Resource-Saving File Management Scheme for Online Video Provisioning on Content Delivery Networks

Wen-Hsing Kuo, *Senior Member, IEEE*, and Yung-Hsuan Lin

Abstract—Content delivery networks (CDNs) have been widely implemented to provide scalable cloud services. Such networks support resource pooling by allowing virtual machines or physical servers to be dynamically activated and deactivated according to current user demand. This paper examines online video replication and placement problems in CDNs. An effective video provisioning scheme must simultaneously (i) utilize system resources to reduce total energy consumption and (ii) limit replication overhead. We propose a scheme called adaptive data placement (ADP) that can dynamically place and reorganize video replicas among cache servers on subscribers' arrival and departure. Both the analyses and simulation results show that ADP can reduce the number of activated cache servers with limited replication overhead. In addition, ADP's performance is approximate to the optimal solution.

Index Terms—Content delivery networks, resource management, video streaming

1 Introduction

In recent years, content delivery networks (CDNs) [1] have been widely implemented to provide scalable cloud services. Fig. 1 shows a typical local CDN whose servers are located in the same place. The request of each visitor (A) is first processed and identified by a gateway server (B) and then directed to a cache server (CS) (C) in the server farm. CSs are typically virtual machines and can dynamically provide various services by executing different contents loaded from a backhaul database (D). This cloud-based architecture can provide a high degree of scalability and flexibility for service provisioning because it adaptively utilizes storage space, computing power, and network bandwidth by activating different numbers of CSs. As mentioned in Refs. [2], [3], [4], the use of CSs is critical because the average loading of a single CS is typically substantially lower than its maximum capability. Therefore, minimizing the number of activated CSs is correlated to, if not equal to, minimizing the total energy consumption because of two reasons. First, supporting an activated virtual/physical machine requires considerable power compared with the dynamic workload of visitors. Second, the system resources (e.g., network bandwidth and CPU time) and power consumption required by each visitor are almost identical. Many related studies, such as [5], [6], [7], [8], [9], [10], have focused on analyzing or reducing the number of activated CSs in a CDN.

CDNs are effective platforms for providing various types of services. Among them, on-demand video provisioning is a popular application, allowing numerous users to arbitrarily request videos from a massive database. Video websites such as YouTube, Vimeo, and DailyMotion are examples. When a visitor arrives and requests a video clip, the system must assign a serving CS and copy a replica of the clip from the backhaul database if the CS does not cache the clip for other visitors. Because each CS has limited capability, the total number of video clips it stores and the total outgoing bandwidth of subscribers it bears is limited by its space and bandwidth constraints. Fig. 2 illustrates an example of this two-dimensional resource allocation problem, where the length and width respectively express the space and bandwidth capacity of a CS, and u and p respectively illustrate users and their requested programs. Here, Visitors 1, 2, and 3 require Video Programs A, B, and C respectively. When User 4 is later directed to this server, an additional unit of bandwidth is required because each video is independently transmitted/played by each user. However, Visitor 4 does not occupy an additional unit of storage space because Program A has already been requested by Visitor 1.

Because of the many time-variant requirements of video clips, intelligently placing videos among CSs and determine their serving subscribers without violating capacity and bandwidth limits is challenging. Typical CDN management schemes in data centers fail to address this video provisioning problem for three reasons. First, rather than being separately required by a specific user/subscriber, a video clip may be simultaneously accessed by numerous online subscribers. Second, the two resource requirements (i.e., bandwidth and storage size) exhibit different characteristics (i.e., when different visitors request the same clip, their storage requirements can be combined, whereas their bandwidth requirements are independent). Third, because video placements and server selections are conducted online as visitors arrive and depart, the migration/management of clip overhead should be limited to provide real-time responsiveness. Therefore, a new resource management scheme that is

• *The authors are with the Department of Electrical Engineering, Yuan-Ze University, Taoyuan, Taiwan.*
E-mail: Whkuo@saturn.yzu.edu.tw, s980537@ee.yzu.edu.tw.

Manuscript received 19 Aug. 2014; revised 1 July 2015; accepted 2 July 2015. Date of publication 15 July 2015; date of current version 16 May 2016.
Recommended for acceptance by J.D. Bruguera.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TC.2015.2456018

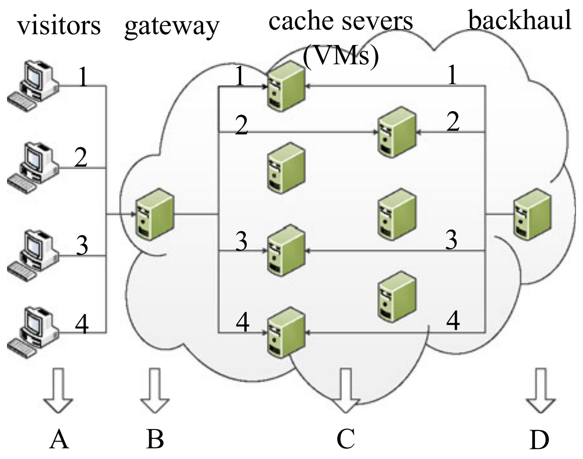


Fig. 1. Illustration of a local CDN.

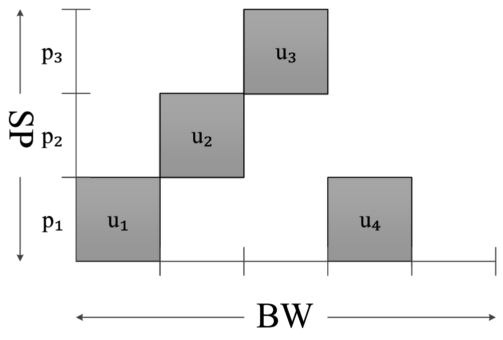


Fig. 2. Example of a CS in a CDN.

specifically designed for provisioning online videos in CDNs is required.

Many studies have been proposed to address different challenges of CDNs. In [11], [12], [13], several feasibility concerns of using virtual machines, including reliability, performance interference, and resource contention, have been discussed. Traditional resource management studies [14], [15], [16] have placed files among a fixed number of servers and focused on goals such as fulfilling users' bandwidth requirement or optimizing server use. In [14], a file placement scheme was proposed for balancing the loading of hard drives in servers. Moreover, in [15], an approach was designed to allocate video files among multiple servers. This approach balances the load and reduces the failure rate of services by deciding the number of video replicas based on server number, video length, and encoding rate. Under similar modeling, a genetic algorithm was proposed in [16]. The discussed methods are based on different assumptions (i.e., fixed number of servers) and objectives (load-balancing) and are thus not suitable for solving our replica placement problem. Some researchers have studied the inner routing between servers or datacenters inside a CDN. In [17], using CDNs to conduct video conferences was discussed. Meng et al. [18] examined server grouping and proposed a scheme that can both reduce the number of switches and improve transmission efficiency. In [19] and [20], routing methods have been proposed among different datacenters of a CDN, thereby lowering carbon footprints and electricity costs and fulfilling users' service requirements. Because we focus on local CDNs where CSs are located in the same place, routing between CSs and datacenters was not the main concern.

Research has also investigated energy and resource saving in CDNs. In [21], user requests were categorized into different classes. To reduce operational costs, the routes of users were established based on the loading and energy costs of each CS. The current study examined a CDN whose CSs are remotely distributed and, thus, faces different challenges and issues. Some studies have focused on reducing the number of activated servers in local CDNs and have had objectives similar to those of our study. The schemes proposed in [2] and [5] place each "workload" among servers based on servers' "degrees of loading." Similarly, the method proposed in [6] allocates heavier workloads to servers with fewer resources to improve resource utilization. This work models the placement problem as the traditional "1-D bin-packing" problem and does not consider the multiple resources (e.g., bandwidth and storage space) of each CS. This type of modeling fails to solve our placement problem, even when generalized to multiple-dimension bin-packing, because it assumes each subscription has independent storage requirements. In [7], a new method called CPA was proposed, which separates CSs into two groups: computation servers and data servers. Under CPA, the requested services are processed on the computation servers, whereas the data is stored on the data servers. This work also has different assumptions and thus cannot be adapted to video stream provisioning. In [8] and [9], capacity management schemes for data centers were discussed. By activating the appropriate number of servers at the appropriate time, the response time and power consumption of the data-center can be reduced. In [10], an analytical model was proposed for balancing throughput performance and power consumption. However, these works have focused on the management of general-purpose machines that serve user requests independently. They do not apply the specific properties of video-on-demand requests, such as combinable space requirements, as we mentioned.

