Cloud Assisted Faster Video Streaming And Sharing

Rohit Dipankar, Nagender Singh Shekhawat, Sachin Tiku,Harshit Gupta
Computer Science and Engineering,
B.M.S. Institute of Technology
Bangalore, India.

Anand R
Asst. Prof
B.M.S. Institute of Technology
Bangalore, India.

Abstract*—While demands on video traffic over mobile networks have been souring, the wireless link capacity cannot keep up with the traffic demand. The gap between the traffic demand and the link capacity, along with time-varying link conditions, results in poor service quality of video streaming over mobile networks such as long buffering time and intermittent disruptions. Leveraging the cloud computing technology, we propose a new mobile video streaming framework, dubbed AMES-Cloud, which has two main parts: AMoV (adaptive mobile video streaming) and ESoV (efficient social video sharing). AMoV and ESoV construct a private agent to provide video streaming services efficiently for each mobile user. For a given user, AMoV lets her private agent adaptively adjust her streaming flow with a scalable video coding technique based on the feedback of link quality. Likewise, ESoV monitors the social network interactions among mobile users, and their private agents try to pre-fetch video content in advance. We implement a prototype of the AMES-Cloud framework to demonstrate its performance. It is shown that the private agents in the clouds can effectively provide the adaptive streaming, and perform video sharing (i.e., prefetching) based on the social network analysis.

Keywords*—*AMoV, ESoV, AMES cloud

I. INTRODUCTION

Over the past decade, increasingly more traffic is accounted by video streaming and downloading. In particular, video streaming services over mobile networks have become prevalent over the past few years .While the video streaming is not so challenging in wired networks, mobile net-works have been suffering from video traffic transmissions over scarce bandwidth of wireless links. Despite network operators' desperate efforts to enhance the wireless link bandwidth (e.g., 3G and LTE), soaring video traffic demands from mobile users are rapidly overwhelming the wireless link capacity.

While receiving video streaming traffic via 3G/4G mobile networks, mobile users often suffer from long buffering time and intermittent disruptions due to the limited bandwidth and link condition fluctuation caused by multi-path fading and user mobility .Thus, it is crucial to improve the service quality of mobile video streaming while using the networking and computing resources efficiently .

II. EXISTING SYSTEM

Cloud computing promises lower costs, rapid scaling, easier maintenance, and service availability anywhere, anytime, a key challenge is how to ensure and build confidence that the cloud can handle user data securely. A recent Microsoft survey found that "58 percent of the public and 86 percent of business leaders are excited about the possibilities of cloud computing. But more than 90 percent of them are worried about security, availability, and privacy of their data as it rests in the cloud."

III. PROPOSED SYSTEM

We propose an adaptive mobile video streaming and sharing framework, called AMES-Cloud, which efficiently stores videos in the clouds (VC), and utilizes cloud computing to construct private agent (subVC) for each mobile user to try to offer "non-terminating" video streaming adapting to the fluctuation of link quality based on the Scalable Video Coding technique. Also AMES-Cloud can further seek to provide "non-buffering" experience of video streaming by background pushing functions among the VB, subVBs and localVB of mobile users. We evaluated the AMES-Cloud by prototype implementation and shows that the cloud computing technique brings significant improvement on the adaptivity of the mobile streaming. We ignored the cost of encoding workload in the cloud while implementing the prototype.

IV. AN EMERGING FRAMEWORK OF CLOUD-ASSISTED MOBILE VIDEO SERVICES

In this article, we will investigate and address the emerging techniques for cloud-assisted mobile video services, which construct private agents for active mobile users in the cloud, in order to offer "non-terminating" and "non-buffering" mobile video streaming service. Private agents are elastically

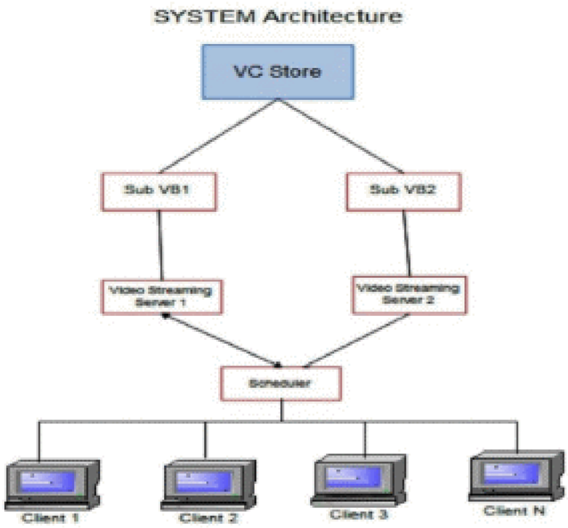

Fig 1 System Architecture [7]

