IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 16, Issue 5, Ver. V (Sep – Oct. 2014), PP 16-23
www.iosrjournals.org

Cloud stream: Delivering high-quality streaming videos through a cloud-based svc proxy

Mr. Gaurav Buddhawar[(1)] Ruchita Munghate[(2)],Vaishali Illulkar[(3)], NehaSahare[(4)]

Namdeorao Poreddiwar College of Engineering and Technology, Gadchiroli
Assistant Professor [(1)], Department of Computer Science and Engineering

Abstract: *Mobile phones are among the most popular consumer devices, and the recent development of 3G networks, rapidly increasing power of personal mobile devices (smart phones, tablets, etc.) is providing much richer contents and social interactions to users. This trend however is restricted by the limited battery lifetime of mobile devices and unstable wireless connectivity, making the highest possible quality of service experienced by mobile users not feasible .The recent cloud computing technology, with its rich resources to compensate for the limitations of mobile devices and connections, can potentially provide an ideal platform to support the desired mobile services. In this paper, we propose the design of a Cloud-based, Mobile social tv system (CloudMoV). The system effectively utilizes both PaaS (Platform-as-a-Service) and IaaS (Infrastructure-as-a-Service) cloud services to offer the living-room experience of video watching to a group of disparate mobile users who can interact socially while sharing the video. To guarantee good streaming quality as experienced by the mobile users with time varying wireless connectivity, we employ a surrogate for each user in the IaaS cloud for video downloading and social exchanges on behalf of the user. Given the battery life as a key performance bottleneck, we advocate the use of burst transmission from the surrogates to the mobile users, and carefully decide the burst size which can lead to high energy efficiency and streaming quality. Social interactions among the users, in terms of spontaneous textual exchanges, are effectively achieved by efficient designs of data storage with Bitable. These various designs for flexible transcending capabilities, battery efficiency of mobile devices and spontaneous social interactivity together provide an ideal platform for Mobile social tv services. We have implemented CloudMoV on Amazon EC2 and Google App Engine and verified its superior performance based on real world experiments.*

I. Introduction

Developments of 3G networks and smart phones enable users to watch video programs by subscribing data plans from service providers. Due to the ubiquity of mobile phones and phone-to-phone communication technologies, data-plan subscribers can redistribute the video content to nonsubscribers. Thanks to the revolutionary "reinventing the phone" campaigns initiated by Apple Inc. in2007, Smartphone's nowadays are shipped with multiple microprocessor cores and gigabyte RAMs; they possess more computation power than personal computers of a few years ago. Apart from common productivity tasks like emails and web surfing, Smartphone's are flexing their strengths in more challenging scenarios such as real time video streaming and online gaming, as well as serving as a main tool for social exchanges. Although many mobile social or media applications have emerged, truly killer ones gaining mass acceptance are still impeded by the limitations of the current mobile and wireless technologies, among which battery lifetime and unstable connection bandwidth are the most difficult ones. It is natural to resort to cloud computing, the newly-emerged computing paradigm for low-cost, and agile, scalable resource supply, to support power-efficient mobile data communication. The cloud can offload the computation and other tasks involved in a mobile application and may significantly reduce battery consumption at the mobile devices, if a proper design is in place. The big challenge in front of us is how to effectively exploit cloud services to facilitate mobile applications.

In this paper, we describe the design of a novel mobile social TV system, CloudMoV, which can effectively utilize the cloud computing paradigm to offer a living-room experience of video watching to disparate mobile users with spontaneous social interactions. In CloudMoV, mobile users can import a live or on-demand video to watch from any video streaming site, invite their friends to watch the video concurrently, and chat with their friends while enjoying the video. It therefore blends viewing experience and social awareness among friends on the go.

Encoding flexibility: Different mobile devices have differently sized displays, customized playback hardware's, and various codec's. Traditional solutions would adopt a few encoding formats

ahead of the release of a video program. But even the most generous content providers would not be able to attend to all possible mobile platforms, if not only to the current hottest models. CloudMoV customizes the streams for

different devices at real time, by offloading the transcoding tasks to an IaaS cloud.

Battery efficiency: A breakdown analysis conducted by Carroll et al. [6] indicates that the network modules (both Wi-Fi and 3G) and the display contribute to a significant portion of the overall power consumption in a mobile device, dwarfing usages from other hardware modules including CPU, memory, etc. We target at energy saving coming from the network module of smart phones through an efficient data transmission mechanism design.