Based on our research, no studies have examined this video placement problem in CDNs. This paper introduces a new problem called resource-saving video placement (RSVP) and proposes a scheme called adaptive data placement (ADP). Through analysis and simulations, we demonstrate the two main advantages of ADP: (i) the worst-case performance difference between ADP and the optimal solution can be guaranteed, and (ii) the replication overhead on each arrival or departure of a visitor is limited. Because ADP is based on common assumptions, it can be applied to various types of CDNs to improve their resource and power efficiency.

The remainder of this paper is organized as follows. Section 2 introduces the system model and notations and formally declares the placement program. In Section 3, the operation of ADP is detailed. Section 4 analyzes the gap between ADP performance and optimal performance and evaluates the ADP operational overhead. The performance and overhead of ADP is also evaluated through simulations in Section 5. Finally, the paper is concluded in Section 6.

2 The Resource-Saving Video Placement Problem

2.1 System Model and Assumptions

Here we introduce the four assumptions of our model. First, each video clip in the backhaul database that serves as the basic management unit has identical storage size and transmission bandwidth. This can be achieved by cutting long programs into many pieces and encoding them with the same codec. Second, all CSs, whether implemented with virtual or physical machines, are assumed to have identical available space and bandwidth, which is a common and reasonable setting in a server farm. Third, the costs of replicating a video clip from the backhaul database to any CS are identical because we focus on local CDNs whose CSs are located approximately. Finally, the overhead of deleting a video replica on any CS is neglected because no data transfer between CSs and databases occurs. All of the previous assumptions are common and realistic and have thus been used in many prior studies [6], [7], [8], [9], [10], [11].

The notations used throughout this paper are as follows. The number of total available video clips in the backhaul database is denoted by integer M. The set of these clips is represented by $\boldsymbol{P} = \{p_1, p_2, p_3 \ldots p_M\}$, where the mth clip is labeled p_m. Similarly, the number of activated CSs (i.e., those available for serving visitors) is N. The set of activated CSs is $\boldsymbol{S} = \{s_1, s_2, s_3 \ldots s_N\}$, where s_n denotes the nth activated CS. Because of the identical size of video clips and the same available resources of each CS, the maximal number of clips each CS can accommodate (i.e., the space capacity) is denoted by SP. Similarly, the maximal number of transmissions each CS can simultaneously support (i.e., the bandwidth capacity) is BW.

Next, we consider the requests of visitors. Let integer I be the number of visiting subscriptions. Each subscriber independently watches a video clip, which is denoted by u_i, $1 \leqq i \leqq I$. Each subscription u can be further expressed by $u = (s_u, p_u)$, where s_u and p_u respectively refer to its responsible CS and requested video clip. To describe each CS's loading condition, we let $\boldsymbol{P}_s = \{p_u | \forall u, s_u = s\}$ be the set of video replicas placed in CS s. Therefore, the residual storage space and bandwidth of each CS s is denoted by $SP_s = SP - |P_s|$ and $BW_s = BW - |\{u | \forall u, s_u = s\}|$. To trace the distribution of each video clip, we use $\boldsymbol{S}_p = \{\mathrm{CS\ s} | \mathrm{p} \in \mathrm{P_s}\}$ to represent the set of servers that contain clip p, and we let $CN(s,p) = |\{u | s_u = s, p_u = p\}|$ refer to the number of subscriptions requesting clip p in CS s.

To manage the resources of CSs, we define the four types of loading conditions later used in the proposed scheme: full (FUL), space full (SPF), bandwidth-full (BWF), and open server (OPS). Type FUL CSs refers to fully loaded CSs with no vacant space and bandwidth (i.e., $\boldsymbol{S}^{FUL} = \{CS\, s | BW_s = 0, SP_s = 0\}$; type SPF CSs refers to those with vacant bandwidth but no empty available space (i.e., $\boldsymbol{S}^{SPF} = \{CS\, s | BW_s > 0, SP_s = 0\}$. CSs with vacant space but no additional bandwidth are called BWF and are denoted by $\boldsymbol{S}^{BWF} = \{CS\, s | BW_s = 0, SP_s > 0\}$. Finally, CSs with both available bandwidth and space are called OPS, which accommodate the replicas that cannot form a complete FUL/SPF/BWF CS. As we detail in Section 3, our proposed scheme always contains one and only one activated OPS CS (which may be empty and contain no replicas). We use s^{OPS} to denote this specific server, whose $\mathrm{BW}_{s^{OPS}} > 0$ and $\mathrm{SP}_{s^{OPS}} > 0$. The set of videos stored in SPF, BWF, and FUL CSs are expressed by $\boldsymbol{P}^{SPF}$, $\boldsymbol{P}^{BWF}$, and $\boldsymbol{P}^{FUL}$, respectively. Clearly, $\boldsymbol{P}^{SPF} = \cup_{s \in S^{SPF}} \{p_s\}$, and so are $\boldsymbol{P}^{BWF}$ and $\boldsymbol{P}^{FUL}$. All notations used throughout this paper are summarized in Table 1.

To illustrate the mentioned notations, Fig. 3 shows an example of a CDN that has four activated CSs (i.e, s_1 to s_4) and seven video clips (i.e., p_1 to p_7). Each CS contains video replicas requested by its respective serving subscriptions: $\boldsymbol{P}_{s_1} = \{p_4, p_5\}$, $\boldsymbol{P}_{s_2} = \{p_2, p_6, p_7\}$, $\boldsymbol{P}_{s_3} = \{p_1, p_2, p_3\}$, and $\boldsymbol{P}_{s_4} = \{p_4, p_5\}$. We use clip p_4 to introduce additional notations. Because p_4 is requested by subscriptions $\{u_5, u_6, u_{16}, u_{17}\}$, which are allocated to both s_1 and s_4, $\boldsymbol{S}_{p_4} = \{s_1, s_4\}$. In s_1, because subscriptions u_{16} and u_{17} are requesting clip p_4, $CN(s_1, p_4) = 2$. CS s_1 is an OPS CS because it has both space and bandwidth vacancy (i.e., $\mathrm{BW_{s_1}} = 2$, and $\mathrm{SP_{s_1}} = 1$), whereas $\mathrm{s_2}$, $\mathrm{s_3}$, and $\mathrm{s_4}$ are SPF, FUL, and BWF CSs respectively.

2.2 Problem Specification

On the basis of the previous notations, our RSVP problem is formally defined as follows:

Definition 2.1.

$$\text{Given } SP, BW, \boldsymbol{P}, \text{ and } p_{u_i} \text{ for all } 1 \leq I \leq I, \tag{1}$$

$$\text{Find } [s_{u_1}, s_{u_2}, \ldots, s_{u_I}] \text{ to minimize } N, \quad \text{Subject to} \tag{2}$$

$$s_{u_i} \in \boldsymbol{S},\ \forall\ 1 \leq i \leq I \tag{3}$$

$$\left|\{u_i | s_{u_i} = s_n\}\right| \leq BW,\ \forall\ 1 \leq i \leq I, 1 \leq n \leq N \tag{4}$$

$$\left|\{p_{u_i} | s_{u_i} = s_n\}\right| = |\boldsymbol{P}_{s_n}| \leq SP,\ \forall\ 1 \leq i \leq I, 1 \leq n \leq N \quad \text{Where} \tag{5}$$

$$|\boldsymbol{S}| = N \tag{6}$$

$$p_{u_i} \in \boldsymbol{P}, \text{ for all } 1 \leq i \leq I. \tag{7}$$

The rationale for this problem is that, given the resource capacity of each CS, the set of video clips, and the clip to which each user subscribes (i.e., (1)), we allocate each subscription to minimize the total number of activated CSs (i.e., (2)). Each subscription is placed into one of the activated CSs (3), whereas the space and bandwidth limit of each CS are not violated ((4) and (5)). Equations (6) and (7) respectively describe the activated server and the clip of the subscription.

