On the Impact of User Mobility on Peer-to-Peer Video Streaming

Igor M. Moraes[1], Miguel Elias M. Campista[1], Jairo L. Duarte[2],
Diego G. Passos[2], Luís Henrique M. K. Costa[1],
Marcelo G. Rubinstein[3], Célio Vinicius N. de Albuquerque[2],
and Otto Carlos M. B. Duarte[1]

[1]GTA/COPPE/POLI
Universidade Federal do Rio de Janeiro
P.O. Box 68504 - 21945-970
Rio de Janeiro, RJ, Brazil

[2]Instituto de Computação - IC
Universidade Federal Fluminense
R. Passo da Pátria, 156 - 24210-540
Niterói, RJ, Brazil

[3]PEL/DETEL/FEN
Universidade do Estado do Rio de Janeiro
R. São Fco. Xavier, 524 - 20550-013
Rio de Janeiro, RJ, Brazil

E-mails:{igor,miguel,luish,otto}@gta.ufrj.br,
{jduarte,dpassos,celio}@ic.uff.br,
rubi@uerj.br

Abstract

Wireless mesh networks (WMNs) are emerging as a promising solution for ubiquitous Internet access with mobility support. In such networks, user mobility may lead to Internet gateway changes and, consequently, impact the performance of continuous media applications. In this article, we investigate the impact of user mobility on the performance of peer-to-peer (P2P) video applications over wireless mesh networks. Peer-to-peer video streaming applications rely on the collaborative behavior of peers to assist the source in delivering multimedia content, reducing costs and increasing the scalability of video distribution. We identify practical issues related to mobility for P2P video streaming implementation in wireless mesh networks, such as addressing and forwarding strategies. Furthermore, we evaluate the performance of different P2P streaming applications as the user walks in our WMN testbed. Results indicate mobile users benefit from the frequent short-lived connections established in modern P2P video sessions.

1 Introduction

Wireless networks are a key technology to provide user mobility. Wireless *mesh* networks (WMNs) are self-organizing networks based on a backbone of stationary wireless routers, which cooperatively provide Internet access to mobile users. Moreover, WMNs are a low cost alternative to extend the Internet as compared to traditional cabled networks. Recently, a great number of research centers and universities [1] are devoting strong efforts to the deployment of wireless mesh networks, providing ubiquitous Internet access in campuses and cities.

Mobility is one of the main reasons for using wireless networks. As entertainment and news updates are part of people's daily lives, users aspire to have access to streaming audio and video applications while on the move in areas illuminated by a WMN. In such case, the application performance perceived by users is affected by variable network connectivity, caused by different factors such as handoffs between routers, varying levels of interference, or protocol reconfiguration. Therefore, it is of utmost importance to evaluate the impact of user mobility on the performance of different protocols and applications in real scenarios.

In this article, we focus on peer-to-peer (P2P) applications, which are a great success in the Internet and in community networks. In P2P, all nodes collaborate to provide the application functionalities. The decentralized P2P architecture increases scalability and provides resilience to node failures [2].

More recently, peer-to-peer techniques are being successfully used for live media streaming [3]. Applications such as SopCast allow a TV channel, radio station, or stored video to be publicly available in the Internet. In P2P video applications, all nodes can send or receive media simultaneously. This cooperative environment is completely different from typical TV and radio broadcasting systems, which are based on the client-server model.

Practical mobility aspects play an important role in analyzing the utility of P2P video streaming applications, especially when compared to traditional client-server applications. In this article, we investigate the behavior of popular P2P video streaming applications in a scenario where a person walks in an area covered by a wireless mesh network. Using a WMN testbed, we analyze the impact of mobility on the performance of different P2P applications, identifying some of the practical problems that arise.

This article is organized as follows. First, Section 2 points out the main characteristics of five of the most popular P2P video applications in the Internet. Then, Section 3 identifies practical aspects of handling user mobility in wireless mesh networks. Section 4 describes our experimentation scenario and discusses the performance measurements obtained. Finally, Section 5 draws the conclusions of the article and identifies open research issues.

2 Peer-to-peer Video Streaming

Video streaming in the Internet has become a popular service in recent years. Nevertheless, video streaming is still a costly service because of the high bandwidth requirements and of the huge number of potential users. Currently, *peer-to-peer* video streaming is emerging as a promising solution for video distribution in the Internet. In these systems, peers have to share their resources in order to assist the server in delivering the video. The key idea is the same of a P2P file-sharing system, but while file-sharing systems target elastic data transfers, streaming systems focus on the efficient delivery of multimedia content under strict bandwidth and timing requirements.