ISSN: 2278-0181

initiated and optimized in the cloud platform. Also the real-time SVC coding is done on the cloud side efficiently. The proposed framework leverages the SVC technique, and offers the scalable and adaptive streaming experiences by controlling the combination of video streams (layers) depending on the feedback of the fluctuating link quality from mobile users.
Also based on the analysis of the SNS activities of mobile users, the proposed framework seeks to pre-fetch the video clips in advance from user's private agent to the local storage of the device. The strength of the social links between users and the history of various social activities can determine how much and which video will be prefetched.

We organize the article as follows: we first explain the cloud agent framework and give details of our proposal on adaptive video streaming and social-aware prefetching. Then the brief video delivery procedure is shown, and we discuss the performance evaluation of our proposal as well. The conclusion of the article is presented in the end.

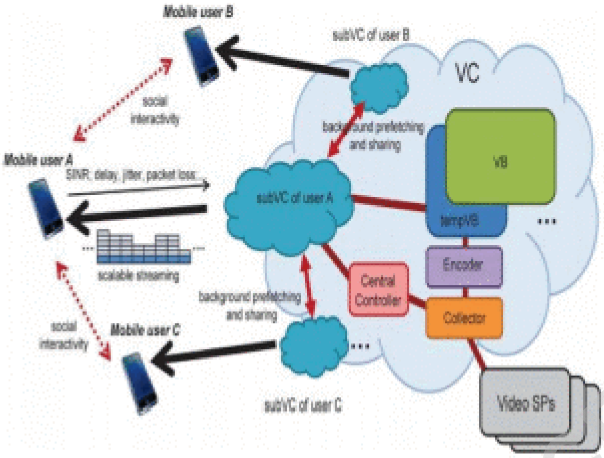

Fig 2 Proposed System [5]

V. SOCIAL AWARE VIDEO PREFETCHING

A. Social Content Sharing

In SNSs, users can subscribe to known friends, famous people, and particular content publishers; also there are various types of social activities among users in SNSs. So we need to define various strength levels for those social activities to indicate the different possibilities that the video shared by one user may be watched by the recipients of his/her sharing activities, so that subVCs can carry out effective background prefetching at subVB and even may push to user's localVB, because after one shares a video, there may be a certain delay that the recipient gets to know the sharing, and initiates to watch [11, 16]. Therefore the prefetching in prior will not impact the users in most cases. Instead, a user can click to see without any buffering delay as the beginning part or even the whole video is already prefetched locally.
The amount of prefetched segments is mainly determined by the strength of the social activities and user's link status. And thus we classify the social activities in current popular SNSs into three kinds, regarding the impact of the activities and the potential reacting priority from the point of view of the recipients [12]:

Direct recommendation: In SNSs, a user can directly recommend a video to particular friend(s) by a short notice message. The recipients of the message may watch it with very high probability. This is considered as "strong".
Subscription: Like the popular RSS services, a user can subscribe to a particular video publisher based on interests. This interest-driven connectivity between the subscriber and the video publisher is considered as "median," because the subscriber may not always watch all subscribed videos.
Public sharing: The activity of watching or sharing a video by a user can be seen by his/her friends in their timeline of activity stream. We consider this public sharing as a "weak" connectivity among users, because many people may not watch the video that one has watched or shared with no specific recommendation.

B. Prefetching Levels

Different strengths of the social activities indicate different levels of probability that a video will be soon watched by the recipient. Correspondingly, we also define three prefetching levels regarding the social activities of mobile users:
"All": The video shared by the **direct recommendations** will be watched with a very high probability, so we propose to pre-fetch the BL and all ELs, in order to let the recipient(s) directly watch the video with a good quality, without any buffering.
"Parts": Because the videos that are published by **subscriptions** may be watched by the subscribers with a not so high probability, we propose to only pre-fetch parts of the BL and ELs segments, for example, the first 10 percent segments.
"Little": The **public sharing** has a weak impact, so the probability that a user's friends (followers) watch the video that the user has watched or shared is relatively low. We propose to only pre-fetch the first BL segment to those who have seen his/her activity in their timeline stream.
The prefetching happens among subVBs and the VB, and more importantly, will be performed from the subVB from the cloud to the localVB of the mobile device depending on the link quality. If a mobile user is covered by Wi-Fi access, due to Wi-Fi's capable link and low price (mostly for free), subVC can push as much as possible in most cases regarding the prefetching levels. However if he/she is with a 3G/4G connection, we propose to downgrade the prefetching level to save energy and money, but users can still benefit from the prefetching effectively for little buffering delay.
For example, as shown in Fig. 2, when user A gets an interesting music video (MV) from his/her subscription, prefetching level "parts" should be chosen; however because A is connected by 3G, the prefetching will be downgraded to "little". User *B* gets direct recommendation of the MV from *A*, and thus *B*'s subVC will pre-fetch the video at the level of "all"; but *B* is also connected by 3G, so only "parts" of the video segments will be pushed. User *C* sees *B*'s sharing activity of the MV while *C* is connected by Wi-Fi, and thus *C*'s subVC will push "little" of the segments to *C*. Note that users themselves can also configure the prefetching conditions on demand.

