Measurement and Analysis of an Internet Streaming Service to Mobile Devices

Yao Liu[1] Fei Li[1] Lei Guo[2] Bo Shen[3] Songqing Chen[1] Yingjie Lan[4]

[1]Dept. of Computer Science
George Mason University
{yliud, lifei, sqchen}@cs.gmu.edu

[2]Dept. of CSE
Ohio State University
lguo@cse.ohio-state.edu

[3]Vuclip
XinLab, Inc.
bshen@vuclip.com

[4]Guanghua School
Peking University
ylan@gsm.pku.edu.cn

Abstract—Receiving Internet streaming services on various mobile devices is getting increasingly popular, and cloud platforms have also been gradually employed for delivering streaming services to mobile devices. While a number of studies have been conducted at the client side to understand and characterize Internet mobile streaming delivery, little is known about the server side, particularly for the recent cloud-based Internet mobile streaming delivery.

In this work, we aim to investigate the Internet mobile streaming service at the server side. For this purpose, we have collected a four-month server-side log on the cloud (with 1,002 TB delivered video traffic) from a top Internet mobile streaming service provider serving worldwide mobile users. Through trace analysis, we find that (1) a major challenge for providing Internet mobile streaming services is rooted from the mobile device hardware and software heterogeneity. In this workload, we find over 3,400 different hardware models with more than 100 different screen resolutions running 14 different mobile OS and 3 audio codecs and 4 video codecs. (2) To deal with the device heterogeneity, CPU-intensive transcoding is used on the cloud to customize the video to the appropriate versions at runtime for different devices. A video clip could be transcoded into more than 40 different versions in order to serve requests from different devices. (3) Compared to videos in traditional Internet streaming, mobile streaming videos are typically of much smaller size (a median of 1.68 MBytes) and shorter duration (a median of 2.7 minutes). Furthermore, the daily mobile user accesses are more skewed following a Zipf-like distribution but users' interests also quickly shift. Considering the huge demand of CPU cycles for online transcoding, we further examine server-side caching in order to reduce the total CPU cycle demand from the cloud. We show that a policy considering different versions of a video altogether outperforms other intuitive ones when the cache size is limited.

Index Terms—Internet Mobile Streaming, Heterogeneity, Popularity, Transcoding

1 Introduction

Recently, mobile devices are getting increasing popularity. For example, according to Canalys, the total number of smartphones sold worldwide in 2011 is 487.7 million [1], which is a 62.7% increase from the previous year (299.7 million). By August 2012, 116.5 million people in the U.S. owned smartphones [2].

Besides general web surfing on the Internet, these days more and more accesses from mobile devices are directed to all kinds of Internet streaming services. For example, YouTube [3] is the among the earliest to provide streaming services to mobile devices such as iPhone. Today both iOS and Android have native support for YouTube. Other popular streaming service providers, including Netflix [4] and Hulu [5], also provide streaming services to subscribed mobile users via APPs built in various mobile operating systems. Placeshifting services like Orb [6] and AirVideo [7] allow mobile users to access media content stored on their home computers. Qik [8] allows users to upload from mobile devices and then broadcast the video content to their friends. Different from the above services, Vuclip [9] lets users search and play all kinds of Internet videos on their mobile devices regardless of their mobile device types.

To understand the key challenges of Internet mobile streaming and the difference from traditional Internet streaming, a number of studies have been performed. As today the majority of Internet mobile streaming services are delivered in a client-server architecture, many studies have focused on the resource consumption and streaming quality received on the mobile device. For example, Xiao et al. [10] studied energy consumption when watching YouTube on mobile devices. Huang et al. [11] investigated fetching policies of different mobile video players, and Finamore et al. [12] examined the potential causes for inferior streaming quality of mobile YouTube accesses.

However, these studies mainly concern about the client side by examining specific devices [10], [11] or via local experiments [12]. As the key to the current Internet mobile streaming delivery services, the server side plays a critical role in the entire streaming delivery process. Unfortunately, so far, little is known about the server side, possibly due to the limited availability of data from the server side. This is particularly true for the modern cloud-based Internet streaming delivery services.

To provide in-depth understanding of the current cloud-based Internet mobile streaming services, in this study, we set to investigate the server side in streaming delivery to mobile devices. For this purpose, we have analyzed a 4-month (from Nov. 2010 to Feb. 2011) server-side workload collected from a top Internet mobile streaming service provider. In this workload, there are about 480 million video sessions with about 1,002 Terabyte video traffic delivered. Through our analysis, we have a number of findings. While the details are presented later in the paper, some highlights are as follows:

- A unique challenge for Internet streaming delivery to mobile devices is rooted from the fact that mobile devices are very heterogeneous. In this workload, we find over 3,400 different hardware models with 109 different screen resolutions running 14 different mobile OS and 3 audio codecs and 4 video codecs. This greatly challenges the traditional Internet streaming delivery infrastructure where the bottleneck often lies in the limited bandwidth.
- To deal with the device heterogeneity, runtime CPU-intensive transcoding is used by the cloud to customize a video to the appropriate versions on the fly for different devices. A video clip could be transcoded into more than 40 different versions in order to serve requests from different devices.
- Compared to videos in traditional Internet streaming, mobile streaming video clips are typically of much smaller size (with a median of 1.68 MBytes) and the video duration is shorter as well (with a median of 2.7 minutes). Furthermore, the daily mobile user accesses are more skewed following a Zipf-like distribution but users' interests also shift quickly, resulting in a stretched-exponential distribution in the long term.

To reduce the huge CPU cycles demanded for transcoding on the fly, we further explore caching on the cloud by trading off storage for CPU cycles. Our study shows that a policy that considers different versions of a video altogether outperforms other intuitive ones (e.g., a file based one) when the cache size is limited. As far as we know, we are among the first to provide a server-side analysis on a Vuclip-like Internet mobile streaming service. Our findings provide new insights and lay some foundations to improve the current Internet mobile streaming delivery.

The rest of the paper is organized as follows. We describe some background and the workload overview in section 2 and study the device hardware and software heterogeneity in section 3. We examine various mobile video properties in section 4 and study the access pattern across different ISPs in section 5. We further explore the trade-off between the storage and the CPU at the server side in section 6. Some related work is described in section 7 and we make concluding remarks in section 8.

2 Background and Workload Overview

To investigate how current cloud-based Internet streaming services are delivered to mobile devices, we have collected server-side log from private cloud rented by one of the largest Internet mobile streaming service providers, Vuclip [9]. Vuclip provides mobile users with the search-and-delivery services. It allows users to search for and watch any videos on any video-enabled mobile phones and devices.

Different from many existing services that only provide streaming services to specific mobile devices, with the powerful cloud platform, Vuclip can serve any type of mobile devices that are capable of streaming playback. Vuclip allows any mobile user to search for interested video available on the Internet, and transcodes them on-demand and on-the-fly based on the type of the mobile device using the cloud CPU cycles. As like most cloud-based services, Vuclip uses a client-server architecture (thus *cloud* and *server* are exchangeable in this paper). To serve different types of mobile devices, Vuclip employs on-demand transcoding on the cloud. Transcoding is a process to convert the requested video clip to the appropriate codecs, format, and size at runtime upon a request so that the video can be properly rendered and played on the requesting mobile device. With elastic cloud resources, Vuclip transcodes a video into different versions by choosing the best audio/video codecs, frame size, frame rate, and quality level combination for the mobile device. According to our analysis, each video was accessed in more than 2 versions on average (as shown in Table 1 and Table 2), and the most popular video was accessed in 44 different transcoded versions (as shown later in Figure 8).

To deliver video content, Vuclip uses the traditional client/server (C/S) architecture. The video file is delivered via pseudo streaming over HTTP. That is, when the requested content is available on the server, the client would issue an HTTP GET request to download the content. A video may be downloaded via several HTTP GET requests with different partial ranges specified (i.e., range requests). To differentiate video requests from HTTP requests, we define a *request* as a single HTTP transfer between the client and the server, and a *session* as the set of requests that are involved in downloading an entire video clip.

The 1-month workload we collected is from Nov. 1st to Nov. 30th, 2010. In this log, there are about 105 million sessions watching more than 4 million different videos. There are a total of about 192 million HTTP requests. The total traffic delivered from the server in these 30

TABLE 1
Summary of 1-month Workload

Workload Length	30 Days
# of Sessions	105,389,370
# of Requests	192,255,173
# of Requests from Mobile Devices	181,556,344
# of Unique Videos Accessed	4,052,740
AVG. # of Versions Per Video	2.31
MAX. # of Versions Per Video	41
Total Traffic Volume	212 TB

days is about 212 TB. Table 1 gives a summary of this workload. Note that among all these requests, some are from desktop/laptop computers instead of mobile devices. In order to focus on the requests from mobile users, we differentiate them in the server log through the User-Agent strings specified in each HTTP request. By analyzing the User-Agent, we find there is a total of 150,072 unique User-Agent strings. Among them, 84,281 (56%) represent mobile devices. However, examining the received requests, we find most of them come from User-Agent strings representing mobile users: more than 94% (181 million out of 192 million) requests are from mobile devices.

With the exclusion of desktop/laptop traffic, Figure 1 gives an overview of the server side traffic in 30 days. Note that the left y-axis represents the total number of requests per day, while the right y-axis represents the total traffic volume per day. During this 1-month period, despite a small decrease in the middle, the number of the requests and the delivered traffic amount kept increasing, indicating the popularity of Vuclip.

