Minimizing Monetary Cost via Cloud Clone Migration in Multi-screen Cloud Social TV System

Yichao Jin and Yonggang Wen
School of Computer Engineering
Nanyang Technological University
{yjin3, ygwen}@ntu.edu.sg

Han Hu
School of Computing
National University of Singapore
huh@comp.nus.edu.sg

***Abstract*—The emergence of multi-screen cloud social TV has the potential to transform TV experience, providing a unified media experience across a diverse set of devices at an affordable cost. One key technology to support unified media experience across multiple screens is to instantiate a virtual machine (VM) as a cloud clone of the user, to manage all his/her media outlets (e.g., TV and smartphone), as implemented in our Cloud-Centric Media Network (CCMN). In this case, as the user shifts his attention from one device to another, the cloud clone can migrate to another location for better quality of experience. In this paper, we investigates the problem of cloud-clone migration for the multi-screen social TV application, minimizing its monetary cost. This problem can be cast into the Markov Decision Process (MDP) framework, to balance a trade-off between the migration cost and the transmission cost. Under this framework, we first derive an upper and lower bound for the optimal monetary cost, by considering a fixed placement policy and an offline policy. We then follow up with an online policy using a dynamic programming approach. Our numerical results indicate, up to 10% monetary cost can be saved, by optimally migrating the cloud clone. Moreover, the cost reduction depends on the length of content-delivery path, the data size associated with VM migration, and the user behavior pattern. These insights would offer operational guidelines to deliver cost effective multi-screen social TV services over CCMN, potentially easing its adoption.**

I. Introduction

Lately TV experience has been dramatically transformed, with the emergence of multi-screen social TV [1]–[3]. First, a traditional "laid-back" video watching experience is combined with "lean-forward" social interactions among peer viewers, resulting in an user-centric viewing environment [1]. Indeed, one salient feature of social TV is to create a living-room TV experience by allowing viewers in remote rooms to converse around contents, via various communication modalities (e.g., text, graphics, audio, and video). Second, social TV offers ubiquitous services that are available at anytime, anywhere, on any device at an affordable cost, with personalized experiences [2]. Finally, with the latest multi-screen or second-screen [3] technology, users can transfer the ongoing sessions from one device to another, without any service interruption. Nonetheless, given its highly regarded value, large-scale deployment of social TV has been limited, if not totally absent.

In response to this market trend, we have designed and implemented a multi-screen social TV system [4]–[6] over a Cloud-Centric Media Network (CCMN) [7]. CCMN leverages the cloud-computing paradigm to transform the media value chain, by encapsulating a set of media services, including distribution, rendering, processing and analytics, into a middleware and exposing them via Application Programming Interfaces (APIs) for application development. The multi-screen cloud social TV is a pilot application on this platform, where one particular feature is *video teleportation* [6]. In particular, one can easily migrate video session back and forward among different devices, with intuitive human-computer interactions. The supporting technology is to instantiate a virtual machine in the cloud as a cloud clone for each user[1]. It maintains all the user sessions, transcodes contents and inserts personalized ads. Moreover, the cloud clone would migrate to different locations in the cloud, to optimize its performance.

In our multi-screen cloud social TV application, one design objective is to minimize the monetary cost, potentially making the service affordable to the general public. One possible deployment scenario of our service is to rent cloud resources from a vendor-neutral provider (e.g., AWS, Azure, etc). In this case, we need to intelligently manage the resource rental cost, dominated by the bandwidth cost. It consists of two parts, including content transmission and cloud clone migration, both of which in turn depends on the location of the cloud clone. Therefore, an optimal policy to migrate the cloud clone should be in place to minimize the total monetary cost.

Similar problems have been addressed by previous research, often with a different cost metrics. In [8], the authors examined the energy tradeoff between video transport and processing for multi-view video streaming, to obtain the optimal location of video processing node with the lowest energy cost. In [9], the authors aimed to balance the tradeoff between content transmission cost and storage cost, to find an optimal content placement strategy with minimal monetary cost. However, none of those works can be directly applied to our problem, because they only investigated the fixed placement method, while we want a dynamic cloud clone migration strategy.