ISSN: 2278-0181

Different mobile users have different patterns of accessing videos [11, 16] that are per-user dependent mainly due to people's different life styles. We capture a realistic trace from the Chinese biggest SNS site, Sina Weibo, and check the access delay of about 2 million people during July 2012. We draw the Probability Distribution Functions of access delays from three randomly selected users in Fig. 3. We find that although user X has very short delays for checking updates in SNSs, user Y and user Z often have access delays for hours, and even up to one or two days. Based on our analysis on the whole user base, the average access delay is about 6.5 hours, and even more than one third of the users have access delay larger than 1 day. These large access delays offer us huge potential to pre-fetch the videos before users access them.

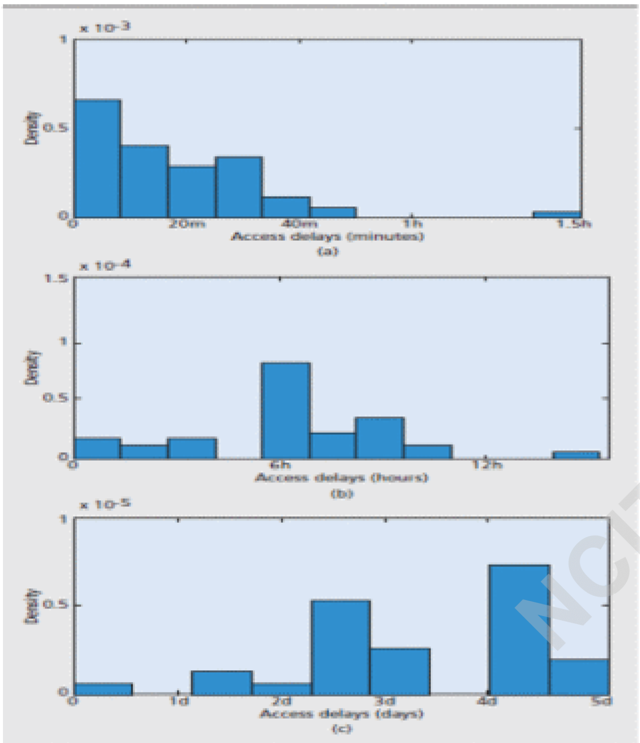

Figure 5.1 Access delays of mobile users to video contents. [3, 8]

VI. VIDEO STORAGE AND STREAMING FLOW

The two parts, cloud-assisted adaptive video streaming and social-aware video prefetching in the framework, have tight connections and will together service the video streaming and sharing: they both rely on the cloud computing platform and are carried out by the private agencies of users; while prefetching, the streaming part will still monitor and improve the transmission considering the link status.

Once a mobile user starts to watch a video via a link, the localVB will first be checked whether there are any prefetched segments of the video. If there is none or just some parts, the client will report to its subVC, and if the subVC has the video, the subVC will initiate for transmission the remaining segments. But once there is no prefetched parts of the video in the subVB, the tempVB and VB in the central VC will be checked. In the case that there is no such a video in tempVB or VB, the collector in VC will immediately fetch the video from external video providers via the link, and the subVC will re-encode in SVC format, taking a bit large delay, and then stream to the mobile user.

