Journal of Information Processing Vol.20 No.3 570–577 (July 2012)

[DOI: 10.2197/ipsjjip.20.570]

Regular Paper

Traffic Management Scheme to Control Content Distribution with Cloud-based Architecture

Haruhisa Hasegawa[1,a)] Noriaki Kamiyama[1,b)] Hideaki Yoshino[1,c)]

Received: September 11, 2011, Accepted: February 3, 2012

Abstract: While information and communication technologies (ICT) are expected to improve energy efficiency, the spread of ICT will also increase power consumption due to higher server loads and increased network traffic. This paper presents a traffic management scheme to control traffic volume and server load by using a virtual machine (VM) in cloud architecture. The proposed scheme solves the inefficiency concerning the difference in number between content that is distributed and that is actually played back. As a result, this scheme reduces network traffic volume by suppressing unnecessary content distribution. This scheme also efficiently reduces total server load by concentrating the load of content being played back onto a smaller number of VMs. We evaluated the load reduction achieved with our scheme and found that the load for distributing content was drastically reduced. Our proposed scheme is expected to contribute to achieving a low-carbon society in the effort to reduce global warming.

Keywords: content distribution network, cloud architecture, telecommunications traffic

1. Introduction

It is becoming increasingly important to achieve a low-carbon society to reduce the effects of global warming [1]. Information and communication technologies (ICT) have been expected to reduce carbon dioxide emissions by promoting teleworking, TV conferences, on-line trading and shopping, and efficient content distribution. However, Internet traffic is still increasing, which in turn increases the power consumption of ICT systems and networks. According to the Cisco Visual Networking Index, the compound annual growth rate of Internet traffic will be 26% from 2008 to 2013 [2]. The Ministry of Economy, Trade and Industry of Japan announced that by 2025 such traffic in Japan would increase to 190 times that in 2006. This increase in traffic will increase the power consumption of ICT equipment by 5.2 times in Japan and 9.4 times worldwide. Therefore, it is necessary to find a way to suppress this increase [1].

Content distribution traffic is predicted to become the major factor in the increase in Internet traffic. It is forecast that 75% of total Internet traffic will be from video and file-sharing services, which will be 40 times that of interactive service traffic [2]. This total Internet traffic is forecast to remain the same. Therefore, it is necessary to improve the efficiency of content distribution to reduce carbon dioxide emissions.

At the same time, the ICT industry is aiming for a "cloud" architecture using virtualization techniques [3], [4], [5]. In cloud architecture, usability is not much affected by the difference in performance levels of user terminals because most processes can be handled at the remote server platform. This characteristic suits ubiquitous environments, and such "cloud services" will spread widely in the near future. For example, a consumer can watch movies on his/her terminal, which may have only a low processing capability, by having the cloud platform process the heavy load instead. However, this architecture increases the total load and power consumption of data centers and networks.

When people consume content, they generally obtain it and then watch it. Therefore, the content must be distributed through a network before being played back. Content distribution is defined as when a consumer becomes the owner of the content. The process of playback is defined as when the consumer becomes the audience of the content. We propose a distribution and playback service with lower power consumption that still satisfies consumers.

Not all content distributed over the network is watched by consumers. Generally, the number of content owners is larger than the audience. The difference in these sizes indicates the inefficiency of content distribution. This means that the conventional method of distributing content, which is assumed to be watched, should be overhauled.

We propose a traffic-management scheme for controlling the traffic volume of content distribution and server load by using a virtual machine (VM) in the cloud architecture. In this scheme, content distribution and playback processes are strictly distinguished. Content distribution is controlled when the number of distribution requests is much larger than that of playback requests. At this time, each consumer can own the content virtually but share it in actuality. As a result, unnecessary content distribution, network traffic, server load, and power consumption can be reduced. The playback load caused by all audiences is concentrated as much as possible on a specific platform or server.

1 NIPPON TELEGRAPH AND TELEPHONE CORPORATION, Musashino, Tokyo 180–8585, Japan
a) hasegawa.haruhisa@lab.ntt.co.jp
b) kamiyama.noriaki@lab.ntt.co.jp
c) yoshino.hideaki@lab.ntt.co.jp

As the number of playback requests increases, the content becomes widely distributed by the copying of the content among consumers' VMs. Our management scheme reduces the unnecessary network traffic and server load of not only content distribution but also playback by concentrating the load on a relatively smaller number of servers. By reducing the number of lower-load servers, fixed power consumption can be avoided. In other words, the server load and traffic are controlled depending on the number of requests for content distribution and playback. Consumers in this scheme share content ownership and the playback loads of servers. This is expected to reduce power consumption in both servers in the cloud and on networks.

2. Categories of Distributed Contents

In general, consumers download content and then play it back. Downloading is the process of moving the content from the source host to the user host that has the capability for playback via a network. The degrees of popularity concerning download and playback do not always correspond. Therefore, content can be categorized as shown in **Table 1**.