In this paper, we formulate the cost-minimization problem in the context of Markov Decision Process (MDP). Specfically, we adopt a Markov process to model the user watching behavior across TV and smartphone. The objective is to minimize the monetary cost of operating the video teleportation service, by migrating the cloud clone to the best location, as the user

[1]We will use virtual machine and cloud clone interchangeably in this paper.

978-1-4799-1353-4/13/$31.00 ©2013 IEEE

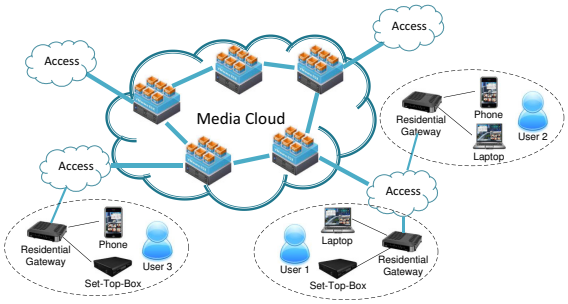
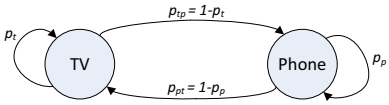


Fig. 1: Multi-screen cloud social TV Architecture

(a) Real user case



(b) Device switching model as a Markov chain

Fig. 2: User behavior model on video session migration

shifts his device. Under this framework, we first consider a fixed placement policy and an offline policy to obtain an upper and a lower bound for the optimal cost. We then propose an online policy based on a dynamic programming approach. Extensive simulations are conduct to evaluate the proposed strategies, with inputs from real traces. Our numerical results suggest, up to 10% monetary cost can be saved in typical scenarios. The cost saving depends on the length of content delivery path, the data size of VM migration and the user behavior pattern. These insights would offer operational guidelines to deliver cost effective multi-screen social TV service over CCMN, potentially easing its adoption.

The rest of this paper is organized as follows. In Section II we present system architecture and problem formulation. In Section III, we proposes three alternative algorithms to derive the optimal cloud clone migration policy. In Section IV, numerical evaluation of system performance is conduct with real-trace. Section V summarizes this paper.

II. System Overview & Problem Formulation

In this section, we first present an architecture of the multi-screen cloud social TV system. Then we focus on the video teleportation feature, including its user behavior model and cost model. Finally, we formulate the minimum-cost cloud clone migration problem under a Markov Decision Process.

A. System Architecture

In Figure 1, we present a systematic view of our cloud based multi-screen social TV system. The system is built upon a media cloud, which provides on-demand media services, including content distribution, media processing, content adaptation and media analytics via APIs. End users are connected via residential gateways to the cloud.

Our system offers a highly-touted multi-screen experience via *video teleportation*, in which one user simultaneously operates multiple devices (e.g., TV and smartphone) and desires to migrate video session from one device to another without interruption. The enabling technology for this feature is that each user is represented by a virtual machine (VM), serving as his proxy in the cloud, to manage all the associated devices and session information. It also offers other functionalities, such as video transcoding and ad-insertion to support personalized multi-screen experience. We call it cloud clone in our research. When substantiated, it locates in a position that best serves the end user. As the user moves around or shift sessions from one device to another, the cloud clone will migrate correspondingly. In addition, the VM migration also aims to minimize its monetary operational cost.

B. System Assumptions

1) Content Processing Model: In this paper, we assume that the cloud clone performs two content processing functions, including video transcoding and ad insertion. Through these processing procedures, a content of size B_o will be changed to B_t in the following two cases.

$B_o \leq B_t$: When the user is consuming the content on a bigger screen (e.g., TV), B_o would be smaller than B_t. In this case, the cloud clone will insert a few personalized ads into the original content, and delivery the combined video streaming to end users. This method has become a key online monetization strategy [10]. At the same time, some set-top boxes may not support the latest video format (e.g., H.264 high profile), requiring the cloud clone to transcode the content into a compatible one with bigger size (e.g., H.264 baseline profile). Note, the transmission cost for the cloud clone to load advertising videos is ignored, since we assume the cloud has pre-loaded all those data at each node.

$B_o > B_t$: When the user is consuming the content on a smaller screen (e.g., mobilephone), B_o would be larger than B_t. The reason is, the video resolution and bitrate for such a device are significantly lower than those of the original one.

2) User Behavior Model: We model user behavior across different devices as a Markov process, which has been widely adopted to characterize a variety of user behaviors on online social activities [11] and IPTV interactions [12]. Without loss of generality, the user is assumed to switch between two devices (TV and phone), thus the model has two corresponding states. This model can be easily extended for more devices.