VII. IMPLEMENTATION AND EVALUATION

We evaluate the performance of the AMES-Cloud framework by a prototype implementation. We choose the U-cloud server (premium) in the cloud computing service offered by Korean Telecom, and utilize the virtual server with 6 virtual CPU cores (2.66 GHz) and 32 GB memory, which is fast enough for encoding 480P (480 by 720) video with H.264 SVC format in 30 fps at real time [9]. In the cloud, we deploy our server application based on Java, including one main program handling all tasks of the whole VC, while the program dynamically initializes, maintains and terminates instances of another small Java application as private agents for all active users. We implement the mobile client at a mobile phone, Samsung Galaxy II, with android system version 4.0. The mobile data service is offered by LG LTE network, while in some uncovered area the 3G network is used. Note that we still use "3G" to indicate the general cellular network. We test in the downtown area, so the practical bandwidth of the mobile link is not as high as we expected, but this won't impact our experiment results. The test video is the Tomb Raider 2012 Trailer in H.264 format with 480P resolution downloaded from YouTube. Its size is 13.849 Mbytes and with a duration of 180 seconds. We first decode it by the x264 decoder into the YUV format, and re-encode it by the H.264 SVC encoder, the Joint Scalable Video Model (JSVM) software of version 9.1 [40]. We just use default settings for the decoding and encoding, and do the H.264 SVC encoding at the virtual server in the cloud.

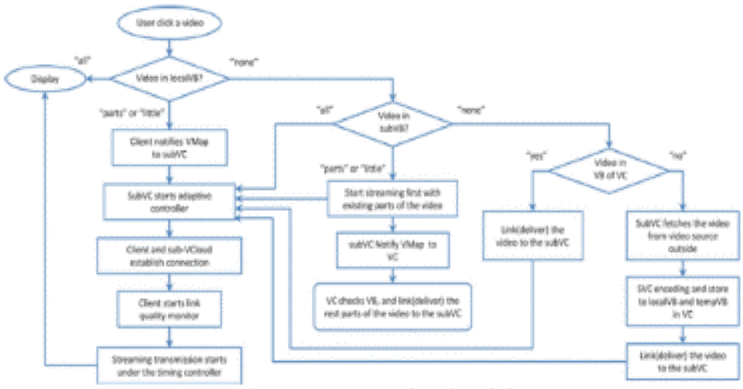

Figure 7.1 System Flowchart [5]

VIII. VIDEO STREAMING IN SUBVC AND VC BY SVC

We evaluate how H.264 SVC works in the cloud regarding the above mentioned SVC configurations. As shown in Fig. 4, because of the strong computational capacity of the cloud computing, the encoding speed is dramatically fast. The best resolution configuration "1:5:2:2" with 5-second temporal

ISSN: 2278-0181

segmentation scheme requires about 560 ms for encoding. For those very short intervals of Twin, the encoding delay is small under 50 ms.

More ELs induce higher overhead due to the duplicated I-frames, and thus we test the storage overhead, which is calculated by the ratio of the extra size of the video segments during SVC encoding to the size of only the low-quality BL. The resolution scheme of "1:1:1:1" has a low overhead around below 10 percent, and "1:2:2:2" with two ELs for each scalability feature has about 17 percent overhead, which is acceptable. However higher resolution like "1:4:2:2" has 61 percent overhead, and "1:5:2:2" has even 120 percent overhead, which is not efficient. Overall, an SVC stream should not contain too many ELs, that is, a high scalability practically brings high overhead.

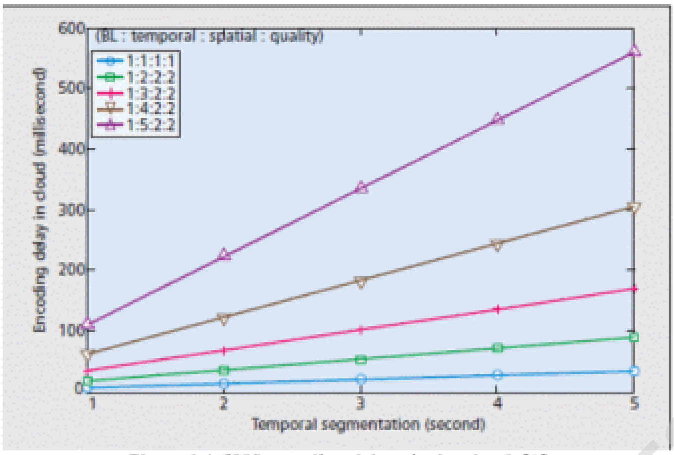

Figure 8.1 SVC encoding delays in the cloud. [1]

IX. PREFETCHING DELAY

In ESoV, video segments can be prefetched among VB, tempVB, and localVBs of the mobile users, based on their activities in SNSs. we evaluate the required delays for different levels of prefetching as shown in Table. 2. We here use the normal resolution configuration of 2 second temporal segmentation by default (the same in following tests). We also set the sharing length of "little" as only the first 5 seconds of the BL and ELs, that of "parts" as the first 15 seconds of the BL and ELs, and that of "all" as all BL and ELs segments.

