YouTube Everywhere: Impact of Device and Infrastructure Synergies on User Experience

Alessandro Finamore
Politecnico di Torino
finamore@tlc.polito.it

Marco Mellia
Politecnico di Torino
mellia@tlc.polito.it

Maurizio M. Munafò
Politecnico di Torino
munafo@tlc.polito.it

Ruben Torres
Purdue University
rtorresg@purdue.edu

Sanjay G. Rao
Purdue University
sanjay@purdue.edu

ABSTRACT

In this paper we present a complete measurement study that compares YouTube traffic generated by mobile devices (smart-phones, tablets) with traffic generated by common PCs (desktops, notebooks, netbooks). We investigate the users' behavior and correlate it with the system performance. Our measurements are performed using unique data sets which are collected from vantage points in nation-wide ISPs and University campuses from two countries in Europe and the U.S.

Our results show that the user access patterns are similar across a wide range of user locations, access technologies and user devices. Users stick with default player configurations, e.g., not changing video resolution or rarely enabling full screen playback. Furthermore it is very common that users abort video playback, with 60% of videos watched for no more than 20% of their duration.

We show that the YouTube system is highly optimized for PC access and leverages aggressive buffering policies to guarantee excellent video playback. This however causes 25%-39% of data to be unnecessarily transferred, since users abort the playback very early. This waste of data transferred is even higher when mobile devices are considered. The limited storage offered by those devices makes the video download more complicated and overall less efficient, so that clients typically download more data than the actual video size. Overall, this result calls for better system optimization for both, PC and mobile accesses.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Miscellaneous; C.4 [**Performance of systems**]: Measurement techniques

General Terms

Measurement, Performance

IMC'11, November 2–4, 2011, Berlin, Germany.

Keywords

Internet Measurement, YouTube, Video Streaming, Mobile Performance, Quality of Experience.

1. INTRODUCTION

Created in 2005 and bought by Google in November 2006, YouTube is the most popular and bandwidth intensive service of today's Internet: it accounts for 20-35% of the Internet traffic [6, 9, 8] with 35 hours of videos uploaded every minute and more than 700 billion playbacks in 2010 [15, 16]. With such a high popularity, it presents a challenge both for the system itself and for the Internet Service Providers (ISP) that need to offer good quality of service for video streaming demands. Therefore the YouTube phenomenon attracted the interest of the research community, with several works [7, 17, 3, 4, 1, 11, 13] focusing on either video characterization, infrastructure, or user behavior.

A second recent change in the way people access the Internet is due to the exploding popularity of mobile devices. Smartphones and Internet tablets are today commonly used both at home and at public places, and the phenomenon is still growing in popularity. Recent estimates forecast that within a few years mobile devices will be the users' preferred choice for accessing the Internet [10] while according to [9, 12] multimedia content represents already a big share of the mobile traffic, with YouTube as the main contributor. Still, mobile operators are struggling with the intrinsically limited capacity of mobile access technologies.

The mix of the two phenomena has serious implications for both content providers and ISPs. Indeed, while YouTube is already commonly accessible on mobile devices from 3G/4G networks, the video encoding rate (and quality) is, by design, much more limited than the one offered to PCs. At the same time, mobile ISPs adopt tariff plans with the explicit goal to limit the amount of traffic a device can consume, a trend that is becoming popular among wired ISPs as well.

In this paper, we focus on the differences and similarities of YouTube usage when accessed from a regular "PC player" or from a "mobile player". The first category includes accesses performed from a regular web browser equipped with the Adobe Flash plugin on a standard PC, e.g., a desktop, a notebook, or a netbook. The second category includes accesses performed through the special mobile version of the YouTube portal, or through the custom application found on devices running Apple iOS, Google Android, or other smartphone operating systems. By dissecting the YouTube traffic observed in operational networks, we explore the impact of devices and corresponding infrastructure synergies on the user experience and the network. Our data sets span over three different countries, including customers in both the United States and Europe, both campus and residential networks, with very different access technologies, i.e., high speed campus LANs, WiFi hot-spots and home Access Points, ADSL and Fiber-To-The-Home links. In each vantage point, we monitored all YouTube traffic for a one-week long period of time.

Our paper exposes the fact that YouTube is highly optimized to deliver video to PC users and considerably more inefficient in serving requests from mobile devices.

To the best of our knowledge, this is the first work to present an in-depth analysis of the YouTube system accessed from mobile devices. Moreover, performing a systematic comparison respect to common PCs, and taking advantage of our unique and heterogeneous data sets, we derive a comprehensive set of results. Our key findings are:

Users access content in the same manner independently of the device used:

- i) content downloaded has the same characteristics both in size and duration in all vantage points;
- ii) the type of device does not modify the way people watch YouTube videos. Most of the watched videos are interrupted within the first 40 seconds, with only 10% of them lasting more than 50% of the actual video duration;
- iii) people stick with default parameters, with negligible voluntary change of video resolution, and marginal fraction of views in full screen mode.

YouTube adopts different mechanisms according to the device used: the download is highly optimized for PC based players, while mobile players show consistently worse performance. This is related to the limited capabilities of the mobile devices but results show that the mechanisms used by YouTube to serve this type of devices cause i) higher access time, ii) lower download rate, iii) more bursty traffic which impair the quality achieved by mobile players.

The amount of traffic downloaded by clients but not used by the player is large:

- i) due to aggressive buffering policies, 25-39% of downloaded traffic is useless when PCs are used; this fraction grows to 35-48% when mobile devices are considered;
- ii) due to a possibly unoptimized implementation in the mobile players, the amount of data transferred exceeds the actual video size in 15-25% of the cases.

The last two findings are striking, and call for a possible system improvement, especially for mobile players. This is of crucial importance considering the growing popularity of mobile devices. Although our data set does not include 3G/4G mobile access clients, we believe this is even more critical for those operators for which the lack of bandwidth is and will be always a bottleneck.

The remaining of the paper is organized as follows: Sec. 2 provides a high level description of the YouTube protocols; the data sets are described in Sec. 3; details on the YouTube traffic characteristics are shown in Sec. 4; Sec. 5 and Sec. 6 detail the user behavior and performance, respectively; related work is discussed in Sec. 7, and a summary of our findings in Sec. 8.

2. YOUTUBE PRIMER

In this section we provide a high level description of the protocol used to retrieve the video content. In general, two phases can be distinguished: 1) content look-up, 2) content download and playback. The first phase is typically performed using a web browser or a custom application running on the local client and querying regular websites (e.g., youtube.com, or a website that embeds a YouTube video). The second phase starts after the user selects the video of interest. This involves resolving the video server hostname and subsequently downloading the video stream from the server. YouTube employs DNS-based mechanisms to direct clients to a server in a data center close to the user. However, in some cases, HTTP-based signaling can be exploited to further redirect the user to other video servers (e.g., in the case of server congestion, or content unavailability) [13]. In this paper, we focus our analysis on the second phase.

YouTube can be accessed from a wide range of devices, each with different capabilities and hardware constraints. Depending on the client device, two mechanisms are used to retrieve the video content:

- PC-player: the client is a regular PC running either a web browser with the Adobe Flash plugin or HTML5 compliant browser[1]. We performed experiments with several browsers and operating systems and found no difference in the traffic they exchange during the second phase. Hence, we will refer to them as PC-player without further distinction[2].
- Mobile-player: the client is a smartphone, an Internet tablet or a set-top-box which uses a custom application[3]. Also in this case we tested different combination of devices running both Apple iOS, Google Android and other operating systems. While several differences are found when considering the first phase, they all behave similarly in the second phase. Therefore, we will refer to them as Mobile-player.

Fig. 1 sketches the temporal evolution of the HTTP messages exchanged between the client and the YouTube servers. The top plot refers to the PC-player while the bottom plot refers to the Mobile-player. Clients exchange messages either with a *front-end server*, *web content servers* or *video servers*. The front-end handles the access to the YouTube portal (www.youtube.com and m.youtube.com), the web content servers provide thumbnails or other content while the video servers are in charge of the streaming. In the HTTP requests the downloaded video is identified by its *videoID* - a unique 11 characters long identifier. In the following we detail the HTTP messages that will be considered in our analysis to study the evolution of a download.

2.1 PC-player

Let us consider a client accessing the www.youtube.com web site from a regular PC using a browser as shown in Fig. 1 (top). We can split the interaction between the browser and the YouTube servers in three steps: (1) video web page retrieval, (2) video prefetch and (3) video download.

During (1), the client downloads the web page describing the video. The HTML document contains a combination of text and other "objects" (e.g., the Adobe Flash player)

[1] http://www.youtube.com/html5

[2] Notebooks and netbooks using regular browsers belong to the PC-player category.

[3] Even if set-top-boxes and TV appliances are hardly mobile, they use the same access mechanism as smartphones, so we consider them in the Mobile-player category.

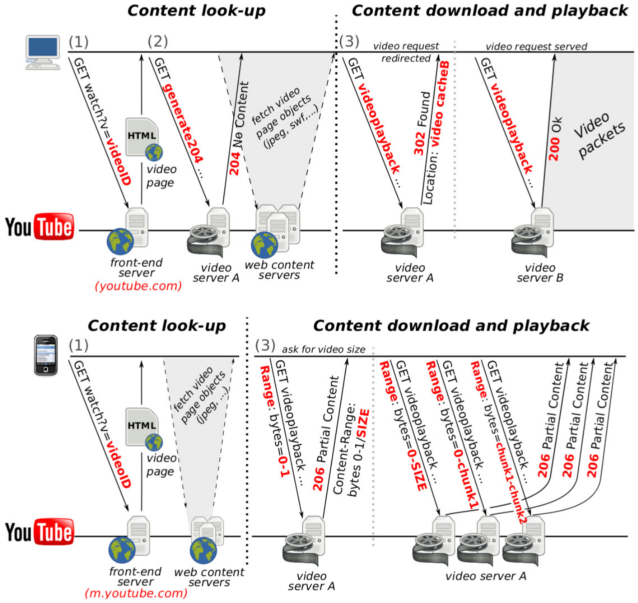

Figure 1: YouTube video download mechanisms. Example of possible evolution when accessing to *youtube.com* from a PC (top) and *m.youtube.com* from a smartphone (bottom).

that the browser needs to fetch to properly display the page. Among the different objects, a Javascript function triggers a `generate204` request sent to the video server that is supposed to serve the video. This starts the video prefetch (2), which has two main goals: first, it forces the client to perform the DNS resolution of the video server hostname. Second, it forces the client to open a TCP connection toward the video server. Both help to speed-up the video download phase. In addition, the `generate204` request has exactly the same format and options of the real video download request, so that the video server is eventually warned that a client will possibly download that video very soon. Note that the video server replies with a `'204 No Content'` response, as implied by the command name, and no video content is downloaded so far.

