Application-aware Video-Sharing Services via Provenance in Cloud Storage

Jinjun Liu, Dan Feng*, Yu Hua, Bin Peng, Pengfei Zuo
Wuhan National Laboratory for Optoelectronics
School of Computer, Huazhong University of Science and Technology
Wuhan 430074, China.
*Corresponding author
{*liujinjun, dfeng, csyhua, pengbin, pfzuo*}@*hust.edu.cn*

***Abstract*—With the wide use of video-editing software, more and more similar videos are uploaded to the video-sharing platforms in cloud storage. The large number of near-duplicate videos would not only make the users not satisfactory, but also consume more resources with service providers. Since the near-duplicate videos have redundancy at the content level, the traditional de-duplication technology is not suitable to address the redundancies. In order to decrease the redundancy, we proposed a novel high performance provenance-based video-sharing system, called Provis. Provis records the provenance of the videos and compresses the near-duplicate videos via storing their provenance which replaces storing their video files to improve the space efficiency. When users attempt to upload the edited videos, Provis can rebuild the videos in the cloud via collecting and uploading their provenance to reduce the network overhead of uploading videos. We implemented Provis and compared its performance with other existing video compression techniques. Our evaluation shows that Provis achieves significant space savings.**

I. Introduction

The widespread use of digital video editors, as well as the wide availability of digital video cameras and smartphone cameras, has produced an increase of similar videos on video-sharing platforms. There are on average 27%, even up to 93% duplicate or near-duplicate (similar) videos via a sample of 24 popular queries from YouTube, Google Video and Yahoo! [1]. The large number of similar videos not only reduce the performance of video-sharing services, but also make the users not satisfactory. We discuss the details from the following three aspects.

First, the videos increase the storage overhead of the video-sharing systems in cloud storage. There are over 100 hours of videos being uploaded every minute on YouTube [2]. This number will increase with fast speed. The partial or complete duplicate videos in the uploaded videos will also increase rapidly. Since the distribution of the total views popularity of videos appears to be Zipf's law [3], these similar videos which occupy the large mounts of storage space will be rarely accessed after a short period from being uploaded.

Second, the videos increase the consumption of the network bandwidth between video-sharing systems and users. Video streamings have dominated global Internet traffic and will account for 79 percentage of all consumer Internet traffic in 2018, from 66 percentage in 2013 [4]. Current web-based video-sharing platforms can only reject the uploaded duplicate videos. When a user uploads a similar video which can be rebuilt via simply editing several videos stored on a remote platform, the user has to upload the video file since the platform can not distinguish the similar video. If we can rebuild the similar video in the cloud, we reduce the network overhead of the uploading of similar videos, and achieve the quick uploading performance.

Third, the videos impact the user experience. For example, most existing video search engines mainly use the associated metadata to index, retrieve and recommend videos. Nevertheless, the coherent metadata of similar videos which are uploaded by users are possibly inaccurate and noisy. The inaccuracy leads to returning the large amounts of near duplicate or irrelevant results and reducing the effectiveness of searching the videos. These similar videos also bring trouble to video classification and video index. The survey of the Accenture revealed that only 45 percentage of video consumers was satisfied with the video services [5].

Several studies have been devoted to avoid the disadvantages of similar videos in terms of the above three aspects. Towards duplicate videos, the de-duplication technology [6] is the common method for eliminating those redundant data. The de-duplication technology is not suitable for the near-duplicate videos due to gauging redundancy at the byte-level. The modification is slight in contents between similar videos, such as resolution changes, encoding format transition, logo insertion and lighting change. Nevertheless, from the point of view of the data stream, the similar videos are entirely different. It is difficult to find the same data blocks between those videos via comparing fingerprint. In the same way, the delta compression (or delta encoding) technology [7] is not efficient for compressing these videos since delta compressors compute the difference between the two files at the byte-level.

Hence, the near-duplicate videos have redundancies at the content level. We can find and compress those videos via comparing semantic. If the modifications from one video to its similar videos can be recorded and repeated automatically, those videos would not need to be directly uploaded and stored. In order to compress these unmanaged redundancies [8], we propose a novel high-performance ***Pro**venance-based **vi**deo-sharing **s**ystem* in cloud storage, called **Provis**. Provis can record and repeat the operations of the modifications (the video-editing process, stored in provenance) from one video to its near-duplicate videos. When these videos need to be uploaded to or stored in cloud storage, Provis merely uploads or stores the provenance. Otherwise, if the provenance of a video which has been deleted for some reasons can be obtained, Provis can regenerate this video. Hence, Provis can recommend more available related videos.

The contributions of this paper are summarized as follows.

High-performance provenance-based video compression. Instead of storing a video, we preserve its provenance to improve the efficient use of storage space of video-sharing system in the cloud. Since the size of a video is far larger than its application-level provenance, the provenance-based video compression is much higher performance than block level de-duplication technology.

Low-overhead provenance-based uploading video. Similar to the contribution above, if we can collect the provenance of a video which is edited offline, we upload the provenance and rebuild the video in the cloud, instead of uploading the video. We adapt this method to reduce the network overhead of uploading videos.

Experimental evaluation via a prototype system. We implemented a prototype system. Though the test, we observe that Provis can save about 80% of storage space via provenance-based video compressing. Since the average size of a video's provenance is about 10KB which is much smaller than the size of the video, Provis can also significantly increase the performance of uploading videos.