Spontaneous social interactivity: Multiple mechanisms are included in the design of CloudMoV to enable spontaneous social, co- viewing experience . First , efficient synchronization mechanisms are proposed to guarantee that friends joining in a video program may watch the same portion (if they choose to), and share immediate reactions and comments. Although synchronized playback is inherently a feature of traditional TV, the current Internet video services (e.g., Web 2.0 TV) rarely offer such a service.

Portability: A prototype CloudMoV system is implemented following the philosophy of "Write Once, Run Anywhere" (WORA): both the front- end mobile modules and the backend server modules are implemented in "100% Pure Java" , with well-designed generic data models suitable for any Big Table-like data store; the only exception is the transcoding module, which is implemented using ANSI C for performance reasons and uses no platform-dependent or proprietary APIs.

II. Related Work

A number of mobile TV systems have sprung up in recent years, driven by both hardware and software advances in mobile devices. Some early systems bring the "living room" experience to small screens on the move. But they focus more on barrier clearance in order to realize the convergence of the television network and the mobile network, than exploring the demand of "social" interactions among mobile users. There is another trend in which efforts are dedicated to extending social elements to television systems [4] [5]. Coppens et al. [4] try to add rich social interactions to TV but their design is limited to traditional broadcast program channels. Oehllberg et al. [5] conduct a series of experiments on human social activities while watching different kinds of programs. Though inspiring, these designs are not that suitable for being applied directly in a mobile environment. Chuahetal. extend the social experiences of viewing traditional broadcast programs to mobile devices, but have yet to deliver a well integrated framework. Schatzetal. Have designed a mobile social TV system, which is customized for DVBH networks and Symbian devices as opposed to a wider audience. Compared to these prior work and systems, we target at a design for a generic, portable mobile social TV framework, featuring co-viewing experiences among friends over geographical separations through mobile devices. Our framework is open to all Internet-based video programs, either live or on-demand, and supports a wide range of devices with HTML5 compatible browsers installed, without any other mandatory component on the devices.

Finally, we are aware of the lack of a richly-featured cloud based mobile social TV system in real life. This iOS-locked application only supports live video channels, and all its social functions are bound to Facebook open APIs. Conversely, the prototype we implement is browser-based and platform independent; it supports both live channels, VoD channels and even personal channels hosted by any user, with wider usage ranges and flexible extensibility. The framework we propose can be readily applied to other cloud-assisted mobile media applications as well.

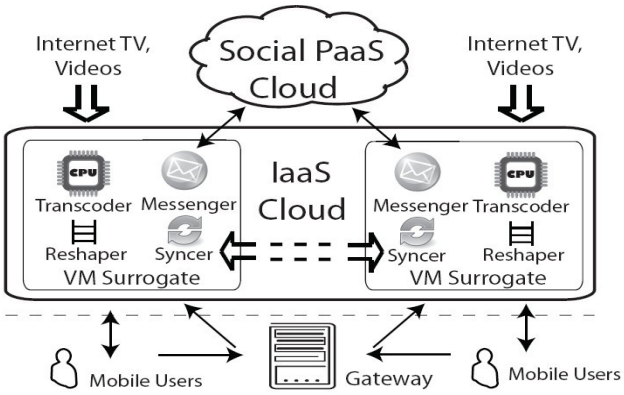


Fig. 1. The architecture of CloudMoV.

III. Cloudmov: Architecture And Design

As a novel Cloud-based Mobile sOcial tV system, Cloud-MoV provides two major functionalities to participating mobile users: (1) **Universal streaming**. A user can stream a live or on-demand video from any video sources he chooses, such as a TV program provider or an Internet video streaming site, with tailored encoding formats and rates for the device each time. (2) **Co-viewing with social exchanges**. A user can invite multiple friends to watch the same video, and exchange text messages while watching. The group of friends watching the same video is referred to as a session. The mobile user who initiates a session is the host of the session. We present the architecture of CloudMoV and the detailed designs of the different modules in the following.

A. Key Modules

The design of CloudMoV can be divided into the following major functional modules.

1. Transcoder. It resides in each surrogate, and is responsible for dynamically deciding how to encode the video stream from the video source in the appropriate format, dimension, and bit rate. Before delivery to the user, the video stream is further encapsulated into a proper transport stream. In our implementation, each video is exported as MPEG-2 transport streams, which is the de facto standard nowadays to deliver digital video and audio streams over loss medium.

2. Reshaper. The reshaper in each surrogate receives the encoded transport stream from the transcoder, chops it into segments, and then sends each segment in a burst to the mobile device upon its request (i.e., a burst transmission mechanism), to achieve the best power efficiency of the device. The burst size, i.e., the amount of data in each burst, is carefully decided according to the 3G technologies implemented by the corresponding carrier.