At this point, the browser handles the control to the Flash player which will manage the actual video download (3). The player sends a HTTP `videoplayback` request to get the video. Note that the same TCP connection previously opened during (2) can be used if HTTP persistent capability is supported between the browser and the Flash plugin. Because of server congestion or lack of content, the server can *redirect* the client to other servers [13]. In this case, the video server replies with a HTTP `'302 Found'` response which specifies the hostname of another video server to contact. The player then resolves the hostname, and sends a new `videoplayback` request. This process can repeat until a valid video server is found. The final video server of the chain replies with the usual HTTP `'200 OK'` response, which initiates the stream of video data to the client.

We highlight that the `generate204` request is a specific optimization that is found only when accessing a video through www.youtube.com. YouTube videos embedded in regular HTML pages do not exploit this, so the player have to resolve the video server hostname yet before sending the first `videoplayback` request.

2.2 Mobile-player

Mobile devices use a different protocol as shown in bottom part of Fig. 1. First, no prefetch message is sent in the content look-up phase. Second, differently from the PC-player case, the video content is downloaded in "chunks", each one requested in a separate TCP connection, using the HTTP `Range` header field to specify the requested portion of the video. The video server then replies with a `'206 Partial Content'` response.

This mechanism is possibly the result of a design choice that tries to cope with the tighter constraints in terms of storage availability for mobile devices. In fact, the mobile devices can hardly buffer the entire video so the player progressively requests portions according to the evolution of the playback.

In the following sections we will investigate the impact of the different mechanisms both on the user experience and on the system infrastructure.

3. DATA COLLECTION

In this section we first introduce the tool we use to collect the traffic, giving a high level description of the algorithms employed to classify the YouTube traffic. Then, we present the collected data sets that we used for the analysis.

3.1 Collection tool

To inspect the network traffic we relied on Tstat [5, 14], an Open Source packet sniffer with Deep Packet Inspection (DPI) capabilities, which implements both traffic classifiers and fine-grained flow-level statistics. Tstat is able to rebuild TCP flows by monitoring packets that are sent and received by clients. Leveraging on this, we improved Tstat so as to identify and we distinguish all possible HTTP messages that can be observed when a client downloads a YouTube video. In this paper, we focus only on the `videoplayback` and `generate204` requests from the client, and distinguish among all possible server replies (HTTP `200`, `204`, `206`, `302` responses). By parsing the URL of the HTTP messages, we can identify PC-player and Mobile-player accesses[4] and extract specific video information, such as the videoID and the video format. We also extract other video related information from the video header, such as the resolution, total duration and size of the video.

In addition to the video properties, we also collect several TCP flow-level statistics, such as the total number of packets and bytes transmitted and received, the total flow duration and the average RTT[5]. Further information on Tstat capabilities as well as the source code can be obtained from [5, 14].

3.2 Data sets

We collected data sets at five vantage points spread across three countries including both Points-of-Presence (PoP) in nation-wide ISPs and University campuses. These data sets

[4]URL requests from mobile devices contain `app=youtube_gdata` or `app=youtube_mobile`. We do not make any further distinction on the type of browser or mobile device used.

[5]RTT is estimated by leveraging TCP acknowledgements [5, 14]

Name	Type	Flows	Vol.[GB]	SrcIP	Videos
US-Campus	Campus	2,172,250	10,898	20,455	446,870
EU1-Campus	Campus	173,024	714	1,203	50,205
EU1-ADSL	Home	740,330	2,615	8,154	189,788
EU1-FTTH	Home	135,907	480	1,136	33,762
EU2-ADSL	Home	830,476	3,688	5,826	205,802

Table 1: Data sets collected.

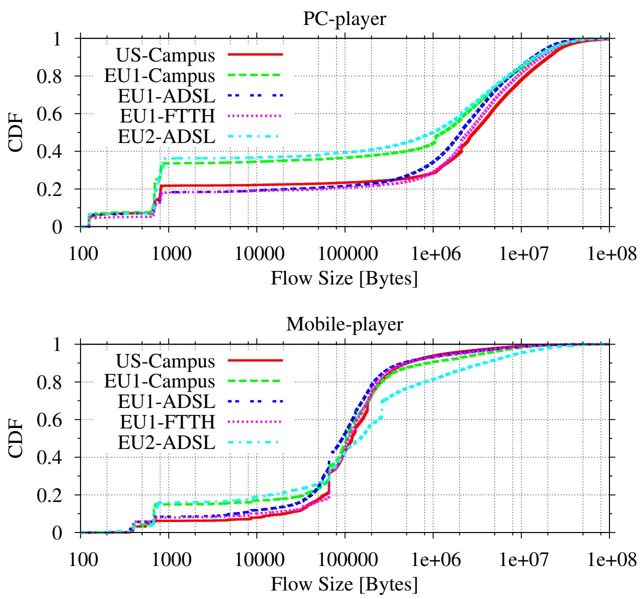

Figure 2: Distribution of the TCP flow size.

represent a unique heterogeneous mix of users and technologies. At each vantage point, we installed a *probe* consisting of a high-end PC running Tstat and monitoring all traffic generated by local clients.

This paper focuses on one-week long traces, collected simultaneously at the five locations, starting at 12:00 AM (local time) on February 25th, 2011. Table 1 summarizes the data sets reporting the name, the type of customers, the total number of YouTube video flows and the corresponding volume of bytes, the number of distinct client IP addresses and the number of videos downloaded. Overall, the data sets account for more than 16 TB of traffic, more than 900,000 videos, and more than 35,000 different client IP addresses.

The Home data sets have been collected from nation-wide ISPs of two different European countries. EU1-ADSL and EU1-FTTH correspond to two different PoPs within the same ISP aggregating users with ADSL and Fiber-To-The-Home (FTTH) access technology. The campus data sets are collected in two different University campus networks, one in the United Stated and the other in Europe. To confirm the popularity of YouTube service, we observe that in all monitored networks the volume of traffic generated by YouTube videos accounts for more than 25% of the total traffic during peak time. Finally, note that the mobile traffic collected in our data sets refers to devices accessing the Internet via WiFi access networks and not via 3G/4G ISPs.

4. FLOW AND VIDEO CHARACTERISTICS

We begin our analysis by giving an overview of the traffic generated by PC-player and Mobile-player clients. Fig. 2 reports the Cumulative Distribution Function (CDF) of TCP flow sizes, i.e., number of bytes (B) received by the client in a flow. Let us focus on the PC-player traffic (top plot). Steps in the CDF clearly show the presence of flows of typical size corresponding to specific HTTP messages: `'204 No Content'` flows are about 120 B long, `'302 Found'` flows are in the [800-1000] B range, while flows containing the `200 OK` responses are typically longer than 80 kB since they contain the actual video data. Interestingly, the initial part of the distribution is different for different probes, with EU1-Campus and EU2-ADSL suffering a higher fraction of redirections (`'302 Found'`) messages. However, the tail of the distribution looks rather similar, suggesting that the size of videos downloaded in the networks monitored is similar. We will detail this better in Sec. 5.

Name	%Flows	%Bytes
US-Campus	32.5	3.5
EU1-Campus	15.6	2.8
EU1-ADSL	27.2	3.9
EU1-FTTH	42.2	6.6
EU2-ADSL	4.2	1.6

Table 2: Fraction of flows and bytes due to mobile terminals.

Looking at results for Mobile-player (bottom plot), we observe that the flow size is similar across data sets, with EU1-Campus and EU2-ADSL still exhibiting higher fraction of redirection messages. However, comparing PC-player and Mobile-player we observe interesting differences: (i) the absence of the prefetching phase causes the `'204 No Content'` responses, of size 120B, to disappear in Mobile-player; (ii) the abundant presence of the HTTP requests using the `Range` header field causes the flows carrying the video data to be one order of magnitude shorter than in PC-player. This is a direct artifact of the video chunking mechanisms and not a difference in the actual video duration and size (see Fig. 3 and Fig. 4). Interestingly, the 500B long flows are due to `'206 Partial Content'` replies to the first `videoplayback` request using the `'Range: bytes 0-1'` header which Mobile-player uses to discover the actual video size from the HTTP response field (see Sec. 2.2).

The effect of the chunking mechanism adopted by Mobile-player has clearly an impact on the number of flows generated by mobile devices to download the content. Table 2 quantifies this by reporting the fraction of flows and bytes that are due to Mobile-player for the different data sets. We can notice that, while Mobile-player traffic is a small fraction of the total volume, it accounts for a much larger fraction of flows. This might pose performance issues on flow-based devices, like NAT boxes or full-state firewalls which keep per-flow state.