The optimal solution to this static problem is intuitive: (i) All unserved video clips are sorted in the increasing order of their subscription numbers. (ii) Starting from the head of the queue, clips are individually placed into a server s until s becomes BWF, SPF, or FUL, or until the queue is emptied. s is then put into the solution set. (iii) If the queue is emptied, the algorithm ends. (iv) If there exists a SPF server $s1$ and a BWF server $s2$ in the solution set, proceed to step (v), otherwise go to step (ii). (v) The clip with the least subscriptions in $s1$ is exchanged with

TABLE 1
Notation Table

$\boldsymbol{P} = \{p_1, p_2, p_3 \ldots p_M\}$	The set of video clips in the system
$\boldsymbol{S} = \{s_1, s_2, s_3 \ldots s_N\}$	The set of activated CSs in the system
p_m	The *m*th video program
s_n	The *n*th activated CSs
M	The number of programs
N	The number of activated CSs
BW	The available bandwidth of each CS
SP	The available space of each CS
$\boldsymbol{U} = \{u_1, u_2, u_3 \ldots u_I\}$	The set of user subscriptions in the system
$u = (s_u, p_u)$	Subscription u which requests program p_u and is placed on CS s_u
$\boldsymbol{P}_s$	The set of video clips stored in CS s
$\boldsymbol{S}_p$	The set of servers containing program p
BW_s	The available bandwidth of CS s
SP_s	The available space of CS s
$CN(s,p) = \lvert\{\text{subscription}\, u \mid s_u = s, p_u = p\}\rvert$	The number of subscriptions served by server s and requesting program p
$\boldsymbol{S}^{SPF} = \{VM\, s \mid BW_s > 0, SP_s = 0\}$	The set of the space full (SPF) servers
$\boldsymbol{S}^{BWF} = \{VM\, s \mid BW_s = 0, SP_s > 0\}$	The set of bandwidth full (BWF) servers
$\boldsymbol{S}^{FUL} = \{VM\, s \mid BW_s = 0, SP_s = 0\}$	The set of full (FUL) servers
s^{OPS}	The open server
$\boldsymbol{P}^{SPF} = \cup_{s \in S^{SPF}} \{p_s\}$	The set of clips stored in the SPF servers
$P^{BWF} = \cup_{s \in S^{BWF}} \{p_s\}$	The set of clips stored in the BWF servers
$\boldsymbol{P}^{FUL} = \cup_{s \in S^{FUL}} \{p_s\}$	The set of clips stored in the FUL servers
$B = I$	The total required bandwidth of all subscriptions
$\mathbb{R}^{OPT}$	The number of activated CSs in the optimal solution
$\mathbb{R}^{ADP}$	The number of CSs in ADP

that with the most subscriptions in *s2*. If *s1* has insufficient bandwidth to accommodate all subscriptions of the clip of *s2*, then it takes the maximal demand it can serve. This process is repeated until *s1* becomes FUL. (vi) *s1* is placed back into the final solution set. *s2* is freed and its clips are returned to the queue, where the same clips are merged. (vii) Go to step (i). This offline solution minimizes the number of CSs because the final allocation is either space-saturated or bandwidth-saturated. In either case, the consumed CSs cannot be further reduced because CSs in the placement have the same resource bottleneck. Since SP are BW are constant and do not vary as $|P|$ changes, the complexity of this algorithm is $O(|P|^2 \, log\,|P|)$ because it is repeated at most $O(|P|)$ times, whereas step

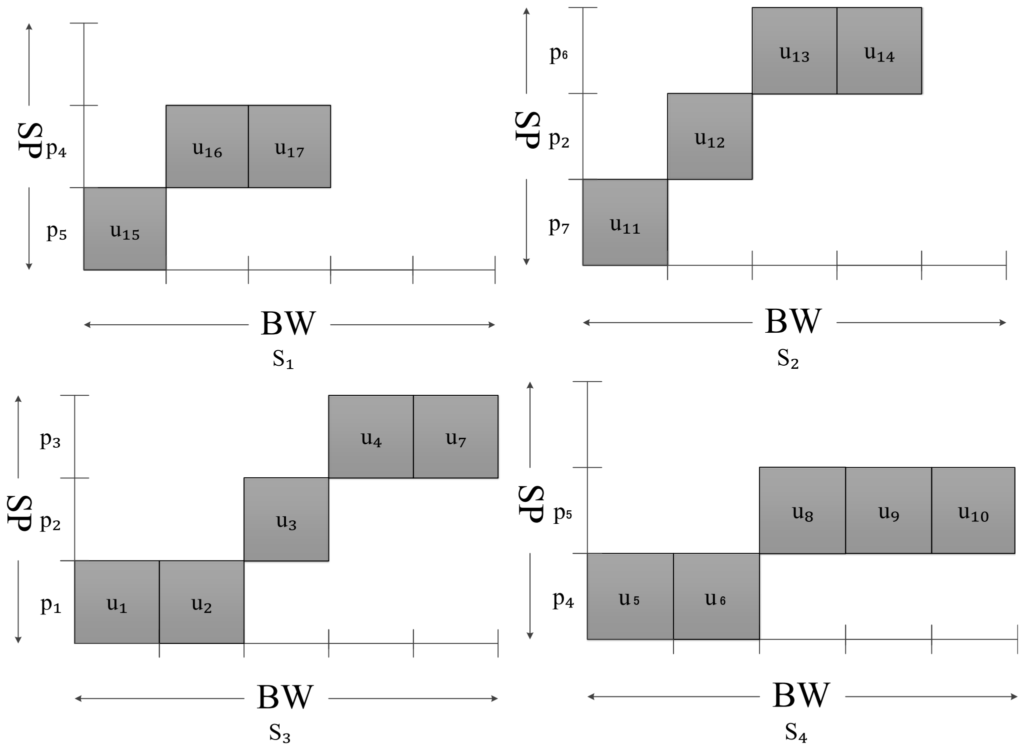


Fig. 3. Example of file placement in a CDN.

(i) has O($|P|log|P|$) complexity. This offline algorithm only serves as a performance reference and is infeasible for addressing time-variant online requirements in the real world because it must locate the whole placement again if any subscriber enters or leaves the system. A reasonable solution should entail limiting the replication overhead to an acceptable level and utilizing the system resources to reduce the number of activated CSs.

3 The Proposed Heuristic: Adaptive Data Placement

To achieve high resource utilization, our proposed scheme, ADP, follows three principles: (i) it maintains only one OPS server in a system to enable most CSs to achieve at least one aspect (i.e., bandwidth or space) of full utilization; (ii) it maintains the exclusiveness of video clips (i.e., allows at most one replica for each clip) among the OPS and SPF servers to improve space efficiency, which we demonstrate in the next section; and (iii) it conducts less physical replication to limit overhead. To increase the readability of the pseudocode, the updating processes of the following variables are not contained in the details of ADP: $\boldsymbol{P}_s$, S_p, BW_s, SP_s, $CN(s,p)$. These parameters can be updated based on their definitions after a subscription is added to or removed from a CS. The only exception is s^{OPS}, which ADP must determine and change during execution. ADP is composed of two main functions: ARRIVE and DEPART, which are respectively executed when a subscription enters and leaves a system. They are detailed in Subsection A. Additional procedures required by DEPART are detailed in Subsection B.

Notably, in the primitive version of ADP, we also considered a periodical readjustment and redistribution process, which periodically swaps subscriptions between BWF and SPF servers to increase the "production" of FUL servers. However, this process yields heavy migration overhead and saves few resources. Therefore, we removed this part from the final version.

3.1 Main Functions: ARRIVE and DEPART

The pseudocode ARRIVE is detailed in Algorithm 1. When a visitor enters the CDN and requests a video clip, the system initiates a subscription u and selects the serving CS s_u based on the current placement of existing subscriptions. ADP first determines whether the same clip (i.e., p_u) exists in any SPF CS, leading to two possible outcomes. First, if the answer is positive, then u is added to this CS to fill up its bandwidth without consuming additional storage space. Otherwise, u is added to s^{OPS}. Here, if s^{OPS} becomes BWF, SPF, or FUL after the addition, then a new empty s^{OPS} is initiated. This ARRIVE process provides fast responsiveness because the initiation of a new CS (i.e., s^{OPS}) occurs only after, not on, a subscription's arrival. Any incoming subscription can be placed into an activated CS (SPF or the OPS) in real-time if the arrival and departure processes of previous subscriptions are completed. To achieve an even shorter latency and address highly bursty demands, a buffer that maintains already activated CSs can be also considered.

Algorithm 1. ARRIVE

ARRIVE(u): decides the serving CS s_u of an incoming subscription requesting clip p_u

```
1: if (∃ CS $s$: $s \in \boldsymbol{S}^{SPF}$ and $\boldsymbol{P}_s \ni p_u$){//find a SPF CS that contains $p_u$
2:     $s_u \leftarrow s$;
3: }else{//all SPF CSs do not contain $p_u$
4:     $s_u \leftarrow s_{OPS}$; //add the subscription into the OPS server
5:     if ($p_u \notin \boldsymbol{P}_{s_{OPS}}$) $\boldsymbol{P}_{s_{OPS}} \leftarrow \boldsymbol{P}_{s_{OPS}} + \{p_u\}$; //copy program $p_u$ from backhaul
6:     if ($BW_{s_u} == 0$ or $SP_{s_u} == 0$) $s_{OPS} \leftarrow$ new server;}
```

Algorithm 2 details the DEPART procedure, which reorganizes the replica placement in CSs when a subscription u leaves the system. Because the system prefers to reduce a subscription from CSs following the priority of "OPS→BWF→SPF→FUL," it determines whether a subscription u' requests the same clip p_u following this preferred order of server types (i.e., it starts by looking for u' at s^{OPS}, then in $\boldsymbol{S}^{BWF}$, $\boldsymbol{S}^{SPF}$, and $\boldsymbol{S}^{FUL}$). These two subscriptions then swap their serving CSs to enable the video clip to depart in this desired priority. The rationale here is that (i) the CSs of higher bandwidth and space utilization should be left as unchanged as possible, and (ii) less operation is required if a subscription leaves from the OPS CS. After u's departure, ADP calls the respective procedures (i.e., leavefromFUL, leavefromBWF, and leavefromSPF) to reorganize the departed CS s_u according to its type $\mathbb{S}_{TYPE}$ (note that no additional process is required for leaving OPS). After the described maintenance is complete, the CS where u departs from (i.e., s_u) is (i) reorganized as a BWF/SPF/FUL CS or (ii) becomes the new OPS; hence, the current OPS is emptied and deactivated during the reorganization process.