Figure 2 shows the hourly mobile streaming access patterns in a day. The figure indicates that hourly accesses peak around 17:00 GMT. Furthermore, the total number of requests and the traffic volume served during peak hours almost double these in non-peak hours. Figure 3 further depicts the hourly pattern from Nov. 8th to Nov. 15th (a week). The figure shows clear peak and off-peak hourly patterns for each day. The figure shows some drop after Nov. 12th. It is likely due to the fact that Nov. 13th was a Saturday and Nov. 14th was a Sunday. We can observe the increase of accesses again on Monday.

TABLE 2
Summary of 4-month Workload

Workload Length	120 Days
# of Sessions	480,905,010
# of HTTP Requests	982,241,100
# of Unique Videos Accessed	10,779,818
# of Unique Files (Versions) Accessed	25,728,606
# of Formats Transcoded	50
AVG. # of Versions Per Video	2.39
MAX. # of Versions Per Video	44
Total Traffic Volume	1,002 TB

After our paper based on 1-month trace is initially published at [13], we are able to further collect 3 more months trace from the Vuclip site. Table 2 gives a summary of the workload. In this 4-month log, from Nov. 2010 to Feb. 2011, there are more than 480 million sessions accessing more than 10 million unique videos. More than 982 million HTTP requests were served by the server. A total of 1,002 TB traffic was delivered from the server. To support heterogeneous mobile devices, videos are served in different versions. In the trace, we observed 50 different formats, and each video is transcoded into 2.39 versions on average. As a result, more than 25 million unique files (versions) were requested during the 4-month period.

Figure 4 shows the accesses during the 4-month period. In this figure, the x-axis represents the 120 days in our trace, the left y-axis represents the total traffic delivered from the server to the mobile devices over each hour, and the right y-axis represents the number of HTTP requests received every hour. Besides a clear daily pattern, we also find that during these 4 months, the number of requests received by the server has been increasing steadily, and so does the traffic delivered from the server. This indicates that Internet mobile video services are becoming more and more popular, and a study of the mobile video access pattern is in urgent need.

3 Characterization of Mobile Device Heterogeneity

3.1 Mobile System Heterogeneity

To provide Internet streaming services to all kinds of mobile devices like Vuclip, a unique challenge is the heterogeneity among mobile devices. Different from the pre-coding approach that was taken by many other service providers to serve specific types of mobile devices, Vuclip employs transcoding, a technique that can customize the video into a proper format for the requesting mobile device at runtime. Although transcoding is very flexible and desirable to serve heterogeneous mobile devices, transcoding demands huge CPU cycles on the fly, where the support with cloud resources is critical.

To get a realistic picture of the mobile device heterogeneity, we retrieve detailed device information from WURFL [14] based on the User-Agents information we have extracted from the 4-month server log. Among the 229,333 User-Agents that represent mobile devices, we are able to get the brand and model information from more than 196,832 (85.8%) distinct User-Agent strings. The rest only have browser information.

As shown in Table 3, accesses to Vuclip in these 4 months came from 3465 different device models. These devices have different screen sizes that can support video playback with different resolution rates. Delving into this, we find that these devices have 109 different resolutions (width and height combinations), ranging from 84×48 to 1600×1200. Figure 5 shows the most

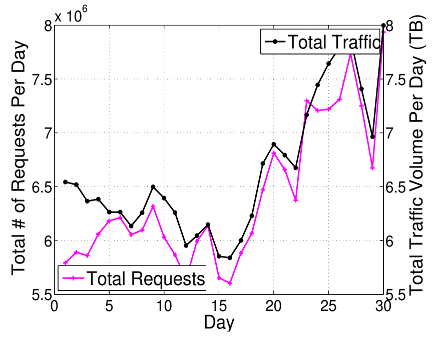


Fig. 1. Daily Accesses in Nov. 2010

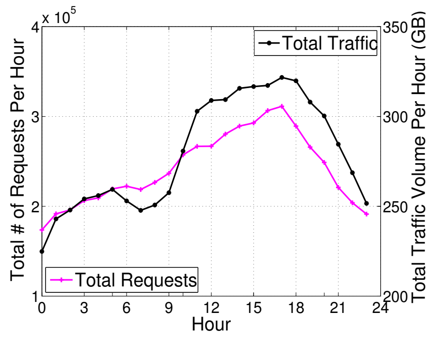


Fig. 2. Hourly Accesses On Nov. 1st 2010

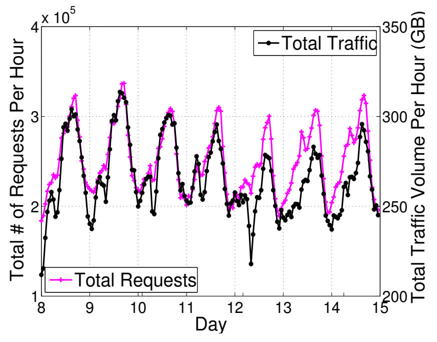


Fig. 3. Weekly Accesses from Nov. 8 to Nov. 15 2010

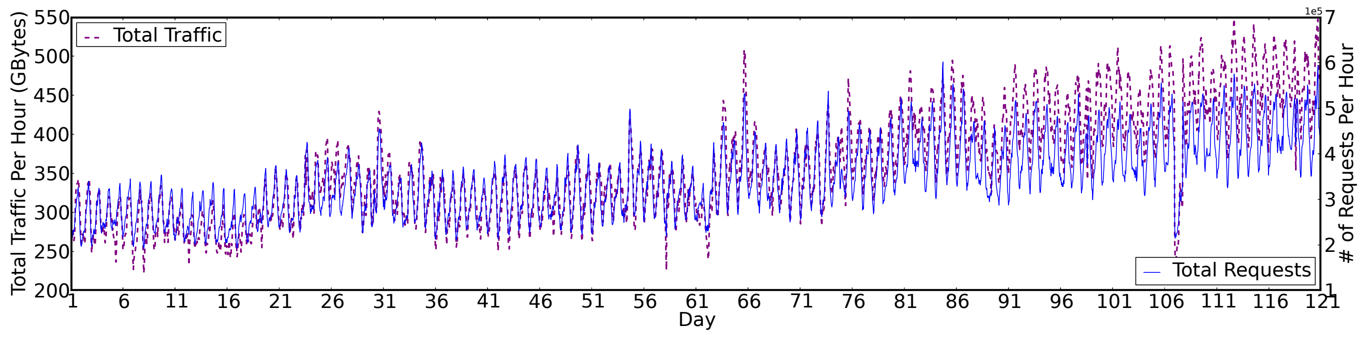


Fig. 4. Hourly Accesses Pattern over 4 Months

TABLE 3
System Heterogeneity of Mobile Devices

Models	3465
Resolution	109
Mobile OSes	14

TABLE 4
Video Codecs (4 Months)

Type	Video	Audio	# of Videos	# of Sessions
ASF	WMV	WMA	871,761	8,798,925
3GP	H.264	AAC	2,288,414	81,091,264
3GP	H.263	AMR	8,013,445	231,990,613
3GP	MPEG-4	AMR	634,753	8,157,977
3GP	MPEG-4	AAC	4,770,913	150,866,231
3GP in Total			10,666,715	472,106,085

popular resolutions, including 320×240, 480×360, and 480×320. They also run on 14 different mobile operating systems.

3.2 Audio/Video Codec Heterogeneity

To play video on a mobile device, both audio and video codecs are required. On different devices, the supported codecs may be different as well. Such heterogeneity would further increase the load for the server if the server conducts transcoding for the mobile device. Note that if such transcoding is done at the client side, it would lead to excessive battery power consumption.

To examine the codec heterogeneity, we further look into the supported audio/video codecs on these 3465 hardware models. We find that typically there are 3 audio codecs being used, namely AAC, AMR, and WMA, and there are 4 video codecs being used, namely H.263, H.264, MPEG-4, WMV. Figures 6 and 7 show the popularity of these codecs. As shown in these figures, AMR is the most popular audio codec, as more than 57.6% devices support it, and H.263 and MPEG-4 are the most popular video codecs.

With 3 audio codecs and 4 video codecs, we expect a total of 12 combinations of different audio/video codecs. In practice, however, not all these combination of the audio and video codec are used. In the workload, we only find 5 combinations. Table 4 shows the 5 video+audio encoding schemes used. For the more than 10 million (10,779,818) unique videos that were accessed in our 4-month log, Table 4 shows that H.263+AMR and MPEG4+AAC are the most popular encoding schemes, accounting for 79.6% of total viewing sessions. This is not surprising as H.263 and MPEG4 are the most widely supported video codecs on the 3465 models of mobile devices.

In addition to different codecs, video files are also encoded into two different formats, i.e, two types of containers, 3GP and ASF. 3GP is the 3GPP file format, which is a multimedia container format defined by the Third Generation Partnership Project (3GPP) for 3G UMTS multimedia services. 3GP is often used on

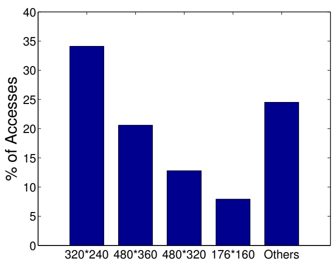


Fig. 5. Most Popular Resolutions

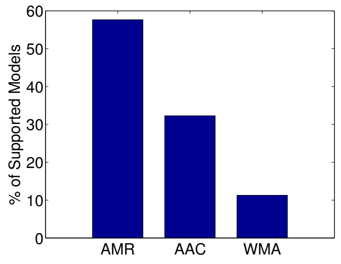


Fig. 6. Audio Codecs

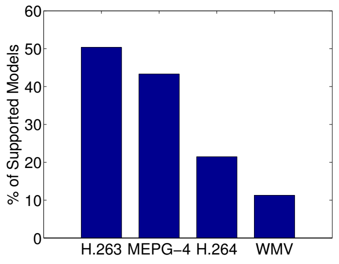


Fig. 7. Video Codecs

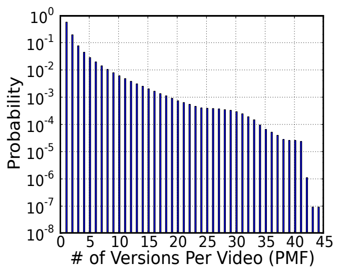


Fig. 8. # of Versions Per Video (PMF)

3G mobile phones. On the other hand, ASF (Advanced Systems/Streaming Format) belongs to Microsoft Media framework and it is a proprietary digital audio/digital video container format. Apparently, 3GP is much more widely used in practice than ASF for mobile videos.