3. Social Cloud. The social cloud is built on top of any general PaaS cloud services with Big Table-like data store to yield better economies of scale without being locked down to any specific proprietary platforms. Despite its implementation on Google App Engine (GAE) as a proof of concept, our prototype can be readily ported to other platforms.

B. Loosely Coupled Interfaces

Similar in spirit to web services, the interfaces between different modules in CloudMov, i.e., mobile users, VM surrogates, and the social cloud, are based on HTTP, a universal standard for all Internet-connected devices or plat forms. Thanks to the loose coupling between users and the infrastructure, almost any mobile device is ready to gain access to the CloudMoV services, as long as it is installed with an HTTP browser. The VM surrogates provisioned in the IaaS cloud cooperate with the social cloud implemented on a PaaS cloud service via HTTP as well, with no knowledge of the inner components and underlying technologies of each other, which contributes significantly to the portability and easy maintenance of the system.

C. Pipelined Video Processing

Streaming of stored contents are supported in CloudMoV. Video processing in each surrogate is designed to work on the fly, i.e., the transcoder conducts real-time encoding from the video source, the

encoded video is fed immediately into the reshaper for segmentation and transmission, and a mobile user can start viewing the video as soon as the first segment is received.

D. Burst Transmissions

3G power states: Different from Wi-Fi which is more similar to the LANed Internet access, 3G cellular services suffer from the limited radio resources, and therefore each user equipment (UE) needs to be regulated by a Radio Resource Control (RRC) state machine .

Different 3G carriers may customize and deploy complex states in their respective cellular networks. Different states indicate different levels of allocated radio resources, and hence different levels of energy consumptions. For ease of implementation, we consider three basic states in our design, which are commonly employed by many carriers, namely CELL DCH (a dedicated physical channel is allocated to the UE in both the uplink and the downlink), CELL FACH (no dedicated channel is allocated but the UE is assigned a default common transport channel in the uplink), and IDLE, in decreasing order of power levels.

IV. Cloudmov: Prototype Implementation

Following the design guidelines in Sec. III, we have implemented a real-world mobile social TV system, and deployed it on the Google App Engine (GAE) and Amazon EC2 clouds, which are the two most widely used public PaaS and IaaS cloud platforms.GAE, as a PaaS cloud, provides rich services on top of Google's data centers and enables rapid deployment of Java based and Python-based applications. Data store, a thin layer built on top of Google's famous Big Table, handles "big" data queries well with linear and modular scalability even for high-throughput usage scenarios.

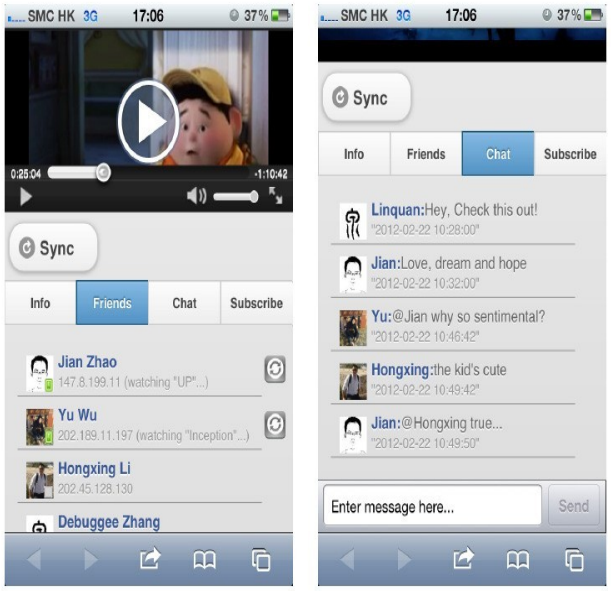
A. Client Use of CloudMoV

All mobile devices installed with HTML5 compatible browsers can use CloudMoV services, as long as the HTTP Live Streaming (HLS) [24] protocol is supported. The user first connects to the login page of CloudMoV, as illustrated in the top left corner of Fig. 3. After the user successfully logs in through the gateway, he is assigned a VM surrogate from the VM pool (the hostnames of available VMs, e.g., ec2-50-16-xx- xx.compute-1.amazonaws.com, are maintained in an in memory table of a MySQL database deployed in the gateway). Surrogate, and welcomed by a portal page as shown on the Right-hand side of Fig. 3. Upon user login, the portal collects the device configuration information by examining the "User- Agent" header values, and this information will be sent to its surrogate for decision making of the video encoding formats.

Fig. 2. Client UI of CloudMoV.

B. VM Surrogates

All the VM surrogates are provisioned from Amazon EC2 web services and tracked by the gateway. We create our own AMI (ami- b6f220df) based on Linux kernel 2.6.35.14, the default image Amazon provides.