Consider now the volume of bytes. Unexpectedly, only less than 6% of YouTube traffic is due to users from mobile devices. The networks we consider offer both wired and WiFi access with large penetration of smartphones, especially in the campus networks. Therefore, one would expect that a large fraction of YouTube accesses is done from such terminals. Our measurements contrast this intuition. Moreover, some recent studies [6, 9] show that multimedia content is responsible for more than 40% of the total volume due to wireless terminals, with YouTube as the main contributor. Our results show that this traffic is little compared to the volume

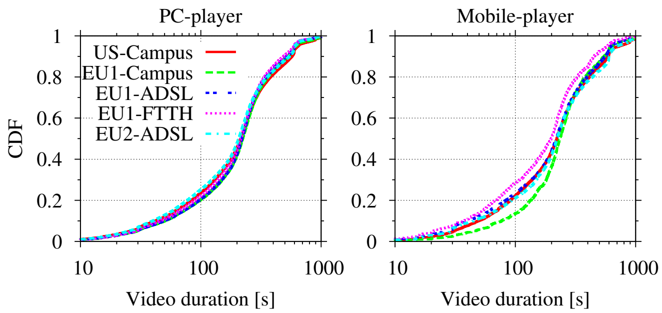

Figure 3: CDF of video duration.

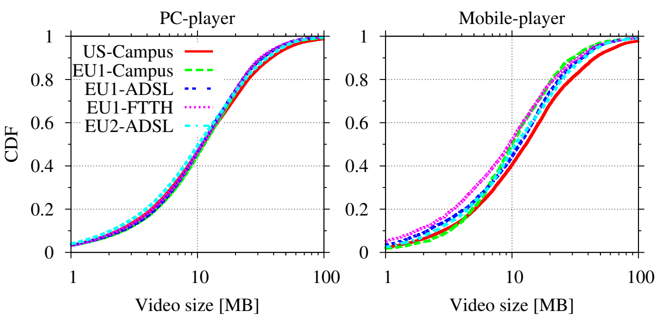

Figure 4: CDF of video size.

generated by wired networks. A possible explanation for this is the fact that wireless users in our networks still prefer to access YouTube videos from standard PC browsers, because of the better user experience compared to smartphones.

4.1 Video content duration and size

Fig. 3 reports the duration, in time, of videos for both PC-player (left) and Mobile-player (right). Note that this corresponds to the duration of the complete video and not to the portion of video watched by the user[6]. Considering the PC-player scenario, and comparing the measurements from the different data sets, we notice that there is great similarity across vantage points so that it is impossible to distinguish among them. For example, in all vantage points, 40% of the videos last less than 3 min, and less than 5% of the videos last more than 10 min.

Consider now the Mobile-player case. We observe a slightly moderate difference among the video duration accessed from different probes (notice the log-scale on the x-axis). Still, 40-50% of all videos accessed from mobile terminals are shorter than 3 minutes, and 5% of videos last more than 10 minutes. Indeed, the Mobile-player and the PC-player CDFs are very similar among them too.

This result shows that people with very different cultural bias (e.g., Europeans vs Americans, students vs residential users), using very different terminals (smartphones vs PCs) and with different Internet access bandwidth (ADSL vs FTTH vs WiFi vs Ethernet) produce and consume the same type of content: short videos which can be quickly watched from YouTube. At the aggregation level that we study, this reflect the distribution of video duration of the YouTube service.

[6]This information is extracted from the video metadata.

ID	Video Codec	Audio Codec	Container	Res.	Name
13	H.263	AMR	3GP	144p	others
17	MPEG-4 ASP	AAC	3GP		others
5	FLV1	MP3	FLV	240p	240p-Fl
36	H.264	AAC	3GP		others
34	H.264	AAC	FLV	360p	360p-Fl*
18	H.264	AAC	MP4		360p-Mp+
43	VP8	Vorbis	WebM		others
35	H.264	AAC	FLV	480p	480p-Fl
44	VP8	Vorbis	WebM		others
22	H.264	AAC	MP4	720p	720p-Mp
45	VP8	Vorbis	WebM		others
37	H.264	AAC	MP4	1080p	others
38	H.264	AAC	MP4	3072p	others

(*) PC-player default format , (+) Mobile-player default format

Table 3: YouTube supported video formats.

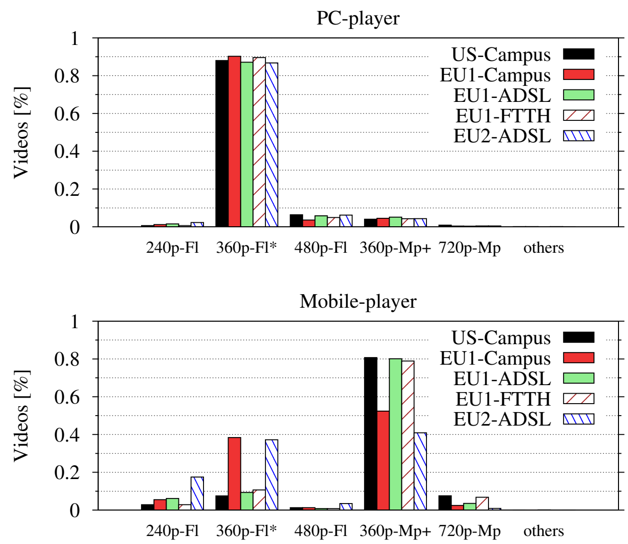

Figure 5: Fraction of videos for popular video format.

Fig. 4 reports the total video size in bytes of the videos that have been seen in our data sets. We find very similar results across traces and devices. This is counterintuitive, since we would expect the distribution to be more variable, e.g., due to the availability of videos with different resolutions, and different encoding formats. In addition, intuition would suggest that the video size would be larger for PC-player than for Mobile-player, to better accommodate the limited resources of smartphones. But this is also not clear in the graph. To understand this better, in the next section we dig into the impact of video codecs and resolutions.

4.2 Video format characterization

A “video” is a complex object that multiplexes encoded video and audio streams. Encoding is done according to different algorithms, and the result is then organized into a container of different type. The combination of the encoding algorithm, video resolution, and the type of container defines the *video format*. A plethora of video formats are available, some of which are proprietary while others are standard.

YouTube supports the formats listed in Table 3. Each format is identified by a unique ID corresponding to the `itag` parameter in the video request. Each ID corresponds to a unique combination of video codec, audio codec, container and resolution. The last column shows the naming convention we used in this paper to identify each format. A marker highlights the YouTube default video format.

The variety of formats reflects the evolution of the system and technology over the last years. In the early days, only Flash Video (FLV) content was supported only at 240p resolution. In 2007 the MP4 container was introduced along with resolution 360p. This switch was driven by the introduction of new devices that did not support FLV videos (e.g., Apple iOS devices). There are also 3GP formats, which are specific for mobile devices, and the more recent WebM formats [15], which are part of the HTML5 specifications. As of today, H.264 video codec is the most widely adopted standard. Note that when the user uploads a new video, the system automatically generates the different video formats and makes them available to download.

At playback time, the user can eventually choose among multiple resolutions via the player graphical user interface. For PCs, the Adobe Flash player presents a menu button listing the available resolutions, e.g., 240p, 360p and 480p. Some Mobile-players instead present a toggle button with the choices of "high" and "low" quality, without explicit indication of the resolutions.

The supported formats do not have the same popularity. Fig. 5 reports the breakdown of video formats considering PC-player and Mobile-player data sets on top and bottom plots, respectively. There is a clear difference respect to the device used to access the video: Flash based formats are largely preferred by PC-player, while MP4 is the preferred container for Mobile-player. This is not surprising considering that Apple iOS products (e.g., iPhone, iPad, iPod touch) cannot handle FLV content. The higher fraction for Flash based formats in the Mobile-player data sets in EU1-Campus and EU2-ADSL may be related to different popularity of devices among certain users (e.g., students) to prefer smartphones running the Android operating system or Windows Mobile which support Flash content.

The default video resolution offered to PC-player is 360p, while Mobile-players tries to retrieve the best available quality according to the network/device capabilities. This causes 720p format (also known as High Definition - HD) to be more popular for Mobile-player than PC-player and this difference is the result of a system design choice.

The previous findings hold true independent from the vantage point, showing the ubiquitousness of the YouTube service. We expect this to change in 3G/4G networks, where the 3GP formats are known to be used and low resolution videos are offered by default.

4.3 Video encoding bitrate

Given a codec and a video resolution, the video quality has a strict relation with the *video encoding bitrate*. It is therefore interesting to observe what is the typical encoding bitrate of YouTube videos. Fig. 6 reports the CDF of the video encoding bitrate for the most important video formats. Each curve aggregates statistics from all videos of the

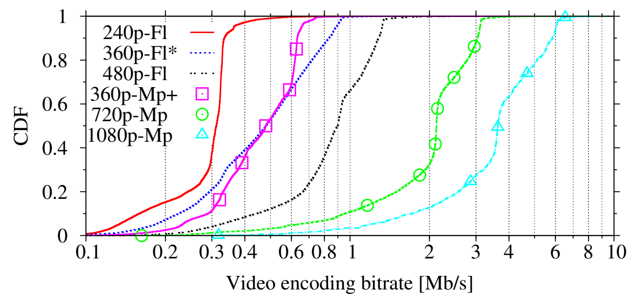

Figure 6: CDF of video encoding bitrate.

data sets (each single data set presents the same distribution, being this a system choice). MP4-based formats are highlighted by line-points patterns. In general, the actual encoding bitrate is the minimum between the maximum allowed bitrate, and the bitrate that allows to achieve the desired quality. The latter depends on the video content, e.g., more static video sequences allow to reach lower encoding bitrate. This is reflected in the curves. For example, consider 240p-Fl (FLV) videos. The sharp knee around 300 kbps is the effect of the maximum bitrate limit, which is reached by 70% of videos. About 30% of videos are instead quality limited. Similarly, 360p-Mp (MP4) videos are configured to not exceed 600 kb/s, with most of the videos being quality limited. In some cases, the maximum bound of the video encoding rate can be violated as shown for example for the 10% of 240p-Fl videos. This can be due to a change in the encoding parameters that happened at some time.