TABLE 5
Video Resolution and Encoding Rate

Quality	Frame Width	Encoding Rate (Kbps)
Low	176	51 - 55
Low	320, 360	71 - 187
High	176	81 - 147
High	320, 360	172 - 335
WiFi	320, 360	358 - 423

Besides the above hardware and software heterogeneity, mobile devices may have different network speed, due to various reasons, such as accessing through cellular network or WiFi. To support different mobile Internet access speed, Vuclip also transcodes video clips into 3 different quality levels: Low Quality, High Quality and WiFi Quality. Table 5 shows the corresponding range of object encoding rate for different quality levels. Consider the variety of resolutions, videos are also customized into 3 different frame widths: 176, 320, and 360. As we can observe from the table that a larger resolution (width) video does not necessarily come with a high encoding rate. On the other hand, a video with a high encoding rate typically comes with a larger resolution.

TABLE 6
Video Quality (4 Months)

Quality	# of Videos	# of Sessions
Low	4,767,532	123,695,613
High	9,710,415	355,047,297
WiFi	270,967	2,162,100

Table 6 further shows the number of videos accessed between Nov. 2010 and Feb. 2011 that are of Low, High, and WiFi Quality as well as the number of their requested sessions. The videos in High Quality are mostly requested: more than 73.8% viewing sessions are for videos encoded with High Quality. Consider that Vuclip transcodes the video content on-demand, it is not surprising that 90% of video contents have at least one version encoded with High Quality. WiFi Quality, however, is the least requested quality level. This is likely due to the relatively slow mobile accessing speed and tiered data plan billing model today.

Since Vuclip transcodes the original video to accommodate mobile devices with different codecs, frame width, and quality level, for the ease of presentation, we use *versions* to refer to different transcoded video files for each video in the rest of the paper. On the other hand, we use *videos* to refer to a set of video clips that correspond to the same content. Figure 8 shows the probability mass function of the number of versions each video has. As shown in the figure, in this workload, about 57% videos have only one version, and about 3.3% videos are accessed in 10 or more versions. The largest version number is 44.

4 Characterization of Mobile Streaming Videos

The previous section has shown that mobile device heterogeneity is a great challenge to the service provider. With such a level of heterogeneity, what kind of video clips are being served is of our great interest. In this section, we further analyze the mobile video clips that we have collected from the server log in order to reveal the commons with and differences from the traditional Internet streaming content.

4.1 Video Playback Duration, File Size, and Formats

Figure 9 depicts the distribution of video playback duration in seconds. In this figure, videos that were accessed in Nov. 2010 are sorted in decreasing order of the playback duration, and the y-axis is in log scale. As shown in the figure, video clips accessed by mobile users are mostly short in terms of playback duration: more than 97% videos are less than 10 minutes long, and the median playback duration is 162 seconds (less than 3 minutes). Compared to the longer duration of traditional Internet streaming video clips, such a shorter duration makes it more feasible for mobile devices because video streaming consumes a lot of limited resources on mobile devices, including the network for data receiving,

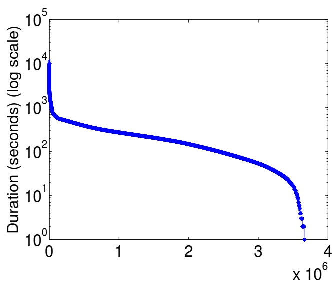


Fig. 9. Video Playback Duration Distribution (1 Month)

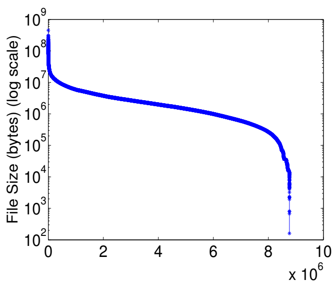


Fig. 10. Video File Size Distribution (1 Month)

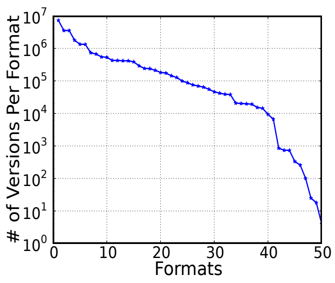


Fig. 11. # of Versions Per Format (4 Months)

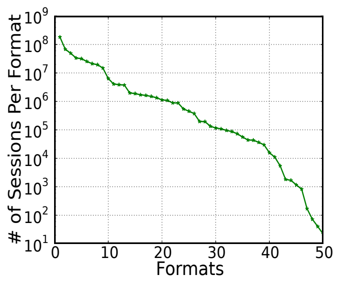


Fig. 12. # of Sessions Per Format (4 Months)

the CPU for decoding, and the display for rendering. Such resource consumption can drain the limited battery power supply at a very high rate.

Correspondingly, Figure 10 shows the file size (bytes) distribution. Again, we sort the video files (versions) based on their sizes in decreasing order. As shown in Figure 10, the video file distribution is similar to that of the duration as shown in Figure 9. Note that, here in this figure, each video may have been accessed in several versions in different formats and file sizes. As we can see, most video files accessed by mobile devices are smaller than 8 MBytes, with a mean file size of 2.78 MBytes and the median file size 1.68 MBytes. This shows that videos accessed by mobile devices are mostly small in terms of bytes. This can reduce the total network transmission for downloading the video file. Note the network interface card could consume 30% to 40% of the total battery power consumed during a streaming session to a mobile device [15], [16].

Moreover, compared to the size of the traditional Internet video files [17], [18], the size distribution we find in this server log is much smaller. This provides a great opportunity for reducing the transcoding cost as we discuss later in section 6.

We then examine how many different versions each format accounts for. Figure 11 shows the result. In this figure, the y-axis shows the total number of versions that are using each format in log scale, while the x-axis shows the different formats ranked by their corresponding number of versions. We find that some formats are significantly more popular than others. Figure 12 further shows the popularity distribution of accesses for different formats used by Vuclip. Similar to Figure 11, some formats are in orders of magnitude more popular than others. During 4 months, accesses for the Top-10 formats as shown in Table 7 account for 94% of total sessions. These results are as expected because of the different popularity of different audio/video codecs supported by different mobile devices.

4.2 Popularity of Mobile Videos

Figure 13(a) shows the popularity pattern of videos accessed site-wide on Nov. 1st. In this figure, the x-axis represents videos ranked by the number of requested sessions in decreasing order, plotted in log-scale, while the y-axis represents the number of viewing sessions of this video, also plotted in log scale. This figure shows that, in log-log scale, the popularity distribution of videos accessed can be well fitted with a Zipf-like distribution

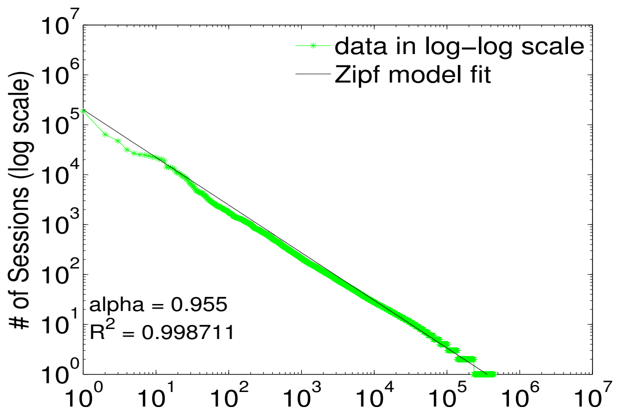
$$y_i \propto \frac{1}{i^\alpha},$$

where i is the popularity rank of the video, y_i is the number of requested sessions for the video, and α is the skewness parameter. Moreover, we find $\alpha = 0.955$ fits our data very well with the goodness of fit value R^2 very close to 1, indicating the popularity distribution is not only Zipf-like, but also very close to the Zipf's law where $\alpha = 1$. Similar patterns have been found for the other days in the workload.