The content in region I of Table 1 is popular among most users and is frequently requested both for download and playback. Since such content is often played back after being downloaded, the number of playback requests is larger than that of downloads. Therefore, a method that does not require download at every playback is preferable for efficiency. The architecture distributes content efficiently via a network, and the user watches it using the user's host. For the content in region I, a method assuming a massive number of download requests is appropriate, such as P2P, P4P, CDN technologies, and so on.

Content in region II is of a niche type and is the focus of enthusiasts. The consumer does not download frequently, but once content is downloaded the consumer often plays it back. Similar to region I, the number of playback requests is larger than the number of downloads. However, a massive number of download requests cannot be assumed. In this case, a content cache that is dedicated for a specific user group may be effective.

Content in region III is not very popular among consumers or is still obscure. The content should be consolidated in the storage center as archival content because it is not requested frequently.

In region IV, content is often requested for download but rarely played back. For example, a consumer may be a collector and collect content. In particular, heavy users behave in this way, i.e., they collect huge amounts of popular content but rarely play it back. It is inefficient to copy content through the network because the user does not play it back. If the users are guaranteed the right to use the content at anytime, it is not important to them whether they actually own it or not.

Table 1 Content categories.

		Requests for download	
		less	more
Requests for playback	more	Region II Niche	Region I Popular
	less	Region III Obscure	Region IV Collectable

Recently, the niche and obscure types of content, i.e., long-tail content, has increased, and the tail has also become longer. It was reported that 24% of the sales of downloadable content in 2007 was for content sold only once, and 91% was for content sold less than 100 times [6]. This means that plenty of content exists in region III. As mentioned above, region III has content that is still obscure and not frequently used but that is nevertheless valuable. The business model for suggesting long-tail content based on each user's preference has been tried in the real world [7]. Such a business model moves obscure content to region I or IV.

As long-tail content is in region III, it is a waste of both network and server resources for the content to be delivered to users in response to all requests. A long-tail business model recommends to many users some content that moves to region II and/or IV. Because the number of download requests is less than that of playbacks in region II, content should be moved to user hosts as mentioned above. In region IV, however, since there are more download requests than playbacks, the network load should be controlled until the content becomes popular and is actually played back. Then, finally, some of the content will move to region I. Here, both download and playback requests are frequent and almost equal in number. It is clear that content has a lifecycle. It is important for efficient content traffic management to distribute long-tail content according to its lifecycle. Therefore, a method is needed to hold content in a concentrated server when the content is first generated and then distribute it gradually in response to the volume of requests for download and playback.

3. Power Consumption of Content Distribution

A content distribution service is provided by a server platform with content and a network that mediates between platforms and users. Generally, the total electric power consumed by a server consists of a fixed amount and an amount that depends on the load [8]. In platform as a service (PaaS), i.e., architecture with virtualized servers, the power consumption of a server can be controlled with regard to the total load of all VMs that belong to the same hardware platform. A cloud platform consists of a massive number of servers. If the server load is concentrated on a smaller number of servers, the rest of the servers do not need to process and can switch to the "sleep" state [9]. As a result, unnecessary fixed power consumption can be suppressed.

In contrast, the power consumption of routers is relatively independent of the load [10]. This means that the power consumption of a server platform can be controlled adaptively by optimizing the entire load across the servers. To reduce the power consumption of a network, the regular traffic volume, which should be assumed by the network designer, must be controlled. In other words, it is difficult for a network to control power consumption by optimizing the traffic load adaptively. As a result, unnecessary content distribution must be avoided, and the total average traffic volume must be suppressed. As mentioned in the previous section, content in regions III and IV is not played back frequently. If content distribution via the network can be avoided until the content is actually played back, the average traffic would be suppressed.

In the next section, we present a traffic control technique to efficiently manage content with low playback frequency. It is also suitable for managing content that is increasing in popularity, i.e., region III to I (via II or IV).

4. Proposed Traffic Management Scheme

We propose a traffic management scheme that distinguishes content distribution and playback processes. Our management scheme reduces the unnecessary network traffic and server load not only of content distribution but also playback.

The basic configuration for this scheme is illustrated in **Fig. 1**. Each user belongs to a cloud service that has a VM dedicated for the user. The user watches or listens to his/her own content through his/her terminal, but the storage to which to save content is shared with other users in the cloud. In ubiquitous environments, a user terminal is not expected to have enough storage and capability to handle richer content. A consumer can play back such content on their terminal by having the cloud platform rather than the terminal process the heavy loads.

Generally, a cloud service provider (CSP) can control the total power consumption with regard to aggregated utilization of VMs. One study defines "cloud" as the data center hardware and software [11]. The datacenter platform consists of multiple servers and storage hardware. The capabilities of the platform are shared among VMs. Since some of the VMs are active and others are inactive simultaneously, the total load of the platform can be utilized efficiently from a statistical viewpoint. For example, if only a smaller number of VMs is utilized in actuality, the load can be concentrated on a smaller number of servers. Then, the fixed power consumption can be reduced by letting unnecessary servers switch to sleep mode.