The rest of the paper is organized as follows. Section 2 presents the background and our motivations. The overview of Provis and four key features are given in Section 3. Section 4 describes our design and implementation details. Section 5 evaluates the performance. Section 6 summarizes related work and Section 7 concludes the paper.

II. Background and Motivation

This section describes the features of application-level provenance and the implementation of video-sharing services in cloud storage, and then presents the motivations of our work.

A. *Application-Level Provenance*

Data provenance, one kind of metadata, is about the history of a data object starting from its original sources. Provenance of a data object can answer the following questions: (1) Where does the object come from? (2) Who is using or used the object? (3) How was the object obtained? Hence, the provenance is a key feature to understand the data and to reuse the data [9].

According to the level of collection, the provenance can be divided into the application-level provenance and the system-level provenance. We mainly focus on the application-level provenance. Since this kind of provenance does not need to collect system information and system calls, the size of provenance is smaller than system-level provenance. Through tracking user-space events, this provenance, as the annotations of data, can be used to help users to search.

The application-level provenance can be used to data reconstruction [10]. There are three advantages being compared with data reconstruction using the system-level provenance. First, as described above, the application-level provenance occupy smaller storage space than system-level provenance. Second, the collecting application-level provenance has much less overhead than collecting system-level provenance. Third, since the reconstruction does not need to concern file system and operation system, its complexity is lower than the system-level. Hence, Provis exploits the application-level provenance to optimize vide-sharing services.

B. *The Video-Sharing Service in the Cloud*

In recent years, many cloud-based services have provided users the opportunity to work and save their files to the cloud. Several video services have been moved to the cloud. For example, NetFlix has stored all of its high-definition video masters on *CloudFront* which integrates with the Amazon S3 and EC2 [11].

With the growth of cloud computing and datacenter technologies, the web-based video-sharing service[1], [3] has been moved to the cloud. Provis achieves the video-sharing service in the cloud storage too. We do this for two reasons. First, the powerful computing resources of the cloud can support the edition and regeneration of videos. Many useful services have offered users the utility to edit videos online in the cloud, such as YouTube, WeVideo, Brainshark, JW Player, etc. Second, much larger memory cache cluster can be used to store active videos' provenance graph, provenance and temporary video files for video search and regeneration.

C. *Motivations*

When we retrieve video on video-sharing platforms in the cloud, we can find there are a large numbers of duplicate or near-duplicate videos. The duplicate videos can be eliminated easily via the data de-duplication technology on the video-vault. Since a slight modification (e.g., encoding format transition) leads to entirely change the video file's bit stream, the near-duplicate videos can not be found and compressed with ease. There exists very strong correlation between the semantics of the near-duplicate videos. If we use a lightweight data structure to record the transformation of the videos' semantics, we can store the similar videos replacing with the semantic changes between them. When we reuse these videos, we can recreate them via the data structure. This method will greatly reduce space overhead of storing them.

If we can intercept the information about the commands and the parameters of users' video-editing operations, we will achieve our design. The application-level provenance can do it. The provenance of videos can be used to record the video-editing processes and reconstruct videos. Hence, we can only store the provenance of a video. Once we need to reuse it, we can recreate it via its provenance.

The online video editors are completely dependent on Internet connection. If users edit videos offline, we can modify the editors to record the users' operations. When the users attempt to upload the edited videos, we can upload their provenance and recreate the videos in the cloud. Since the cost of transmitting a data is much more than computing it, this method will greatly reduce the network overhead of uploading videos and save energy.

Since the sizes of videos are much larger than the application-level provenance commonly, the provenance can

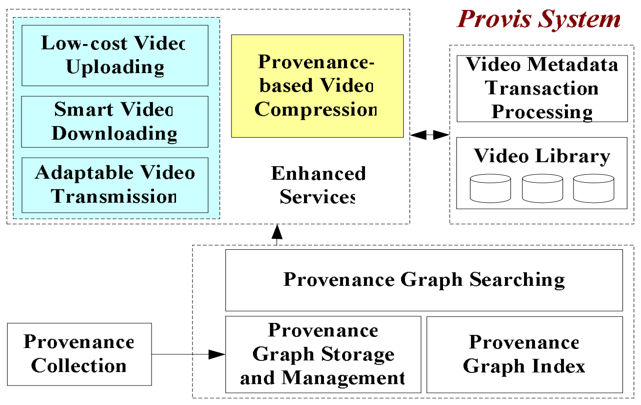
Provis System
Low-cost Video Uploading
Smart Video Downloading
Adaptable Video Transmission
Provenance-based Video Compression
Enhanced Services
Video Metadata Transaction Processing
Video Library
Provenance Graph Searching
Provenance Collection
Provenance Graph Storage and Management
Provenance Graph Index

Fig. 1. The Provis Architecture

be kept for a longer period. When a user deletes a video for some reasons except the security, we can still recreate this video via its provenance for some time. In some application scenarios, this method will improve the performance of video-sharing service. When we preserve all provenance, we can recreate several videos which have been deleted and provide more related videos.

III. Provis System

A. Overview

Figure 1 shows the logical diagram of Provis. Provis provides three enhanced services for users and meanwhile stores the provenance information of videos instead of the video files to decrease the storage consumption of system.