Algorithm 2. DEPART

DEPART(u): reorganizes the file placement of CSs when a subscription u leaves the system

```
1: //find a subscription $u'$ which takes over $u$'s position
2: if (∃ subscription $u'$: $p_{u'} = p_u$ and $s_{u'} = s_{\mathrm{OPS}}$) break;
3: else if (∃ subscription $u'$: $p_{u'} = p_u$ and $s_{u'} \in S^{BWF}$) break;
4: else if (∃ subscription $u'$: $p_{u'} = p_u$ and $s_{u'} \in S^{SPF}$) break;
5: else $u' \leftarrow u$;
6: swap $s_u, s_{u'}$;
7: $\mathbb{S}_{TYPE} \leftarrow type\_of(s_u)$; //save the type of$s_{u'}$ before leaving
8: disconnect $u$ from $s_u$;
9: //Manage $s_u$ according to its $\mathbb{S}_{TYPE}$
10: switch $\mathbb{S}_{TYPE}$ {
11: case OPS: break;
12: case BWF: leavefromBWF($s_u$); break;
13: case SPF: leavefromSPF($s_u$); break;
14: case FUL: leavefromFUL($s_u$); break;
15: }
16: /* check OPS state */
17:     if ($BW_{s_u} SP_{s_u} > 0$)){// $s_u$ has both available bandwidth and space
18: deactivate $s_{OPS}$;
19: $s_{OPS} \leftarrow s_u$;
20: }
```

3.2 Procedures

We first describe the procedure MOVE, which migrates subscriptions between CSs and is executed in reorganization procedures leavefromBWF, leavefromSPF, and leavefromFUL. As shown in Algorithm 3(a), this procedure moves n subscriptions of clip p from a source CS s_{src} to a destination CS s_{dest}. It first determines whether s_{dest} already contains the replica of clip p; if not, then p is replicated from the backhaul. The procedures then switch the serving CS of these n subscriptions from s_{src} to s_{dest}.

Next, we introduce the departure process for different types of CSs. If a subscription leaves from s_{OPS}, then the system performs no additional operations. However, when a subscription has left BWF CS s_{LEAVE}, as shown in Algorithm 3(b), the system moves one subscription from s_{OPS} to s_{LEAVE} to fill the bandwidth of s_{LEAVE} if s_{OPS} is not empty. Otherwise, s_{LEAVE} migrates the subscriptions to the SPF CSs, which already contain these clips, and becomes the new OPS. Here, the number of subscriptions moved from s_{LEAVE} to the SPF CS s' is $\min(CN(s_{LEAVE}, p), BW_{s'})$ because it must not surpass the total number of existing subscriptions in s (i.e., $CN(s_{LEAVE}, p)$,) as well as the available bandwidth of s' (i.e., $BW_{s'}$). Algorithm 3(c) details the procedure executed when a subscription leaves SPF CS s_{LEAVE}. If s_{LEAVE} has no available space after the departure, then it remains SPF and no additional process is necessary. Otherwise, the system determines whether the current s_{OPS} is empty; if so, then s_{LEAVE} becomes the new s_{OPS}. Otherwise, the system migrates one subscription from s_{OPS} to s_{LEAVE} to fill up s_{LEAVE}'s storage space and causes it to become SPF or FUL again.

Finally, we detail the process when a subscription leaves a FUL CS s_{LEAVE}. The system considers two conditions: (i) one available storage space emerges after the departure (i.e., $SP_{s_{LEAVE}} == 1$), and (ii) CS s_{LEAVE}'s storage space remains full (i.e., $SP_{s_{LEAVE}} == 0$). In condition (i), the system reorganizes clips the same as when a subscription leaves from a BWF CS because in these two conditions, CS s_{LEAVE} has one unit of available bandwidth and any available storage space. In condition (ii), the system first determines whether another SPF CS s' has common replicas with s_{LEAVE}; if so, then one of s''s subscriptions (i.e., u') is moved to s_{LEAVE} to cause it to be FUL again. Afterward, s' may remain SPF or become the new OPS. If replicas in s_{LEAVE} and all SPF CSs are mutually exclusive, then the system moves one common replica from s_{OPS} to s_{LEAVE} to make it FUL again.

4 Performance Analysis

In Subsection A, we analyze the migration overhead of ADP. In Subsection B, we introduce a lemma and then use it to discuss the performance of ADP.

4.1 Migration Overhead

According to our fourth assumption, when a subscription is added to a CS that already has the same video replica, the system does not have to replicate the clip from the backhaul. Therefore, when analyzing ADP overhead, the replication should be counted only when a CS includes a clip not required by its existing subscriptions. In the following theorem, we prove that, in our proposed method, at most one video clip is replicated to CSs on any arrival or departure of a subscription.

Algorithm 3. Procedures of ADP

(a) Procedure MOVE

MOVE(s_{src}, s_{dest}, n, p):moves n subscriptions of clip p from CS s_{source} to s_{dest}

```
1: if ($CN(s_{dest}, p) == 0$ ) copy $p$ from backhaul to $s_{dest}$;
2: for (int n' = 0;n'<n;n'++){
3: find a subscription $u$ where $p_u == p$ and $s_u == s_{src}$;
4: $s_u \leftarrow s_{dest}$;
5: }
```

(b) Procedure leavefromBWF

leavefromBWF(s_{LEAVE}): reorganize the placement after a subscription leaves from a BWF CS s_{LEAVE}

```
1: if ($\boldsymbol{P}_{s_{OPS}}! = \emptyset$){
2: find a clip $p$ in $s_{OPS}$;
3: $move(s_{OPS}, s_{LEAVE}, 1, p)$;
4: }else{/* move common programs to SPF */
5: for(each clip $p : p \in (P_{s_{LEAVE}} \cap \boldsymbol{P}^{SPF})$) {
6: for(each CS $s' : s' \in \boldsymbol{S}^{SPF}$ and $\boldsymbol{P}_{s'} \ni p$)
   $move(s_{LEAVE}, s', \min(CN(s_{LEAVE}, p), BW_{s'}), p)$;
7: }
8: }
```

(c) Procedure leavefomSPF

leavefromSPF(s_{LEAVE}): reorganize the placement after a subscription leaves from a SPF CS s_{LEAVE}

```
1: if ($SP_{s_{LEAVE}} == 1$ && $\boldsymbol{P}_{s_{OPS}}! = \emptyset$){
2: find a program $p$ in $s_{OPS}$;
3: $move(s_{OPS}, s_{LEAVE}, \min(CN(s_{OPS}, p), BW_{s_{LEAVE}}), p)$;
4: }
```

(d) Procedure leavefromFUL

leavefromFUL(s_{LEAVE}): reorganize the placement after a subscription leaves from a FUL CS s_{LEAVE}

```
1:  if ($SP_{s_{LEAVE}} == 1$){//has storage space after leaving
2:     leavefromBWF($s_{LEAVE}$)
3:  }else{//has no storage space after departure
4:     if ($\exists$ subscription $u'$: $(p_{u'} \in \boldsymbol{P}_{s_{LEAVE}}) \wedge (s'_u \in \boldsymbol{S}^{SPF})$){//has
       common clips with SPF
5:     $s' \leftarrow s'_u$; //mark this SPF CS as $s'$
6:           $move(s', s_{LEAVE}, 1, p_{u'})$; // move one common
             subscription to $s_{LEAVE}$
7:     if ($SP_{s'} > 0$ && $\boldsymbol{P}_{s_{OPS}} \neq \emptyset$)){//if $s'$ has an empty space
       after the migration
8:        find a clip $p$ in $s_{OPS}$; //becomes SPF
9:        $move(s_{OPS}, s', \min(CN(s_{OPS}, p), BW_{s'}), p)$;
10:    }else if(($SP_{s'} > 0$ && $\boldsymbol{P}_{s_{OPS}} = \emptyset$){
11:         $s_{LEAVE} \leftarrow s'$; //$s'$ is set as the new OPS in the end
12:    }
13:    }else if ($\exists$ subscription $u'$: $(p_{u'} \in \boldsymbol{P}_{s_{LEAVE}})$ $\wedge(s'_u = s_{OPS})$)
       {//has a common clip with OPS
14:      $move(s_{OPS}, s_{LEAVE}, 1, p_{u'})$; // move it from OPS to the
         departing CS $s$
15:    }
16: }
```

Theorem 4.1 *ADP conducts at most one physical replication (i.e., copying a clip from the backhaul server to a CS) on any single arrival or departure of a subscription).*

Proof. We prove this lemma by considering all possible conditions when a subscription u is added to a CS s, two possibilities are created in the ARRIVE function:

(a-1) s is SPF (lines 1 and 2 of ARRIVE): This occurs only when s already contains p_u, thus requiring no physical replication.