It is known that the higher is the resolution, the higher is the bitrate. For example, the 360p videos (currently the default choice) do not exceed 1 Mb/s video rate, while 480p video bitrate goes up to 1.5 Mb/s. 720p and 1080p require up to 3 Mb/s and 6 Mb/s respectively. This allows to speculate on the impact of YouTube switching to higher resolution by default. For example, defaulting to 480p would correspond to almost double the amount of traffic due to YouTube, with possibly large impact on both the YouTube CDN and on ISP networks. Going to 720p as the default choice would correspond to multiply by a factor of 4 the offered traffic. Given that YouTube already accounts for more than 20% of Internet traffic and assuming the user demand remains the same, this would correspond to a critical traffic surge that might impair the YouTube service itself.

5. USER BEHAVIOR AND IMPLICATIONS

In this section we focus our attention on the way people watch videos from the YouTube system, observing if they interact with the GUI, e.g., switching resolution or going in full screen mode, and which portion of the video people actually watches. Both have interesting implication on the workload the system has to handle and the efficiency it achieves in serving the requests. We first introduce the concept of *video session* which is required to characterize the user's behavior.

5.1 Methodology

As we have already seen in Sec. 4, video can be downloaded in multiple TCP connections, which is predominantly the case for Mobile-player. To illustrate this, Fig. 7 shows

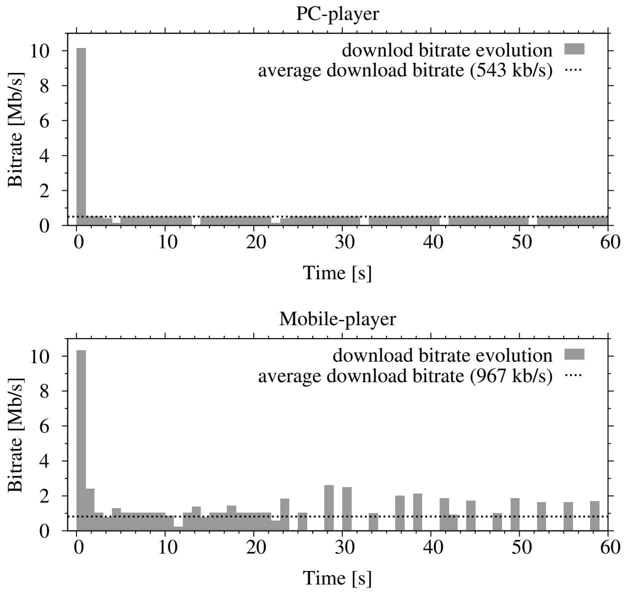

Figure 7: Evolution over time of the download bitrate for a video encoded at 540 kb/s.

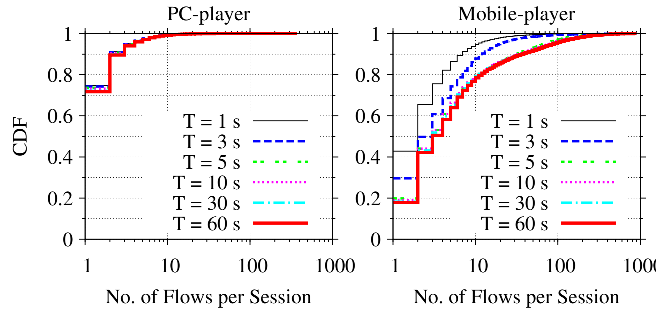

Figure 8: Sensitivity of the number of TCP connections per session for different values of T. EU1-ADSL data set.

	PC-player			Mobile-player		
Data set	0	1	>1	0	1	>1
US-Campus	95.10	4.60	0.30	99.75	0.19	0.05
EU1-Campus	96.62	3.12	0.27	99.28	0.61	0.10
EU1-ADSL	95.27	4.45	0.28	99.63	0.28	0.09
EU1-FTTH	95.73	3.99	0.28	99.39	0.42	0.19
EU2-ADSL	95.14	4.40	0.46	98.07	1.36	0.57

Table 4: Percentage of resolution switch.

the bitrate evolution obtained downloading the same video from a PC (top) and a mobile device (bottom) inside the EU1-Campus network. In both cases, the server starts sending an initial burst of data at a very fast rate to quickly fill the play-out buffer at the player. This mechanism is conventionally called "fast-start". The server then starts shaping the rate as observed in [2]. Note that this is a server-based shaping mechanism in which the client has no role (neither application layer nor TCP flow control messages are sent). For PC-player, after the initial burst, the download proceeds within the same single TCP connection, whose throughput is practically equal to the average video encoding rate (dotted line). Note that the average download rate is computed discarding the initial burst.

For Mobile-player instead, the bitrate evolution is more bursty. This is a consequence of leveraging different TCP connections to download chunks of video. Indeed, from second 23 and on, the mobile terminal aborts the ongoing TCP connection, and starts requesting chunks of video on separate TCP connections. They last about 1 second and are separated in time by about 2 seconds of silence. Since a new TCP connection is used, the server enters the "fast-start" phase, which is early interrupted when the client aborts the underlying TCP connection. We believe this mechanism is due to a client-side buffer management policy which abruptly interrupts the TCP connection when the play-out buffer is filled up. The client then re-starts the download when the buffer depletes below a certain threshold. This results in an inflation of TCP connections, and a possible inefficient download.

The early abortion of the TCP connection can be due to other causes as well. For example, a resolution change or a fast-forward in the video are handled by aborting the current download and starting a new one for both PC-player and Mobile-player. Finally, the initial control messages possibly sent on separate TCP connections are also fundamentals to capture the dynamics of the download.

To capture the variety of behaviors for downloading a video, we use the concept of "video session", i.e., a mechanism to group all connections related to the download of the same content. More specifically, a video session corresponds to the set of TCP connections that i) share the same source IP address and *videoID* and ii) are separated by a silence period shorter than T seconds. For instance, two connections c_1 and c_2 belong to the same session if the difference in time between the beginning of c_2 (time of the TCP SYN packet) and the end of c_1 (time of the last packet observed) is smaller than T.

Fig. 8 reports the number of connections per session for different values of T. The EU1-ADSL data set is considered, but other data sets show identical results. The choice of T is not critical for PC-player, while $T > 5$ s is required to properly aggregate Mobile-player connections. In the following, we set $T = 60$ s, a conservative choice to better capture users actions that could happen after the download has been completed but while the playback is still running.

Fig. 8 also shows the impact of the Mobile-player mechanisms in the number of connections per session. While for PC-player, only 2% of the sessions have more than 6 connections, for Mobile-player more than 4% of the sessions involve more than 100 connections.

5.2 Resolution switch

Given the above definition of a session, a change of video resolution is easily detected by observing requests with the same *videoID*, but different video format.

Table 4 reports the percentage of sessions involving zero, one or more than one resolution switch for both PC-player and Mobile-player. Surprisingly, results show that a resolution switch happens for less than 5% of PC-player sessions. This is an example of inertia, where users stick with the default video format. This also shows that users are possibly

Data set	Low-to-High	High-to-Low
US-Campus	95.7	4.3
EU1-Campus	86.1	13.9
EU1-ADSL	93.9	6.1
EU1-FTTH	90.5	9.5
EU2-ADSL	83.6	16.4

Table 5: Percentage of resolution switch breakdown for PC-player.

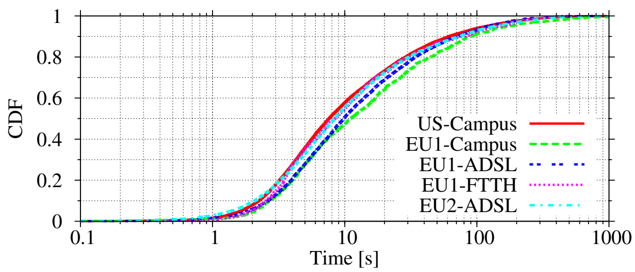

Figure 9: Time at which Low-to-High resolution switch happens.

not interested in this feature or they are unaware of it. For Mobile-player the choice of resolution is either hidden or not available, and a marginal fraction of users exploit it.

We can further classify the resolution changes in "Low-to-High" and "High-to-Low" considering the involved video resolution. Table 5 reports the breakdown. We can see that the Low-to-High resolution switch is largely predominant, with more than 80% (for all traces) being a 360p-Fl to 480p-Fl switch. Interestingly, when the full screen playback is enabled, the Flash player automatically switches from 360p-Fl to 480p-Fl (the converse is not true). Combining Tables 4 and 5, we can conjecture that full screen mode is not popular, but it is the main cause of resolution switch.

As final note, the largest majority of High-to-Low switch are 360p-Fl to 240p-Fl. This suggests that those are triggered by the user because of bad performance. EU2-ADSL and EU1-Campus show a slightly larger High-to-Low switch fraction. As we will see in the following, those are the two vantage points with slightly worse performance.

To complete the analysis, we investigate when the resolution switch is triggered. Fig. 9 shows the CDF of the time between the session start and Low-to-High resolution switch. Due to buffering at the player, this is an overestimate of the actual switch time. 50% of these events happens in the first 10 seconds, while only 10% of users trigger them after 1 minute. In terms of video size, more than 80% of the switches happens in the first 20% of the video data download while only 5% occurs in the second half of the video. The same consideration holds for High-to-Low changes. Overall we can conclude that resolution changes are usually performed at the very beginning of the playback.

We find surprising that results are practically identical in all data sets despite differences in users habits and cultures.

5.3 Fraction of watched video

We now focus our attention on the time a user spends watching a video. To measure this, we leverage the fact that the player abruptly aborts the video download if the user changes the web page on the browser (or custom player).

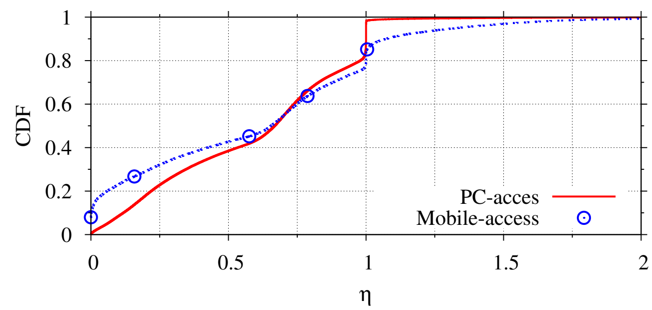

Figure 10: CDF of the fraction of downloaded video bytes. EU1-ADSL data set.