In this paper, we assume the cloud is managed based on the "pay as you go" [11] model, where a VM is active only while

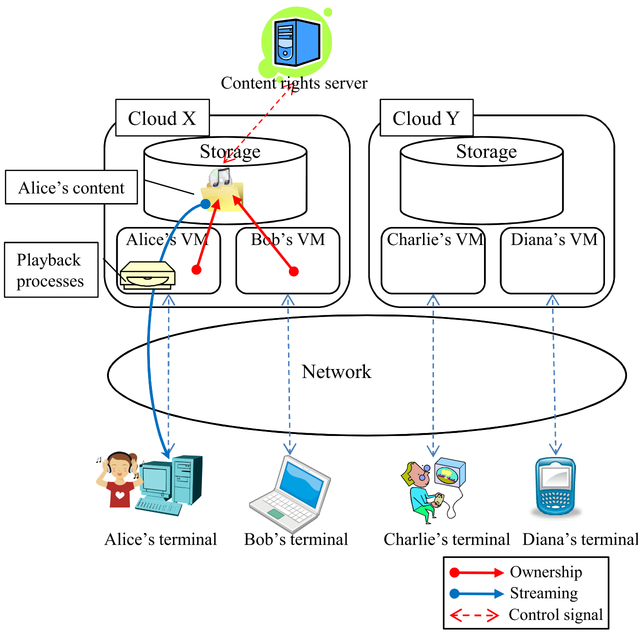

Fig. 1 Basic system configuration for proposed scheme.

the user pays to utilize the cloud resources. The upper limit of resources that each VM can utilize is regulated by the contract. Then, the model allows the cloud to determine how many VMs are active and to estimate the upper limit of the total necessary resources. The cloud can assign necessary resources to VMs because it knows which VMs are active. However, if some resources are in sleep mode, the capability of the cloud decreases. Because each VM is assigned sufficient necessary resources while it is active, the sleep mode of VM does not affect the content playback speed. On the other hand, the shared storage must be kept in the awake mode even in the "pay as you go" model because it takes time to "wake up" the storage in sleep mode when a user wants to play the content in it. Here the situation is categorized into two cases of whether or not the content owner and the user (requester) who wants to play it are in the same cloud. (a) The content is stored in the same cloud that both the owner and requester belong to. All of the storage that stores content possibly requested by active VMs should be in the awake mode. Because both VMs belong to the same cloud, it is possible for the cloud to know whether they are active or inactive. (b) The content is stored in a cloud that the owner belongs to, which is a different one from the cloud the requester belongs to. In this case, it is difficult for the owner's cloud to know whether the VM of the requester is active or not. Therefore, the storage shared among multiple clouds must be in the awake mode.

In typical cloud services, users save their own content in the storage of the cloud to which they belong. If the user wants to watch or listen to content, he or she requests the VM to play it back and can experience a streaming-like service via the user terminal. In this case, the terminal behaves like a thin client. We define "ownership of content" as follows: a user owns content when the user can play it back anytime he or she requests, and can also discard it at anytime.

The Over the top (OTT) VoD services are becoming increasingly common as a means of distributing content. Consumer generated media (CGM) services are a means of providing content that is uploaded by general consumers. All the content distribution of both services is managed by the application service providers. By contrast, our proposed scheme will provide a content distributing platform for consumers. It will make it possible for users to distribute their own content via their own VMs. For example, users will be able to distribute content that was generated by themselves and that is partially or completely scrambled. The users can also sell the necessary information to descramble it. The cloud provider can acquire the demand for long tail content while suppressing their distribution load. This, as a result, is expected to increase the value of long tail content generated by consumers by storing the content in common storage in clouds and to make it possible for such content to be utilized by other consumers.

In a cloud service, the storage is also virtualized, and it does not need to be explicitly clear where the content is stored. In Fig. 1, Alice owns content in her VM, i.e., the storage of cloud X. Bob can own the same content by sharing it with Alice if he has the right to access Alice's saved content at anytime without requiring another user's permission, and he can discard that right at any time. In this scheme, although the number of content owners increases, the load for copying content is not always required. Only the control message to manage the right of ownership is necessary. This can suppress the load and traffic volume even while content ownership spreads. Alice and Bob do not need to belong to the same cloud. For example, if the centralized content rights server manages the right of ownership, the same content can be shared among multiple clouds. We define the following methods to achieve the scheme described above.

(1) Publish

A user obtains or creates new content and then saves it as original content in the storage of the user's VM. The content can be shared because the storage is virtualized, and it can be accessed by other VMs if permitted. The user (content holder) who has the original content registers the content to the content rights server (registrar) using the "publish" method. The user's VM advertises that the content can be accessed by other VMs and indicates which cloud it is stored in by publishing it. Then the published content becomes sharable with other users who notice the permission and location of the content.