(a-2) $s = s_{OPS}$ (lines 4 to 6 of ARRIVE): In this case, if $p_u \notin P_{s_{OPS}}$, then one replication is made to copy p_u from the backhaul. Otherwise, no physical replication is necessary. Therefore, at most one replication is conducted. In all arrival cases, the replication is thus processed at most *one* time.

Next, we study the possible conditions of departure. Because the situations are more complicated than the arrival cases, we study all possible conditions and subconditions:

(d-1) line 11 in DEPART: no migration is performed when the subscription leaves from s_{OPS}, and thus no replication overhead occurs.

(d-2) line 12 in DEPART (leavefromBWF is called):

(d-2–1) lines 2 to 3 in leavefromBWF ($P_{s_{OPS}} \neq \emptyset$): Function MOVE is called once. Because at most one clip is moved, the replication overhead is at most one.

(d-2–2) lines 4 to 8 in leavefromBWF ($P_{s_{OPS}} = \emptyset$): All subscriptions having common replicas are moved to SPF CSs. Because the replicas of these subscriptions are already in SPF CSs, no physical replications are committed during the movement, and thus the replication overhead is zero.

(d-3) line 13 in DEPART (leavefromSPF is called): MOVE is called at most once in leavefromSPF. Therefore, at most one replica is copied to a CS.

(d-4) line 14 in DEPART (leavefromFUL is called):

(d-4–1) line 2 in leavefromFUL ($SP_{s_{LEAVE}} == 1$): Same as (d-2). Therefore the overhead is at most one.

(d-4–2) lines 3 to 16 in leavefromFUL ($SP_{s_{LEAVE}} == 0$):

(d-4–2–1) lines 5 to 12 in leavefromFUL ($\exists$ subscription u': $(p_{u'} \in P_{s_{LEAVE}}) \wedge (s'_u \in S^{SPF})$): The first MOVE yields no replication because the replica $p_{u'}$ already exists in both s_{LEAVE} and s'_u. Because the only replication overhead the system may yield is the MOVE function in line 9, the replication overhead is at most one.

(d-4–2–2) lines 13 to 15 in leavefromFUL ($\nexists$ subscription u': $(p_{u'} \in P_{s_{LEAVE}}) \wedge (s'_u \in S^{SPF})$): The system yields no replication overhead because it moves a common replica only in line 14. Therefore, at most one video clip is copied from the backhaul.

Based on the mentioned branches, we conclude that physical replication always occurs at most once on a departure event. Therefore, on any arrival and departure of any subscription, the replication overhead of ADP is at most one. □

4.2 Performance Bound between ADP and the Optimal Solution

Next, we analyze the performance (i.e., the number of active CSs in the system) of the proposed scheme. We first show the property of "replica exclusiveness" among OPS and all SPF CSs in Lemma 4.2; in Theorem 4.3, we then find the difference between the performance of the proposed scheme and the optimal performance based on this lemma.

Lemma 4.2. *In ADP,* $\forall \text{clip}\, p, \left|S_p \cap \{S^{SPF} \cup \{s^{OPS}\}\}\right| \leq 1$ *(i.e., in a CDN managed by ADP, for all video clips, at most one replica is in the OPS CS and all SPF CSs.*

Proof. We prove this lemma by induction. In other words, we must (i) show that the lemma (i.e., $\forall \text{clip}\, p, \left|S_p \cap \{S^{SPF} \cup \{s_{OPS}\}\}\right| \leq 1$) holds initially, and (ii) prove that it remains satisfied after an arrival or departure of any subscription. Clearly, (i) is true because at the beginning, where no subscriptions in any CS are found, $S_p = \emptyset$ for all clip p.

Next, we study the conditions after an arrival and departure. As discussed in Theorem 4.1, when a subscription u arrives, it must be added to an SPF CS (a-1) or s_{OPS} (a-2). Lemma 4.2 still holds in (a-1) because no CSs change any replica after the arrival. In case (a-2), no replication would occur if $p_u \in P_{s_{OPS}}$ before the arrival. Therefore, all CSs contain the same video clips, thus indicating that the lemma remains true. However, if $p_u \notin P_{s_{OPS}}$, then it must not have belonged to the OPS and SPF originally and, thus, exists only in s_{OPS} after the addition. In either case, the lemma holds. Because the condition remains true in both (a-1) and (a-2), it holds after a subscription's arrival.

Next, similar to Theorem 4.1, we consider the replica combination under all possibilities. Here, we let $P\prime$ and $S\prime$ respectively denote the sets of replicas and servers after the reorganization process. Given that the OPS and all SPF servers have exclusive clips initially, the lemma holds after each of the following conditions:

(d-1) line 11 in DEPART: $s'_{OPS} = s_{OPS}$, $P'_{s_{OPS}} \subseteq P_{s_{OPS}}$, $S^{SPF'} = S^{SPF}$, and $\forall s \in S^{SPF}, P'_s = P_s$.

(d-2–1) lines 2 to 3 in leavefromBWF: $s'_{OPS} = s_{OPS}$, $P'_{s_{OPS}} \subseteq P_{s_{OPS}}$, $S^{SPF'} = S^{SPF}$, and $\forall s \in S^{SPF}, P'_s = P_s$.

(d-2–2) lines 4 to 8 in leavefromBWF: $s'_{OPS} = s_{LEAVE}$, $P_{s_{LEAVE}} \cap P^{SPF'} = \emptyset$, $S^{SPF'} \subseteq S^{SPF}$, and $\forall s \in S^{SPF'}$, $P'_s = P_s$.

(d-3) leavefromSPF is called: Two cases are possible:

(i) The IF statement in line 1 of leavefomSPF is satisfied. Two subconditions are considered. First, if $CN(s_{OPS}, p) > BW_{s_{LEAVE}}$, then $s'_{OPS} = s_{OPS}$, $S^{SPF'} \subseteq S^{SPF}$, and $\forall s \in S^{SPF'} \cup \{s'_{\text{OPS}}\}, P'_s = P_s$ (i.e., s_{LEAVE} becomes FUL). Otherwise, $s'_{OPS} = s_{OPS}$, $P'_{s_{OPS}} = P_{s_{OPS}} - \{p\}$, $S^{SPF'} = S^{SPF}$, $P'_{s_{LEAVE}} = P_{s_{LEAVE}} + \{p\}$, and $\forall s \in S^{SPF'} - \{s_{LEAVE}\}, P'_s = P_s$ (i.e., all replicas of p are migrated to s_{LEAVE}, and s_{LEAVE} remains as SPF).

(ii) Otherwise, no migration is made. Therefore, $s'_{OPS} = s_{OPS}$, $P'_{s_{OPS}} = P_{s_{OPS}}$, $S^{SPF'} = S^{SPF}$, and $\forall s \in S^{SPF}, P_s = P'_s$

In either case, the lemma holds.

(d-4–1) line 2 in leavefromFUL: Same as (d-2).

(d-4–2–1) lines 5 to 12 in leavefromFUL: Three cases are possible:

(i) The IF in line 7 of leavefromFUL is satisfied: the operation among the OPS and SPF server s' is identical to case (i) of (d-3). Therefore, the lemma also holds in this case.

(ii) The IF in line 10 leavefromFUL is satisfied: $s'_{OPS} = s_{LEAVE}$, $\boldsymbol{P}'_{s_{OPS}} \subseteq \boldsymbol{P}_{s_{LEAVE}}$, $\boldsymbol{S}^{SPF'} = \boldsymbol{S}^{SPF} - \{s_{LEAVE}\}$, and $\forall s \in \boldsymbol{S}^{SPF'} \cup \{ s'_{OPS}\}$, $\boldsymbol{P}'_s = \boldsymbol{P}_s$ (i.e., the original ops is empty and s_{LEAVE} becomes the new OPS).

(iii) Both IFs are not satisfied (i.e., $SP_{s'} = 0$): In this case, no modification is made.