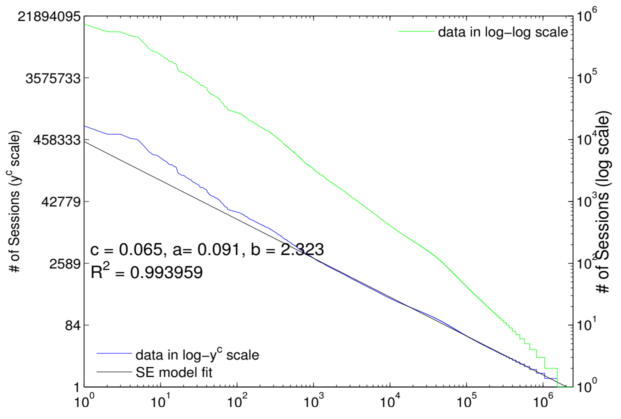
The Zipf-like distribution is known to be efficient in modeling web traffic, and is the premise for efficient web caching. Specifically, α is an indicator of request concentration, and proxy caching can be more efficient with a larger α value. For example, it was reported in [19] that α varies between 0.64 and 0.83 for web traffic, while it tends to be smaller for media traffic (for example, work [18] reports 0.56 for YouTube traffic). Different from previous measurement studies where data were collected at edge locations (e.g., one university campus), the mobile video accesses are highly concentrated at the server side as indicated by the larger α value observed. Such discrepancy is reasonable as collecting traffic at edge locations can only reflect the the access pattern of users from one specific area (e.g., one university

TABLE 7
Most Popular Formats

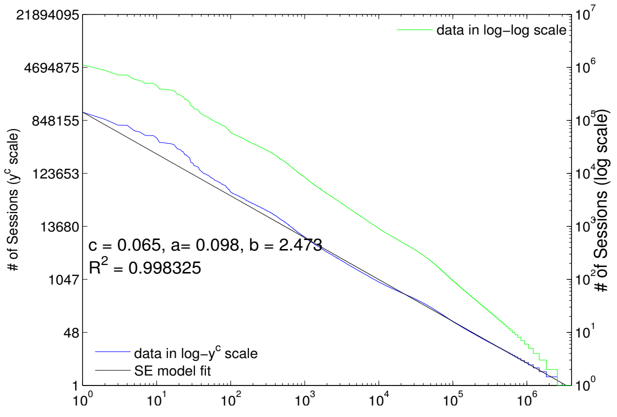
Type	Video	Audio	Frame Width	Quality
3GP	H.263	AMR	176	High
3GP	MPEG-4	AAC	320	High
3GP	H.263	AMR	176	Low
3GP	MPEG-4	AAC	480	High
3GP	H.264	AAC	480	High
3GP	H.264	AAC	480	Low
3GP	MPEG-4	AAC	320	Low
3GP	MPEG-4	AAC	480	Low
3GP	H.264	AAC	320	High
ASF	WMV	WMA	176	High



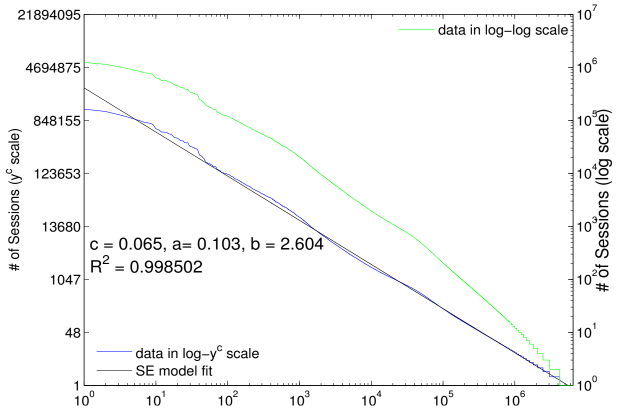
(a) One Day (Nov. 1 2010)



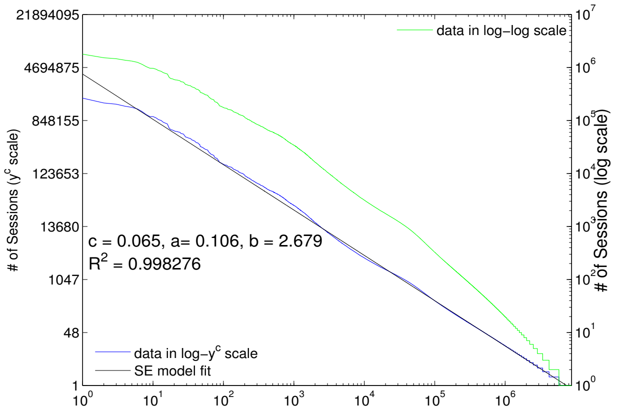
(b) Two Weeks (Nov. 1 to Nov. 14)



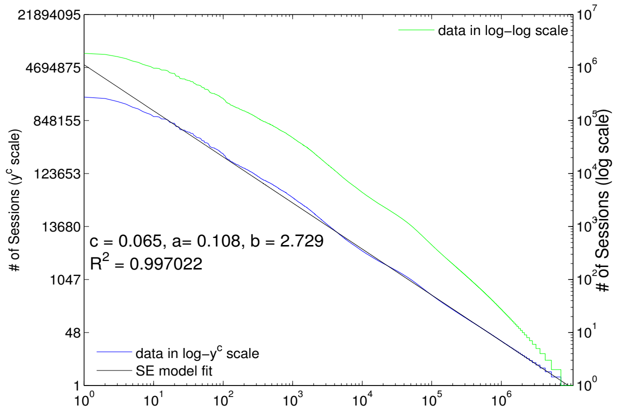
(c) One Month (Nov.)



(d) Two Months (Nov. to Dec.)



(e) Three Months (Nov. to Jan.)



(f) Four Months (Nov. to Feb.)

Fig. 13. Video Popularity Distribution

campus), while the server logs can provide a complete and global view of the video popularity. Furthermore, the more concentrated accesses also mean caching at the server side is more effective than caching at the edge/client side, if caching at the server side is needed. Note for content delivery, caching at the server side is typically not for reducing network traffic as caching at the client side.

While Figure 13(a) shows short-term (one day) popularity distribution, we further examine the distribution of popularity over longer term. Figure 13(b) to (f) show the corresponding distribution over different periods of time. In these figures, the left y-axis is in powered scale while the right y-axis is in log scale. The x-axis is in log scale as well. As shown in the figures, the video popularity over different periods (two weeks, one month, two months, three months, and four months) all deviate from a straight line in log-log scale, meaning not a Zipf-like distribution. Instead, they can roughly be fitted with a stretched exponential (SE) distribution as shown by the left y-axis in powered (by a constant c) scale [20]. With SE distribution, the rank distribution function can be expressed as

$$y_i^c = -a \log i + b.$$

An SE distribution is fit by several parameters as shown in the figure. For example, parameter c is also called the *stretch factor*, which characterizes the median file size of workload [20]. It was reported that for media workloads with a median file size $<$ 5MBytes, the stretch factor is ≤ 0.2. Our analysis confirms this with a c of 0.065 and a median file size of 1.68 MBytes. Parameter a in an SE distribution increases with the duration of workload as well as the ratio of media request rate to new content birth rate, and it causes the distribution to deviate from a straight line in log-log scale. From Figures 13(b) to (f), we find that parameter a increases from 0.091 (two weeks) to 0.108 (four months). This confirms that the parameter a in an SE distribution increases over time [20]. On the other hand, the stretch factor c remains an invariant, as the file size distribution of the workload also remains stable during this time period.

The stretched exponential distribution has been used to characterize many natural and economic phenomena, as well as the access patterns of Internet media traffic. It was shown that under an SE distribution, media caching is much less efficient than under a Zipf distribution [20]. This poses a new challenge if long-term caching is needed on the server side.

4.3 Popularity of Different Video Versions

We have shown in Figure 13(a) that the daily video popularity follows a Zipf-like distribution. However, as discussed before, each video may be accessed by very diverse mobile devices, resulting in multiple transcoded versions. We thus further examine the popularity distribution of all versions accessed on Nov. 1st where each version is counted as a distinct object. Figure 14(a) shows that when different versions are considered as different objects, the popularity cannot be well-fitted with the Zipf distribution. On the one hand, due to the increased number of video versions (2.31 versions per video on average over 1 month), the skewness factor α decreases from around 0.95 to 0.7. On the other hand, as shown in the figure, the accesses to the Top-1000 versions are much larger than what Zipf predicts, indicating significant deviation from Zipf-law.

Over medium term (e.g., one week and two weeks), we find that the popularity distribution can be well fitted with the Zipf distribution, with the goodness of fit R^2 very close 1 as shown in Figures 14(b) and (c). But in longer terms, similar to the popularity of different videos, the popularity pattern of different versions converges into SE distribution as shown in Figures 14(d), (e) and (f). This is because the popularity distribution is non-stationary over long terms, and the most popular versions cannot keep up with the same popularity as new videos and new versions join the system.

TABLE 8
Summary of 1-month Video Accesses

Average Daily New Videos	92 K
Average Daily Accesses Videos	502 K
Percentage of Daily New Videos	18%
Average Daily Accessed Videos Difference	292 K
Percentage of Average Daily Difference	58%
Average Daily Requested Sessions	3512 K
Total Accessed Videos in 30 Days	4052 K
Percentage of New Videos in 30 Days	68%

4.4 Popularity Evolution

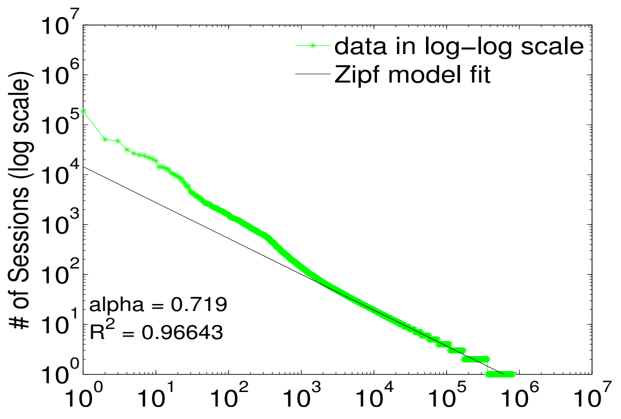
We have shown that mobile users' daily accesses for mobile videos are highly concentrated, but monthly accesses patterns are flatter. In this subsection, we further examine how video popularity changes over time, aiming to shed light on such popularity changes. We first study the commons between accesses in consecutive days, and then, more specifically, we consider the temporal locality characteristics of these accesses.

Table 8 summarizes the daily accesses and the corresponding videos that were requested in the system based on our 1-month log. According to our analysis, 292K out of 502K (58%) unique video clips accessed daily are not accessed in the previous day on average. Among these 292K video clips, about 92K are new video clips. This indicates that about 18% video clips accessed every day are new. The rest 200K (40%) are unpopular ones that were in the system, but were not frequently accessed. In total, new video clips account for about 68% of total unique video clips accessed during 30 days. Since new video clips are generated at a high rate of 18%, this confirms the implication of SE-distribution that a monthly static caching scheme may not be so efficient as a more frequently updated one.