Each VM has its own private storage as well as the conventional cloud architecture. Other VMs are prohibited from accessing data that are held in private storage. In other words, private content is protected as safely as it would be in the conventional architecture. The content will only be available to other users if the content owner publishes the content, which means that it will be stored in the public storage space and made available to other users. The content should be safe against malicious and repetitive attacks. In our proposed architecture, user terminals are not allowed to access content directly. Because only VMs can access content, a malicious attack would only be caused by another VM. For example, a malicious VM might send "subscribe" or "refer" requests massively. Within a single cloud, such attacks can be detected by finding a concentration of specific content. It is also relatively easy for the cloud to determine which VM is malicious. It is difficult to detect an inter-cloud attack only within the cloud that stores the content. However, the content rights server can find signal concentration and identify which cloud is carrying out the repetitive attack and send the information to the clouds both perpetrating and receiving the attack. The cloud being attacked can control the number of signals from the cloud where the malicious VM is. The attacking cloud can analyze the VMs and find out which one is malicious and turn it off. Therefore, it is possible to control the signals caused by repetitive attacks through the collaboration between content rights servers and clouds.

(2) Subscribe

A user who wants to obtain content that has already been published requests it from the registrar using the "subscribe" method. If "subscribe" is permitted, the same content can be shared among users.

The conditions for permission are defined by the original content owner. The conditions for the copied content are inherited from the original content. The conditions can include the policy on allowing "subscribing," such as that copying is not allowed between clouds. The conditions are registered in the content rights server by the original content owner. The content rights server judges whether a "subscribe" request should be allowed based on the conditions, and permits it if the requester's profile corresponds to the conditions.

Here, the VM that has published the content and the one that subscribes to the content do not always need to belong to the same cloud. It is not necessary for the original content to be copied to the VM that subscribes to it since the registrar only tells the cloud to open the right of ownership to the VM by sending control signals. At this step in the "subscribe" process, the VM does not copy the content, but knows that other VMs have reserved the right to play it. Since our proposed scheme restricts the copying process at the "subscribe" step, the judgment of whether or not the content should be copied is done at the "refer" step described in the next subsection. This is decided depending on the number of concurrent users playing the same content. This condition and process are described in the next subsection. Therefore, power consumption is relatively lower because only the load to control the right of ownership is required. This will also keep the network traffic volume relatively lower until the number of playback requests increases.

In this scheme, shared content must not be modified by any user. If a user modifies the content, the content should be treated as new and original; in this case, the modified content must be newly stored in the storage. The problem of synchronization occurs among users who are sharing the same original content if it is modified by one of them. Here, such users are called "subscribers." The ownership is virtualized as described above in this subsection, and the subscribers cannot see that the original content was changed. Since a modification of content involves the original and modified content, the users can also choose to subscribe to the modified content. The user who modified the content should publish it as new and notify the other subscribers and the original content owner. The "publish" must be based on the conditions of permission, which should be equal to or stricter than the original conditions. The subscribers may want to cancel the "subscribe" for original content. The original content, however, must be kept in the shared storage as long as a subscriber for it remains.

If the owner deletes the content, it becomes impossible for the subscribers to utilize it. If the owner hopes to delete the modified content derived from the original one, it should be deleted. In that case, the genealogy tree must be managed by the content rights server. Only if the owner allows the subscribers to utilize the content will the content be kept in the shared storage. In that case, at least one of the subscribers will inherit the ownership. The owner may also try to delete content while other subscribers are in the process of playing it back. In this case, there are two options. The first option is that the content is eliminated from the storage at the same time the owner tries to delete it. As a result, the playing process will suddenly be aborted. The other option is that the ownership will be temporarily inherited by the subscribers who are playing the content. After the playback is completed, the inheriting subscriber deletes it. However, in this case, there is the risk that the subscriber may not actually delete the content.

(3) Refer

When a user (requester) wants to play back the content, a "re-

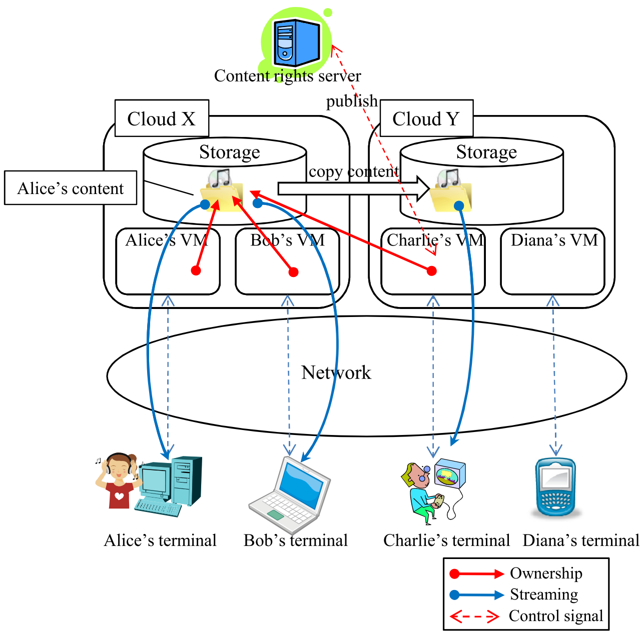

Fig. 2 Content distribution between two clouds.

fer" request is sent from the requester's VM to the registrar. Then, the registrar orders the VM of the content holder to play back the content for the requester. The content can be provided as a media stream to the requester from the cloud to which the content holder belongs. As the number of "refers" for a specific content holder increases, the load of that cloud becomes heavier. At the same time, the situation indicates that the niche content in the long-tail of a distribution is becoming popular and that the content should be distributed more widely.