In the three mentioned cases, the lemma holds.

(d-4–2–2) lines 13 to 15 in leavefromFUL: $s'_{OPS} = s_{OPS}$, $\boldsymbol{P}'_{s_{OPS}} \subseteq \boldsymbol{P}_{s_{OPS}}$, $\boldsymbol{S}^{SPF'} = \boldsymbol{S}^{SPF}$, and $\forall s \in \boldsymbol{S}^{SPF'}$, $\boldsymbol{P}'_s = \boldsymbol{P}_s$.

The previous discussion proves that if the lemma holds initially, then it remains true after any subscription's arrival or departure. Because we have proven that: (i) the lemma is true at the beginning (i.e., empty CSs), and (ii) it holds after any subscription's arrival or departure, we conclude this lemma is always true during ADP execution. □

Next, we use Lemma 4.2 to analyze the worst-case performance difference between ADP and the optimal solution. Some additional notations are used in this theorem. Let the total required bandwidth and storage space of all existing subscriptions in the system be B and M respectively. The lower bound (LB) of the required number of the active CSs is $\mathbb{R}^{\mathrm{LB}} = \lceil \max(\frac{B}{\mathrm{BW}}, \frac{M}{\mathrm{SP}}) \rceil$ because, regardless of the placement, the total bandwidth and storage space of the subscriptions can be no less than the total available resources of the CSs currently activated. The number of the CSs of the optimal solution and our proposed scheme ADP is denoted by $\mathbb{R}^{OPT}$ and $\mathbb{R}^{ADP}$ respectively.

Theorem 4.3. $\mathbb{R}^{ADP} - \mathbb{R}^{OPT} \leq min(|\boldsymbol{S}^{BWF}| + |\boldsymbol{S}^{FUL}|, |\boldsymbol{S}^{SPF}|)$ *(i.e., the performance difference between ADP and the optimal solution is bounded).*

Proof. Because $\mathbb{R}^{OPT} \geq \mathbb{R}^{LB}$, we know that

$$\mathbb{R}^{ADP} - \mathbb{R}^{OPT} \leq \mathbb{R}^{ADP} - \mathbb{R}^{LB}. \tag{8}$$

Next, in the file placement of ADP, the bandwidth of all FUL and BWF CSs are fully loaded. Therefore,

$$B \geq \mathrm{BW} \cdot (|\boldsymbol{S}^{BWF}| + |\boldsymbol{S}^{FUL}|). \tag{9}$$

Similarly, because the replicas contained by all SPF and OPS CSs differ (Lemma 4.2), and the storage space of SPF CSs are fully utilized:

$$M \geq \mathrm{SP} \cdot (|\boldsymbol{S}^{SPF}|). \tag{10}$$

Based on (9) and (10), we know that the performance of the LB should be

$$\begin{aligned}\mathbb{R}^{LB} &= \left\lceil max\left(\frac{B}{BW}, \frac{M}{SP}\right)\right\rceil \\ &\geq max(|\boldsymbol{S}^{BWF}| + |\boldsymbol{S}^{FUL}| + 1, |\boldsymbol{S}^{SPF}| + 1).\end{aligned} \tag{11}$$

By placing (11) into (8), we have

$$\begin{aligned}&\mathbb{R}^{ADP} - \mathbb{R}^{OPT} \leq \mathbb{R}^{ADP} - \mathbb{R}^{LB} \\ &\leq [|\boldsymbol{S}^{BWF}| + |\boldsymbol{S}^{FUL}| + |\boldsymbol{S}^{SPF}| + 1] \\ &\quad - \max(|\boldsymbol{S}^{BWF}| + |\boldsymbol{S}^{FUL}| + 1, |\boldsymbol{S}^{SPF}| + 1) \\ &= min(|\boldsymbol{S}^{BWF}| + |\boldsymbol{S}^{FUL}|, |\boldsymbol{S}^{SPF}|).\end{aligned} \tag{12}$$

□

Although this theoretical worst-case performance difference is no higher than 50 percent, as we show in the simulations, the actual bound is generally much tighter. This is because in the placement of ADP, CSs are always dominated by either BWF or SPF CSs, according to the distributions of the subscriptions' requesting clips. Another type of CSs occupies only a small portion, making the actual performance bound more approximate to the optimum than the worst case.

5 Numerical Results

5.1 Simulation Settings

Because we assume the incoming event of each subscription is independent, the interval between each set of two consecutive subscriptions is exponentially distributed. To simulate the time-variant amount of traffic during a day, the arrival rate per hour is expressed as $q(t) = 80\,000(\cos(\frac{2\pi t}{24}) + 2)$, $0 \leq t \leq 24$, where t is the hour of a day. The program each incoming subscription requires is distributed using Zip's distribution [22], where the probability of requiring the ith program is $\frac{i^{-\alpha}}{\sum_{j=1}^{M} j^{-\alpha}}$. As [6] suggested, $0.271 \leq \alpha \leq 1$; therefore, we set α as 0.6. The lifetime of each subscription (i.e., the duration for which each visitor watches the chosen video) is uniformly distributed in [0,600] seconds because the video clip was 1 min in duration. These settings are commonly used in similar simulations such as [6]. Next, after primitive simulations, we set $SP = 1250$ and $BW = 10\,000$, which is the "range of interest" of the simulation because the space and bandwidth resources of a CS are more balanced under this setting. As subscribers arrive and leave, the majority of CSs can be either bandwidth-limited or space-limited. Therefore, the complete behavior of ADP can be observed.

We used the simulation results from the second day to enable the CSs to not be empty in the beginning and to create a more realistic environment. The results (e.g., the number of active CSs and replication times in Figs. 5, 6, and 7) are averaged 100 times for any single setting and recorded for every minute of the simulation.

5.2 Video Concentration

We first adjust the values of M (i.e., the total number of video clips) to 5000, 1.0 000, and 50 000, to observe ADP behavior under different degrees of video concentration. As shown in Fig. 4, when M = 5000, subscriptions tend to be more concentrated on popular clips. However, more unpopular clips are found in the system when M is higher. The type composition of CSs under various Ms- is shown in Fig. 5. When M is lower (M = 5000 in Fig. 5a), the type of CSs is typically dominated by BWF. As M increases, SPF CSs become more dominant (Figs. 5b and 5c). Regardless of

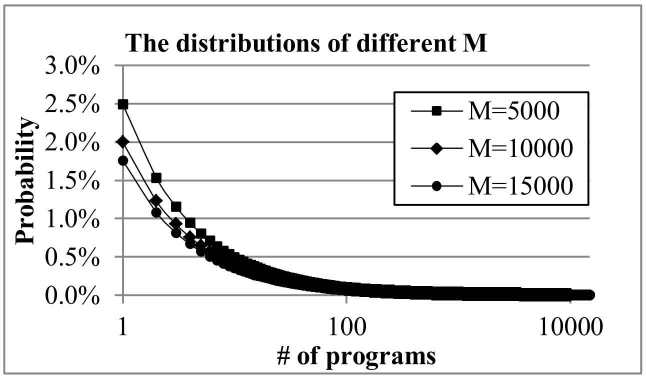
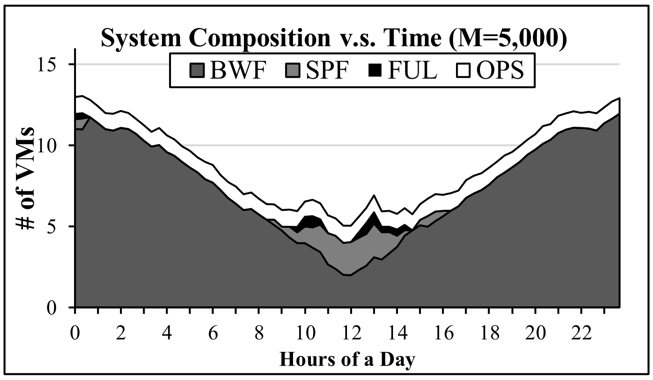
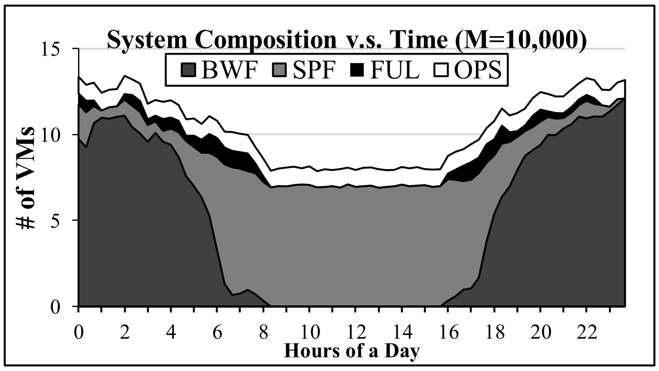


Fig. 4. Program distribution.