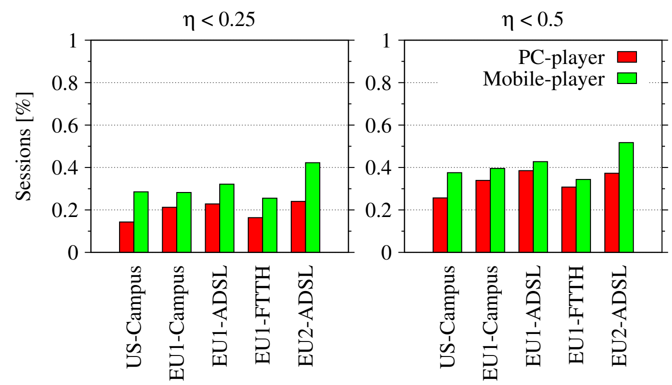

Figure 11: Fraction of video downloads having $\eta < 0.25$ and $\eta < 0.50$

Let η be the fraction of downloaded video within a session with respect to the actual video size. If $\eta < 1$, then the user did not watch the entire video[7].

Fig. 10 shows the CDF for η for EU1-ADSL data set. Two observations hold: i) about 80% of video sessions are abruptly interrupted; ii) Mobile-player results show that the player can download *more* data than the video size ($\eta > 1$). We will investigate this better next.

To better compare results, Fig. 11 details the fraction of video downloads having $\eta < 0.25$ and $\eta < 0.5$ for different data sets. Interestingly, results are similar for all vantage points, with users on Mobile-player consistently aborting earlier than users on PC-player. Fig. 12 details the absolute and relative time at which the user stops watching the video on the left and right plots, respectively. It shows that people tend to abort the playback very soon, with 60% of videos being watched for less than 20% of their duration. This can be due to a mismatch between the users' interests and the content they find on YouTube. Notice that this is also an interesting fact that could be exploited to better handle the content distribution among the CDN nodes, e.g., caching only some portion of each video. The impact of Mobile-player versus PC-player is very limited, testifying

[7]By checking the `Range` header field for requests, we filter out those sessions in which the user fast-forward the playback to a position outside the already buffered portion of the video.

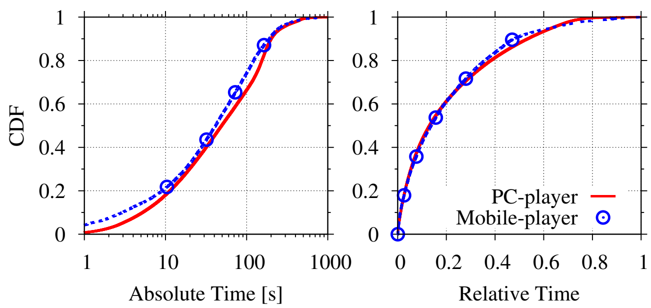

Figure 12: CDF of absolute (left) and relative (right) portions of watched videos for sessions with incomplete downloads. EU1-ADSL data set.

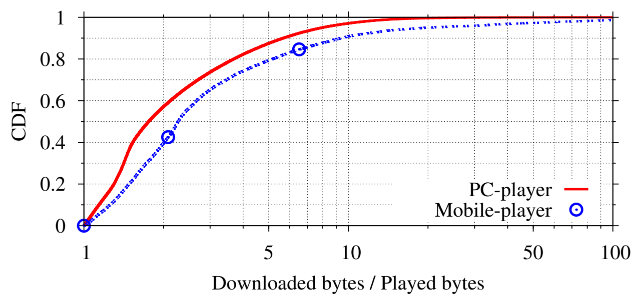

Figure 13: CDF of ratio between downloaded bytes and played bytes. EU1-ADSL data set.

that the probability of aborting the playback is not biased by the device but it is related to users' habits.

5.4 Impact of buffering policy and user early abort

Consider now all video data already buffered at the player at the time the user aborts the playback. That data has been downloaded in vain. Fig. 13 precisely quantifies this by reporting the ratio among downloaded bytes and the amount of bytes possibly consumed by the player. The latter is evaluated as $session\_duration * average\_encoding\_bitrate$, assuming that the playout started immediately after the first byte has been received, and that data is consumed at the video encoding bitrate. Since the initial buffering is neglected, our estimation can be considered as a lower bound. Notice also that we cannot evaluate the amount of wasted data for sessions which already have completed the download.

In spite of this, results are dramatic for PC-player: 40% of sessions download more than two times the amount of data that was possibly watched. This is the result of aggressive buffering policies adopted by YouTube servers (recall that server-side shaping is adopted) [2]. Even worse, the Mobile-player waste is higher, with 20% of sessions downloading more the 5 times the amount of possibly watched data.

This waste could be reduced by implementing better streaming policies, e.g., the server sends chunks of the video by explicitly request by the client. Alternatively, a more accurate prediction of the fraction of the video that a user will watch can be leveraged to avoid transferring useless data.

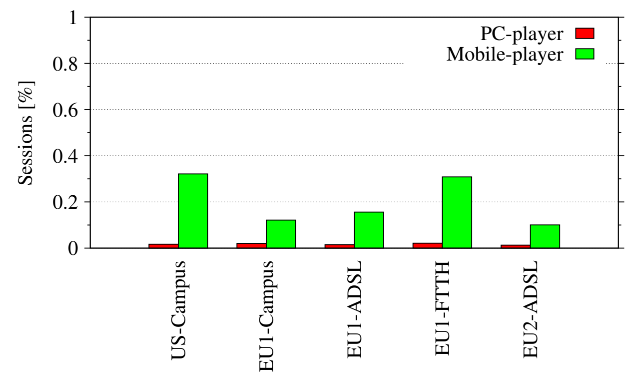

Figure 14: Fraction of sessions downloading more than the entire video.

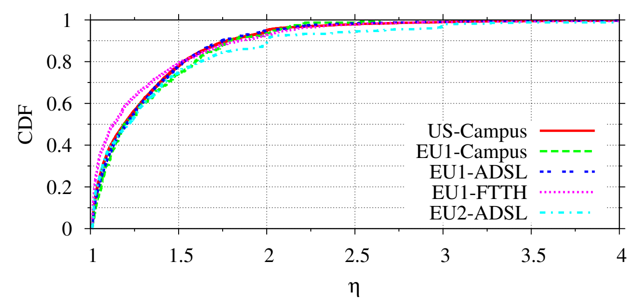

Figure 15: Ratio of downloaded data versus video size for sessions with $\eta > 1$. Mobile-player.

5.5 Sessions downloading more than the video size

Let us now focus on sessions where $\eta > 1$. Intuitively, those should be very limited, since one would expect that the player should not download more data than the total video size. Fig. 14 shows the fraction of sessions for which this happens. Only sessions with no resolution switch are considered. For PC-player, less than 2% of sessions show this phenomenon. We have found that the exceeding amount of volume is possibly related to users watching the same video multiple times causing the player to re-download the video. Overall, this effect is marginal.

For Mobile-player instead we observe that 15-30% of sessions download more than the actual video size. Performing some active experiments, we have confirmed at least two causes for this: 1) in case of backward seeks, the player has to re-download the same content because it has been already discarded from the player's local buffer. This does not happen for PC-player which caches the entire video; 2) the aggressive chunk-based download mechanism is source of inefficiency: the Mobile-player often requests chunks bigger than needed, i.e., requesting from a desired position x up to the end of the video. The server then sends data from position x at a high rate, quickly filling up the Mobile-player buffer, e.g., up to position y. Being the buffer full, the player application abruptly closes the underlying TCP connection. However TCP had already received some data at the transport-layer receiver buffer up to position $y' > y$. The data $y' - y$ is thus discarded. When the application buffer

Data set	PC-player	Mobile-player
US-Campus	39.17	47.9
EU1-Campus	36.91	38.1
EU1-ADSL	24.93	38.7
EU1-FTTH	38.43	53.5
EU2-ADSL	29.27	35.6

Table 6: Average percentage of wasted bytes considering peak hour with respect to useful data.

depletes, the player requests data from position y and not from position y'. Considering the download of $y' - y$, the aggressive server buffering policy coupled with player limited buffering capabilities is thus origin of inefficiency.

To quantify the waste of traffic due to this, Fig. 15 reports the CDF of the ratio of downloaded data versus nominal video size for sessions with $\eta > 1$. 50% of sessions download 25% more data, and 4% of the sessions downloads more than the twice of the video size.

5.6 Wasted video data

Table 6 quantifies the overall percentage of wasted bytes with respect to useful data. It includes both the effect of aggressive buffer management and of chunk based video retrieval mechanisms. Measurements refer to the peak-hour time, when YouTube traffic peaks to several hundreds of Mb/s in most vantage points. Results show that the amount of traffic downloaded by clients but not used by players is comparable with the useful data traffic. For example, for US-Campus, the wasted traffic in a single hour amounts to 28.8 GB and 1.5 GB for PC-player and Mobile-player, respectively, corresponding to more than 67 Mb/s of constant waste. This is a large amount of wasted bandwidth both from the perspective of the ISP and the YouTube CDN.

We have performed experiments on mobile devices connected to a 3G network using both iOS and Android devices. The problem shows up exactly in the same way, with clients downloading a lot more data than the video played and the video size. We did not observe differences among different devices or operating systems. This is an issue that might be particularly critical given the increasing popularity of YouTube accesses from mobile devices in 3G/4G networks.

6. STREAMING PERFORMANCE

In this section we focus on the streaming performance considering two metrics that reflect the user experience: *startup latency* and *bitrate ratio* between download rate and video encoding rate. We then pinpoint possible causes that can impair them.