Figure 2(a) captures a real use case when an user shifts his attention between TV and mobilephone. This scenario can be modelled by a Markov process with two states, as illustrated in Figure 2(b). In this model, the state transition matrix is completely determined by p_t (the probability in which the user uses TV in both the current and the next time slot) and p_p (the probability in which the user uses mobilephone in both the current and the next time slot). Accordingly, we have $p_{tp} = 1 - p_t$, $p_{pt} = 1 - p_p$, where p_{tp} and p_{pt} are the user device switching probability between TV and phone.

3) Cost Model: In supporting cloud clone migration, the system would incur three cost components, including a transmission cost, a migration cost (if any) and a processing cost. The transmission cost infers to the bandwidth cost when videos are transmitted from the source to the user. The migration cost corresponds to the bandwidth consumed for the cloud clone migrates from one node to another. The processing cost corresponds to the resources consumed when the cloud clone responses to user requests, and processes the requested contents. Note the processing cost is constant no matter where the cloud clone is placed. As a result, we only focus on the transmission cost and the migration cost in this paper.

In this research, we consider an operational model in which the media service provider rents cloud resources from a vendor-neutral cloud provider (e.g., Amazon AWS and Windows Azure). We adopt the on-demand pricing model in this work, because media service is often real-time.

For a chosen content, we assume a content delivery path from the source to the end user. The path length is L, and there are overall $L+1$ nodes including the source node, the end user node and $L-1$ intermediate nodes. Node along the path is indexed as k according to its hop distance to the source. Note that, the cloud clone can locate at neither the user side nor the content source. We assume the monetary cost incurred by transmission is proportional to the traversed hop distance and the content size. When the cloud clone is at k, $k \in \{1, ..., L-1\}$, the transmission cost in a time slot is,

$$C_{tr}(k) = \alpha B_o k + \alpha B_t (L-k), \tag{1}$$

where α is the per hop price to transmit per GB data, B_o is the original content size, and B_t is the content size after the cloud clone processes this content.

The monetary cost for cloud clone migration, includes both VM processing cost to initialize and complete the migration, and the transmission cost to send the session data to its new destination. Therefore, the monetary cost charged by migrating cloud clone from one node k to another node k' is,

$$C_{mig}(k, k') = \alpha V_{mig} |k - k'| + \beta, \quad k \neq k', \tag{2}$$

where $k' \in \{1, ..., L-1\}$ denotes the hop distance from the cloud clone to media source after migration, V_{mig} is the migrated data volume, and β is the additional price to initialize and complete each migration.

C. Markov Decision Process Formulation

Using the system models, we introduce Markov decision process (MDP) to formulate the cloud clone migration problem. Specifically, the MDP formulation is a 4-tuple including the system state set, the scheduling action set, the state transition matrix and the cost function.

System States: We define a system state at time slot t as $s_t = (l_t, u_t)$ by jointly considering the location of cloud clone and active user device type. $l_t \in \{1, 2, ..., L-1\}$ denotes the location of cloud clone at time slot t. $u_t \in \{TV, Phone\}$ denotes the user device type at time slot t. As a result, this system state set $\mathcal{S} = \{s_1, ..., s_T\}$ can represent both the changes initiated by the user, and the corresponding cloud clone migrations made by the system, where T denotes the period that the user is interacting with our system.

Cloud Clone Migration Actions: The scheduler action set $\mathcal{A} = \{a_1, ..., a_T\}$ defines the destination of the cloud clone migration at each time slot. Specifically, we model the action a_t as the migration decision at time slot t, where $a_t = k$, $k \in \{1, ..., L-1\}$, denotes the cloud clone migrates from its current location l_t to a new place $l_{t+1} = k$. We treat $l_t = l_{t+1}$ as the case that no migration is taken at time slot t.