(a) "Friend tab" (b) "Chat tab"

Fig. 3. "Friend" and "Chat" tabs. computation involved, we propose to implement all the video processing related tasks using ANSI C, to guarantee the performance. In particular, we install FFmpeg together with libavcodec as the groundsill library to develop the transcoding, segmentation and reshaping modules on the VM surrogates. We have also installed a Tomcat web server (version 6.5) to serve as a Servlet container and a file server on each surrogate. Both FFmpeg and Tomcat are open source projects.

Once a VM surrogate receives a video subscription request from the user, it downloads the video from the source URL, and processes the video stream by transcoding and segmentation, based on the collected device configurations by the portal.

C. Data Models in the Social Cloud

We use GAE mainly as the back-end data store to keep the transient states and data of CloudMoV, including users' online presence status, social messages (invitation and chat messages) in all the sessions.

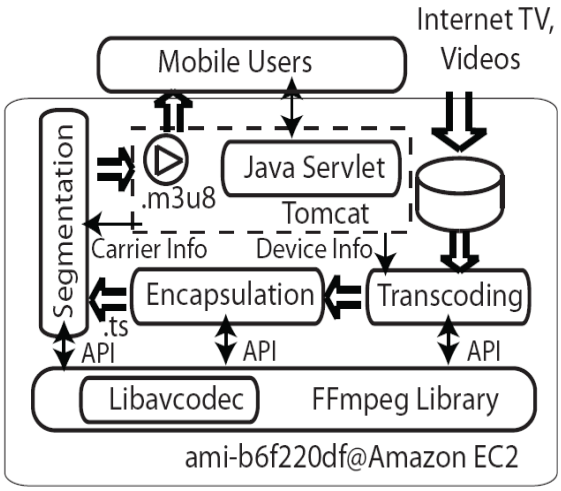


Fig. 4. Streaming architecture in each customized VM image (ami-b6f220df).

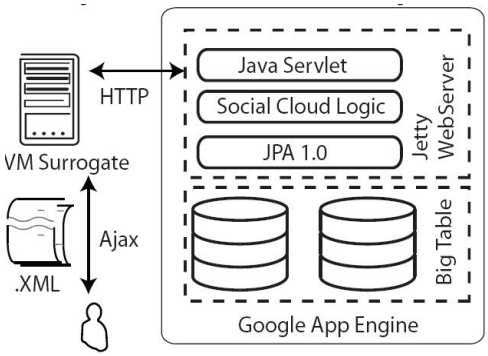


Fig. 5. Social message exchanges via Google App Engine.

V. Real-World Experiments

We carry out both unit tests and performance evaluations of CloudMoV deployed on Amazon EC2 and Google App Engine, using a number of iPhone 4S smart phones (iOS 5.01) as the mobile clients, which have been registered on the Apple developer site. The gateway is implemented on a Virtual Private Server (VPS) hosted by Blue host.

A. Measuring the RRC States

We first design measurement experiments to discover the timeout values of the critical inactivity timers employed in 3HK's 3G network, as discussed in Sec. III-D. We enable logging functions on a fully charged iPhone 4S and use the Mobile Safari (the HTML5-compatible browser on iPhone) to watch a YouTube video using CloudMoV services.

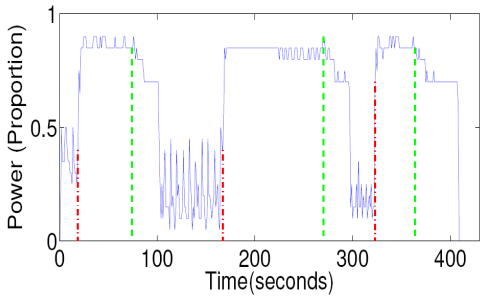


Fig. 6. Power consumption over time on an iPhone 4S

B. Impact of Burst Size on Power Consumption

The technique of video segmentation is widely employed in video streaming applications, but mostly for ease of distribution and not for battery efficiency at potential mobile users. Apple Inc., which proposed the HTTP Live streaming protocol, suggests 10-second-playback segments, which has been followed in many streaming applications. We find this segment size is problematic and can drain the battery of a mobile device quickly. In Fig. 8, we compare the power consumption levels when burst transmission intervals of 10 seconds and 60 seconds are used, respectively, for the iPhone 4S to stream a 10- minute YouTube flash video (.flv).