6.1 Startup latency

We define the startup latency as the time elapsed between the first `videoplayback` request and the first received packet containing actual video payload. This corresponds to a lower bound of the delay experienced before the actual video playback starts since the initial buffering time at the player is ignored. The latter indeed is implementation dependent and hard to know. We prefer thus to focus on a simpler, more precise and accurate measure.

Fig. 16 reports the fraction of sessions with the startup latency higher than a certain threshold. Given that we are interested in studying the user experience, we selected

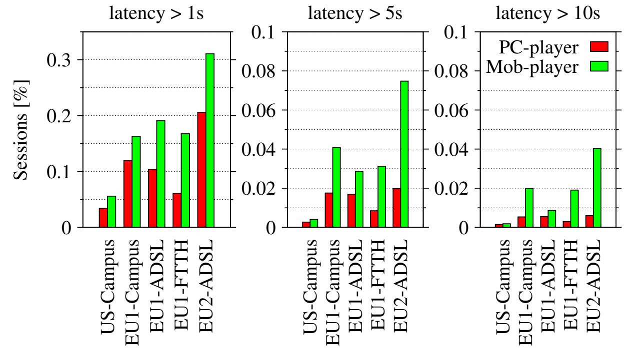

Figure 16: Fraction of sessions with high startup latency. Note the different y-axis across plots.

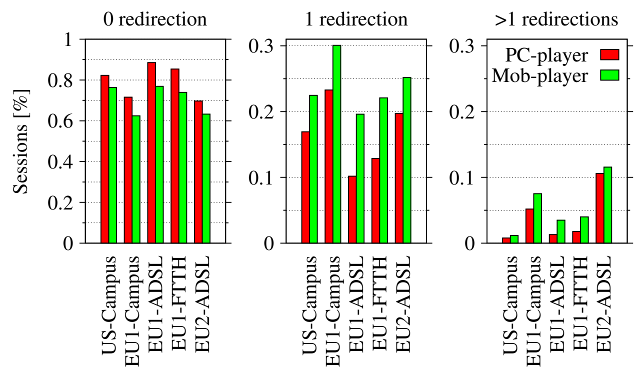

Figure 17: Fraction of sessions suffering redirections. Note the different y-axis across plots

threshold values that can be appreciated by the user, i.e., 1, 5 and 10 s. Results show that the performance is heterogeneous across the data sets, with Mobile-player suffering larger delays. For example, in the US-Campus less than 5% of sessions suffer a startup latency higher than 1 s. In the EU1-Campus instead, more than 10% of sessions start after 1 s with 2% of them suffering a startup latency higher than 5 s.

We found that the delay is due to a combination of causes. **Redirections**: Video sessions can suffer from a different number of redirections. Each redirection involves i) a DNS query to resolve the hostname of the next video server, ii) the opening of a new TCP connection, iii) a new video query. The network distance between the client and the server plays also a significant role, since YouTube CDN is likely to direct clients to video servers with the closest RTT. However, in case of redirections, the final server may not be the closest one in RTT.

Fig. 17 reports the fraction of sessions affected by redirections. More than 70% of PC-player session does not suffer from redirections in all data sets, while Mobile-player sessions are more likely to be redirected. Understanding why this is happening is difficult. A possible cause of redirection is due to cache miss. [13] already showed that a cache miss at the closest data center, may cause a redirection to a farther away data center. However, following requests for the same content are directly served by the clos-

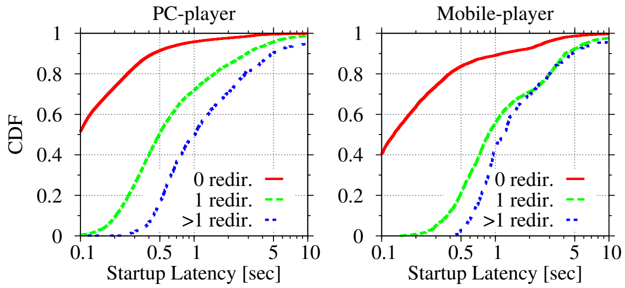

Figure 18: CDF of startup latency for different number of redirections. EU1-ADSL data set.

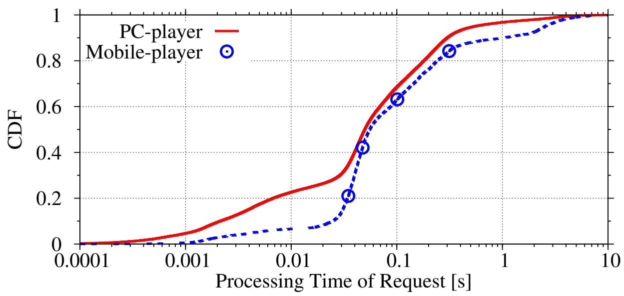

Figure 19: CDF of the video query processing time. EU1-ADSL data set.

est cache. This hints to a caching mechanism based on pull schemes. Since Mobile-player videos are less frequently requested than PC-player videos because of the different video format adopted, a cache miss is possibly more likely to happen. Thus more redirections can occur.

Fig. 18 depicts the impact of the redirections on the startup latency. We can see that the higher is the number of redirections in the video session, the higher is the startup delay. Considering PC-player (left plot), 92% of sessions that do not suffer redirections exhibit a startup latency smaller than 500 ms. When one redirection is faced, only 50% of sessions start within 500 ms. If more than 1 redirection is faced, more than 82% of sessions have a startup latency of at least 500 ms, with 10% of sessions suffering a latency higher than 5 s. Trends are similar for mobile player (right plot) where in general startup latencies are higher than in the case of PC-player.

Video request processing time: Another possible cause of large startup time can be the server processing time, i.e., time needed by the video server to process a video request. To estimate it, we compute the time between the last `video-playback` request sent by the client and the first video packet sent by the server. To eliminate the network delay we subtract the RTT.

Fig. 19 reports the CDF of the estimated processing time for both PC-player and Mobile-player in the EU1-ADSL data set. Other data sets show similar trends. We can see that 50% of the requests are served within $<$ 50 ms. A sharp knee around 30 ms is present and a heavy tail is found with processing time growing up to 5 s. The distribution reflects the time required by the cache to retrieve the requested content from the back-end before serving it. Very low latencies can be related to the video being already

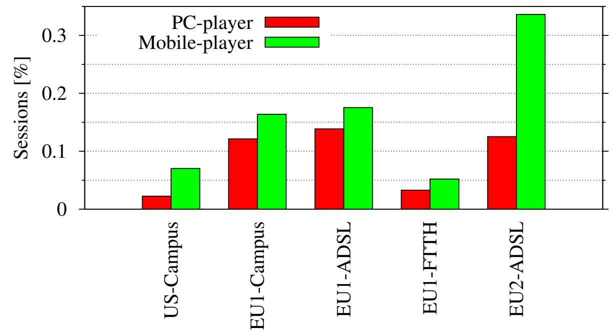

Figure 20: Fraction of sessions with bitrate ratio$<$ 1.

cached in the server memory; values in [30,300] ms can be related to disk access latency; finally values larger than 300 ms can be due to congestion in the back-end or to packet loss recovered by lengthy TCP timeout, or to rare content that has to be fetched from some slower storage system. The fact that Mobile-player responses require higher processing time can again be explained by the less frequent requests of Mobile-player video content. Note also that the prefetching mechanism implemented by PC-player could also speed-up the content retrieval (see Sec. 2).

6.2 Bitrate ratio

The download bitrate of the video has a fundamental role in defining the quality of the video playback. In fact, if data is not received fast enough, buffer “under-run” events will be suffered, causing the video playback to pause. To measure the smoothness of the playback, we define the *bitrate ratio* as the ratio between the average session download bitrate and the video encoding bitrate. The first corresponds to the total amount of bytes downloaded aggregating flows of the same video session, divided by the time between the first and the last video packet. According to this definition, a bitrate ratio smaller than 1 is a clear sign of impaired performance.

Fig. 20 reports the fraction of sessions with a bitrate ratio lower than one. Some interesting observations hold: first, the access technology has a clear impact on the performance with the ADSL networks performing worst for more than 10% of the downloads with respect to the other networks. For instance, compare EU1-ADSL and EU1-FTTH (the latter offers 10Mb/s full duplex access capacity). Both refer to customers of the same ISP in the same city. Still, EU1-ADSL customers suffer worse performance. Unexpectedly, EU1-Campus performs also quite bad. Further investigation revealed that this is the result of a local University network policy that limits the bandwidth of subnets of some dorms. Most of the sessions having poor performance are indeed coming from those subnets. Fig. 20 shows that Mobile-player presents consistently lower performance than PC-player. This can be due to the presence of a WiFi network that is used by Mobile-player devices. The shared WiFi connection can indeed impair the download throughput. This is the case for US-Campus.

Other causes of reduced performance can be related to the YouTube infrastructure performing worst when serving Mobile-player requests. Consider EU2-ADSL, in which more

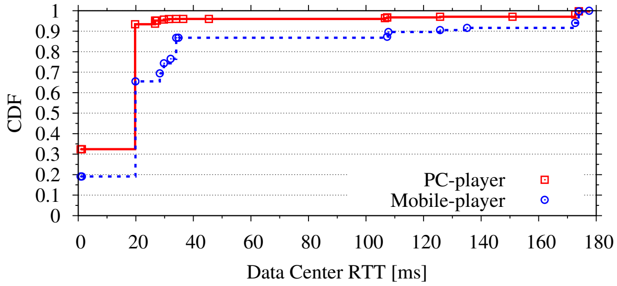

Figure 21: CDF of the fraction of bytes delivered by each data center. EU2-ADSL data set.

than 32% of Mobile-player sessions are performing poorly versus less than 13% of PC-player sessions. We pinpoint that Mobile-player impaired performance is related to the YouTube system. Consider Fig. 21. It reports the CDF of the fraction of bytes downloaded by different video servers respect to the RTT to the EU2-ADSL vantage point. Each point in the figure aggregates video servers that belong to the same CDN data center as in [13]. We found that EU2-ADSL clients can use a data center which is very close to the vantage point (RTT $<$ 1 ms). However, it can only serve 35% of the PC-player sessions. The majority of sessions are indeed served by a second data center which is 20 ms far from the vantage point. For PC-player, these two data centers handle 96% of video requests. However, due to the lower popularity of Mobile player accessed content, 35% of Mobile-player sessions are served by other data centers, 10% of which are found outside Europe and suffer RTT $>$ 100 ms. These sessions can be impaired by network congestion and can exhibit lower download bitrate. Finally, recall the Mobile-player chunking mechanism. The cost of opening a new TCP connection to request a new chunk becomes significant when the RTT is in the order of hundreds of ms, impairing the download bitrate too.