State Transition: The transition from state $s_t = s$ to $s_{t+1} = s'$ is determined by both the user behavior model and the cloud clone migration decision. Since the user decision is independent of the cloud clone migration, we have the transition probability $\mathcal{P}_{a_t}(s, s')$, that action a in state $s = (l, u)$ at time t will lead to state $s' = (l', u')$ at time slot $t+1$ as,

$$\begin{aligned} \mathcal{P}_{a_t}(s, s') &= Pr_{(s_{t+1}=(l',u')|s_t=(l,u),l'=a_t)} \\ &= Pr_{(l_{t+1}=l'|l_t=l,l'=a_t)} Pr_{(u_{t+1}=u'|u_t=u)}, \end{aligned} \tag{3}$$

where $Pr_{(u_{t+1}=u'|u_t=u)}$ can be obtained from the transition matrix of user behavior model, and $Pr_{(l_{t+1}=l'|l_t=l,l'=a)}$ is determined by the action policy $\pi(s_t) = a_t$.

Cost Function: We define the cost function $\mathcal{R}_{a_t}(s, s')$ as the total monetary cost consumed during the period from time t to $t+1$. This cost includes both media transmission cost and cloud clone migration cost. As a result, we have,

$$\mathcal{R}_{a_t}(s, s') = C_{tr}(k) + (1 - \delta(k - k')) C_{mig}(k, k'), \tag{4}$$

where $\delta(x)$ is the indicator function that $\delta(x) = 1$ when $x = 0$, and $\delta(x) = 0$ otherwise. Thus $1 - \delta(k - k')$ indicates whether the migration should be taken at time slot t.

Optimization Objective: The goal is to find an optimal migration policy $\pi(s_t) = a_t$, so that the cost can be minimized. This objective can be further formulated as an unconstrained optimization problem over a finite time horizon, as,

$$\min_{a_t} \sum_{t=0}^{T-1} \mathcal{R}_{a_t}(s_t, s_{t+1}). \tag{5}$$

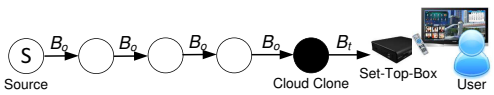
III. Migration Strategies for Cloud Clone

In this section, we start with three simple case studies, to illustrate the problem and the fundamental trade-off between transmission cost and migration cost. Following that, we propose three alternative methods to solve this problem.

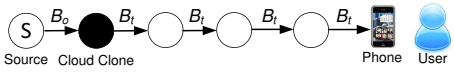
A. Case Studies for Cloud Clone Placement

Figure 3 illustrates three cases for optimal cloud clone placement. In these cases, we consider the content delivery path as a line topology. Node S serves as the media source, and the black node serves the cloud clone.

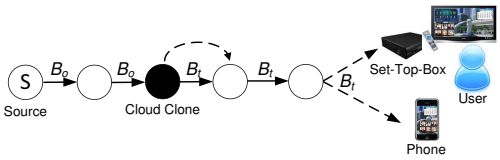
1) Case A: In this case, as shown in Figure 3(a), the end user is always consuming the content on one device with a larger screen (i.e. TV) and the retargeted content has a larger size than that of the original content (i.e., $B_o < B_t$). It can be shown that the optimal location for the cloud clone would be the node nearest to the user along the delivery path, to minimize the transmission cost.



(a) Optimal cloud clone placement when $B_o < B_t$



(b) Optimal cloud clone placement when $B_o \geq B_t$



(c) Possible cloud clone scheduling when B_t changes

Fig. 3: Case studies on cloud clone migration

2) Case B: In this case, as shown in Figure 3(b), the end user is always consuming the content on one device with a smaller screen (i.e., smartphone) and the retargeted content has a smaller size than that of the original content (i.e., $B_o > B_t$). It can be shown that the optimal location for the cloud clone would be the node nearest to the content source along the delivery path, to minimize the transmission cost.

3) Case C: In this case, as shown in Figure 3(c), the user switches between two devices when consuming the same content. In the two phases, the optimal location for the cloud clone could change in the process to minimize the total cost.

B. Fixed Placement & Cost Upper Bound

In this subsection, we propose a fixed placement strategy for the cloud clone. We assume that the cloud clone does not migrate during the content consumption period T. In this case, the original problem is translated into a new problem of finding one location along the content delivery path that would result in the minimum transmission cost. The optimal cloud clone location can be founded by an exhaustive searching approach, in which we simply compute the cost for all $L-1$ locations and choose the one with the least cost.

Since the optimal location is a special case, the monetary cost resulted from this fixed placement policy is an upper bound for the minimum cost of the original migration problem.