In this scheme, if the number of concurrent users of the same content exceeds the threshold, the next requester must copy the original content to his or her own VM while playing it back. After the requester completes the copying, the VM publishes the content to the registrar. Upon completion of copying the content, the "publish" must be the same as the original setting, which was determined by the original content owner. When this scheme is actually implemented, the user's policy should be used as the permission setting. For example, the user may wish for the content to be distributed only in the same cloud as the one the user belongs to. In that case, the original setting must be inherited by the copied one. Therefore the publish action does not change the original setting, but registers the publisher as an owner of the copied content. Because the content source increases in number whenever the number of concurrent copying processes exceeds the threshold value, the number of content servers increases with the number of requesters. The threshold can therefore be decided to have one or more values as a parameter.

The scheme is illustrated in **Fig. 2**. The threshold is set to 2 as an example in the figure. At first, Charlie only has the right to access Alice's original content. However, with his request to play it back, the number of requesters has exceeded the threshold. Then Charlie's VM copies the content while it plays it back.

The load for content distribution is reduced as the number of subscribers who share the same content becomes larger. On the other hand, as the number of users playing the content simultaneously increases, the load of the cloud also increases. The threshold arbitrates between these two characteristics. The threshold needs to be determined not by the load of each VM, but by one of the common elements of the cloud, mainly storage, I/O interface, or common CPU. The upper limit of the threshold can be set to the value of the processing capability divided by the number of content items in order to design the most robust performance for maximum load. The threshold can be adjusted dynamically according to the load of the common cloud element to maintain efficiency. Since it is better to copy the content before the load is too high, the threshold should be set lower. If copying it to another cloud occurs prior to it being copied within the same cloud, the load becomes distributed among the clouds.

In our proposed traffic management scheme, content distribution is controlled when the number of distribution requests is much larger than that of playback requests. In the traditional content distribution service [12], a content download is triggered by a user's request for it. In that case, the number of downloads and requests correspond, but the downloaded content is not always played back by the users. We introduced the "playback ratio" as the ratio at which the user plays back the content after obtaining it. The playback ratio expresses the ratio between the number of distribution requests and that of playback requests. The playback ratio affects the performance of this scheme because its efficiency depends on the shared ownership between multiple users of content that is not played back often. In the next section, we clarify how the playback ratio affects the performance quantitatively by comparing the conventional and proposed cloud architecture. The scheme reduces the amount of unnecessary network traffic and server load of not only content distribution but also playback by concentrating the load on a relatively smaller number of servers.

In Table 1, regions I and II have more requests to play back per subscribe request than regions III and IV. From an implementation viewpoint, since it is efficient for a cloud to smooth the peak load, frequent copying of content should be avoided while massive playing processes are running. There is an option to copy such content beforehand that it is known from the empirical information that the playing back is concentrated. In other words, this option is applied for the situation when the content of IV is becoming that of I in Table 1. This policy can be implemented by copying the content that has a large number of "refer" requests in advance. When the content of II is becoming that of I, there is no problem since the number of "subscribes" becomes large in our scheme. It is not necessary to worry about the content in III until it becomes that of II or IV.

It is generally required that both the user terminal and VM have sufficient capability to play content. One concern with the proposed scheme is that the capabilities of the terminals and VMs are not uniform. When the VM does not have the necessary capability to play the content, the playback experience is degraded. If the necessary capability is clear, each VM can estimate whether it can play the content or not in advance. ITU-T Recommendation H.264 defines the profile and level to indicate the required performance of the playback device [13]. The VM that conforms to the level of the content is required to be capable of playing back content for that level and for all lower levels. For example, Amazon

EC2 has multiple instances of VMs with various processing capabilities such as the number of CPUs and the memory size [14]. The information on which instance is necessary becomes clear by benchmarking the capability related to the content level. In our scheme, the information should be registered with the content rights in the content rights server. In clouds based on the "pay as you go" model [11], the processing capability can be changed in the contract between the user and the cloud provider. When the VM does not conform to the content level, the user can request the cloud to increase the capability. In our scheme it must be assumed that user terminals also have different processing speeds. There are two options for this issue depending on the terminal performance. If the terminal has relatively lower capability, the VM on the cloud should process most of the load in order to reduce the load of one terminal. Therefore, the transcoding and rendering should be processed by the VM. In contrast, if the terminal has higher capability, the VM does not decode the content but transmits data directly to the terminal. Thin client architecture has started to adopt similar technologies [15], [16]. In this case, the playback experience can be robust against packet loss or delay using forward error correction (FEC) [17]. Here, the terminal can also estimate whether the content can be processed using H.264 information.