Overall, measurements presented in this section show that Mobile-player performance result generally less efficient than PC-player. The intrinsically smaller popularity of Mobile-player accessed videos poses additional challenges to the YouTube CDN infrastructure, which results highly optimized for PC-player as of now.

7. RELATED WORK

We consider three categories of works related to us.

YouTube Videos Characterization: These works have focused on characterizing various aspects of YouTube videos as well the usage patterns. On the one hand, [7] and [17] characterized video popularity, durations, size and playback bitrate, as well as usage pattern statistics such as day versus night accesses and volume of traffic considering a campus network. On the other hand, [3] and [4] crawled the YouTube site for an extended period of time and performed video popularity and user behavior analysis. Further, [3] compares YouTube to other video providers such as Netflix and [4] investigates social networking in YouTube videos. In contrast, our work is focused on the comparison between PC-player and Mobile-player downloads and goes deeper in the characterization of the content also taking advantage of the heterogeneous set of users and networks monitored.

YouTube Infrastructure Studies: These works characterize the YouTube video delivery infrastructure [1, 11, 13]. [11] shows that most YouTube videos are distributed from a single data center in the U.S. [1] shows that a few data centers in the U.S. were in charge of distributing the videos around the world. Finally, [13] shows that data centers spread around the world are in charge of distributing the video, and that latency between clients and servers plays a role in content server selection. In contrast, our work is focused on understanding the difference on video delivery between mobile devices and PCs. In particular, we show how mobile devices control the video download rate while in the case of PC-player, the download rate is controlled by the server [2].

User Behavior on Mobile Devices: More recently, there have been several works characterizing high level usage patterns of mobile devices [9, 6, 12]. [9] shows that the number of mobile devices doubled between 2009 and 2010 and that more than 80% of mobile devices traffic is HTTP, with multimedia traffic alone accounting for more than 30% of HTTP. [6] compares the content and flow characteristics of mobile devices and PCs traffic. Using a DPI tool, the authors are able to show that YouTube alone accounts for more than 35% of the Internet traffic. In contrast to these works, we go much deeper into the behavior and performance of users accessing YouTube from mobile devices. In addition, we highlight problems caused by the YouTube infrastructure when delivering videos to mobile devices.

8. CONCLUSIONS

Considering a large and heterogeneous data set of YouTube traces, we have presented our findings about user behavior when watching videos and how the type of user device and infrastructure influence the performance of the playback.

Interestingly, users access YouTube in a very similar manner, independent of their location, the device they use, and the access network that connects them. In addition, they typically watch only a small portion of the video, and stick to default player configurations.

While YouTube guarantees very good playback quality by means of aggressive buffering policies, we pinpointed sources of unnecessary data transfer, and the potential for future performance optimization. For example, a less aggressive buffering mechanism could be used to limit the amount of unnecessary traffic transferred to users' devices but never played back due to early abort by the users. For mobile devices, besides the adoption of the prefetching scheme that is useful to speed-up the video playback, a more precise control of the buffering is essential to avoid duplicate transmission of data. Finally, CDN caching schemes can be improved by leveraging the fact that only a fraction of videos are actually watched by users.

As future work, we aim at extending our conclusions by including traces collected from 3G/4G networks to precisely quantify the impact of YouTube traffic in such environments. Indeed, the issues we highlighted observing wireline access networks could be even more critical for 3G/4G wireless access networks, where bandwidth is precious. Similarly, an interesting direction of future research is to extend our analysis to consider other popular video streaming content distributors.

Acknowledgments

This work was funded in part by NSF Cybertrust 0715833 and by the European Union through the FIGARO project (FP7-ICT-258378). We would like to thank the ISPs and the IT Department of both campuses for their support, and also the shepherd and the anonymous reviewers for their constructive comments.

9. REFERENCES

[1] V. K. Adhikari, S. Jain, and Z.-L. Zhang. YouTube Traffic Dynamics and its Interplay with a Tier-1 ISP: an ISP Perspective. In *IMC '10: Proceedings of the 10th ACM Internet Measurement Conference*, pages 431–443, Melbourne, Australia, 2010.

[2] S. Alcock and R. Nelson. Application Flow Control in YouTube Video Streams. *SIGCOMM Computer Communication Review*, 41:24–30, April 2011.

[3] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon. I Tube, You Tube, Everybody Tubes: Analyzing The World's Largest User Generated Content Video System. In *IMC '07: Proceedings of the 7th ACM Internet Measurement Conference*, pages 1–14, San Diego, California, USA, 2007.

[4] X. Cheng, C. Dale, and J. Liu. Statistics and Social Network of YouTube Videos. In *IWQoS '08: 16th International Workshop on Quality of Service*, pages 229–238, Enschede, The Netherlands, 2008.

[5] A. Finamore, M. Mellia, M. Meo, M. M. Munafò, and D. Rossi. Experiences of Internet Traffic Monitoring with Tstat. *IEEE Network*, 25(3):8–14, 2011.

[6] A. Gember, A. Anand, and A. Akella. A Comparative Study of Handheld and Non-Handheld Traffic in Campus Wi-Fi Networks. In *PAM '11: Proceedings of the 12th International Conference on Passive and Active Measurement*, pages 173–183, Atlanta, Georgia, USA, 2011.

[7] P. Gill, M. Arlitt, Z. Li, and A. Mahanti. YouTube Traffic Characterization: A View From The Edge. In *IMC '07: Proceedings of the 7th ACM Internet Measurement Conference*, pages 15–28, San Diego, California, USA, 2007.

[8] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian. Internet Inter-Domain Traffic. In *SIGCOMM '10: Proceedings of the ACM Special Interest Group on Data Communication*, pages 75–86, New Delhi, India, 2010.

[9] G. Maier, F. Schneider, and A. Feldmann. A First Look at Mobile Hand-Held Device Traffic. In *PAM '10: Proceedings of the 11th International Conference on Passive and Active Measurement*, pages 161–170, Zurich, Switzerland, 2010.

[10] The Mobile Internet Report. `http://www.morganstanley.com/institutional/techresearch/mobile_internet_report122009.html`.

[11] M. Saxena, U. Sharan, and S. Fahmy. Analyzing Video Services in Web 2.0: a Global Perspective. In *NOSSDAV '08: Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, pages 39–44, Braunschweig, Germany, 2008.

[12] M. Z. Shafiq, L. Ji, A. X. Liu, and J. Wang. Characterizing and Modeling Internet Traffic Dynamics of Cellular Devices. In *SIGMETRICS '11: Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems*, pages 305–316, San Jose, California, USA, 2011.

[13] R. Torres, A. Finamore, J. R. Kim, M. Mellia, M. M. Munafò, and S. G. Rao. Dissecting Video Server Selection Strategies in the YouTube CDN. In *ICDCS '11: Proceedings of the 31th IEEE International Conference on Distributed Computing Systems*, pages 248–257, Minneapolis, Minnesota, USA, 2011.

[14] Tstat Home Page. `http://tstat.polito.it`.

[15] YouTube Blog: Mmm mmm good - YouTube videos now served in WebM. `http://youtube-global.blogspot.com/2011/04/mmm-mmm-good-youtube-videos-now-served.html`.

[16] YouTube Press Room, `www.youtube.com/t/press_statistics`.

[17] M. Zink, K. Suh, Y. Gu, and J. Kurose. Characteristics of YouTube Network Traffic at a Campus Network - Measurements, Models, and Implications. *Computer Networks*, 53(4):501–514, 2009.

Summary Review Documentation for

“YouTube Everywhere: Impact of Device and Infrastructure Synergies on User Experience”

Authors: A. Finamore, M. Mellia, M. Munafò, R. Torres, S. G. Rao

Reviewer #1

Strengths: Well written and clear paper that answers most of the questions in each areas covered: video content characterization, user behavior and performance.
Brings attention to a critical point about current waste in bytes downloaded.

Weaknesses: Three additional points are not covered:
- Behavior on cellular networks
- Differences between devices/clients within a category (e.g. iPhone vs. Android)
- How is this different from other content providers (e.g. daily motion)?

Comments to Authors: Great paper! Not only is the paper doing a good job at characterizing YouTube traffic, I am particularly pleased to see that attention is brought to the wasted network resources caused by applications.

As mentioned in the weaknesses, your paper would be stronger, if you could get your hands on traces from a cellular network to do a more comprehensive study. A comparison with other similar content provider would be interesting, but this would be another paper.

However the point about comparison with different devices or clients within a category could be covered. You mention at the beginning are typically similar. But are there any differences? For instance, regarding the duplicate range request that leads to waste that might be due to a video buffer size. Do we see the same issue for Apple and Android smart phones? You might see differences there and it would be critical to point it out.

Figure 2, flow size: you are saying that PC vs. smartphone is not an apple to apple comparison because smart phones use range requests and the video will be split into multiple TCP flows. What about you stitch together each video and then compare the video sessions size in bytes?

Resolution switch: you mention users that are not aware of the feature. I think this is just another example of inertia, people always use the default settings (similar to the opt in / opt out results for organ donations).