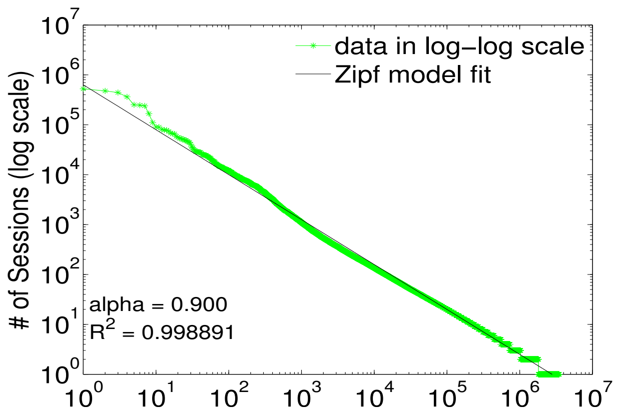
Unlike traditional video on-demand streaming systems, Vuclip has a larger repository as well as a faster new content generation rate. We next examine if temporal locality is helpful in predicting what will be popular in the future in such a highly dynamic system.

Figure 15 keeps track of the Top-100, Top-500, Top-1000, Top-2000, Top-5000, and Top-10000 videos that were accessed the most on the first day of our trace, Nov. 1st, 2010. We examine how the popularity of these top videos would evolve over these 4 months. We find that the Top-100 video list has the fastest drop-out rate, as only fewer than 20% of the videos would remain in the list after 13 days, while it takes 54 days for Top-500, 80 days for Top-1000, and 109 days for Top-2000 video lists. After 119 days, about 30% of videos on Top-5000 lists and 32% of videos Top-1000 list would still remain on such lists.

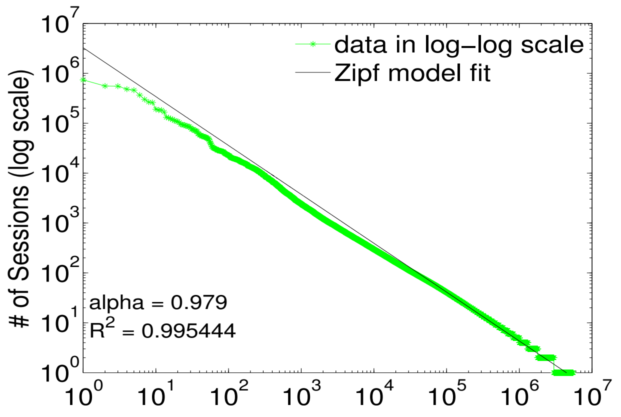
Compared to the change of popular videos, Figure 16 shows the changes of popular versions. We find that the drop-out rate for Top-100 and Top-500 version lists are similar to Figure 15. However, the Top-1000, Top-2000,



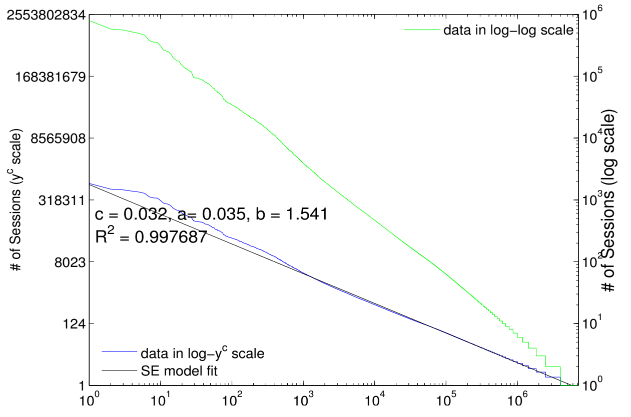
(a) One Day (Nov. 1 2010)



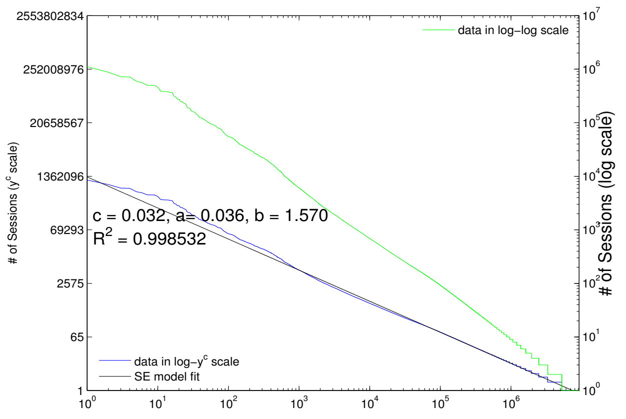
(b) One Week (Nov. 1 to Nov. 7)



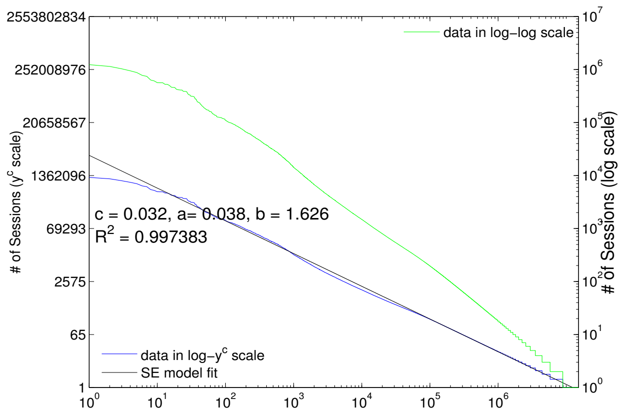
(c) Two Weeks (Nov. 1 to Nov. 14)



(d) Three Weeks (Nov. 1 to Nov. 21)



(e) One Month (Nov.)



(f) Two Months (Nov. to Dec.)

Fig. 14. Version Popularity Distribution

Top-5000, and Top-10000 version lists are seeing faster drop-out rate than that in Figure 15. As a result, only fewer than 20% videos on these lists would remain. This indicates version popularity changes more dynamically than video popularity.

Having shown how videos' popularity would change over time, we further study how the list of popular videos changes everyday. Figure 17 shows the percentage of videos in the top lists that are different from the previous day. We find that such a change rate varies between 15% and 37%, which is faster than the traditional VoD system [17]. In comparison, Figure 18 shows that the change rates of Top-100 and Top-500 version lists are similar to the top video lists, while the change rates in Top-1000, Top-2000, Top-5000, Top-10000 version lists are higher than the corresponding top video lists.

4.5 Correlation Between Popularity and Video Length

We have shown in section 4.1 that the mobile video files are often short and the video popularity in a short-term has a Zipf-like distribution. Thus one may wonder whether shorter videos get more accesses. To examine