2.1 Data Delivery

Peer-to-peer video streaming applications are classified according to the type of distribution graph they build [2]. Tree-based overlays use tree distribution graphs, where the root is the video source. Each node joins the tree and then receives data from a parent node, the source, or a peer who has previously subscribed. The main advantages of the tree topology are low latency and low control overhead when peer changes are rare. On the other hand, when peer changes are frequent, control overhead is high because the tree needs to be reconstructed upon each change. Moreover, discontinuities in reception may happen, and therefore peers must buffer video packets during the tree reconstruction to avoid losing content. In mesh-based overlays, a mesh topology is used. Data is divided into chunks and each peer exchanges information about its own chunks with a subset of peers. A peer can download chunks from several peers concurrently, turning the mesh topology less sensitive to peer failures and departures. Nevertheless, control overhead tends to be higher because peers must exchange chunk maps to request content from other peers. Moreover, service performance depends on the buffer size of the nodes because video chunks that might be received out of order must be stored.

2.2 P2P Video Applications

We have chosen five of the currently most popular P2P video streaming applications, namely ESM, PPLive, SopCast, PPStream, and TVAnts[1]. ESM uses tree-based data delivery whereas the other applications employ a mesh topology.

ESM (End System Multicast) [4] was one of the first P2P video applications to appear. Actually, the original goal of ESM is to implement multicast functionalities in the end systems instead of the routing layer. A part of the system called Narada creates a mesh topology that interconnects the participating nodes according to estimation of Round-Trip Time (RTT) between nodes. From this topology, a distribution tree is created, where the root is the video source and branches are low latency links up to the leaves. ESM is one of the few P2P video applications that are open source.

PPLive, SopCast, PPStream, and TVAnts are proprietary, with no specification or source code available. They all use a mesh-based architecture and work similar to BitTorrent. A node subscribes to the system and receives addresses of a set of super-nodes, which are nodes responsible of keeping track of peers that already downloaded or are currently downloading a specific video chunk. When a node gets in contact with peers indicated by a super-node, the node receives a buffer map from each of them, containing the video chunks that the peer has. After receiving the maps, the node chooses a subset of peers to request the video chunks from.

Nowadays, PPLive is probably the most popular P2P video streaming application. The major difference between PPLive and BitTorrent is on the

---

[1] The applications can be downloaded from `http://esm.cs.cmu.edu`, `http://www.pplive.com/en`, `http://www.sopcast.org`, `http://www.ppstream.cn`, and `http://tvants.en.softonic.com`.

time requirements PPLive must guarantee. SopCast is also very popular, partially because of the services it provides to producers of content. In SopCast, anyone is allowed to register and broadcast a TV channel in the Internet. PPStream and TVAnts are also gaining popularity among users.

P2P streaming applications use different transport protocols and, depending on those protocols, they are more or less adapted to mobility. ESM uses TFRC (TCP Friendly Rate Control) as a transport protocol, which is rate-controlled UDP and achieves TCP-friendly throughputs [4]. Given that most P2P applications are not open source, traffic monitoring has been performed [2, 5] to identify the transport protocols used. Silverston and Fourmaux [5] classify transport protocols according to the type of traffic (video or signaling). PPLive uses TCP for video transmission and TCP and a few UDP datagrams for signaling. SopCast uses primarily UDP for most of its functions but also makes use of a few TCP connections. PPStream relies only on TCP, whereas TVAnts uses both TCP and UDP for video transmission and signaling. The main characteristics of the cited applications are summarized in Table 1.

One would expect that applications that strongly rely on the use of TCP would face problems in our mobile wireless mesh scenario. The connection-oriented nature of TCP, coupled with its congestion control mechanism turns its adaptation to wireless networks non-trivial [6]. However, the nature of P2P video sessions is based on a large number of short-lived TCP connections and we claim that this feature fits well mobile scenarios.

Table 1: Main characteristics of the analyzed P2P applications.

Application	**Data delivery structure**	**Transport protocol**	**Source code**	**Own channel broadcast**
SopCast	mesh-based	UDP/a few TCP	proprietary	yes
PPLive	mesh-based	TCP/a few UDP	proprietary	no
TVAnts	mesh-based	TCP/UDP	proprietary	no
PPStream	mesh-based	TCP	proprietary	no
ESM	tree-based	TRFC	open source	yes

3 Wireless Mesh Networks

Wireless mesh networks (WMNs) are a cost-effective alternative for ubiquitous network access given their user mobility support and low installation cost. Currently, WMNs are being widely deployed in cities, university campuses, commercial buildings, and underserved communities. WMNs are characterized by the presence of a backbone typically composed of stationary wireless routers that provide backhaul access to nodes not within range of gateways to the wired infrastructure, and interconnect isolated LANs. The backbone uses multihop communications to maintain network connectivity and to forward traffic to and from users.

In WMNs, users are potentially mobile and connect to the backbone via routers playing the role of access points. During handoff, network connectivity may be interrupted and performance of protocols degraded [7]. The handoff duration typically depends on the addressing scheme and the forwarding strategy employed by WMNs, as discussed in the following sections.