Resolution switch: you claim that full screen mode is the main cause of resolution switch. You have not proved that to the reader. Is this your assumption?

Fraction of video watched: you should also highlight that once a video is fully downloaded, you assume that it is fully watched, right? Because at that point, you have no way of knowing what the user does. If that is the case, your estimates of wasted bytes are a lower bound. To be fair, you should point out that you cannot bring the number of wasted bytes down to zero because you still have to do buffer some of the data.

Video wasted data with duplicate content: FWIW I have personally noticed the same behavior on apple devices on cellular networks in the US across many providers (not just YouTube). You might want to make this clearer in Section 5.5 (test it by yourself), that this is not limited to YouTube.

Reviewer #2

Strengths: The paper does a nice study of YouTube experience on mobile and PC-based devices. The effort collects data from multiple vantage points, observes a significant number of flows, and makes some interesting observations with the data.

Weaknesses: Some of the conclusions appear more conjectures than reality and are not validated.

Comments to Authors: I thought the effort to characterize YouTube performance over different platforms is quite commendable. The authors did a very nice job of exploring different aspects of user behavior when watching these videos. Some of the observations, e.g., wasted bytes, use of default player configs, etc. are quite nice to learn about.

However, there were also a few observations that seem unclear:
- For instance, the authors point out that mobile devices use more flows than PC-based devices, and this might lead to poor performance at NATs or other stateful firewalls. Is this really true? Is the bulk on mobile device traffic video, and do such divides really get impacted?
- The authors claim that people from different locations (US and Europe) and with diverse Internet access speeds like to watch the same type of video, i.e., short videos. I wonder what is the real cause and effect. Is it not likely that most of the videos on YouTube are short videos? I would be careful about the conclusions being drawn from the data.
- The authors point out that students prefer cheaper phones, e.g., Android or Windows based. Are they really much cheaper than the iOS phones? Is this validated well?

Despite these issues, I thought the paper does make some very nice observations about video consumption on different platforms. The issue of resolution switch and the high download of bytes is interesting and I would encourage the authors to communicate this to the device vendors or operators involved. If this happens in

cellular networks, it is really an unfortunate waste of resources and should be fixed.
Overall, a nice paper and an interesting read.

Reviewer #3

Strengths: The authors analyze five very interesting and appropriate data sets. They identify interesting properties of YouTube delivery on the two studied platforms. More importantly, to me, they show that today's YouTube content delivery mechanisms are wasteful since in a large number of cases they lead to the download of more data than the video size itself. And this waste is more amplified for the mobile player.

Weaknesses: I would have loved to see similar analysis on mobile platforms - the authors only have a small experiment on this. Other than that, I think the paper makes a nice contribution.

Comments to Authors: There has been a tremendous amount of work on the characterization of the YouTube service. As such this paper falls in a rather well-published area in the measurement community. However, what this paper does, which has not been done in the past, is to study how the delivery of the YouTube service differs depending on the client platform used. And the authors clearly demonstrate that the YouTube application on mobile or set top box platforms, and the PC equivalent actually can lead to some differences in terms of performance.

The findings from the characterization study are not that impressive, I would say. Yes, the content consumed appears to be similar, with the exception of the resolution. But, interestingly to me, the authors clearly demonstrate that the performance DOES depend on the platform itself. And of course the mobile player needs to be able to deal with far more variability in the transmission medium than its PC equivalent.

Even more interestingly, it appears that the selected content delivery mechanism on mobile player can introduce significant waste, in some cases leading to the download of twice as much content as the video size itself!!!! That to me is an important aspect of YouTube that certainly deserves more attention especially when we target networks scarce in resources, like wireless networks.

The one thing I would have liked to see is a similar dataset collected on 3G/4G phones, but I think the authors have done enough. And they do have a small experiment on this aspect anyway.

Finally, I really liked the latency impact studied in the last subsection of 6.

All in all, I find that this work makes a significant contribution and I do hope that the authors find a way to reach YouTube and communicate their findings on the behavior of Mobile Player.

More detailed comments:
- Section 5.2 - the authors have not really proved that the main cause for resolution switch is full screen. At least not in the first paragraph after Figure 9. You need to study it in a little more detail before you can reach such a conclusion.
- In Section 5.6 you introduce y and y'. I would have liked to see the results on this section based on those metrics y and y' for a more precise specification of the quantity under study. For instance, is the waste of traffic in Fig 15 the sum of y'-y?

Reviewer #4

Strengths: Interesting comparison of PC vs mobile-device characteristics on Youtube across a number of dimensions. The paper also exposes how mobile devices are handled by YouTube.

Weaknesses: Unclear whether all mobile devices/applications behave the same. Some of the main observations claimed by the authors are probably misguided. The paper presentation and writing could be better.

Comments to Authors: Overall, I believe that this is an interesting paper that provides evidence on the differences of how desktop versus mobile devices are handled by YouTube, which is a significant fraction of traffic.

The paper highlights that a significant fraction of downloaded bytes may be wasted since many views are aborted before the end of the video. Further, the paper shows that mobile devices suffer higher latencies mainly due to redirection, although to me it is unclear why this is happening - the authors argue that this is due to the popularity of mobile videos but this is never truly examined in the paper and does not appear as a convincing argument.

There are a number of things that remain unclear in my opinion:

- You group all mobile devices together, but you never really examine that all mobile applications or clients do behave in a similar fashion. You claim in several places within the paper that indeed this is the case, but this is not truly verified but rather appears as author intuition.

- Why is the claim that users across institutions and devices do watch similar type of videos significant? At the aggregation level that you study the service, this simply reflects the distribution of video durations and sizes in the YouTube service. It is not a user-specific property as it is clearly shown by your results. Hence, I find Section 4 of limited interest.

There are several problems with the paper writing throughout the paper, especially grammatical and syntax mistakes.

In Section 2, you mention that you observe "no differences" regarding the second phase. Differences with respect to what?

Section 4.1: There is something wrong with the text here. Figures do not show differences but the text claims "slightly large difference", unless I missed something.

Section 4.3: I find the arguments that the network might not be able to handle better encodings a bit far-fetched. Please provide concrete evidence, or tone down the arguments.

Section 5.3: What about stop events without moving to another web page? Can you observe these?

Section 5.5: I could not really follow your discussion here towards the end of the second paragraph. It needs re-writing.

Section 5.6: Without concrete evidence on 3G performance it is hard to follow the argument that performance is the same. Since you don't have 3G data I would avoid providing vague arguments without concrete numbers on what the experiments and results were.

Section 6.1: How was the RTT measured?

Reviewer #5

Strengths: - There are a couple of interesting observations in this paper (access of YouTube is independent from the type of device and access technology, the study of flow control/adaptation mechanism of YouTube for different devices, the observation that bulk data is delivered in vain to the end-users). Despite the fact that there are a couple of anecdotal evidences it is nice to monitor how YouTube works in a systematic fashion.
- Diverse sets of devices, access technologies and locations of users and solid measurement methodology.
- A large number of downloads was considered.

Weaknesses: I like the analysis of the datasets, but there are a few aspects that have to be considered (please check my comments to the authors).

Comments to Authors: In the wireless setting, both the bandwidth and the energy are scarce resources. You may want to add a few references and comment on this in your study. Here are a few references that might be of your interest:

- "Energy consumption of mobile YouTube: quantitative measurement and analysis, in Proceedings of the Second International Conference on Next Generation Mobile Applications, Services and Technologies" NGMAST '08.
- "Cellular Data Network Infrastructure Characterization and Implication on Mobile Content Placement" Proc. SIGMETRICS 2011.

There are a couple of measurement studies that unveil how YouTube delivers bulk data to end-users; in the beginning there is a burst and then the bandwidth in throttled. For example see this paper:

- "Application Flow Control in YouTube Video Streams", ACM CCR April 2011.

You may want to comment on these results and elaborate why you observe that the amount of traffic downloaded by clients that is not used by them is significant. Is this due to the fact that the download speed is higher than the watch speed, or is it due to the fact that the mechanism cannot accurately estimate (overestimate) the minimum bandwidth that is needed so that the end-users will not complain about the quality of the video.

Did you observe special cases where the application flow mechanism of YouTube is not accurate? Does this happen for popular or unpopular content? Is this related to the proximity of the YouTube server to end-user?

You should comment on the time-of-day effect in your datasets. Do you observe that users stop the session faster during the peak hours than during non-peak hours? How does the flow control behave during the peak hours and the non-peak hours?

The related work is not very well written and should go over a few more iterations.

Response from the Authors

We would like to deeply thank the reviewers for their valuable comments. In the final version of the paper, we have integrated most of the reviewers' suggestions. In the following, we highlight the major changes.

We updated some figures adding the y-label as to better express the reported measurements. We also clarified some statements in our paper, rewrote sentences and smoothed claims by making them more conjecture-like as suggested by the second reviewer. In some cases, as for the full screen resolution switch, we provided further numbers to support our conclusions.

A common suggestion was to investigate further the impact of different mobile operating systems and to verify if the problem related to the amount of wasted data downloaded is similar considering different devices. In the original data sets, we are not able to distinguish the O.S. or the device used. This does not allow us to precisely quantify the differences. However, we performed active experiments, as reported in Sec. 5.6, with both Android and iOS devices, and concluded that they exhibit similar behavior and suffer from similar deficiencies.

All reviewers suggested improvements to the paper by including measurements collected from 3G/4G cellular networks. We agree with this requirement and we are actively pursuing the issue with mobile operators as possible partners but unfortunately we do not have such data set for now. However, in the final version of the paper we have strengthened the results by performing other active measurements using mobile phones. Findings confirm the trends already observed in the paper.

Finally, we have added a paragraph to the conclusion section, discussing some of the interesting avenues for future work. In particular, besides the analysis of 3G/4G cellular networks, we have left the assessment of the impact on our results if we consider measurements of different devices, time-of-day effects and the comparison with other streaming services as future analysis.