and simulation results.

MPR is easy to be implemented in terms of traffic splitting and can be extended to general networking systems. However, it may not achieve the maximum utilization of network resource as we just simply schedule the retransmitted packets to the minimum-delay path, with no considerations of the relationship between the packet loss rate and the available bandwidth on the minimum-delay path. Specially, if the packet loss rate is large while the available bandwidth is small, the retransmitted packets maybe arrive later as compared wtih those along the maximum-bandwidth path. Therefore, in the future, we plan to design dynamic assignment for the retransmitted packets and also the original packets and evaluate its effectiveness.

References

1. Amir, Y., Danilov, C., Kaplan, M.A., Musaloiu-Elefteri, R., Rivera, N.: On redundant multipath operating system support for wireless mesh networks. In: Proc. IEEE SECON (2008)
2. Apostolopoulos, J., Wong, T., Tan, W., Wee, S.: On multiple description streaming with content delivery networks. In: Proc. IEEE INFOCOM (2002)
3. Bohacek, S., P.Hespanha, Leeand, J., Lim, C., Obraczka, K.: A new TCP for persistent packet reordering. IEEE/ACM Trans. Netw. **14**(2), 369–382 (2006)
4. Chen, L., Heinzelman, W.B.: QoS-aware routing based on bandwidth estimation for mobile ad hoc networks. IEEE J. Selected Areas Commun. **23**(3), 561–572 (2005)
5. Cuetos, P., Ross, K.W.: Adaptive rate control for streaming stored fine-grained scalable video. In: Proc. ACM NOSSDAV (2002)
6. Ford, A., Raiciu, C., Handley, M.: TCP extensions for multipath operation with multiple addresses (2009). http://tools.ietf.org/html/draft-ford-mptcp-multiaddressed-02
7. Gupta, D., Wu, D., Mohapatra, P., Chuah, C.: Experimental comparison of bandwidth estimation tools for wireless mesh networks. In: Proc. IEEE INFOCOM (2009)
8. Han, H., Shakkottai, S., Hollot, C.V., Srikant, R., Towsley, D.: Multi-path TCP: a joint congestion control and routing scheme to exploit path diversity in the internet. IEEE/ACM Trans. Netw. **14**(6), 1260–1271 (2006)
9. Handley, M., Raiciu, C., Bagnulo, M.: Outgoing packet routing with MP-TCP (2009). http://tools.ietf.org/html/draft-handley-mptcp-routing-00
10. Hsiao, P., Kung, H.T., Tan, K.: Video over TCP with receiver-based delay control. In: Proc. ACM NOSSDAV (2001)
11. Hunter, J.: The exponentially weighted moving average (1986). Quality Technol.
12. Jain, R., Chiu, D., Hawe, W.: A quantitative measure of fairness and discrimination for resource allocation in shared systems. DEC Research Report TR-301 (1984)
13. Jurca, D., Frossard, P.: Packet selection and scheduling for multipath video streaming. IEEE Trans. Multimedia **9**(3), 629–641 (2006)
14. Kandula, S., Katabi, D., Sinha, S., Berger, A.: Dynamic load balancing without packet reordering. In: Proc. ACM SIGCOMM (2007)
15. Karn, P., Partridge, C.: Improving round-trip time estimates in reliable transport protocols. In: Proc. ACM SIGCOMM (1987)
16. Kim, T., Ammar, M.H.: Receiver buffer requirement for video streaming over TCP. In: Proc. IEEE ICNP (2006)
17. Leung, K., Li, V., Yang, D.: An overview of packet reordering in transmission control protocol (TCP): problems, solutions, and challenges. IEEE Trans. Multimedia **18**(4), 522–535 (2007)
18. Li, J., Blake, C., Morris, R.: Capacity of ad hoc wireless networks. In: Proc. ACM MOBICOM (2001)
19. Lim, H., Xu, K., Gerla, M.: TCP performance over multipath routing in mobile ad hoc networks. In: Proc. IEEE ICC (2003)
20. Marina, M.K., Das, S.R.: On-demand multipath distance vector routing in ad hoc networks. In: Proc. IEEE ICNP (2001)
21. McAuley, A.J., Manousakis, K., Kant, L.: Flexible QoS route selection with diverse objectives and constraints. In: Proc. IEEE IWQoS (2008)
22. van der Merwe, J., Sen, S., Kalmanek, C.: Streaming video traffic: characterization and network impact. In: Proc. the seventh international web content caching and distribution workshop (2002)
23. Nguyen, T., Cheung, S.: Multimedia streaming using multiple TCP connections. ACM Trans. Multimedia Comput. Commun. Appl. **4**(2), 1061–1076 (2008)
24. Nguyen, T.P., Avideh, Z.: Multiple sender distributed video streaming. IEEE Trans. Multimedia Special Issue Streaming Media **6**(2), 315–326 (2004)
25. Padhye, J., Firoiuy, V., Towsley, D., Kurose, J.: Modeling TCP throughput a simple model and its empirical validation.pdf. In: ACM SIGCOMM (1998)
26. Phatak, D.S., Goff, T.: A novel mechanism for data streaming across multiple IP links for improving throughput and reliability in mobile environments. In: Proc. IEEE INFOCOM (2002)
27. Radunovic, B., Gkantsidis, C., Gunawardena, D., Key, P.: Horizon: balancing TCP over multiple pahts in wireless mesh network. In: Proc. ACM MOBICOMM (2008)
28. Radunovic, B., Gkantsidis, C., Key, P., Rodriguez, P.: An optimization framework for opportunistic multipath routing in wireless mesh networks. In: Proc. IEEE INFOCOM (2008)