3.1 Addressing

Addressing is a challenging issue for mobile networks. The address modifications and the use of private IP addresses impact the protocols performance, possibly leading to connectivity losses and application disruptions.

3.1.1 Private IP vs. Public IP

From data collected in 2005, more than 50% of broadband users in the United Kingdom and in the United States are behind Network Address Translator (NAT) devices [8].

One problem related to the use of NAT is that computers accessing the Internet via NAT devices can make outbound connections to computers with public IPs, but typically cannot receive inbound connections. Furthermore, when a computer that is behind a NAT device moves to a new location that is also served by another NAT device, TCP connections may be broken because of the time required to reconfigure port mapping. Consequently, applications may suffer with packet losses and disruptions.

In peer-to-peer streaming applications, all participating nodes need to be bidirectionally reachable. As NAT often prevent peers from establishing connections with each other, the efficiency of video distribution is seriously impacted.

Currently, most peer-to-peer applications rely on manual port configuration to allow communication between peers behind NAT devices. Nevertheless, some techniques have been proposed to allow peer-to-peer TCP applications. Most of them are based on layering another address on top of the IP address. For example, the Simple Traversal of UDP through NATs

(STUN) protocol [9] and its TCP version (STUNT) [10] use the Session Initiation Protocol (SIP) URI to first communicate with a STUN/STUNT server. Then, the server reports back to the STUN/STUNT client what is the public IP address of the NAT device and what port is opened by the NAT device to allow incoming traffic. Although a number of techniques to traverse NAT devices exist, none of them is ideal in a mobility scenario and, as a consequence, the coexistence of NAT and mobility remains an open issue.

3.1.2 Dynamic IP vs. Fixed IP

Internet applications assume that the IP address of a given node does not change during long periods of time. This assumption, however, does not hold for mobile networks, because mobile users can change addresses as they move from one network to another. Thus, established connections do not survive the address change, because they rely on fixed source and destination IP addresses. If the mobile user uses a dynamic IP, the application may experience connectivity losses during the process of acquiring a new IP address. On the other hand, if the mobile user has a fixed IP address, his address must be kept through tunnels in order to support the active connections, using Mobile IP, for example. In the WMN scenario, an alternative to tunneling is to run an ad hoc routing protocol in each mobile user.

Wireless mesh networks typically use routing protocols to forward data. Each node maintains a routing table to correctly forward packets. If a user node, which is also running a routing protocol, moves across the backbone, the topology changes and, thus, the routing tables should be updated. If the mobile user has a fixed IP, the handoff is impacted only by the convergence time of the routing protocol. On the other hand, if the mobile user uses dynamic IP, the time to acquire a new IP address or to establish a tunnel should be added to the convergence time.

4 Video Streaming in WMNs

In this section, we present the results of our experiments. First, we evaluate the performance of a client-server application under client mobility. Then, we analyze the behavior of different P2P applications in the same scenario. Our measurements were collected in an indoor testbed at the GTA laboratory of the Federal University of Rio de Janeiro, Brazil. The testbed is composed of two desktops, one laptop, and five wireless routers. The desktops are equipped with IEEE 802.11b/g PCI wireless interfaces with external omni-directional antennas and Fast Ethernet cards. The laptop is equipped with an IEEE 802.11b/g PCMCIA wireless card. The two desktops play the role of gateways providing Internet access to the mobile user. Both desktops run Debian Linux OS and the laptop runs Windows XP because most P2P applications are not implemented in Linux. For the client-server experiments, another Windows Vista desktop, connected to the gateways via a wired network, is used as a server. The wireless routers are LinkSys WRT54G running Linux OpenWRT. These wireless routers and the two gateways form the WMN backbone.

In our experiments, we launch a packet sniffer on the laptop and on the desktops. Afterwards, we start the P2P application and select a TV channel or start the client-server application. To evaluate the impact of user mobility, we limit our analysis to the events that occur inside the wireless mesh network, by comparing the traffic that arrives at the gateways with that of the laptop. As soon as the application buffer gets full, this warmup phase is ended and a person carrying the laptop starts walking from point A to point D and then comes back to A (Figure 1). This mobility scenario aims at reproducing a user walking through different gateways. We expect with this mobility scenario that the user will change of Internet gateway at least twice, from GW_1 to GW_2 and then again to GW_1 on his way back. Upon returning to point A, we quit the application, entering the termination phase, and stop all packet sniffers.

All testbed nodes use IEEE 802.11g in ad-hoc mode and run the OLSR (Optimized Link-State Routing) protocol with the ETX (Expected Transmission Count) routing metric. The gateways announce default routes using HNA (Host and Network Association) messages implemented in OLSR. Therefore, the laptop chooses the gateway according to the path computed by the routing protocol. In our testbed, the laptop uses a fixed private IP address. Thus, the gateways use NAT to allow users to access the Internet. We have fixed the IP address of the laptop to avoid DHCP delays.