5. Evaluation

We assumed the traffic model below and evaluated the number of connections with regard to copying and playing back content. The number does not include connections for sending only control messages, i.e., "publish," "subscribe," and "refer."

The number of connections indicates not only the network traffic volume but also the server load, and the number of copy and playback processes, respectively, is almost proportional. It was reported that the access frequency distribution of video content within each continuous 24-hour period obeyed the Zipf distribution and that the skew parameter θ obeyed the normal distribution with an average of 0.199 and a standard deviation of 0.07, in a VoD service on the Internet [18]. Hence, we assumed the Zipf distribution with $\theta = 0.199$ for $q(m)$ and set $q(m) = c/m^{1-\theta}$, where c is the normalization constant to make $\sum_{m=1}^{M} q(m) = 1$ [12]. We made the request for content generation proportional to the frequency distribution and the number of users (100,000). Each user request followed a Poisson distribution with an average possibility of 0.1. We defined the playback ratio as the rate at which the user plays back the content after obtaining it. The number of connections for copying and playback is shown in **Fig. 3**.

The horizontal axis is the "refer" threshold described in the previous section. When the threshold is set to 0, all "refer" requests initiate the copying of content from the content holder's VM to the requester's VM. The playback ratio was given as a constant of 0.4. As the threshold increases, the number of copying connections decreases. Because we assumed that the content distribution had long-tail characteristics, the number of copying connections was reduced to less than that in inverse proportion to the threshold, even when the threshold was 2. The number of connections for playback was not reduced because connections are required to access the content in cloud architecture.

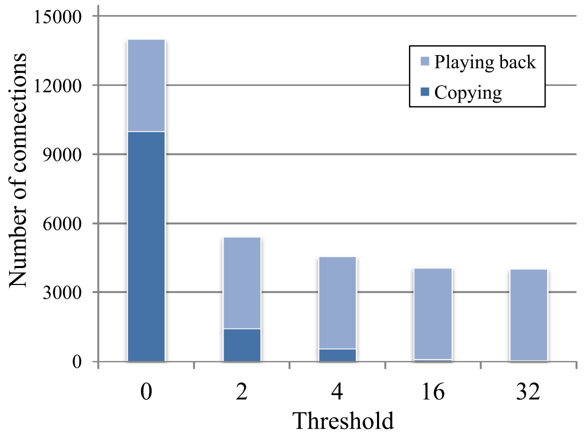

Fig. 3 Number of connections for copying and playing back versus threshold.

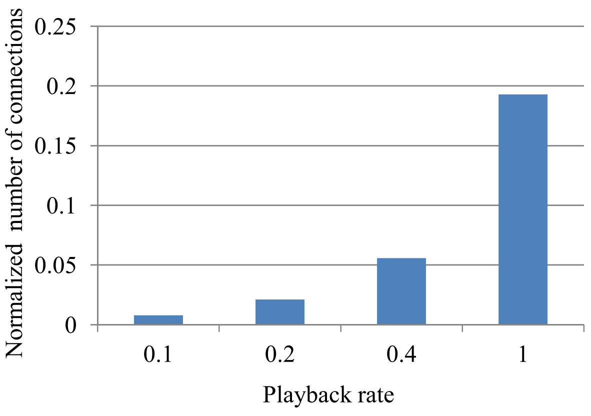

Fig. 4 Number of connections for copying content playback rate.

Figure 4 shows the number of connections for copying content versus the playback ratio, which was set to 0.1, 0.2, 0.4, and 1.0. In the figure, the number of connections is normalized by that with a playback ratio of 1 and threshold of 0. The threshold was set to 4. As the playback ratio increases, the number of connections also increases. When the playback ratio is 1.0, all the distributed content is played back. This will happen for most content for which the number of user requests exceeds the threshold that is copied. For a relatively smaller number of niche content, the number of requests does not exceed the threshold. Such niche content suppresses copying, and the number of copying connections decreases. If the playback ratio is smaller, this scheme reduces the number of connections drastically.

Content distribution using cloud architecture is underway in the market. In the conventional architecture, users can store their downloaded content in a private storage space and access it remotely from a ubiquitous environment with various user terminals via the Internet [19]. Our proposed scheme can reduce the load of downloading content compared to the conventional scheme. We compared the performance of the conventional scheme and the proposed scheme; the results are shown in **Figs. 5** and **6**. These Figures show the number of connections used for copying content versus the number of contents as a parameter of threshold, which varied from 2 to 16. The number of connections was normalized by that of the conventional cloud architecture in order to clarify the difference in performance between the proposed and conventional architectures. The number of contents

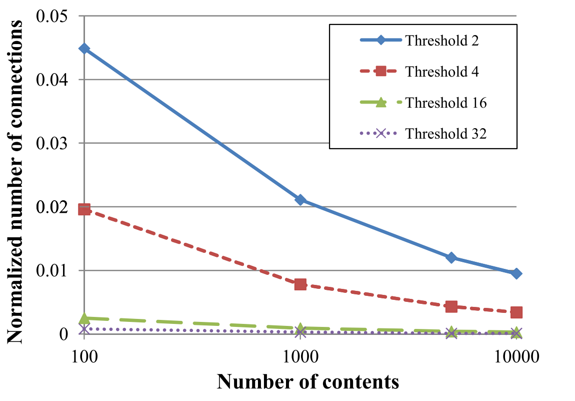

Fig. 5 Number of connections for copying content versus number of contents (playback ratio was set 10%).