Provis includes three components: 1) video provenance processing, this part collects the provenance of videos from client-side or server-side, and creates a provenance graph to index, search and store the provenance of the videos. 2) four enhanced services, Provis provides low-cost video uploading, smart video downloading and adaptable cloud-based video transmission for users, and also provides provenance-based video compression to reduce the overhead of storage in the video-sharing system. 3) the metadata of video and video service platform, this part includes two modules, the video metadata transaction processing and the video library. The first module finishes several common metadata operations of videos, such as metadata indexing, metadata searching and metadata storing. The second one provides the accessing and the storage of videos. Details of the first two components of Provis are described in the next section.

B. User View

From the viewpoint of the users, Provis can improve the performance of three key functions of video-sharing services by using the video provenance. The first enhanced service is the low-cost video uploading. When a user edits one or several videos (called the original videos) to new videos (called the target videos), Provis collects the provenance of the new videos. If the user attempts to upload these new videos, he/she first uploads the IDs of the original videos and the target videos and the provenance to the cloud. A cloud-side server of Provis judges whether the target videos can be rebuilt via provenance searching. Then, the server notifies the client-side of Provis the IDs of the original videos which need to be uploaded. The user's machine will only upload those videos and other videos will be rebuilt on the servers. Since the size of the application-level provenance is much smaller than the size of video, uploading the provenance by replacing the video files greatly reduces the cost of network.

The second enhanced service is smart video downloading. When a user attempts to download a video, he/she will send a request to a server of Provis. This request includes the video's ID and some information about device for playing videos. After receiving the request, the server determines whether the video can be delivered to the user. If the video is actually nonexistent, or is not comfortable for the user's machine or network environment, Provis will provide several similar videos which come from the provenance graph searching to the user. Then, the user chooses one video and downloads it from Provis.

The third enhanced service is adaptable cloud-based video transmission. When a user shares a video to his/her friends through Provis, he uploads the video to a cloud-side server of Provis and sets the video-sharing group. When his/her friends attempt to download this video, Provis will examine whether the video is available to the friends. If the video is unavailable due to the unsuitable viewing result or the low communication bandwidth between the server and the friends, Provis will recommend several related videos like the above mentioned way.

C. System View

From the viewpoint of the system, in order to improve the efficient utilization of the cloud storage space, Provis leverages the provenance of videos to compress near-duplicate videos. Provis sets a parameter for every video in the provenance graph. The parameter determines whether Provis compresses the video. The parameter relates with the tradeoff between the cost of storing the video and cost of reconstructing the video. If the former cost is higher than the later cost, Provis only stores the video provenance instead of the video file. When a user attempts to access this video, Provis will reconstruct the video in the cloud. Due to the strong computing power of the cloud, reconstructing video can not reduce the QoS of accessing video.

IV. Design and Implementation

A. Time-aware Provenance Graph

Since most video-sharing platforms do not remove videos, the lifetime of videos is infinite. Nevertheless, with the time passing, fewer and fewer people will access the videos, the videos eventually may not be accessed anymore after a certain point in time [12]. Jaipahkdee et al. have found that the average active lifetime span of videos was from 54.22 days to 96.38 days [13]. Xu et al. have found that this span has become more and more shorter [12].

If a video passed its life span, the index information of the video, the metadata and video file can be moved to slower and cheaper storage media. The hierarchical storage structures of the video index, the metadata and file are efficient. Similarly,

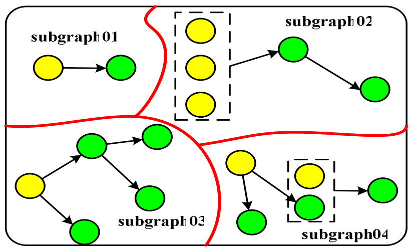

Fig. 2. Provenance Graph and Provenance Subgraph (A node represents a video. The yellow nodes are the head nodes of provenance chains. A directed edge, called provenance edge, means a causal relationship between two entities (nodes or sets of nodes))

only the part of the video provenance graph stores on the cache. As time goes on, the provenance of several videos can be pushed to next level storage media. Hence, Provis builds a time-aware provenance graph.

1) Provenance Collection: In order to obtain specially appointed provenance, the different particular points in the software stack need to be modified. For example, through modifying the kernels of operation systems, PASS [14] can intercept system calls made by applications and investigate provenance. Burrito [15] captures user activity context to automatically collect provenance of applications.

Provis collects the application-level provenance via modifying specific video-editor. Since the online video editing deserved more and more attention by application developers with the dramatically growing trends of online video applications, Provis can collect the application-level provenance in two scenerios, the offline video editing and the online video editing. In the first scene, when a user processes the videos via a modified video-editing software, the software automatically stores the video-editing commands locally. When this user attempts to upload the edited videos, the provenance are uploaded. In the second scene, Provis captures the commands on cloud (system) side, and records some timestamps, those information compose the primitive provenance.

2) Provenance Graph Indexing: After Provis captures the primitive provenance, some pre-processes need to be done. In order to recognize the duplicate video, Provis uses a compact video representation (video fingerprint) to uniquely identify a video, called *V*ideo *ID*entifier (VID). Youtube uses a string with 11 characters to identify a video [16]. Some studies have focused on computing the video fingerprint [2]. Provis needs to replace the filename (or an assembly of video's storage path and filename) with VID in wild provenance.