29. Liang, S., Cheriton, D.: TCP-RTM: using TCP for real time applications. In: Proc. IEEE ICNP (2002)
30. Sarolahti, P.: Multipath interface in the socket API (2009). http://tools.ietf.org/html/draft-sarolahti-mptcp-af-multipath-00
31. Scharf, M., Ford, A.: MPTCP application interface considerations (2009). http://tools.ietf.org/html/draft-scharf-mptcp-api-00
32. Sripanidkulchai, K., Maggs, B., Zhang, H.: An analysis of live streaming workloads on the internet. In: Proc. ACM SIGCOMM Conference on Internet Measurement (2004)
33. Stevens, W.: TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms (1997). http://ietf.org/rfc/rfc2001.txt
34. Wang, B., Kurose, J., Shenoy, P., Towsley, D.: Multimedia streaming via TCP: an analytic performance study. ACM Trans. Multimedia Comput. Commun. Appl. **4**(2) (2008)
35. Wang, B., Kurose, J., Shenoy, P., Towsley, D.: Multipath live streaming via tcp: scheme, performance and benefits. ACM Trans. Multimedia Comput. Commun. Appl. **5**(3) (2009)
36. Wei, W., Zakhor, A.: Path selection for multi-path streaming in wireless ad hoc networks. In: Proc. IEEE ICIP (2006)
37. Wu, W., Demar, P., Crawford, M.: Sorting reordered packets with interrupt coalescingr. Comput. Netw. **53**(15), 2646–2662 (2009)
38. Xu, D., Chen, Y., Xiong, Y., Qiao, C., He, X.: On finding disjoint paths in single and dual link cost networks. In: Proc. IEEE INFOCOM (2004)
39. Ye, Z., Krishnamurthy, S.V., Tripathi, S.K.: Effects of multipath routing on TCP performance in ad hoc networks. In: Proc. ICC (2003)
40. Zhibin, W., Ganu, S., Raychaudhuri, D.: IRMA: integrated routing and MAC scheduling in multi-hop wireless mesh networks. In: Proc. IEEE WiMesh (2006)
41. Zhang, M., Karp, B., Floyd, S., Peterson, L.: RR-TCP: a reordering-robust TCP with DSACK. In: Proc. IEEE ICNP (2003)
42. Zhu, C., Corson, M.S.: QoS routing for mobile ad hoc networks. In: Proc. IEEE INFOCOM (2002)