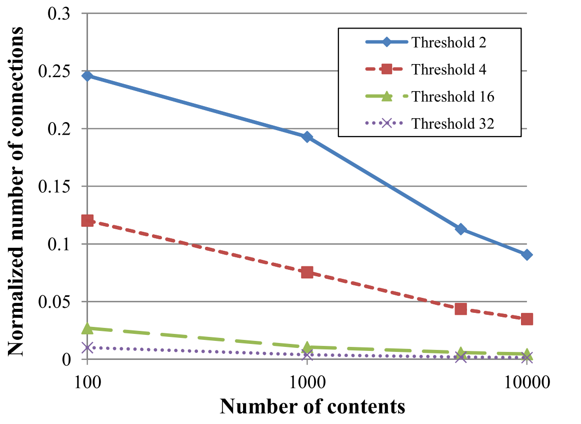

Fig. 6 Number of connections for copying content versus number of contents (playback ratio was set to 50%).

was varied from 100 to 10,000. The playback ratio was respectively set to 10% and 50% for Figs. 5 and 6. Our proposed architecture significantly reduced the number of copied contents in the cloud. In Fig. 5, when the threshold was set to 2, the number of copies was less than 5% of that of the conventional architecture. It is clear in Fig. 6 that even when the playback ratio was higher, the number of copies became less than 25% of the conventional architecture. As the number of contents increases, the number of connections decreases, even with the same threshold. The reason for this is that the number of niche contents increases as the total content increases. As a result, the niche contents are not requested for playback much, and the number of connections, i.e., traffic load, is suppressed.

In Fig. 6, the number of copies has a knee at the point where the number of contents was 1,000 with a threshold of 2. When the threshold was set to a relatively smaller value, copying was suppressed for only the long tail content. Therefore, most of the content is copied when the total number of contents is smaller. When the number of contents becomes larger, more content is suppressed from being copied according to this scheme.

6. Conclusion

As the distribution of rich content continues to increase, IT service providers such as ISPs, ASPs, and CSPs will need to increase the numbers of routers, switches, and servers they have. Adding facilities will lead to an increase not only in their investment but also in power consumption, so traffic should be reduced to avoid this. We presented a traffic management scheme to solve the inefficiency concerning the difference in number between content distribution and playback. Popular and niche contents are distributed in a network. Most content is categorized as niche content in the long-tail of a distribution. It is important to establish a traffic management scheme that can adapt to the differing characteristics of niche and popular contents. The proposed scheme controls the traffic efficiently with regard to content popularity.

Our proposed scheme makes content distribution more efficient by utilizing the difference in frequency between downloading and playing back. However, it is not suitable for real-time transmission such as broadcasting because the contents are not stored, and they are watched almost simultaneously. This means that there is no difference in frequency between downloading and playing back. For broadcasting it is efficient to share network resources among users, for example, multicast technologies. In contrast to that, our proposed scheme is based on sharing cloud resources.

In order to share content, it is necessary to coordinate the rights between users. In our proposed scheme, the coordination is done by the content rights server, which is preferably a third, neutral organization separate from clouds and users. It will also need to manage the security of content. It is necessary to manage security so that the content cannot be opened more widely than the range registered in the content rights server. The rights server needs to verify the user before permitting access to the content. For example, PKI based technologies can be used for authentication and certification [20].

The power consumption of routers is relatively constant since they do not depend much on the offered load. Therefore, to reduce power consumption, it is more effective to reduce the average traffic of all network facilities. Our proposed scheme reduces the network load by controlling traffic depending on the number of requests for content distribution and playback. This also reduces the server load of contents distribution. If the load is concentrated on relatively fewer servers, idle servers can switch to sleep mode to reduce power consumption. Our scheme is designed for cloud architecture, where content is stored in the servers of a cloud and managed by the VM of each user. For content distribution, it is not always necessary to copy content from a content holder's VM to a requester's VM. Content can simply be shared among VMs. The content holder's VM can play back content instead of the requester's VM. This concentrates the load on a relatively smaller number of VMs.

Therefore, our scheme reduces the network traffic, server load, and power consumption of IT systems. Our proposed scheme will thus contribute to achieving a low carbon society and aid in the fight against global warming.

References

[1] Ministry of Economy Trade and Industry: Green IT Initiative in Japan, available from ⟨http://www.meti.go.jp/english/policy/GreenITInitiativeInJapan.pdf⟩ (Oct. 2008).