Through editing the videos, several videos can compose a number of disjoint video sets. Similarly, there are some disconnected provenance subgraphs. As shown in Figure 2, the provenance graph includes four provenance subgraphs (*subgrap01*, *subgrap02*, etc), a yellow node means a head node of subgraph and a green node means a new generated video. Each subgraph gets a unique numeric to identify itself, called *S*ubgraph *ID*entifier (SID). Since users like to merge some classic video clips which come from several videos to a new video, a video's origin may be a set of videos. Figure 2 shows that the video (the first green node) can be obtained via editing three videos (the left three yellow nodes) in the *subgraph02*.

As described above, a provenance graph can be divided into many subgraphs. In order to improve the performance of provenance graph searching, the index of provenance graph is organized into a two-level structure in Provis. The first Level builds the provenance subgraph index structure, the second level indexes all of provenance graph nodes in one subgraph.

3) Provenance Graph Storage and Management: Since the provenance graph of videos is time-aware, Provis uses a hierarchical storage structure of the videos' provenance graph and adopts a kind of the temporal locality replacement policy also. The size of the provenance subgraph is varied. The larger subgraph means more attention in current web-based video-sharing systems. In accordance with the last access time and the size of each provenance subgraph, Provis adopts a simple LRU-like algorithm to schedule the provenance graph of videos.

Since a edited video may come from several videos as described above, the provenance graph is not a common directed graph. Provis uses a key-value structure to store and manage the provenance graph. Towards the management of provenance graph, our implementation includes adding and updating operation about nodes and edges, and also includes the adding and merging operation of subgraphs. The size of provenance of a video is always a lot smaller than itself. If a video is deleted by its uploader, its provenance will continue to be stored. The reason of video deletion, security or space limitation, determines whether the video can be accessed by other users.

4) Provenance Graph Searching: Provis uses the VID of the target video as the key to quickly obtain the provenance of the video. Nevertheless, when a user attempts to download a video which is uncomfortable for his/her view machine or is unavailable, Provis can smartly recommend the other videos which adjacent to the video in the provenance graph. Provis needs to find the ancestors of one video and its descendants via the provenance graph searching. The VIDs of the original videos can be used as the secondary key to achieve above-mentioned functions.

B. Four Enhanced Services of Provis

1) Effective Provenance-based Video Compression: Provis can compress the videos via storing their files replacing with only storing their provenance to improve the space-efficiency. Because the size of a video's provenance is always far smaller than its video file, this method actually uses recomputation as a replacement for storage. Adapting this method mainly considers three respects in Provis.

The first one is "*When does the system launch the compression operation?*". Compression is often performed to get a further storage space saving. The video compression operation can be periodically executed like the System Sweeper. Because of the overhead of video regeneration, A administer can choose many periods of a light server load as a point for launching compression. The second one is "*Which videos can be compressed?*". The decision to compress a video depends on the comparing between the cost of video storage and the cost of video regeneration. Later in the next section we perform a cost analysis about it. The last one is "*How to execute the compression operation?*". We mainly discuss this point in this subsection. The provenance-based video compression includes

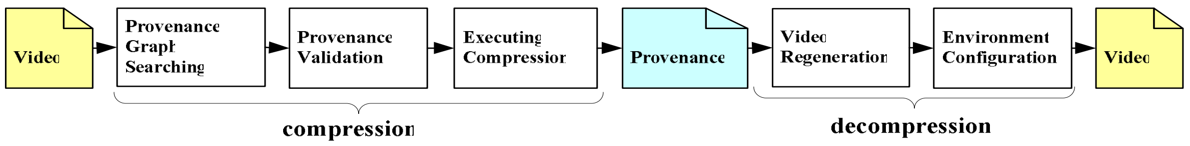

Fig. 3. The Function Module Diagram of Provenance-based Video Compression

three function modules, and the decompression includes two modules, as shown in Figure 3.

As discussed above, the module of provenance graph searching can find all of the provenance paths to the node (video) which needs to be compressed in provenance graph. In order to ensure the reliability of provenance, Provis uses a module, called Provenance Validation. Provis defines *weak* and *strong* validation. The first definition only verifies whether all of the ancestral video files and the provenance are stored on the servers. The second definition not only assures these information is stored, but also chooses a provenance path which has the minimal overhead of regenerating video to recreate the video. The choice of the validation method depends on the integrity of provenance data. If a video passes the validation, the next step is to execute video compression. Provis will delete the video and set a flag, called *video_exist*, in the provenance graph to indicate that this video is not stored.

When a user requests to access a video which has been compressed, Provis will decompress (recreate) this video. This part includes two modules, the *Environment Configuration* and the *Video Regeneration*. The first module needs to prepare the data which includes provenance and ancestor files and software. The software and its vision can be ignored within a short period of time. Nevertheless, Towards recreating a video which has been compressed a relatively long time ago, it is necessary to store the relational softwares. The another one completes three tasks: (1) choosing the provenance path with the minimal cost, (2) executing the commands in provenance to recreate video, (3) deleting the temporary files and setting the flag *video_exist*.