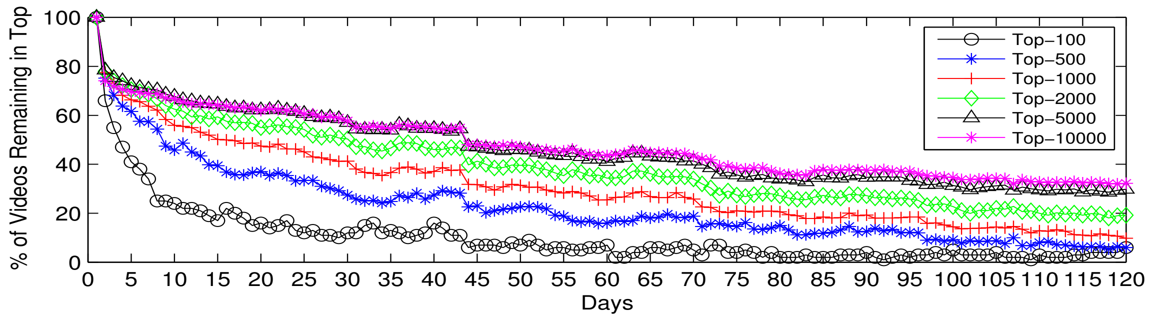


Fig. 15. % of Videos Remaining in Top

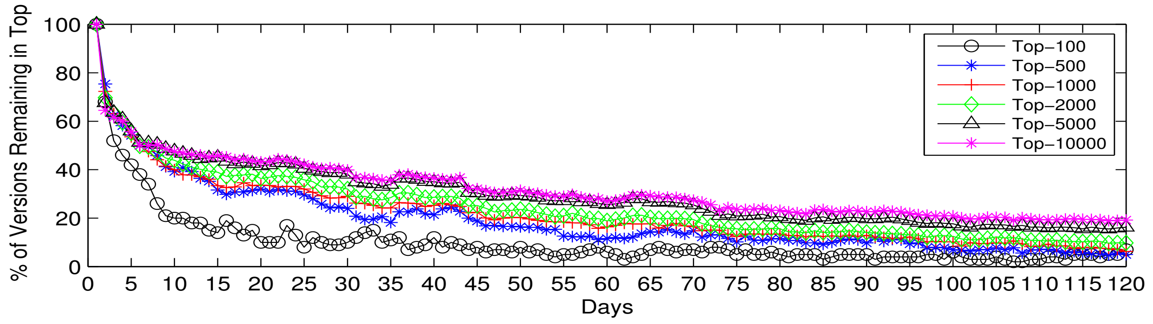


Fig. 16. % of Versions Remaining in Top

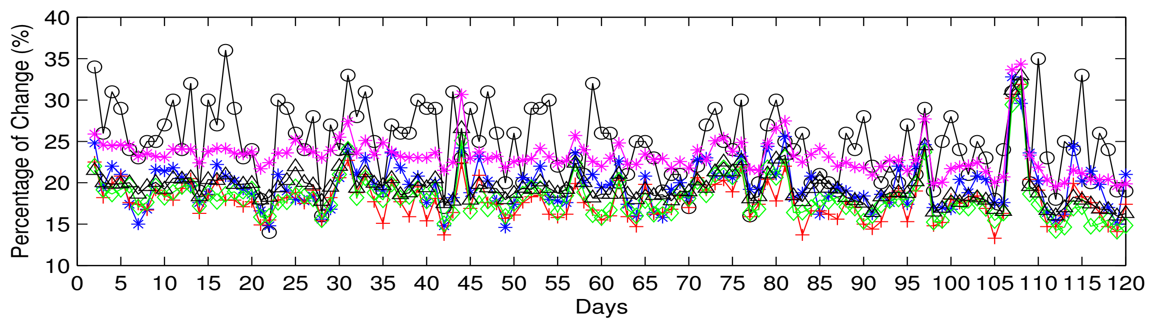


Fig. 17. % of Change in Top Video Lists

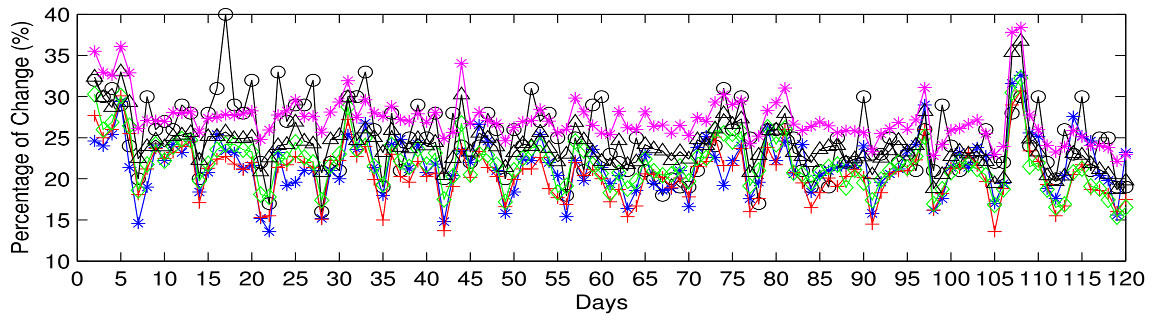


Fig. 18. % of Change in Top Version Lists

this, we group video files into 1 minute interval based on their lengths. Figure 19 shows the total number of requested sessions decreases as the video length increases, and videos shorter than 5 minutes account for about 60% total requests. We further calculate the correlation between video popularity and video length, and examine if shorter videos tend to be more popular. Our results show that the correlation coefficient is 0.006, which indicates the correlation is weak. We have also conducted similar tests based on versions, and the results are similar. This indicates the large percentage of requests for short videos are due to the large population of short video contents instead of user preferences.

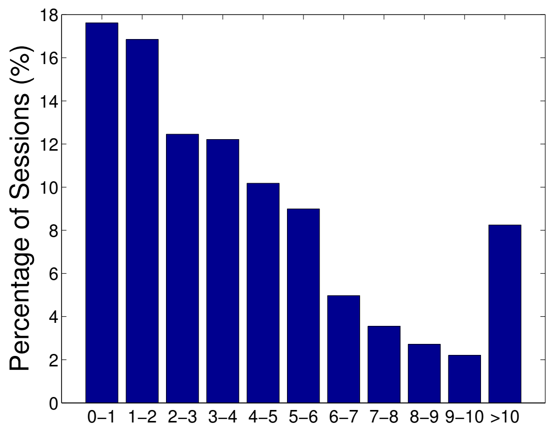


Fig. 19. % of Sessions for Videos of Different Length (minute)

TABLE 9
Average # of Versions per ISP

ISP	Versions per Video
#1	1.68
#2	2.14
#3	1.53
#4	2.08
#5	1.41
#6	1.52

5 Characterization of Accesses based on ISPs

In this section, we further examine if the accesses at ISP-level exhibit similar characteristics as observed at the server-side.

5.1 Popularity Distribution within ISP

We focus on the top six ISPs that contributed to the most traffic volume in one month. For the ease of presentation, we anonymize the identities of the ISPs, and refer to them as ISP #1 to ISP #6 ranked by the amount of their traffic. Figure 20 shows the popularity of videos accessed on Nov. 1st across six ISPs. The rest days all exhibit similar popularity pattern. Despite the fact that the total number of requests from six ISPs are different, the unique number of videos accessed are in the same order of magnitude.

It is shown that the access patterns of ISPs #3, #4, #5 and #6 are also Zipf-like, which can be well fitted with R^2 very close to 1. α values are smaller than as observed site-wide, varying between 0.66 and 0.72, which is close to traces from web proxy [19], and higher than observed from other media systems [18]. This shows that mobile video accesses are less concentrated at ISP side compared to at the server side (Section 4.2).

The access patterns of ISPs #1 and #2 cannot be well-fitted with Zipf distribution, because the top videos are accessed more frequently than the Zipf model predicts.

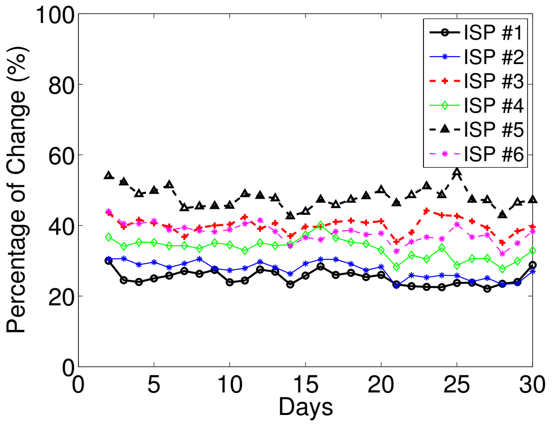


Fig. 21. % of Change in Top-1000 Videos

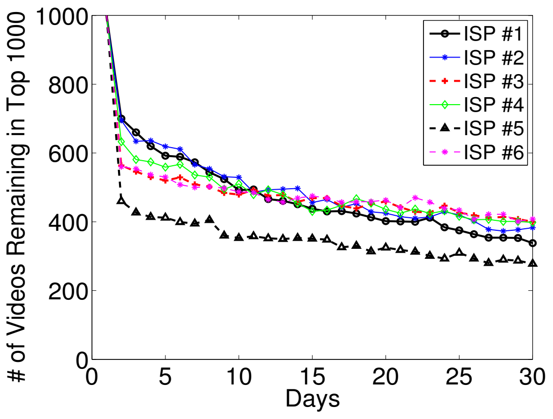


Fig. 22. # of Videos Remaining in Top-1000

As one of the concerns about serving videos to mobile users is the device heterogeneity, we further examine how users from different ISPs access the videos. Table 9 shows the average number of versions per video accessed from each ISP in one month. We find that versions accessed by each ISP is smaller compared to site-wide (2.31 versions per video as we show in Section 3). This indicates that there is less diversity within ISP compared to site-wide. As a result, the daily distribution of version popularity per ISP is very similar to the popularity of video popularity shown in Figure 20.

5.2 Popularity Evolution within ISP

Figure 21 shows the percentage of change in Top-1000 requested videos in the 6 ISPs. Compared to Figure 17, it can be seen that the shifting of user interest within ISPs (varying between 25% and 50%) is faster than site-wide (around 20%). This indicates if prefetching is employed at the ISP side, it would be less effective in some ISPs. We further examine the object hit ratio if the Top-1000 accessed videos are prefetched into the ISP's proxy. Table 10 shows the results based on our 1-month trace.

It is surprising that the effectiveness of prefetching can be largely different across the six ISPs. For example, the hit ratio is higher than 0.7 for ISP #1 and ISP #2, and