[2] Cisco: Cisco Visual Networking Index: Forecast and Methodology, 2008–2013, available from ⟨http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf⟩ (accessed 2009-06-09).

[3] Amazon, available from ⟨http://aws.amazon.com/ec2, http://aws.amazon.com/s3⟩.

[4] Google apps, available from ⟨http://www.google.com/apps/intl/en/business/index.htm⟩.
[5] Microsoft azure, available from ⟨http://www.microsoft.com/windowsazure/⟩.
[6] Elberse, A.: Should you invest in the long tail?, *Harvard Business Review*, pp.88–96 (July-August 2008).
[7] Anderson, C.: *The long tail: Why the future of business is selling less of more*, hyperion books (2006).
[8] Fan, X., Weber, W.-D. and Barroso, L.A.: Power Provisioning for a Warehouse-sized Computer, *Proc. ACM International Symposium on Computer Architecture ISCA 2007* (June 2007).
[9] Chen, G., He, W., Liu, J., Nathm, S., Rigas, L., Xiao, L. and Zhao, F.: Energy-Aware Server Provisioning and Load Dispatching for Connection-Intensive Internet Services, *Proc. 5th USENIX Symposium on Networked Systems Design and Implementation NSDI 2008* (Apr. 2008).
[10] Chabarek, J., Sommers, J., Barford, P., Estan, C., Tsiang, D. and Wright, S.: Power Awareness in Network Design and Routing, *Proc. 27th IEEE Conference on Computer Communications INFOCOM 2008*, pp.457–465 (Apr. 2008).
[11] Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Kats, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I. and Zaharia, M.: A View of Cloud Computing, *Comm. ACM*, Vol.53, No.4 (Apr. 2010).
[12] Kamiyama, N., Mori, T., Kawahara, R., Harada, S. and Hasegawa, H.: ISP-Operated CDN, *IEEE Global Internet Symposium 09, INFOCOM Workshops* (Apr. 2009).
[13] ITU-T Recommendation H.264: Advanced video coding for generic audiovisual services (June 2006).
[14] Amazon: Amazon EC2 Instance Types, available from ⟨http://aws.amazon.com/ec2/pricing/⟩.
[15] Citrix: Deliver rich local-like multimedia content from XenApp and XenDesktop, available from ⟨http://hdx.citrix.com/hdx-mediastream⟩.
[16] Microsoft: Remote Desktop Client Experience, available from ⟨http://technet.microsoft.com/en-us/library/dd560636(WS.10).aspx⟩ (June 2009).
[17] Wah, B.W., Su, X. and Lin, D.: A survey of Error-Concealment Schemes for Real-time Audio and Video Transmission over the Internet, *Proc. IEEE International Symposium on Multimedia Software Engineering*, pp.17–24 (Dec. 2000).
[18] Yu, H., Zheng, D., Zhao, B. and Zheng, W., Understanding User Behavior in Large-Scale Video-on-Demand Systems, *ACM EuroSys 06*, (2006).
[19] Amazon: Amazon cloud drive, available from ⟨https://www.amazon.com/clouddrive/learnmore⟩.
[20] ITU-T Recommendation X.509: Information Technology – Open Systems Interconnection – The Directory: Authentication Framework (Aug. 2005).

Haruhisa Hasegawa is a Senior Research Engineer, Supervisor, Group Leader, Traffic Solution Group, Communication Traffic & Service Quality Project, NTT Service Integration Laboratories. He received his B.E., M.E., and Dr. of Information and Computer Science from Waseda University, Tokyo, in 1988, 1990, and 2001, respectively. He joined NTT in 1990 and has been researching traffic engineering and management in telecommunications networks. From 2000 to 2003, he was engaged in network management at the network operation center of NTT East. He is a member of IEEE Communications Society and IEICE Japan.

Noriaki Kamiyama received his M.E. and Ph.D. degrees in communications engineering from Osaka University in 1994 and 1996, respectively. From 1996 to 1997, he was with the University of Southern California as a visiting researcher. He joined NTT Multimedia Networks Laboratories in 1997. Since then, he has been engaged in research concerning pricing methods for IP networks, content-distribution systems, optical networks, network topology design, and IP traffic measurement. He is currently in NTT Service Integration Laboratories. He is a member of IEEE, IEICE and the Operations Research Society of Japan.

Hideaki Yoshino received his B.Sc., M.Sc. and D.Sc. degrees in information science from the Tokyo Institute of Technology, Tokyo, Japan, in 1983, 1985 and 2010, respectively. He joined NTT Laboratories in 1985 and has been engaged in communication traffic and service quality research. As a visiting scholar, he stayed at the University of Stuttgart, Germany, during 1990–1991. He is currently an executive researcher, supervisor and serves as a project manager of NTT Service Integration Laboratories, where he is conducting research and management on traffic and quality for future communications networks and services. Dr. Yoshino is a member of IEEE, IEICE Japan, and the Operations Research Society of Japan.