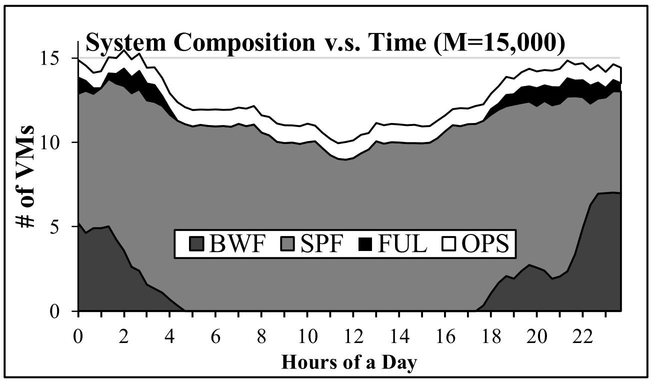
the value of M, more SPF CSs tend to be under lower popularity (i.e., Hours 10 to 15), and more BWF CSs tend to be under higher popularity (i.e., Hours 0 to 5, 10 to 24). This is



(a) M = 5000

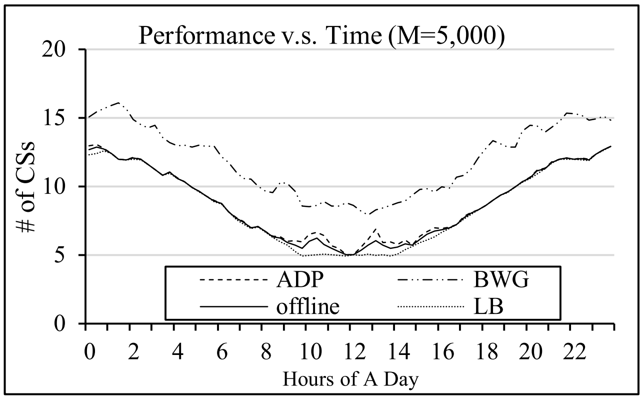


(b) M = 10 000

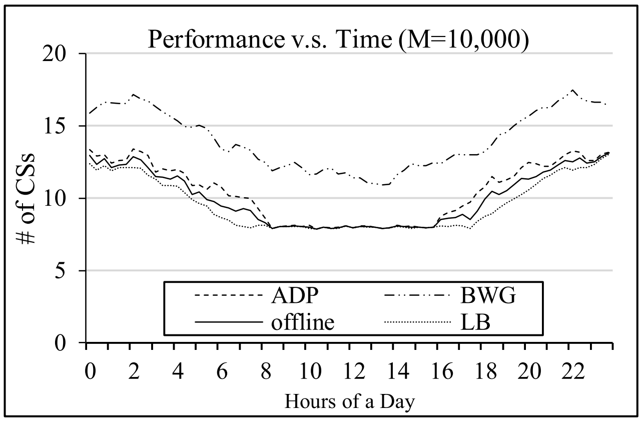


(c) M = 15 000

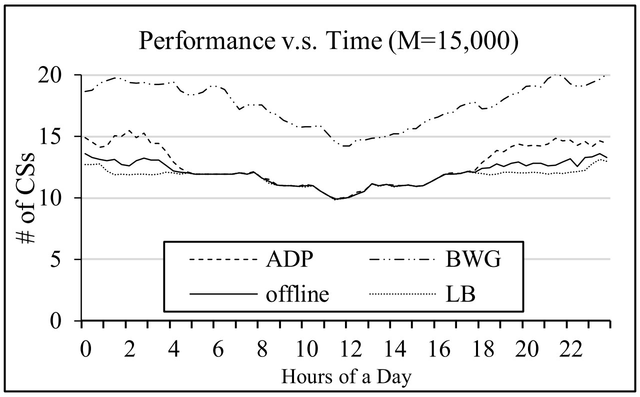
Fig. 5. Server composition of ADP under various video distributions.



(a) M = 5000



(b) M = 10 000



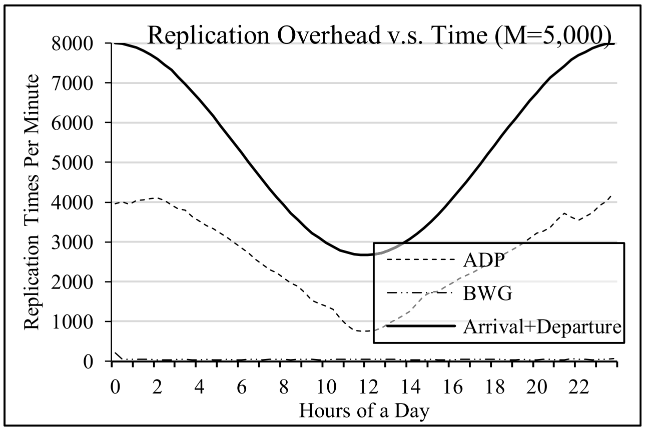
(c) M = 15 000

Fig. 6. Performance of various placement strategies.

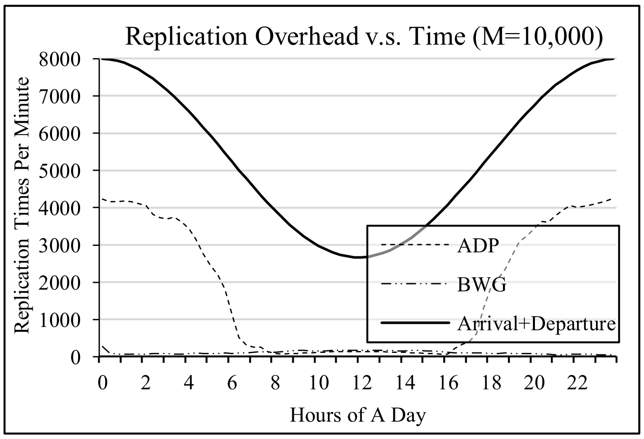
because lower popularity leads to fewer subscriptions to each program, which increases the difficulty of filling up the bandwidth of each CS. In the following simulations, we use these three M values (i.e., various degrees of program distributions) to observe the system's behavior.

5.3 Performance-Resource Consumption

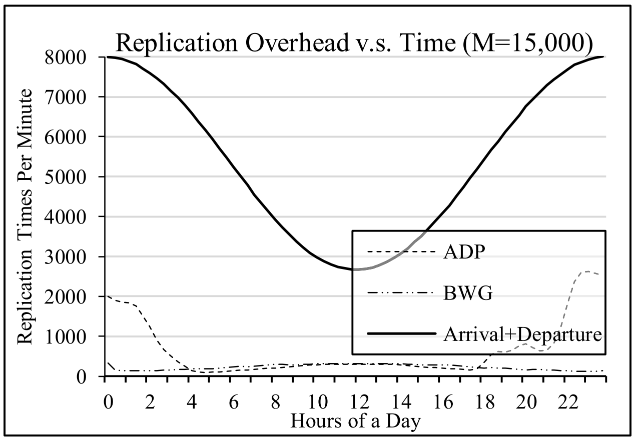
Next, we compare the performance (i.e., the number of CSs ADP requires) with other approaches. As mentioned, no studies have attempted to address this dynamic allocation problem. Therefore, we compare ADP with a primitive approach called bandwidth-greedy (BWG). In BWG, each arrival subscription is placed into the CS that has the least residual bandwidth (i.e., the CS having the lowest BW_s) to fully use the bandwidth of the existing active CSs before



(a) M = 5000



(b) M = 10 000



(c) M = 15 000

Fig. 7. Overhead of various approaches.

initiating a new one. When a subscription leaves the system, BWG performs no organization but deletes the replica that is not used by other subscriptions. The performance of the "offline solution" to the RSVP problem, as we introduce in Section 2, is also presented as a performance reference. Finally, we observe the LB mentioned in Theorem 4.3. The results of LB, ADP, offline, and BWG under different M are shown in Fig. 6.

The results of our study are critical for several reasons. First, compared with BWG, ADP costs substantially less CSs, thus indicating the effectiveness of ADP in saving system resources and consumed power. In Fig. 6c, our proposed scheme can save up to approximately 40 percent of CSs (from 20 to 12 at 10 p.m.). Second, ADP typically demonstrates performance approximate to the offline solution and LB. The average and standard deviation of the performance difference between ADP and the offline (optimal) version are respectively 1.60 and 2.6 percent when M = 5000; 3.76 and 3.48 percent when M = 10 000; and 5.01 and 6.19 percent when M = 15 000. ADP incurs a larger difference when the number of SPF and BWF CSs are more balanced in the system (e.g., 10–14 when M = 5000, 5–9/16–20 when M = 10 000, and 0–4/18–24 when M = 15 000). The worst-case performance difference is approximately 20 percent, which occurs at 2 a.m., M = 15 000. This fact corresponds to Theorem 4.3: the performance difference is $\min(|S^{BWF}| + |S^{FUL}|, |S^{SPF}|)$. Therefore, this difference is mild when SPF or BWF CSs dominate the system but becomes more pronounced when both types of CSs are more balanced. Third, under a higher M, the performance curve of ADP is flatter, showing that the number of CSs is less affected by the arrival rate of subscriptions. This is because when SPF CSs dominate the system (i.e., Fig. 5 c), the number of CSs is mainly decided by the space of the clips in the system and is less sensitive to the time-varying bandwidth requirement (i.e., subscriber number of each clip).