C. Offline Algorithm & Cost Lower Bound

In this subsection, we investigate an offline algorithm, based on dynamic programming approach, to solve the optimal migration problem. Specifically, we assume that the scheduler has the knowledge of all user behaviors $\mathcal{U} = \{u_1, ..., u_T\}$ in advance. In this case, we have,

$$\tilde{\mathcal{P}}_{a_t}(s, s') = \begin{cases} 1, & u = u_t, u' = u_{t+1}, l' = a_t; \\ 0, & otherwise \end{cases}. \quad (6)$$

We denote $v(s_t)$ as the minimal aggregated cost from s_t to s_T. Using value iteration in the backward induction, we have,

$$v(s_t) = \min_{a_t}\{\sum_{s_{t+1}} \tilde{\mathcal{P}}_{a_t}(s_t, s_{t+1})(\mathcal{R}_{a_t}(s_t, s_{t+1}) + v(s_{t+1}))\}, \quad (7)$$

where $v(s_T) = 0$ gives an initial value. By using this iterative equation, we can find the optimal policy, given by,

$$\pi(s_t) = \arg\min_{a_t}\{\sum_{s_{t+1}} \tilde{\mathcal{P}}_{a_t}(s_t, s_{t+1})(\mathcal{R}_{a_t}(s_t, s_{t+1}) + v(s_{t+1}))\}. \quad (8)$$

Since the scheduler has perfect information, it follows that the monetary cost resulted from this offline algorithm is a lower bound for the original clone migration problem. We will verify this result with numerical simulations in Section IV.

D. Online Algorithm

This subsection proposes an online algorithm, based on dynamic programming, to solve the problem. In this case, the scheduler only knows the transition matrix of user behavior in advance and can observe the user pattern as time evolves.

Similar to the offline method, we define $v(s_t)$ as the minimal expected aggregated cost from s_t to s_T as,

$$v(s_t) = \min_{a_t}\{\sum_{s_{t+1}} \mathcal{P}_{a_t}(s_t, s_{t+1})(\mathcal{R}_{a_t}(s_t, s_{t+1}) + v(s_{t+1}))\}, \quad (9)$$

where $\mathcal{P}_{a_t}(s_t, s_{t+1})$ is the transit probability as Eq. (3).

By still using a backward Bellman equation, we can derive the optimal policy as,

$$\pi(s_t) = \arg\min_{a_t}\{\sum_{s_{t+1}} \mathcal{P}_{a_t}(s_t, s_{t+1})(\mathcal{R}_{a_t}(s_t, s_{t+1}) + v(s_{t+1}))\}. \quad (10)$$

It can be shown that the minimum cost resulted from this online algorithm falls between the lower bound and upper bound derived in previous two subsection, as verified by performance evaluation in next section.

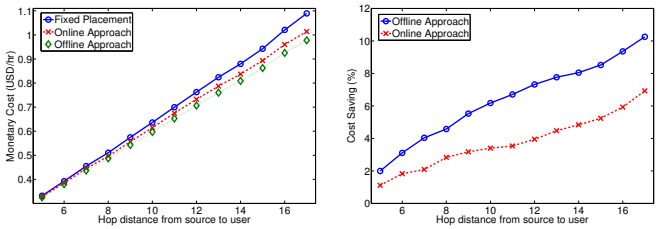
IV. Performance Evaluation

This section verifies the performance of those algorithms via numerical simulations, based on real application scenarios.

A. Experimental Settings

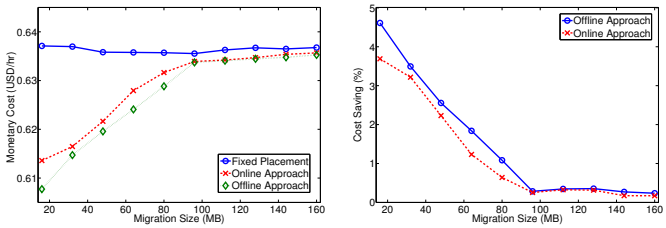
We adopt the price information from Windows Azure [13]. Specifically, the per-hop price to transmit one Gigabyte data is $\alpha = 0.12$ USD/GB, the price to migrate cloud clone (i.e., create a new VM, and delete the previous one after all the sessions have been migrated) is $\beta = 0.02$ USD, if we use the extra small VM instance to implement cloud clone.