The video server used in the client-server experiments is a PC running the VLC application configured with HTTP. The mobile user also runs VLC. The video in the server was previously recorded from a SopCast channel to guarantee that the videos used in the experiment have similar rates.

We use four P2P video streaming applications in our tests: SopCast (version 2.0.4), PPStream (version 2.2.25), PPLive (version 1.9.21), and TVAnts (version 1.0.0.59). We do not include the results of ESM because it was not able to connect to any channel at the time we performed the measurements. We perform five measurements with each application using popular channels: TV Macumb@ for SopCast, Hunan TV for PPStream and PPLive, and CCTV-1 for TVAnts. The video streams have rates between 350 and 400 kbps.

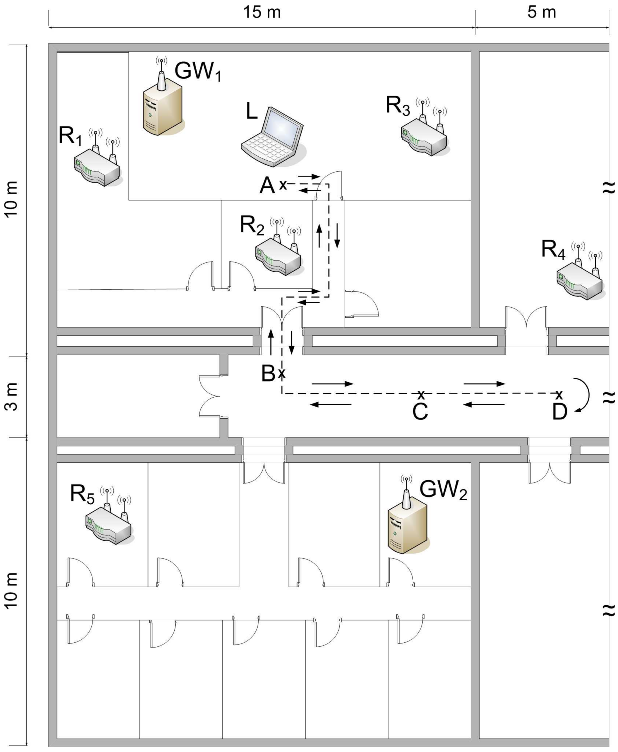


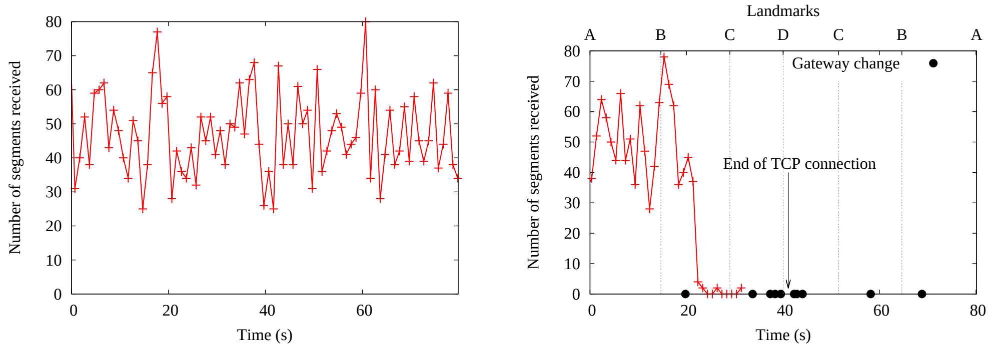
Figure 1: Trajectory of the experiment.

In addition to network performance metrics, we conduct a subjective analysis to verify the quality of the video reception on the mobile client. First, we record the video stream received by the mobile client, which is called user sample. We also record the video stream received by a user placed at the same wired network of the two gateways, called reference sample. The user and the reference samples are shown to an audience of 30 spectators. Each spectator gives a rate from 1 to 5, where 5 is the best, according to the perceived quality of the video.

As previously mentioned, according to the transport protocol used, the applications behave differently in the presence of client mobility and NAT. In the following, we focus on the measurements obtained with SopCast and PPStream, which are representative of applications based on UDP and TCP, respectively. Results obtained with all applications are summarized in Table 2 of Section 4.4. We analyze both download and upload traffics.