We can see that prefetching supported by the cloud computing is significantly fast. When prefetching via wireless links, it takes several seconds. However it is obvious that in most cases [26], [38] a recipient of the video sharing may not watch immediately after the original sharing behaviour, that is normal users have significant access delay gaps, so this prefetching transmission delay won't impact user's experience at all, but will bring "non-buffering" experience in fact when the user clicks to watch at a later time.

X. WATCHING DELAY

We test how long one user has to wait from the moment that one clicks the video in the mobile device to the moment that the first streaming segment arrives, which is called as "click to play" delay. As shown in Fig. 8, if the video has been cached in localVB, the video can be displayed nearly immediately with ignorable delay. When we watch video which is fetched from the subVC or the VC, it generally takes no more than 1 second to start. However if the user accesses to AMES-Cloud service via the cellular link, he will still suffer a bit longer delay (around 1s) due to the larger RTT of transmission via the cellular link.

For the cases to fetch videos which are not in the AMES Cloud (but in our server at lab), the delay is a bit higher. This is mainly due to the fetching delay via the link from our server at lab to the cloud data center, as well as the encoding delay. In practical, there are be optimized links in the Internet backbone among video providers and cloud providers, and even recent video providers are just using cloud storage and computing service. Therefore this delay can be significantly reduced in practice. Also this won't happen frequently, since most of the popular videos will be already prepared in the AMES-Cloud.

XI. CONCLUSION

In this paper, we discussed our proposal of an adaptive mobile video streaming and sharing framework, called AMES-Cloud, which efficiently stores videos in the clouds (VC), and utilizes cloud computing to construct private agent (subVC) for each mobile user to try to offer "non-terminating" video streaming adapting to the fluctuation of link quality based on the Scalable Video Coding technique. Also AMES-Cloud can further seek to provide "non-buffering" experience of video streaming by background pushing functions among the VB, subVBs and localVB of mobile users. We evaluated the AMES-Cloud by prototype implementation and shows that the cloud computing technique brings significant improvement on the adaptivity of the mobile streaming.

The focus of this paper is to verify how cloud computing can improve the transmission adaptability and prefetching for mobile users. We ignored the cost of encoding workload in the cloud while implementing the prototype. As one important future work, we will carry out large-scale implementation and with serious consideration on energy and price cost. In the future, we will also try to improve the SNS-based prefetching, and security issues in the AMES-Cloud.

REFERENCES

[1] "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2011–2016," CISCO, 2012.

[2] Y. Li, Y. Zhang, and R. Yuan, "Measurement and analysis of a large scale commercial mobile Internet TV system," in *Proc. ACM Internet Meas. conf.*, 2011, pp. 209–224.

[3] T. Taleb and K. Hashimoto, "MS2: A novel multi-source mobile-streaming architecture," *IEEE Trans. Broadcasting*, vol. 57, no. 3, pp. 662–673, Sep. 2011.

[4] X. Wang, S. Kim, T. Kwon, H. Kim, and Y. Choi, "Unveiling the bit torrent performance in mobile WiMAX networks," in *Proc. Passive Active Meas. Conf.*, 2011, pp. 184–193.

[5] A. Nafaa, T. Taleb, and L. Murphy, "Forward error correction adaptation strategies for media streaming over wireless networks," *IEEE Commun. Mag.*, vol. 46, no. 1, pp. 72–79, Jan. 2008.

[6] J. Fernandez, T. Taleb, M. Guizani, and N. Kato, "Bandwidth aggregation- aware dynamic QoS negotiation

ISSN: 2278-0181

for real-time video applications in next-generation wireless networks," *IEEE Trans. Multimedia*, vol. 11, no. 6, pp. 1082-1093, Oct. 2009.

[7] T. Taleb, K. Kashibuchi, A. Leonardi, S. Palazzo, K. Hashimoto, N. Kato, and Y. Nemoto, "A cross-layer approach for an efficient delivery of TCP/RTP-based multimedia applications in heterogeneous wireless networks," *IEEE Trans. Veh. Technol.*, vol. 57, no. 6, pp. 3801–3814, Jun. 2008.

[8] K. Zhang, J. Kong, M. Qiu, and G. L. Song, "Multimedia layout adaptation through grammatical specifications," *ACM/Springer Multimedia Syst.*, vol. 10, no. 3, pp. 245–260, 2005.

[9] M. Wien, R. Cazoulat, A. Graffunder, A. Hutter, and P. Amon, "Real-time system for adaptive video streaming based on SVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1227–1237, Sep. 2007.

[10] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 9, pp. 1103–1120, Sep. 2007