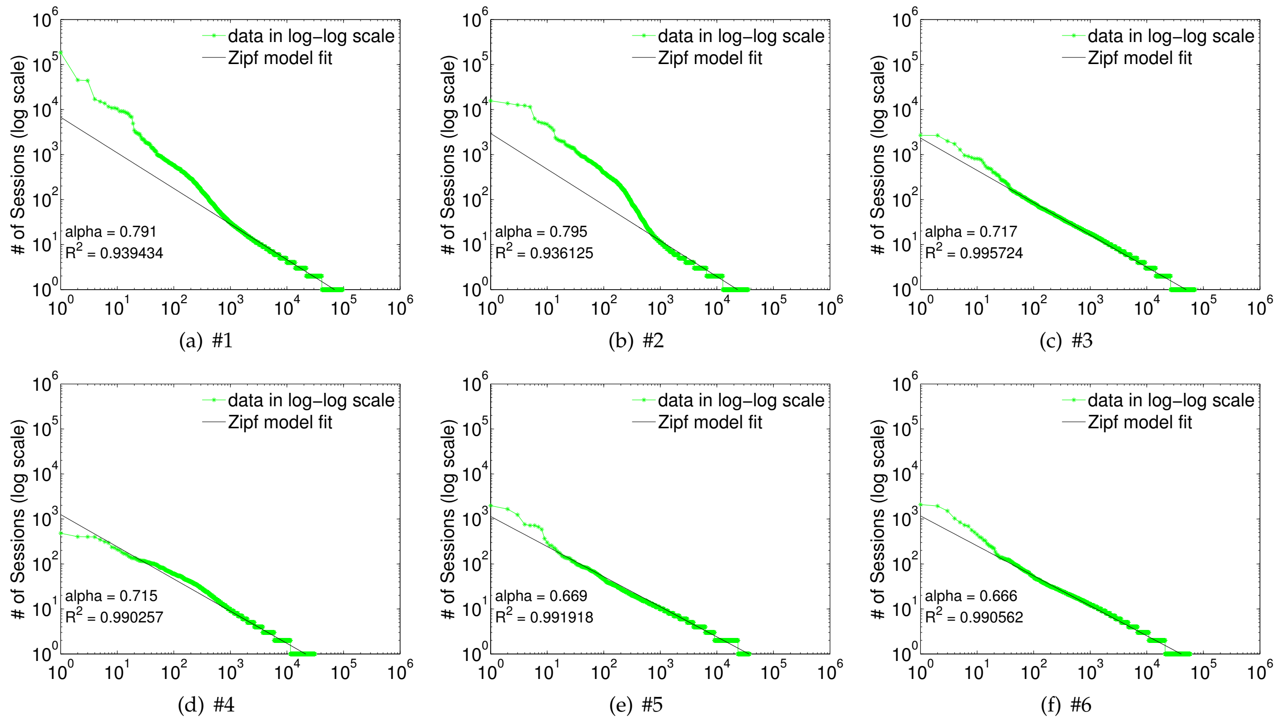


Fig. 20. Daily Video Popularity Distribution per ISP

TABLE 10
Prefetching Top-1000 Videos the Previous Day

ISP	Hit Ratio
#1	0.71
#2	0.75
#3	0.23
#4	0.33
#5	0.22
#6	0.21

could be as low as around 0.2 for ISP #3, #5, and #6. The underlying reason, as we presume, is two-fold. On one hand, the percentage of change is relatively small for ISP #1 and #2 (less than 30%), so that more videos would remain in top accessed list. On the other hand, as we have shown in Figure 10 of the TPDS manuscript, the daily access pattern of ISP #1 and #2 are much more concentrated than the other four.

However, despite the higher rate of change in top videos in ISP #3, #4, #5, and #6, the long-term temporal locality across all 6 ISPs as shown in Figure 22 are similar to that site-wide (Figure 15): around 30% to 40% videos would remain in Top-1000 list throughout the 29 days. And the number of videos remain in Top-1000 in ISP #3, #4, #5, and #6 becomes even more stable after the initial days of change, even though they have higher initial purge out rate.

It remains unclear why user interests in the four ISPs changes faster. Our conjecture is that this is related to the different presentation models and different recommendation systems different ISPs use to present the Vuclip site to the users. We will leave this for future work.

6 Trade-off between CPU and Storage

As aforementioned, in order to conduct on-demand transcoding to serve all kinds of heterogeneous mobile devices, Vuclip has to rely on a rented private cloud platform. While the cloud can provide elastic services with sufficient CPU cycles for online transcoding, it challenges service providers both technically and economically with the growing popularity of Internet mobile streaming services. This is because Vuclip will be charged more if more CPU cycles have been used. It is thus very desirable to reduce the huge demand of CPU cycles for such transcoding.

In the previous section, we have shown that the mobile users' accesses are more concentrated (skewed) than those in the traditional Internet streaming services. Thus, caching at the server side, sometimes called reverse caching, could be explored to temporarily cache some transcoded objects so that on-the-fly transcoding would not be necessary if the same type of mobile devices access the same video. Such full-object caching is possible for mobile videos because mobile video objects are typically smaller with a median of 1.68 MBytes as we showed before. Note that different from the traditional caching objectives such as web proxy caching for reducing the network traffic, caching at the server side here is to reduce the CPU cycles demanded for transcoding. That is, a trade-off between the storage and the CPU.

On the other hand, we have also shown that mobile users' interests shift quickly (Section 4.4). This provides some hints for cache replacement. Regardless whether the cache is implemented via disk and/or memory, the cache replacement policy is the key to the cache performance. Typical cache replacement strategies (such as popularity-based policies) may work, however, the complexity added by Vuclip-like services comes from the fact that a video often has multiple transcoded versions. Intuitively, these different versions could be considered as separate objects in the cache. However, if we consider video popularity, different versions of a same video are internally related. Therefore, we next explore different replacement strategies via simulations for Vuclip-like systems.

6.1 Replacement Strategies

A simple strategy is to ignore the internal relationship of different versions of a video, and consider each version as a distinct object. Under this assumption, the existing web proxy cache replacement policies can be adopted. Since we are dealing with video objects, we thus first consider a *version-popularity* based replacement policy, in which a utility function is defined as the ratio of the version access number to the storage size occupied by that version. The version with the least utility is the victim to be purged from the cache.

On the contrary, if we consider that different versions of a video are related because along the diminishing popularity of a video, all of its versions may get fewer and fewer accesses, we can also consider a policy in which all different versions of a video are bundled together as one object in the cache. Taking the popularity of this object as the sum of popularity of all its versions, we can design a *video-popularity* based replacement policy, in which the entire object (with all versions) is replaced once it is identified as the victim based on the utility function defined as the ratio of the total access number to this video to the total size of occupied storage.

With the above two strategies, naturally, one may wonder if a *hybrid* policy could perform better. That is, neither considering the version independently as the version-popularity based policy does, nor considering all versions of a video as one object as the video-popularity based policy does. In a hybrid strategy, a utility function can be defined for each video as in the *video-popularity* based replacement policy, but the victim is the least popular version of the least popular video.

6.2 Simulation Results

To study the effectiveness of these different strategies, we conduct trace-driven simulations using the collected workload to compare their performance. With the accesses of Nov. 1st, Figure 23(a) shows that when the cache size is smaller than 27% of the total size of accessed objects of that day, a video-popularity based replacement policy can achieve the highest cache hit rate, and save roughly about 55% CPU cycles, while a version-popularity based strategy performs the worst. This is likely due to the highly concentrated video access pattern as shown in Figure 13(a) compared to less concentrated version access pattern shown in Figure 14(a). The hybrid strategy performs consistently worse than the video-popularity based one, but still a little better than the version-popularity based policy. These results are consistent with our analysis on the video and version popularity. When the cache size increases, the version-popularity based policy has more flexibility in choosing the best version to cache, and thus achieves the best cache hit ratio among the three.

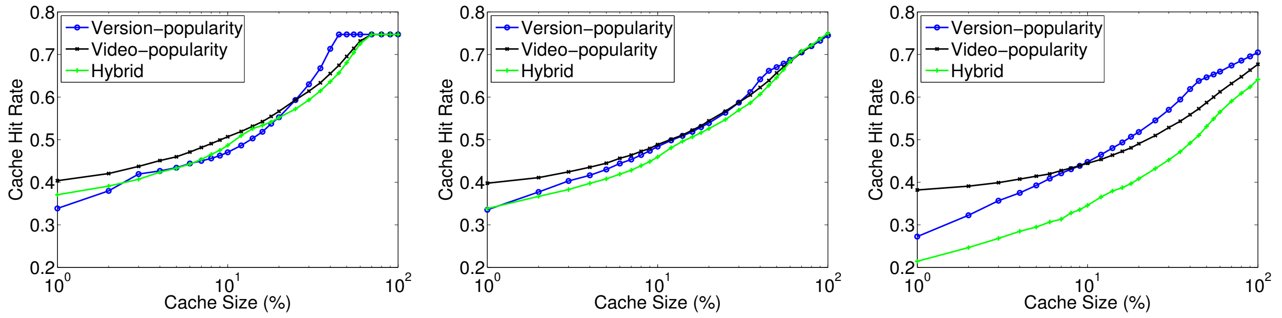
Figure 23(b) further shows the results when one-week trace from Nov. 1st to Nov. 8th was simulated. The cache size percentage used in this simulation is based on total accessed objects of Nov. 1st. We find that the cache performance over one week can reach as high as that of a day. However, the cache performance over a month as shown in Figure 23(c) is worse. This is because with an SE distribution for monthly video accesses, caching is much less efficient than with a Zipf distribution for daily accesses.

7 Related Work

The Internet has witnessed the sharp increase of Internet video traffic in the recent years with all kinds of Internet streaming systems, such as VoD and Internet P2P-based streaming systems. Lots of research has been conducted to study these Internet streaming systems. For example, Yu et al. [17] examined server logs of a traditional VoD system with a total of over 6700 unique videos, and analyzed user access patterns, session length, and video popularity. Yin et al. presented the access logs of a live VoD system [21], which shows different user and content properties compared to [17]. Krishnappa et al. collected Hulu traffic at a campus edge network, and examined the potential benefits of performing caching and prefetching at edge networks [22]. For P2P-based streaming systems, Wu et al. investigated P2P streaming topologies in UUSee [23]. Huang et al. conducted a large scale measurement to study the PPLive-based on-demand streaming [24]. More recently, video service providers are leveraging multiple data centers in the cloud to serve videos. For example, Netflix is running all its operations today on Amazon EC2 [25]. YouTube also leverages Google's CDN for video streaming [26].

Along the increasing popularity of user-generated content (UGC), studies have also been conducted to characterize UGC videos. For example, Cha et al. studied user behaviors and video popularity of YouTube, and compared them with non-UGC content from Netflix [27]. Work [18] examined the traffic characteristics of YouTube at a campus edge network.

With the rapid increase of Internet-capable mobile devices in recent years, mobile Internet video services and accesses are surging. A few studies have been conducted to investigate the performance of mobile streaming applications. Focusing on the resource utilization for



(a) Daily Cache Performance (b) Weekly Cache Performance (c) Monthly Cache Performance

Fig. 23. Cache Performance over Different Time

receiving streaming data on mobile devices, Xiao et al. studied the power consumption of mobile YouTube [10]. Finamore et al. collected traffic from several edge locations and studied the potential reasons for the inferior streaming experience of mobile YouTube users [12]. Previously, we have also conducted measurements to study the resource utilization of different streaming approaches to mobile devices [16]. Furthermore, in order to save battery power, we had designed and implemented BlueStreaming, a system that can leverage low-power of Bluetooth to help P2P streaming to mobile devices [28]. Collecting data from an ISP providing cellular data services, Erman et al. analyze mobile video traffic, focusing on different streaming methods used, prevailing encoding rates, and user behaviors [29]. However, their study is limited to one ISP. Li et al. collect server-side log of an iOS application that uses HTTP live streaming (HLS) for video accesses, and analyze the user behaviors and access patterns [30]. Different from their study that focuses on iOS and HLS only, we base our study on a video service that can be accessed via a much larger variety of mobile devices.