2) Low-cost Video Uploading: When a user attempts to upload a video without editing operations, Provis computes a content-aware hash signature as the video's VID, and then sends this VID to the server. After receiving the VID, Provis needs to judge whether the video is a duplicate video via searching the video metadata index. If this VID exists, Provis sends a message to the user and points out that this video is a duplicate video. Otherwise, the user has to upload the video and its relational information.

When a user attempts to upload a video which was edited by himself/herself, The VIDs of the video and its original videos, and the provenance information are sent to Provis. After receiving those data, Provis firstly validates whether the original videos are stored. The following three circumstances are to occur according to the result of validation. First, all of the original videos can be found in the system. If the cost of uploading the video is more than the cost of recreating the video, Provis will recreate the video as same as above provenance-based video decompression. If the provenance uploaded are not repeated, it will be added to the provenance graph. Then, Provis will send a *Upload_Done* message to the user. Second, only part of the original videos can be found in the system. Provis will tradeoff between the cost of uploading video and the sum cost of uploading the absent videos and recreating video to determine whether the video is recreated in cloud storage. Third, no original videos can be found in the system. The video will be directly uploaded to the cloud.

3) Smart Video Downloading and Adaptable Cloud-based Video Transmission: The video downloading is one of the most common operations. Nevertheless, there are two scenarios which often happen during a user downloads videos. 1) After a user downloads a video from web server with difficulty, the user finds that the video is probably not comfortable for the user's machine, such as, resolution and format. 2) The video which the user attempts to download is unavailable for some reason, nevertheless, there are the relevant near-duplicate videos which are usable for the user. Since the videos which are uploaded by users commonly have poor metadata information, it is difficult to find their relevant near-duplicate videos via metadata search. Hence, if the provenance of the videos can be collected, Provis can recommend their relevant videos via provenance graph search.

Currently, the transmission of big files between two users has developed into the three party transfer in the cloud. The sender firstly transfers the data to cloud. And then the recipient downloads the data from cloud. The two above situations will still occur. For example, a user sends a high-definition video to another one. If the video-viewing device of the recipient is a mobile device, such as smartphone, the video may be unavailable or unsuitable for the recipient. If the near-duplicate videos with different definitions or formats can be found via provenance graph search, Provis can automatically send the comfortable video to the recipient.

If the reason of deleting a video by users is the video upload size limit of the video-sharing platform, this video can be recreated and reused. Hence, more relevant near-duplicate videos can be obtained via provenance graph search. Provis can find the current or the past related shared videos.

C. Cost Analysis

The cost analysis is an important part of Provis. The results of analysis are used to determine whether video uploading replaces video regeneration in the function module of *Low-cost Video Uploading*, and whether video storage replaces with video compression in the function module of *Provenance-based Video Compression*. In order to clearly describe the cost analysis in Provis, we defined some common parameters which are shown in Table I.

1) Storage and Computation Cost Analysis: When recreating video is cheaper than to store it for a given period of time, provenance-based video compression will be cost effective. According some researches on video access pattern [17], we can predict the time interval between consecutive accesses of

TABLE I. NOTATION USED IN COST ANALYSIS

Label	Description
C_C	Cost of compute per hour
C_N	Cost of data transfer per GB
C_S	Cost of storage per GB per hours
M	Ancestor files of video V, M $\Rightarrow$ V
V	Target video
τ	Time overhead of recreating video V
S_M, S_V, S_P	Size of M , Size of V, Size of provenance

the video V which will be compressed. We assume this time is T_p (hours).

If we do not compress the video V, the cost of video, called $C_{uncompressed}$ is mainly the cost of storage, so,

$$C_{uncompressed} = C_S \times S_V \times T_p$$

If we compress the video V, the cost of video, called $C_{compressed}$ includes the cost of recreating and the cost of provenance storage, so,

$$C_{compressed} = C_C \times \tau + C_S \times S_P \times T_p$$

It will be cost effective to compress the video V, when:

$$C_{compressed} < C_{uncompressed}$$

$$\Rightarrow C_C \times \tau + C_S \times S_P \times T_p < C_S \times S_V \times T_p$$

$$\Rightarrow C_C \times \tau < C_S \times S_V \times T_p \text{ (Commonly, } S_P \ll S_V)$$

$$\Rightarrow \frac{C_C \times \tau}{C_S \times S_V} < T_p \quad (1)$$

Since the time interval between twice system compressions is short, the left value of the equality (1) is constant. With the passage of time, the popularity of a vide will decline, and T_p will increase. Once Provis checks a video's T_p is larger than the left value of the equality (1), the video will be compressed.

2) Transmission Cost Analysis: If we directly upload the video V, the cost, called C_{upload_video} is mainly the cost of transfer, so,

$$C_{upload\_video} = C_N \times S_V$$

If we upload the provenance of the video V, the cost, called $C_{upload\_provenance}$, includes the cost of recreating video and the cost of uploading provenance, so,

$$C_{upload\_provenance} = C_C \times \tau + C_N \times S_P$$

It will be cost effective to upload the video V, when:

$$C_{upload\_provenance} < C_{upload\_video}$$

$$\Rightarrow C_C \times \tau + C_N \times S_P < C_N \times S_V$$

$$\Rightarrow C_C \times \tau < C_N \times S_V \text{ (Commonly, } S_P \ll S_V)$$

$$\Rightarrow \frac{C_C}{C_N} < \frac{S_V}{\tau} \quad (2)$$

The left value of the equality (2) is also constant. The right value reflects the complexity of recreating video. Most of online video-editing systems provide few simple edit operations, such as cutting video, merging clip and adding music. This phenomenon exposes that most of person only use some simple edit commands. Hence, we speculate that most uploading videos can adapt our method.