We use Cisco's Media Experience Engine [14] as a reference to obtain the bitrate information. Specifically, we define $b_o = 1500$ kbps as the bitrate of live streaming sources with 640x480 resolution delivered by Veoh Network (a popular Internet television provider). We set $b_t^t = 2000$ kbps as the bitrate, after the cloud clone inserts video advertisements into the original video, and transcodes it into AVC (Advanced Video Coding) SD (Standard Definition) output with 640x480 resolution for TV viewing. We set $b_t^p = 400$ kbps as the bitrate when the source is transcoded into iPhone compliant MPEG-4 output with 480x320 resolution for iPhone player.



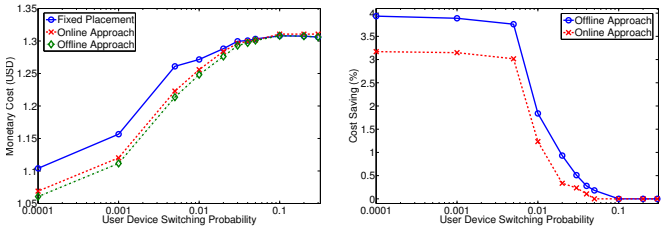
(a) Minimal cost per hour (b) Cost saving

Fig. 4: Minimal monetary cost vs. delivery path length



(a) Minimal cost per hour (b) Cost saving

Fig. 5: Minimal monetary cost vs. migration size



(a) Minimal cost per hour (b) Cost saving

Fig. 6: Minimal monetary cost vs. user behavior pattern

We assume each experiment lasts for 2 hours, and each time slot is 10 seconds, since the video teleportation operation could take around 10 seconds [4]. Thus, there are in total $T = 720$ slots in each period, and $B_o = 1.875$ MB, $B_t^t = 2.5$ MB, and $B_t^p = 0.5$ MB in each time slot. In every experiment, we run all the algorithms for 1000 rounds. At each round, we generate the user behavior trace according to the Markov model in section II-B. Only the average values are reported.

B. Monetary Cost

In this subsection, we evaluate the cost resulted from different algorithms. Our focus is to understand the impact of various system parameters on the monetary cost, ultimately providing operational guidelines for deployment.

1) Delivery Path Length: Figure 4 shows the monetary cost and the cost savings compared to the fixed placement, as a function of delivery path length L, where $p_t = p_p = 0.99$, and $V_{mig} = 8$ MB.

We have a few observations from this experiment. First, the cost resulted from the online algorithm always falls between the cost resulted from the fixed placement policy and the offline algorithm, as suggested in our analysis in Section III. This observation can be also applied to Figure 5 and 6. Second, the monetary cost increases almost linearly as the delivery length L increases. This observation can be traced by the linear cost model as in Eq. (1) and (2). Finally, compared to the fixed placement, our algorithm provides significant cost saving, which increases as the length of the content delivery path increases. For example, when $L = 4$ the optimal costs from three algorithms are almost the same, whereas the cost saving reaches more than 10% when $L = 17$. In the latter case, if the service would be offered to 100 users for one year, the monetary cost saving can be as much as $100,000.

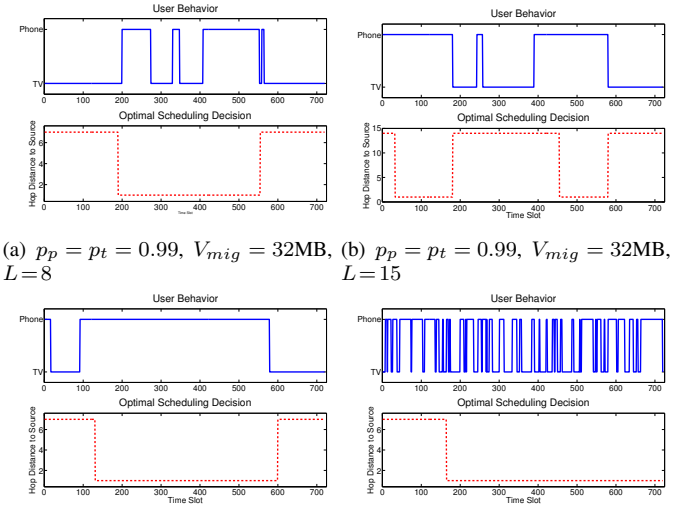
2) Migration Size: Figure 5 illustrates the monetary cost and the cost savings, in terms of the migration size V_{mig}, where $p_t = p_p = 0.99$, and $L = 10$.