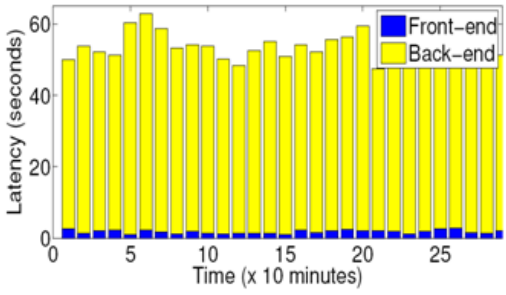


Fig. 7. Average user sign-in latencies over time.

C. Sign-in Latency into the System

When a user signs into the CloudMoV system via the login gateway shown in Fig. 3 and gets identified, the gateway will request a virtual machine instance from the IaaS cloud to be the user's surrogate. The sign-in process finishes when the surrogate is initialized and the user is connected to the surrogate. In this experiment, five mobile users repeatedly join the system and log off as soon as the respective surrogate is initialized.

D. Startup Latency of Video Playback

We evaluate the transcoding performance on the surrogates in CloudMoV, first by measuring the playback startup latency on the surrogates, from the time when the video subscription request is received from the mobile user to the time when the first transcoded burst segment is generated.

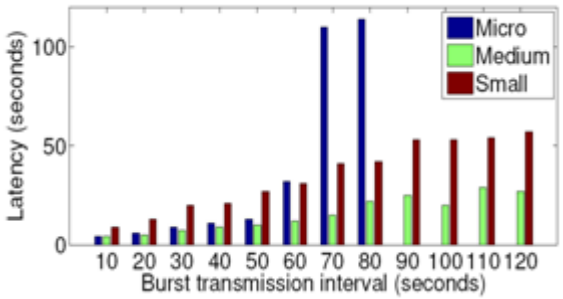
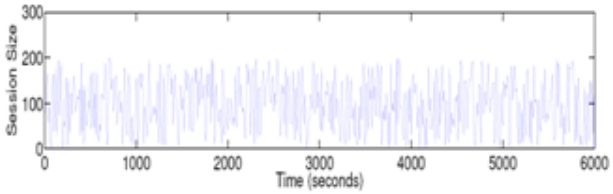


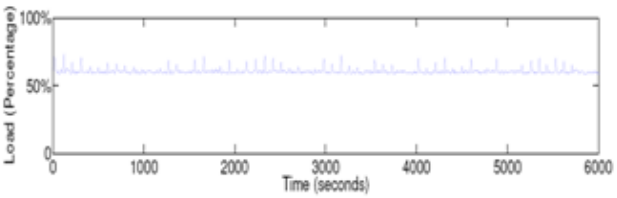
Fig. 8. Startup latency at different burst sizes.

VI. Concluding Remarks And Future Work

This paper presents our view of what might become a trend for mobile TV, i.e., mobile social TV based on agile resource supports and rich functionalities of cloud computing services. We introduce a generic and portable mobile social TV framework, CloudMoV that makes use of both an IaaS cloud and a PaaS cloud. The framework provides efficient transcoding services for most platforms under various network conditions and supports for co-viewing experiences through timely chat exchanges among the viewing users. By employing one surrogate VM for each mobile user, we achieve ultimate scalability of the system. Through an in-depth investigation of the power states in commercial 3G cellular networks, we then propose an energy-efficient burst transmission mechanism that can effectively increase the battery lifetime of user devices. For implementing social interactions, most Big Table- like data stores (including GAE) support memcache which is a highly efficient



(a) Session size over time



(b) Workload of the session host over time

Secondary storage on the data stores. We seek to integrate memcache support into CloudMoV, by possibly me caching the data (e.g., chat histories) of sessions where friends chat actively, so as to further improve the query performance. To sustain the portability of the system, we will stick to standard API interfaces, i.e., JCache (JSR 107), in
our system.

References

[1]. M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," IEEE Pervasive Computing, vol. 8, pp.14–23, 2009.
[2]. S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in Proc. of IEEE INFOCOM, 2012.
[3]. W. Zhu, C. Luo, J. Wang, and S. Li, "Multimedia cloud computing," IEEE Signal Processing Magazine, vol. 28, pp. 59–69, 2011.
[4]. T. Coppens, L. Trappeniners, and M. Godon, "AmigoTV: towards a social TV experience," in Proc. of EuroITV, 2004.
[5]. N. Ducheneaut, R. J. Moore, L. Oehlberg, J. D. Thornton, and E. Nickell, "Social TV: Designing for Distributed, Sociable Television Viewing," International Journal of Human-Computer Interaction, vol. 24, no. 2, pp. 136–154, 2008.
[6]. A. Carroll and G. Heiser, "An analysis of power consumption in as smartphone," in Proc. of USENIXATC, 2010.