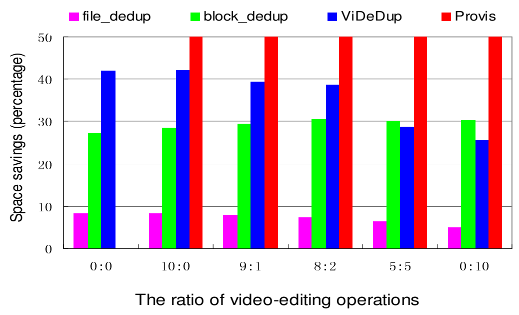

Fig. 4. Compression Ratio vs the Video-editing Operation

V. PERFORMANCE EVALUATION

A. Experiment Setup and DataSets

To test our approach, the cloud-side of Provis prototype is implemented on one server of a cluster with hundreds of servers. Each server includes an Intel E5620, 11GB of main memory, 1.7TB ST3250310AS disk, and a high-bandwidth network. The client-side of Provis prototype runs on a personal computer. We modified the part program of FFmpeg to capture the video-editing provenance as the video editor in Provis. Provis uses the Berkeley DB to index and store the provenance graph.

In order to verify our algorithm in some widely accepted application scenarios, we evaluated our Provis prototype via using a representative video set [1]. Since the videos with time duration over 10 minutes were removed from the video set, we download hundreds of videos with the above method. The format of videos is AVI and their sizes are from 1MB to 2GB.

B. Result and Analysis

1) Video Compression Ratio: There are several technologies which can be used to compress the data for saving the storage space. The file level de-duplication, the block level de-duplication and ViDeDup [8] are utilized mainly in the video-sharing systems. We verify the compression performance of Provis via comparing these three methods with our work.

We divide the video-editing operations into two types, the video transcoding and other video-editing commands. Two reasons make us to do it. First, towards the providers of video-sharing service, the online video-transcoding has become important in reducing the net overhead of video viewing and storage cost of videos [11]. This kind of operation is the base of other video-editing operations also. Second, towards the video-editing users, the other operations are more important. Since the transcoding operations of the videos can not change the semantic content of the videos, nevertheless, the great mass of other operations can change the semantics of the videos. For example, the operation of extracting audio will lose a portion of visual information. Current users always use many original videos to create a new video for interest, celebration, or other goals.

Figure 4 shows the performance of several compression methods after only a step of vide-editing operation, such as converting AVI format videos to MP4. The ratio of video-editing operations means the ratio between the transcoding operations and other operations. For example, "0:0" means all of the videos are not edited and "8:2" means that eighty

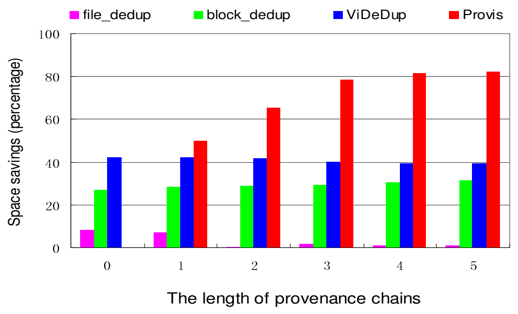

Fig. 5. Compression Ratio vs the Length of Provenance Chains

percent of editing commands is the operation of transcoding. In order to deliver more visually appealing the compression performance, we use the percentage of the space savings to describe the compression ratio. From the Figure 4, we can include two points: (1) when we can not get the provenance of video (the ratio is the "0:0"), ViDeDup is the best method to compress the videos. The reason is that ViDeDup can find and de-duplicate the near-duplication videos via exacting and comparing the semantic contents of the videos. (2) If we can get the video-editing provenance, no matter how the ratio changes, Provis can save almost invariable 50% of storage space.

Toward viral videos, the probability of being edited apparently is more. In order to measure the impact of the length of provenance chain on compression ratio, we assume that a provenance edge stores the information about a command of video-editing operations. a provenance edge in a provenance path is deemed to a step of provenance chain. As shown in Figure 5, with the increase of the length of provenance chain, the compression ratio of the file level de-duplication reduces quickly, and the compression ratio of the ViDeDup and block level de-duplication almost remains unchanged. Nevertheless, Provis can provide a growing performance of saving storage space and about 80% of the storage space can be saved.

2) Storage Overhead of Provenance Graph: Since Provis only preserves the commands and the video-editing parameters and does not store the environmental variables of system and the other metadata of videos, the storage overhead of provenance graph satisfies a linear growth with the increase of the video-editing operations. Our test confirmed this. If we store a command in each edge of provenance graph, we observe that the space consumption relies on "linear" growth, as shown in Figure 6. We also find that the average space overhead of storing a edge is about 100B. If the average video-editing steps are 100 (Obviously, most of the videos do not need so many steps of video-editing operations), We can get the average size of the provenance of a video is about 10KB ($100 \times 100B$). This size is much smaller than the sizes of the videos. Hence, Provis can greatly improve the performance of uploading videos. We assume that 1 million videos are uploaded everyday, and 27% of which is the edited videos, We can get the storage overhead of provenance graph per day is 2.7GB ($27\% \times 10^6 \times 10KB$). This cost is acceptable to the cloud storage providers.