4.1 Client-Server Application

Figure 2 shows the performance of the client-server application when the client is stopped or moving. Points A to D in Figure 2(b) show the instants the client crosses the trajectory landmarks of Figure 1. The black dots down on the X-axis mark the instants where the gateway changes. In both scenarios, the client only establishes one TCP connection with the server. When the client is static, this connection is active during all the experiment (Figure 2(a)). On the other hand, when the client is mobile, the number of TCP segments received rapidly decreases just after the first gateway change. After a few seconds the TCP connection ends, as indicated in Figure 2(b), stopping the video reception. For this case, the audience gives a score equal to 3.68 (with standard deviation $\sigma = 0.75$) for the reference sample and equal to 1.35 ($\sigma = 0.55$) for the user sample, showing that the client-server application is not well suited for mobile environments. By relying on just one connection, the application disrupts because the server resets the connection. On the other hand, a P2P application that continuously establishes new short-lived connections and maintains multiple active connections is more suitable for mobile environments, as the performance analysis in the next sections demonstrates.



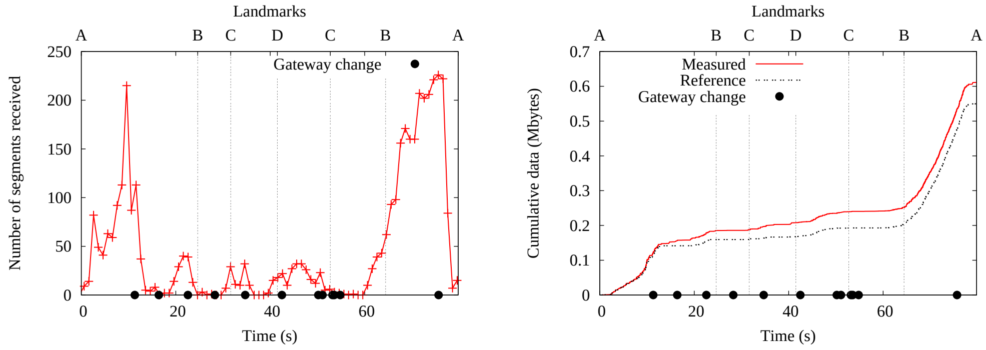
(a) Client is stopped.

(b) Client is moving.

Figure 2: Network measurements with client-server application.

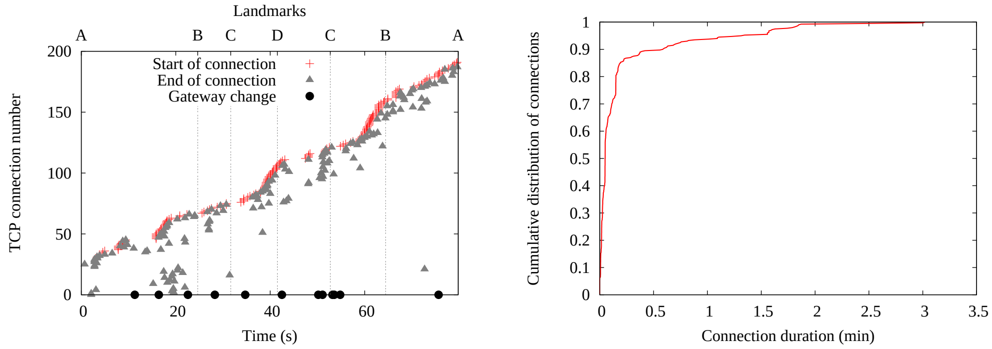
4.2 P2P TCP-based Application: PPStream

Figure 3 presents the results of one experiment run with PPStream, an application that only uses TCP as transport protocol. All figures except Figure 3(d) show the performance of the application during the movement from point A to D and then back to point A. As expected, PPStream is the application most sensitive to gateway changes. Figure 3(a) plots the number of TCP segments received per second during the experiment. We can see that there is a period during which the routing protocol is adapting to the client's movement. As routes are recomputed, there are more than two gateway changes, even though the client keeps walking in a scenario with only two gateways. We can observe that the number of received TCP segments sharply decreases, reaching zero sometimes. Figure 3(b) plots the cumulative number of TCP bytes received by the client as it moves across the WMN gateways. The *reference* curve plots the cumulative number of bytes transmitted by the gateways without taking into account retransmissions, whereas the *measured* curve shows the cumulative number of bytes received by the client including retransmissions. When the number of transmission failures in the wireless network is higher the curves are farther from each other. The plateaus occur when the TCP connections are stale. Figure 3(c) plots the beginning (crosses) and the end (triangles) of each TCP connection. As the warmup and the termination phases are omitted in this figure, some connections are already opened before the start of movement and others are not closed before the end of the movement. The fact that most of the TCP connections are short lived contributes to the robustness of P2P applications, where peers may enter or leave the network at any time. These short-lived connections are better illustrated in Figure 3(d), which plots the cumulative distribution function of TCP connection durations including warmup and termination phases. P2P streaming applications also benefit from short-lived connections to handle user mobility. Because the data stored in the application buffer comes from numerous connections, and only a few of them are active and therefore, affected when the gateway changes, the video streaming is not severely impaired. The audience, in this case, rates the reference sample with 4.08 ($\sigma = 0.68$) and the user sample with 2.45 ($\sigma = 0.77$). It is also worthy noting that during gateway changes, the rate of new connection openings decreases. Figure 3(e) plots the inter-arrival time of received TCP segments. We observe that during gateway changes, the number of received segments decreases, as the time between successive receptions grows. In the case of TCP, its congestion control also contributes to reduce the source transmission rate. Because peers also act as servers, we also analyze upload traffic traces. Figure 3(f) shows that TCP traffic also suffers with transmission failures in the upload direction.