5.4 Overhead

Finally, we observed the replication overhead of ADP, which is counted only when a video clip is copied to a CS that does not contain its replica. We recorded the total replication times of ADP and BWG in each minute and also showed the total arrival and departure of subscriptions at the baseline. The overheads of the two approaches are shown in Fig. 7. The results demonstrate that the overhead of ADP is higher because it reorganizes the replica placement when subscriptions leave CSs. However, BWG replicates a clip to a CS only when a subscription arrives and does not conduct any additional process on departure. The results also show that the overhead is limited by the total arrival and departure times of subscriptions, as we proved in Theorem 4.1. The results demonstrate that saving on the number of active CSs has a price (i.e., higher replication overhead than BWG), which is though limited and acceptable.

Based on the results in Fig. 5, we also observed that the ADP overhead is milder when SPF CSs are more dominant (i.e., higher M). This is because under this condition, new subscriptions are more likely to be placed in an existing CS, which already contains its replica. When most CSs in the system are BWF, the system is more likely to replicate the clip to a new CS when a subscription arrives because BWF CSs have no available bandwidth.

6 Conclusion and Discussion

In this paper, we examine an online video placement scheme for superior utilization and energy-saving in cloud delivery networks. We introduce a new problem that dynamically places incoming video subscribers to CSs to limit the number of active machines as well as the replication overhead. This problem considers both transmission bandwidth and storage space constraints and is modeled in a general manner. It can therefore be applied effectively to various types and scales of CDNs. By classifying servers into different types, our proposed ADP scheme places and reorganizes video subscriptions on their arrival and departure. Through analysis, we demonstrate the effectiveness of ADP regarding performance and overhead. The worst-case overhead of ADP is limited, and the performance difference to the optimum is bounded. The outstanding performance of ADP is also evidenced by the simulations. The results show that ADP significantly outperforms the compared scheme under various conditions and maintains performance approximate to the optimal solution. In addition, the replication overhead of the system is also limited. To the best of our knowledge, ADP is the only scheme that addresses this placement problem and provides all the mentioned advantages.

We notice that other types of approaches may effectively reduce the server number (although they might produce heavier migration overhead; e.g., by analyzing and predicting the incoming subscriptions or using machine learning to redistribute replicas). In the future, we will study these approaches to further improve and generalize our scheme.

Acknowledgments

This paper is supported by the Ministry of Science and Technology, Taiwan under grant number MOST103–2221-E-155–019-MY3.

References

[1] L. M. Vaquero, L. R.-Merino, J. Caceres, and M. Lindner, "A break in the clouds: Towards a cloud definition," in *Proc. ACM SIGCOMM Comput.Commun. Rev.*, 2009, pp. 50–55.

[2] L. Tsai and W. Liao, "Cost-aware workload consolidation in green cloud datacenter," in *Proc. IEEE CLOUDNET*, 2012, pp. 29–34.

[3] L. A. Barroso and U. Holzle, "The case for energy-proportional computing," *IEEE Comput.*, vol. 40, no. 12, pp. 33–37, 2007.

[4] P. Bohrer, E. N. Elnozahy, T. Keller, M. Kistler, C. Lefurgy, C. McDowell, and R. Rajamony, "The case for power management in web servers," *Power Aware Computing*. Kluwer Academic Publishers, pp. 261–289, 2002.

[5] B. Li, J. Li, J. Huai, T. Wo, Q. Li, and L. Zhong, "EnaCloud: An energy-saving application live placement approach for cloud computing environments," in *Proc. IEEE CLOUD*, 2009, pp. 17–24.

[6] Y. Ho, P. Liu, and J. Wu, "Server consolidation algorithms with bounded migration cost and performance guarantees in cloud computing," in *Proc. IEEE Utility Cloud Comput.*, 2011, pp. 154–161.

[7] M. Korupol, A. Singh, and B. Bamba "Coupled placement in modern data centers," in *Proc. IEEE Int. Parallel Distrib. Processing*, 2009, pp. 1–12.

[8] A. Gandhi, M. H.-BALTER, and R. Raghunathan, "AutoScale: Dynamic, robust capacity management for multi-tier data centers," *ACM Trans. Comput. Syst.*, vol. 30, no. 4, p. 14, Nov. 2012.

[9] A. Krioukov, P. Mohan, S. Alspaugh, L. Keys, D. Culler, and R. H. Katz, "NapSAC: Design and implementation of a power-proportional web cluster," in *Proc. First ACM SIGCOMM*, 2010, pp. 102–108.

[10] Z. Zhou, F. Liu, H. Jin, B. Li, B. Li, and H. Jiang, "On arbitrating the power-performance tradeoff in SaaS clouds," in *Proc. IEEE INFOCOM*, 2013, pp. 872–880.

[11] W. Deng, F. Liu, H. Jin, X. Liao H. Liu, and L. Chen, "Lifetime or energy: Consolidating servers with reliability control in virtualized cloud datacenters," in *Proc. IEEE CloudCom*, 2012, pp 18–25.

[12] F. Xu, F. Liu, L. Liu, H. Jin, B. Li, and B. Li, "iAware: Making live migration of virtual machines interference-aware in the cloud," *IEEE Trans. Comput.*, vol. 63, no. 12, pp. 3012–3025, Nov. 2014.

[13] F. Xu, F. Liu, H. Jin, and A. V. Vasilakos, "Managing performance overhead of virtual machines in cloud computing: A survey, state of the art, and future directions," in *Proc. IEEE*, 2014, vol. 102, no. 1, pp. 11–31.

[14] A. Nimkar, C. Mandal, and C. Reade, "Video placement and disk load balancing algorithm for vod proxy server," in *Proc. IEEE Internet Multimedia Syst. Architecture Appl.*, 2009, pp. 1–6.

[15] X. Zhou and C. Xu, "Optimal video replication and placement on a cluster of video-on-demand servers," in *Proc. Int. Conf. Parallel Processing*, 2002, pp. 547–555.

[16] J. Guo, Y. Wang, K. Tang, S. Chan, W. M. Wong, P. Taylor, and, M. Zukerman, "Evolutionary optimization of file assignment for a large-scale video-on-demand system," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 6, pp. 836–850, Jun. 2008.

[17] Y. Feng and B. Li, "Airlift: Video conferencing as a cloud service using inter-datacenter networks," in *Proc. IEEE Int. Conf. Netw. Protocols*, 2012, pp. 1–11.

[18] X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.

[19] P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav, "It's not easy being green," in *Proc. ACM SIGCOMM*, 2012, pp. 211–222.

[20] Z. Zhou, F. Liu, Y. Xu, R. Zou, H. Xu, J. C. S. Liu, and H. Jin, "Carbon-aware load balancing for geo-distributed cloud services," in *Proc. IEEE Modelling, Anal. Simul. Comput. Telecommunication Syst.*, 2013, pp. 232–241.

[21] H. Xu and B. Li, "Cost efficient datacenter selection for cloud services," in *Proc. IEEE Int. Conf. Commun. China*, 2012, pp. 51–56.

[22] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, "YouTube traffic characterization: A view from the edge," in *Proc. ACM Internet Measurement Conf.*, 2007, pp. 15–28.

Wen-Hsing Kuo received the BS degree and the PhD degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 2002 and 2008, respectively. He joined the Department of Electrical Engineering, Yuan Ze University, Taoyuan, Taiwan, as an assistant professor in 2008. His research interests include next-generation wireless networks, cloud computing, and Internet of Things. He served as a Technical Program Committee member in many international conferences, including Globecom, ICCAS, WiMob, VTC, and ICC. One of his papers was elected as "the best journal paper of 2011" by the Multimedia Communication Technical Committee, IEEE Communication Society. He was awarded the "Outstanding Young Scholar Research Project" by the Minister of Science and Technology of Taiwan in 2014, and he was also the inventor of several Taiwan and US patents. He is a senior member of the IEEE.

Yung-Hsuan Lin received the MS degree in electrical engineering from Yuan Ze University, Taiwan, in 2014. His research interests include cloud computing and multimedia communications.