3) Time Overhead of Provenance Graph Search: During the video regeneration, Provis needs to cost the additional time to do provenance graph search. For example, Provis need to find the original videos of a video which is attempted to

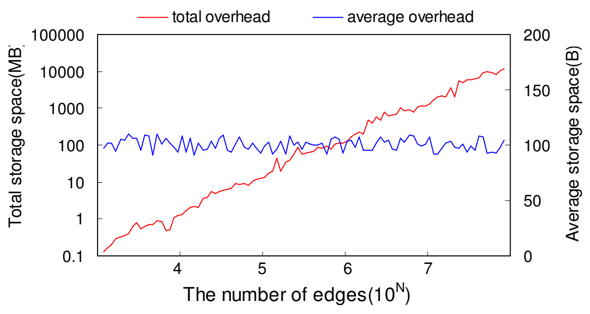

Fig. 6. The Storage Overhead of Provenance Graph

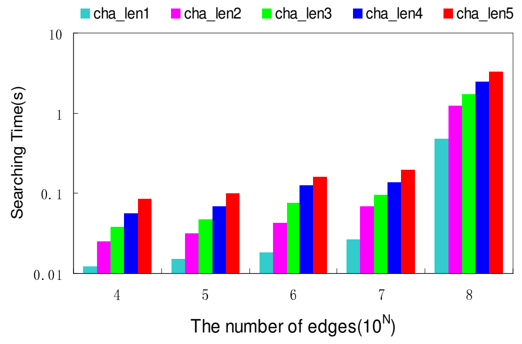

Fig. 7. The Time Overhead of Provenance Graph Search

recreate via provenance graph search. With the growth of provenance graph, the time overhead is also increased, as shown in Figure 7. Nevertheless, when the provenance graph is too large and need to store a portion of them into the disk, the time cost will significantly increase. Hence, there is a huge benefit with a reasonable large memory.

4) Time Overhead of Video Regeneration: As mentioned earlier, The time overhead of a video regeneration mainly includes two parts, the searching time of its provenance and the execution time of the video-editing rebuilding. We have measured the first one above. We mainly measure the second one in this subsection.

The sizes of the videos which we choose are close to the five levels (100KB, 1MB, 10MB, 100MB and 1000MB). We do eight kinds of video-editing commands which are the most commonly used, as shown in Figure 8. Since the command of the concatenating videos need to transcode the original videos into the videos which have the same quality, the command of the concatenating videos can be considered to be a special transcoding command. We can find that the transcoding commands obviously take longer than other commands, about three orders of magnitude. The time of transcoding is far greater than the total execution time of hundreds of video-editing operations and provenance graph searching. Since the time delay of transcoding operations can be accepted by end users in the current video-sharing services, we think the time delay of the video regeneration can be accepted too.

VI. Related Work

Due to the ever growing numbers of near-duplicate videos, Many studies have focused on the near-duplicate video detection and elimination for effective copyright protection, search, retrieval, and tagging [18]. There are several methods to determine whether two videos are near-duplicate. Those methods

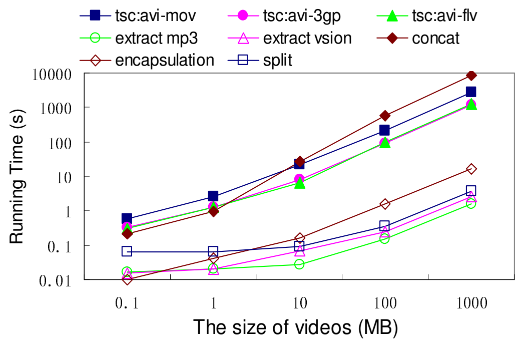

Fig. 8. The Average Running Time of Different Video-editing Operations

can be used to find some similar videos, nevertheless, need large number of computation to construct and compare the signatures of videos.

ViDeDup [8] compares the ordinal signature to find and de-duplicate similar videos. The video-signature computation and the pairwise video-comparison need a lot of overhead in ViDeDup, like the methods above. This method can address the videos with same content and different formats or frame resolution or frame rate. Provis can compress similar video via adding few space overhead of storing provenance, and also find the videos with similar semantic contents.

Recent efforts have mainly focused on using the provenance of videos to improve the security and performance of video-sharing systems. In order to help consumers to decide how much to trust a video, VEIL embeds the provenance of a video clip into the data itself [19]. Halaschek-Wiener et al. [20] used provenance information to annotate videos and to enrich the user's browsing experience on semantic web.

The space-time tradeoff had been analyzed and leveraged in some potential applications. Adams et al. [21] designed a cost model to strike a tradeoff between storing a result and computing a result. The cost model includes three key areas (the semantic meaning of the result, marginal costs and the forecasting of price) in cloud computing. Atish Kathpal et al. [22] achieved storage efficiency via potentially eliminating some video versions which can be rebuilt using on-the-fly transcoding in video-sharing systems. A part of similar videos are only found and compressed in this method like ViDeDup. Since the full video-editing processes can be recorded via collecting provenance on user-side or cloud-side, Provis can compress more similar videos and reduce the network overhead of uploading videos.