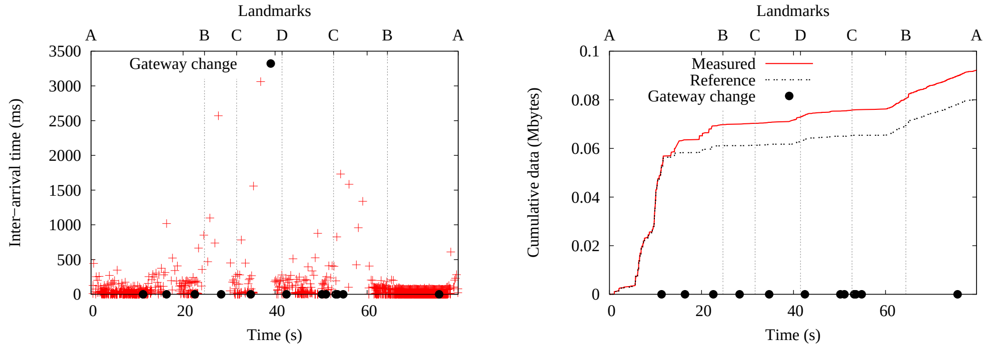
(a) TCP segments received.

(b) Cumulative TCP bytes received.



(c) TCP connections duration.

(d) CDF of TCP connections duration.



(e) TCP segments inter-arrival time.

(f) Cumulative TCP bytes sent.

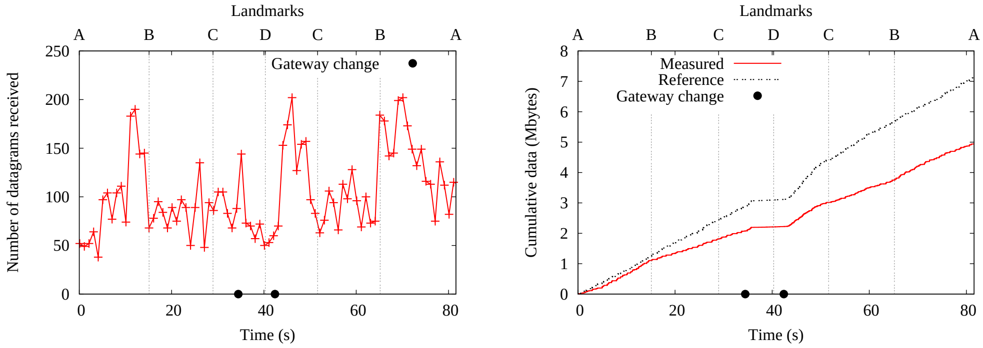
Figure 3: Network measurements with PPStream.

4.3 P2P UDP-based Application: SopCast

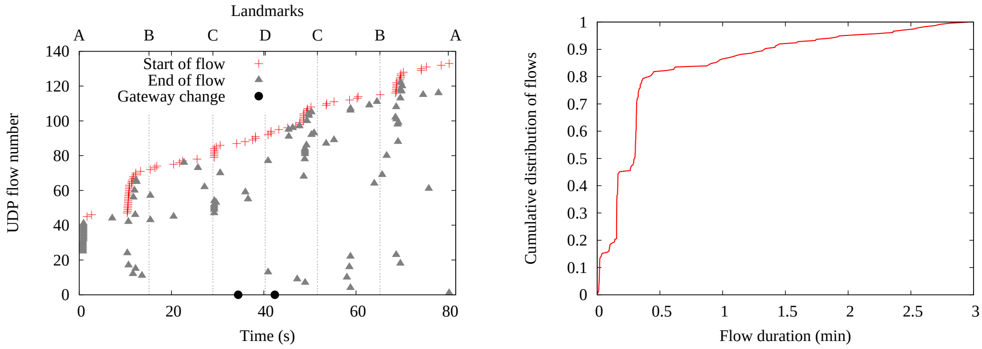
Figure 4 presents the results of one experiment run with SopCast, which mostly uses UDP. Typically, UDP-based applications have shown fewer losses because of gateway changes compared to TCP-based applications. Even though there are losses when the gateway changes, the received data do not reach zero as seen in the experiment plotted in Figure 4(a). With SopCast, the video did not freeze as opposed to the TCP-based application (PPStream). The audience gives a score equal to 3.77 ($\sigma = 0.62$) for the reference sample and equal to 3.24 ($\sigma = 0.62$) for the user sample. This behavior is also reflected in the cumulative plot of UDP datagrams presented in Figure 4(b). In this figure, the reference curve plots the cumulative number of bytes sent by both gateways and the measured curve plots the cumulative number of bytes received by the client. Note that these curves do not present long plateaus as Figure 3(b). Figure 4(c) plots the beginning and the end of different UDP flows received by the mobile client. Figure 4(d) presents the cumulative distribution function of UDP flow durations. As for TCP connections in PPStream, most of the UDP flows are short-lived, which means that less flows suffer with gateway changes. Another important remark is that the inter-arrival time between UDP datagrams is less sensitive to gateway changes than in the case of TCP-based applications (Figures 3(e) and 4(e)). Differently from TCP-based applications, gateway changes do not result in upload traffic losses in applications that mostly use UDP as transport protocol, as seen in Figure 4(f). This is because, regardless of using NAT or not, the IP address of the destination peer remains the same even when the user changes the associated gateway. Therefore, once the datagrams sent by the mobile user are received by at least one gateway, the delivery rate is not affected. In download direction, otherwise, Internet peers have to change the destination IP address to the new associated gateway as the user moves. In this case, the destination IP address is equal to the address of the associated gateway using NAT. This operation can take sometime, often resulting in transmission failures, independently of the transport protocol. As TCP traffic is bidirectional, the traffic is affected by user mobility even when data is transmitted in upload direction. Hence, the combination of gateway changes, NAT, and TCP always results in transmission failures when segments are sent to the previous associated gateway.