In this study, we have investigated the commons and differences of mobile Internet streaming services with/from the traditional Internet streaming services. Our study reveals a critical challenge in Internet mobile streaming services is the hardware and software heterogeneity of mobile devices. Our analysis also shows different access patterns of mobile videos from traditional Internet streaming videos. Furthermore, we have also shown that caching at the server side with a proper replacement policy can significantly reduce the resource consumption for Vuclip-like Internet streaming systems in dealing with heterogeneity. An earlier version [13] of this manuscript is published in the proceedings of INFOCOM 2012.

8 Conclusion

The wide adoption of mobile devices in practice has made pervasive Internet streaming possible. With elastic resources, cloud platforms have also been increasingly employed for Internet streaming delivery. While a number of studies have been conducted to examine the streaming services from the client's perspective, in this work, we have studied the Internet mobile streaming services from the server side via 4-month server-side log collected on a cloud from one of the largest Internet mobile streaming service providers. Through detailed analysis, we have shown the great hardware and software heterogeneity of mobile devices, different characteristics of mobile videos, and different user access patterns from those in traditional Internet streaming services. As the great challenge that Vuclip-like system faces is the huge demand of CPU resources for online transcoding to deal with heterogeneity, we show that caching at the server side with a proper replacement policy can effectively trade-off limited storage size for great savings on CPU cycles. These results provide some basic guidelines for building and optimizing future Internet mobile streaming systems.

9 Acknowledgement

We appreciate constructive comments from anonymous referees. The work is partially supported by NSF under grants CNS-0746649, CNS-1117300, CCF-0915681, CCF-1146578.

References

[1] "Smart phones overtake client PCs in 2011," http://www.canalys.com/newsroom/smart-phones-overtake-client-pcs-2011.
[2] "comScore Reports August 2012 U.S. Mobile Subscriber Market Share," http://www.comscore.com/Press_Events/Press_Releases/2012/10/comScore_Reports_August_2012_U.S._Mobile_Subscriber_Market_Share.
[3] "Mobile YouTube," http://m.youtube.com.
[4] "Netflix ," http://www.netflix.com.
[5] "Hulu," http://www.hulu.com.
[6] "OrbLive," http://itunes.apple.com/app/id290195003.
[7] "Air Video," http://itunes.apple.com/app/id306550020.
[8] "Qik," http://www.qik.com.
[9] "Vuclip," http://m.vulip.com/.
[10] Y. Xiao, R. Kalyanaraman, and A. Yla-Jaaski, "Energy Consumption of Mobile YouTube: Quantitative Measurement and Analysis," in *Proc. of NGMAST*, 2008.
[11] J. Huang, Q. Xu, B. Tiwana, Z. M. Mao, M. Zhang, and P. Bahl, "Anatomizing Application Performance Differences on Smartphones," in *Proc. of MobiSys*, 2010.
[12] A. Finamore, M. Mellia, M. Munafo, R. Torres, and S. G. Rao, "YouTube Everywhere: Impact of Device and Infrastructure Synergies on User Experience," in *Proc. of ACM IMC*, 2011.

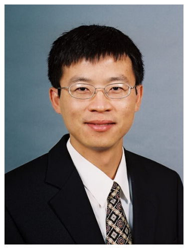
[13] Y. Liu, F. Li, L. Guo, B. Shen, and S. Chen, "A Server's Perspective of Internet Streaming Delivery to Mobile Devices," in *Proc. of IEEE INFOCOM*, 2012.
[14] "WURFL," http://wurfl.sourceforge.net/.
[15] J. Adams and G. Muntean, "Adaptive-Buffer Power Save Mechanism for Mobile Multimedia Streaming," in *Proc. of IEEE ICC*, 2007.
[16] Y. Liu, L. Guo, F. Li, and S. Chen, "An Empirical Evaluation of Battery Power Consumption for Streaming Data Transmission to Mobile Devices," in *Proc. of ACM Multimedia*, 2011.
[17] H. Yu, D. Zheng, B. Y. Zhao, and W. Zheng, "Understanding User Behavior in Large-Scale Video-on-Demand Systems," in *Proc. of EuroSys*, 2006.
[18] P. Gill, M. Arlitt, Z. Li, and A. Manhanti, "YouTube Traffic Characterization: A View From the Edge," in *Proc. of ACM IMC*, 2007.
[19] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web Caching and Zipf-like Distributions: Evidence and Implications," in *Proc. of IEEE INFOCOM*, 1999.
[20] L. Guo, E. Tan, S. Chen, Z. Xiao, and X. Zhang, "The Stretched Exponential Distribution of Internet Media Access Patterns," in *Proc. of PODC*, 2008.
[21] H. Yin, X. Liu, F. Qiu, N. Xia, C. Lin, H. Zhang, V. Sekar, and G. Min, "Inside the Birds Nest: Measurements of Large-Scale Live VoD from the 2008 Olympics," in *Proc. of ACM IMC*, 2009.
[22] D. K. Krishnappa, S. Khemmarat, L. Gao, and M. Zink, "On the Feasibility of Prefetching and Caching for Online TV Services: A Measurement Study on Hulu," in *Proc. of PAM*, 2011.
[23] C. Wu, B. Li, and S. Zhao, "Exploring large-scale peer-to-peer live streaming," in *IEEE Transactions on Parallel and Distributed Systems*, June 2008.
[24] Y. Huang, T. Z. Fu, D.-M. Chiu, J. C. Lui, and C. Huang, "Challenges, design and analysis of a large-scale p2p-vod system," in *Proc. of ACM SIGCOMM*, Seattle, WA, USA, August 2008.
[25] "Netflix in the Cloud," http://velocityconf.com/velocity2011/public/schedule/detail/17785.
[26] R. Torres, A. Finamore, M. Mellia, M. Munaf, and S. Rao, "Dissecting Video Server Selection Strategies in the YouTube CDN," in *Proc. of IEEE ICDCS*, 2011.
[27] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "I Tube, You Tube, Everybody Tubes: Analyzing The Worlds Largest User Generated Content Video System," in *Proc. of ACM IMC*, 2007.
[28] Y. Liu, F. Li, L. Guo, Y. Guo, and S. Chen, "BlueStreaming: Towards Power-Efficient Internet P2P Streaming to Mobile Devices," in *Proc. of ACM Multimedia*, 2011.
[29] J. Erman, A. Gerber, K. Ramakrishnan, S. Sen, and O. Spatscheck, "Over The Top Video: The Gorilla in Cellular Networks," in *Proc. of ACM IMC*, 2011.
[30] Y. Li, Y. Zhang, and R. Yuan, "Measurement and Analysis of a Large Scale Commercial Mobile Internet TV System," in *Proc. of ACM IMC*, 2011.

Yao Liu is a PhD student in Computer Science at George Mason University. She received the B.S. degree in Computer Science from Nanjing University, China in June 2007. Her research interests include multimedia systems and peer-to-peer networks.

Fei Li is an assistant professor of Computer Science at George Mason University. He joined George Mason University in August 2007. He received his B.S. degree in Computer Science in July 1997 from Jilin University in China, M.S., M.Phil., and Ph.D. in February 2002,in May 2007 and in February 2008, respectively, all in Computer Science from Columbia University. Dr. Li's research interests are online and approximation algorithm design and analysis, and applied algorithms for energy-efficient computing, networked systems, and cloud computing.

Lei Guo is a research scientist in Ohio State University. He received the BS degree in space physics and the MS degree in computer science from the University of Science and Technology of China in 1996 and 2002, respectively. He received the PhD degree in computer science and engineering from the Ohio State University in 2007. After that, he worked in Yahoo! and Microsoft as a senior member of technical staff on the systems and algorithms of social search and social networks. His research interests include big data processing and clouding computing, multimedia systems, mobile computing, social networks, P2P networks, and Internet measurement and modeling.

Bo Shen received the B.S. degre in computer science from Nanjing University of Aeronautics and Astronautics, China and the Ph.D. degree in computer science from Wayne State University, Detroit MI. He is now the vice president of engineering at Vuclip. Before that, he was a Senior Research Scientist with Hewlett-Packard Laboratories. His research interests include multimedia signal processing, multimedia networking and content distribution systems. He has published over 50 papers in prestigious technical journals and conferences. He holds seven U.S. patents with many pending. Dr. Shen has been on the Editorial Board for IEEE TRANSACTIONS ON MULTIMEDIA from 2006 to 2008. He served as the Lead Guest Editor for IEEE TMM Special Section on Multimedia Applications in Mobile/Wireless Context. He also served on Program Committee for a number of technical conferences including SIGMM.



Songqing Chen received the PhD degree in computer science from the College of William and Mary in 2004. He received the M.S. and B.S. degrees in Computer Science from Huazhong University of Science and Technology in 1999 and 1997, respectively. He is currently an associate professor of computer science at George Mason University. His research interests include the Internet content deliver systems, Internet measurement and modeling, operating systems and system security, and distributed systems and high performance computing. He is a recipient of the US NSF CAREER Award and the AFOSR YIP Award.

Yingjie Lan is an Assistant Professor at Peking Univeristy. His research interest includes robust networks, competitive analysis of online algorithms, operations management, revenue management and supply chain management.