VII. Conclusions

Currently, the performance and the user experience of video-sharing services have suffered from the more and more similar videos on video-sharing platforms in the cloud. The similar videos can not be efficiently de-duplicated or compressed via the traditional methods. In this paper, we proposed a novel prototype system, called Provis. Provis leverages the provenance of videos to reconstruct and reuse them. Though the test, we observe that Provis can save about 80% of storage space via provenance-based video compressing, and the time delay of video regeneration can also be accepted. Hence, Provis can not only improve the storage efficiency, but also provide the low-cost video uploading.

Acknowledgments

The work was supported in part by National Basic Research 973 Program of China under Grant 2011CB302301 and National Natural Science Foundation of China (NSFC) under Grant 61025008, 61173043 and 61232004.

References

[1] X. Wu, A. G. Hauptmann, and C.-W. Ngo, "Practical elimination of near-duplicates from web video search," in *ACM Multimedia 2007*.

[2] Y. Cai, L. Yang, W. Ping, F. Wang, T. Mei, X.-S. Hua, and S. Li, "Million-scale near-duplicate video retrieval system," in *ACM Multimedia 2011*.

[3] S. Mitra, M. Agrawal, A. Yadav, N. Carlsson, D. Eager, and A. Mahanti, "Characterizing web-based video sharing workloads," *ACM Transactions on the Web (TWEB)*, vol. 5, no. 2, p. 8, 2011.

[4] C. V. N. Index, "Forecast and methodology, 2013c2018 (june 10, 2014)."

[5] Accenture, "Video-over-internet consumer survey 2013: Multi-tasking and taking control," Online: http://www.accenture.com/SiteCollectionDocuments/PDF/Accenture-Video-Over-Internet-Consumer-Survey-2013.pdf.

[6] W. Xia, H. Jiang, D. Feng, and Y. Hua, "Silo: A similarity-locality based near-exact deduplication scheme with low ram overhead and high throughput." in *USENIX ATC 2011*.

[7] R. C. Burns and D. D. Long, "Efficient distributed backup with delta compression," in *ACM IOPADS97*.

[8] A. Katiyar and J. Weissman, "Videdup: an application-aware framework for video de-duplication," in *USENIX HotStorage 2011*.

[9] L. Carata, S. Akoush, N. Balakrishnan, T. Bytheway, R. Sohan, M. Selter, and A. Hopper, "A primer on provenance," *Communications of the ACM*, vol. 57, no. 5, pp. 52–60, 2014.

[10] Y. Xie, D. Feng, Z. Tan, and J. Zhou, "Design and evaluation of a provenance-based rebuild framework," *IEEE Transactions on Magnetics*, vol. 49, no. 6, pp. 2805–2811, 2013.

[11] Y. Feng, B. Li, and B. Li, "Jetway: minimizing costs on inter-datacenter video traffic," in *ACM Multimedia 2012*.

[12] X. Cheng, J. Liu, and C. Dale, "Understanding the characteristics of internet short video sharing: A youtube-based measurement study," *IEEE Transactions on Multimedia*, vol. 15, no. 5, pp. 1184–1194, 2013.

[13] W. Jaipahkdee and C. Srinilta, "Data placement in heterogeneous storage of short videos," *World Academy of Science, Engineering and Technology*, pp. 773–777, 2009.

[14] K.-K. Muniswamy-Reddy, D. A. Holland, U. Braun, and M. I. Seltzer, "Provenance-aware storage systems." in *USENIX ATC*, 2006.

[15] P. J. Guo and M. Seltzer, "Burrito: Wrapping your lab notebook in computational infrastructure." in *TaPP*, 2012.

[16] A. Finamore, M. Mellia, M. M. Munafò, R. Torres, and S. G. Rao, "Youtube everywhere: Impact of device and infrastructure synergies on user experience," in *ACM SIGCOMM 2011*.

[17] G. Gursun, M. Crovella, and I. Matta, "Describing and forecasting video access patterns," in *IEEE INFOCOM 2011*.

[18] J. Liu, Z. Huang, H. Cai, H. T. Shen, C. W. Ngo, and W. Wang, "Near-duplicate video retrieval: Current research and future trends," *ACM Computing Surveys (CSUR)*, vol. 45, no. 4, p. 44, 2013.

[19] A. Gehani and U. Lindqvist, "Veil: A system for certifying video provenance," in *IEEE ISM 2007*.

[20] C. Halaschek-Wiener, J. Golbeck, A. Schain, M. Grove, B. Parsia, and J. Hendler, "Annotation and provenance tracking in semantic web photo libraries," in *Provenance and Annotation of Data*. Springer, 2006, pp. 82–89.

[21] I. F. Adams, D. D. Long, E. L. Miller, S. Pasupathy, and M. W. Storer, "Maximizing efficiency by trading storage for computation," in *HotCloud*. USENIX Association, 2009.

[22] A. Kathpal, M. Kulkarni, and A. Bakre, "Analyzing compute vs. storage tradeoff for video-aware storage efficiency," in *USENIX HotStorage 2012*.