4.4 Summary of the Experiments

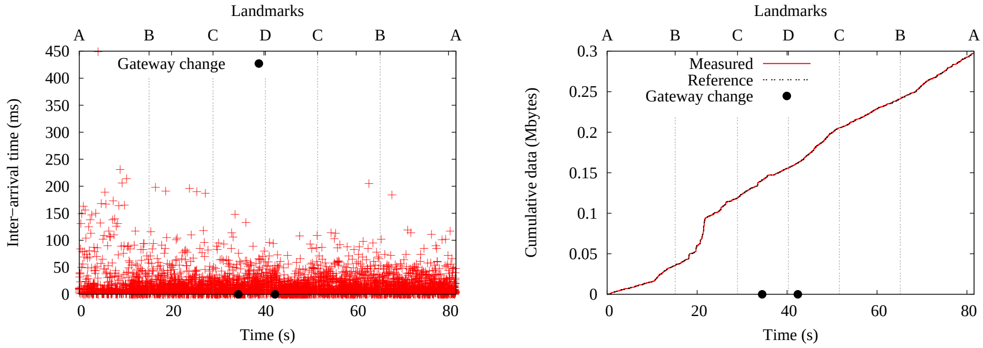
Table 2 summarizes the measurements obtained with SopCast, PPStream, PPLive, and TVAnts. Each measurement was repeated five times for each application. Table 2 shows the average and the standard deviation of some measurements of interest. In our experiments, the number of UDP sources of SopCast is higher than the number of TCP sources of PPStream. Nevertheless, UDP sources in PPStream were not observed and TCP sources in SopCast are often observed at the beginning of each experiment. The UDP delivery ratio is defined as the number of UDP datagrams transmitted by the



(a) UDP datagrams received. (b) Cumulative UDP bytes received.



(c) UDP flows duration. (d) CDF of UDP flows duration.



(e) UDP datagrams inter-arrival time. (f) Cumulative UDP bytes sent.

Figure 4: Network measurements with SopCast.

gateways divided by the number of UDP datagrams received by the mobile node. With TCP sources, computing the delivery ratio is not straightforward because a segment is retransmitted until it is delivered, or a timeout occurs. Therefore, a segment loss would only happen after multiple retransmissions. In addition, retransmissions can also be caused by lost acknowledgments.

As in our scenario we would like to measure transmission failures, we define the TCP retransmission ratio as the number of first attempts to send a segment divided by all attempts, including possible retransmissions. Note that the delivery ratio of UDP datagrams in download direction is the most affected because sources do not use congestion control algorithms as in the TCP case. On the other hand, in the upload direction the delivery ratio is high, confirming the results shown in Sections 4.2 and 4.3. TCP traffic presents transmission failures in both download and upload directions because it is bidirectional and the client is behind a NAT device. The number of gateway changes is similar for all applications because it depends on the routing protocol, which is the same in all experiments. Moreover, the rate of new connections or flows per second contributes to the robustness of the applications in mobile scenarios.

Table 2: Experiment summary.