This experiment reveals a few insights. First, the cost resulted from the fixed placement remains the same when V_{mig} changes, because fact that there is no migration based on this policy. Second, a threshold effect is observed, where the cost remains constant when the migration size is beyond the threshold. Specifically, for the set of chosen parameters in the experiment, when $V_{mig} < 100$ MB, as V_{mig} increases, the migration cost grows; when $V_{mig} > 100$ MB, the monetary cost remains constant and is close to that from the fixed placement. This effect can be understood as follows. When the migration size is beyond a threshold, the migration cost would be higher than the minimum transmission cost when the cloud clone locates in the best location. In this situation, it would be better if the cloud clone stays at its current location, reducing to the case for the fixed placement policy.

3) User Behavior Pattern: Figure 6 illustrates the monetary cost and the cost savings compared to the fixed placement, as a function of user device switching probability $p_s = p_{tp} = p_{pt}$, where $L = 10$, $V_{mig} = 64$ MB, and $p_p = p_t$. This experiment also reveals a threshold effect, in which the system behaves differently on the two sides of the threshold transit probability ($p_s = 0.01$ in this case).

On one hand, when $p_s > 0.01$, the cost savings are very limited. Because if the user switches his device too often, the high migration cost would prevent the cloud clone moving such often. And the migration policy reduces to the fixed placement policy. As such, the cost saving diminishes as the transit probability increases.

On the other hand, when $p_s < 0.01$, the minimum cost grows, but the cost saving slightly decreases, as p_s increases. This can be understood as follows. When the device switching probably is relatively small, the the time spent on one device dominates the other device; and as p_s increases, the relevant weight on one device starts to diminish. For the fixed placement policy, the chance in which the cloud clone is not placed in the optimal location increases, resulting in a higher transmission cost; for the online algorithm, the frequency in which the cloud cloud migrate is smaller than that for device switching, resulting in a higher cost as in the fixed strategy. The flat curve for the cost saving can be understood by examing a saving function $\Delta(k, k')$ by migrating cloud clone



(a) $p_p = p_t = 0.99$, $V_{mig} = 32$MB, $L = 8$ (b) $p_p = p_t = 0.99$, $V_{mig} = 32$MB, $L = 15$

(c) $p_p = p_t = 0.99$, $V_{mig} = 16$MB, $L = 8$ (d) $p_p = p_t = 0.9$, $V_{mig} = 32$MB, $L = 8$

Fig. 7: Examples of the optimal migration scheduling policy

from location k to k', defined as,

$$\Delta(k, k') = t_m(C_{tr}(k') - C_{tr}(k)) - C_{mig}(k, k'), \tag{11}$$

where t_m is the time period from now to the next migration (or the end of period T if there is no further migration). By substituting Eq. (1) (2) into Eq. (11), we obtain,

$$\Delta(k, k') = \begin{cases} \alpha(k' - k)(t_m(\bar{B}_t - B_o) - V_{mig}) - \beta, & k' \geq k \\ \alpha(k' - k)(t_m(\bar{B}_t - B_o) + V_{mig}) - \beta, & k' < k \end{cases}, \tag{12}$$

where $\bar{B}_t$ is the average retargeted content size during t_m,

$$\bar{B}_t = (t_p B_t^p + (t_m - t_p) B_t^t)/t_m, \tag{13}$$

and t_p is the time the user spends on phone during t_m.

In this case, the cost saving is determined by $(\bar{B}_t - B_o)$, which is in turn determined by t_p. Since t_p can not vary in a large scale when $p_s < 0.01$, the cost saving remains almost the same, when p_s changes.

C. Optimal Migration Polices

In this subsection, we investigate the optimal migration policies through numerical simulations. Figure 7 presents four examples of the online cloud clone migration scheduling.