	Applications			
	SopCast	**PPLive**	**TVAnts**	**PPStream**
Experiment duration (s)	80.80 (σ=1.56)	79.62 (σ=0.52)	78.25 (σ=3.75)	78.27 (σ=5.02)
Number of source peers (UDP/TCP)	110.40 (σ=15.07)/ 1.00 (σ=0.63)	104.25 (σ=4.81)/ 60.75 (σ=6.53)	32.20 (σ=2.40)/ 13.40 (σ=1.62)	-/48.4 (σ=3.20)
UDP deliv. ratio (%) (download/ upload)	79.53 (σ=16.44)/ 99.99 (σ=0.01)	66.18 (σ=2.87)/ 99.80 (σ=0.33)	66.46 (σ=1.59)/ 98.82 (σ=0.97)	-/-
TCP retrans. ratio (%) (download/ upload)	62.50 (σ=41.46)/ 100.00 (σ=0.00)	93.92 (σ=2.07)/ 97.31 (σ=1.23)	92.48 (σ=4.40)/ 95.63 (σ=2.03)	97.47 (σ=0.70)/ 95.73 (σ=2.70)
Number of gateway changes	6.40 (σ=2.58)	10.00 (σ=2.06)	8.00 (σ=1.67)	8.00 (σ=3.58)
Rate of new connec./ flows	0.03 (σ=0.01)/ 0.84 (σ=0.02)	1.90 (σ=0.01)/ 1.07 (σ=0.01)	0.51 (σ=0.01)/ 0.31 (σ=0.01)	1.44 (σ=0.05)/-

5 Conclusion

In this article, we have investigated the feasibility of peer-to-peer (P2P) video streaming applications running on wireless mesh networks. P2P streaming is already a huge success in the Internet, thanks to its decentralized nature and larger scalability when compared to server-based video distribution. Moreover, wireless mesh networks (WMNs) are an enabling technology for ubiquitous access to the Internet. Nevertheless, as we have discussed in this paper, there are different open issues on dealing with practical aspects of mobile P2P streaming using WMNs.

The first part of this paper outlines practical issues related to addressing and to the forwarding strategy used. Then, we describe our experiments with four of the most popular P2P video streaming applications, namely, SopCast, PPLive, PPStream, and TVAnts. We collected traffic traces from those applications running in a WMN testbed in our laboratory. The analysis of those traces has shown that UDP-based applications allow a smooth adaptation to the WMN environment. Nevertheless, applications that use TCP connections can also be feasible especially if using numerous short-living connections. The use of short-living connections is a characteristic of P2P streaming applications, where peers receive media content from multiple sources. Thus, the P2P distributed architecture can contribute to video streaming, independent of the transport protocol.

Acknowledgment

The authors would like to thank CNPq, CAPES, Faperj, and FUJB for partially funding this research.

References

[1] M. E. M. Campista, D. G. Passos, P. M. Esposito, I. M. Moraes, C. V. N. de Albuquerque, D. C. M. Saade, M. G. Rubinstein, L. H. M. K. Costa, and O. C. M. B. Duarte, “Routing metrics and protocols for wireless mesh networks,” *IEEE Network*, vol. 22, no. 1, pp. 6–12, Jan. 2008.

[2] A. Sentinelli, G. Marfia, M. Gerla, L. Kleinrock, and S. Tewari, “Will IPTV ride the peer-to-peer stream?” *IEEE Communications Magazine*, vol. 45, no. 6, June 2007.

[3] D. A. G. Manzato and N. L. S. da Fonseca, “An incentive mechanism for peer-to-peer networks with live streaming,” in *IEEE GLOBECOM*, Nov. 2006.

[4] Y.-H. Chu, S. G. Rao, S. Seshan, and H. Zhang, “A case for end system multicast,” *IEEE Journal of Selected Areas in Communications*, vol. 20, no. 8, pp. 1456–1471, 2002.

[5] T. Silverston and O. Fourmaux, “Measuring P2P IPTV systems,” in *International workshop on Network and Operating Systems Support for Digital Audio & Video (NOSSDAV)*, June 2007.

[6] A. A. Hanball, E. Altman, and P. Nain, “A survey of TCP over ad hoc networks,” *IEEE Communications Surveys & Tutorials*, vol. 7, no. 3, pp. 22–36, 2005.

[7] Y. Amir, C. Danilov, M. Hilsdale, R. Musaloiu-Elefteri, and N. Rivera, “Fast handoff for seamless wireless mesh networks,” in *ACM Mobisys*, June 2006, pp. 83–95.

[8] P. Rodriguez, S.-M. Tan, and C. Gkantsidis, “On the feasibility of commercial, legal P2P content distribution,” *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 1, pp. 75–78, Jan. 2006.

[9] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy, “STUN - simple traversal of user datagram protocol (UDP) through network address translators (NATs),” IETF Network Working Group RFC 3489, Mar. 2003.

[10] S. Guha, Y. Takeda, and P. Francis, “NUTSS: A SIP-based approach to UDP and TCP network connectivity,” in *ACM SIGCOMM*, Aug. 2004.