We notice an interesting observation in all the examples, that the optimal cloud clone location is either at the nearest or the furthest node to user, regardless of delivery path length, migration size and user behaviors. This can be understood from Eq. (12), the cost saving function $\Delta(k, k')$. Specifically, the scheduler can check whether $\max \Delta(k, k') > 0$ to trigger a migration. If a migration is required, the place which leads to the maximal cost saving, is selected as the destination. It can be seen that, $\Delta(k, k')$ is a linear function of $(k' - k)$. As a result, if $\Delta(k, k') > 0$, it is advantageous to migrate the cloud clone to the furthest node to maximize $(k' - k)$, so that the cost saving can be maximized. Otherwise, migration is not required. Hence it is optimal to place the cloud clone at either the nearest or the furthest node to user.

V. Conclusion and Future Works

This paper investigated the problem on minimizing monetary cost via cloud clone migration in our social TV implementation. We formulated it as a Markov Decision Problem, to balance a trade-off between the transmission cost and the migration cost. Under this framework, we first consider a fixed placement policy and an offline policy to obtain an upper and lower bound for the optimal cost. We then proposed a more practical online policy. The results indicated, up to 10% cost can be saved in typical use scenarios, by optimally migrating the cloud clone. The savings can be affected by the delivery path length, the migration size and user behavior pattern. Moreover, we also found the optimal cloud clone location is either at the nearest or the furthest node to the user.

This work only considered the traffic from the source to the users. The results might not hold if the traffic are from both sides (e.g., video chatting when watching). We will investigate it in our future works. Besides, our multi-screen social TV system has been implemented on top of a private cloud at NTU. It has been exposed to over 200 students in practice. We will collect real user behaviors to verify our strategy.

Acknowledgment

The authors would like to thank anonymous reviewers for their insightful comments. This work benefits significantly from technical discussions with Dr. Marie-Jose Montpetit from MIT Media Laboratory, Prof. Dapeng Wu and Prof. Xiaolin Li from University of Florida, Prof. Jianwei Huang from the Chinese University of Hong Kong. This work was supported in part by Singapore MOE Tier-1 Grant (RG 31/11) and a research gift fund from Microsoft Research Asia.

References

[1] M. Montpetit and M. Médard, "Social television: Enabling technologies and architectures," *Proceedings of the IEEE*, pp. 1395–1399, 2012.
[2] Z. Yu, X. Zhou, and et al., "A hybrid similarity measure of contents for TV personalization," *Multimedia systems*, vol. 16, pp. 231–241, 2010.
[3] P. Cesar and et al., "Usages of the secondary screen in an interactive television environment: Control, enrich, share, and transfer television content," *Changing television environments*, pp. 168–177, 2008.
[4] Y. Jin and et al., "Inter-screen interaction for session recognition and transfer based on cloud centric media network," in *IEEE ISCAS*, 2013.
[5] Y. Jin, T. Xie, and et al., "Multi-screen cloud social TV: Transforming TV experience into 21st century," in *ACM MM, Demo, (Accepted)*, 2013.
[6] Lianhe Zaobao, "NTU develops video teleportation technology," http://www.zaobao.com.sg/edu/pages5/edulive120814.shtml, 2012.
[7] Y. Jin, Y. Wen, G. Shi, G. Wang, and A. Vasilakos, "CoDaaS: An experimental cloud-centric content delivery platform for user-generated contents," in *IEEE ICNC*, 2012, pp. 934–938.
[8] J. Llorca, K. Guan, and et al., "Energy efficient delivery of immersive video centric services," in *IEEE INFOCOM*, 2012, pp. 1656–1664.
[9] Y. Jin, Y. Wen, and et al., "Towards monetary cost effective content placement in cloud centric media network," in *IEEE ICME*, 2013.
[10] T. Mei, X. Hua, L. Yang, and S. Li, "VideoSense: towards effective online video advertising," in *ACM MM*, 2007, pp. 1075–1084.
[11] V. S. Tseng and K. W. Lin, "Efficient mining and prediction of user behavior patterns in mobile web systems," *Information and Software Technology*, vol. 48, no. 6, pp. 357–369, 2006.
[12] V. Gopalakrishnan, R. Jana, and et al., "Understanding couch potatoes: measurement and modeling of interactive usage of IPTV at large scale," in *ACM IMC*, 2011, pp. 225–242.
[13] "Azure pricing," http://www.windowsazure.com/en-us/pricing/details/.
[14] Cisco, "User guide for cisco media experience engine 3000," http://www.cisco.com/en/US/docs/video/mxe/2_x/2.0/user/guide/mxe_20_ug.pdf.