A Distributed Cache Update Deployment Strategy in CDN

To cite this article: Xinhua E and Binjie Zhu 2018 J. Phys.: Conf. Ser. 1004 012031

View the article online for updates and enhancements.

You may also like
- Nodes detection in CDN with enhanced naive bayes tree
  Ruijie Hou, Changyong Guo, Zhaoxin Zhang et al.
- A Three-level Location-inventory Problem With Perishable Product
  Zhengqiang Wu, Weili Li and Lijing Song
- Distributed Load Balancing for the Adaptive Video Streaming Using CDN with Ring System of Server Consolidation by Circulant Topology
  H A Nekrasov and A Yu Romanov
A Distributed Cache Update Deployment Strategy in CDN

Xinhua E1,2 and Binjie Zhu2

1Faculty of Information Technology, Beijing University of Technology, Beijing, 100124, China
2China Mobile Group Beijing Company Limited, Beijing, 100007, China

*Elson1900@163.com

Abstract. The CDN management system distributes content objects to the edge of the internet to achieve the user's near access. Cache strategy is an important problem in network content distribution. A cache strategy was designed in which the content effective diffusion in the cache group, so more content was storage in the cache, and it improved the group hit rate.

1. Introduction
Nowadays services such as Video-on-Demand and live TV programming available over Internet. Users want to have access to high quality video at any times. Content Delivery Network (CDN) is a solution that has been used to delivery traditional Web page content and Video content [1-7]. The CDN management system distributes content objects to the edge of the internet to achieve the user's near access. Cache strategy is an important problem in network content distribution. A cache strategy was designed in which the content effective diffusion in the cache group, so more content was storage in the cache, and it improved the group hit rate.

2. Cache update deployment strategy
The cache server group can only store partial content objects. However, if contents requested by Client are not available, a distributed cache group can pull those content objects on a globe scale. The cache server that stores content objects at the first time is called as an intra-group primary source server. The system will spread this content object to other servers in the group via certain methods according to dynamic changes requested by clients. Then, they also become intra-group servers of this content object after storage. The rules and methods of how servers in the distributed cache group to pull content objects from intra-group primary source servers or origin servers determine their deployment in the cache group, which is called as a cache update deployment strategy.

Based on previous analysis, most of current researches on cache strategies aim at cache replacement algorithms or cache deployment methods with certain architectures. This paper proposed a cache deployment strategy that is suitable to cache server groups from the perspective of improving degree of collaboration. This method allows various cache server nodes to make cache update decisions instead of adopting centralized control modes.

The cache server group is composed by some close servers. If contents are not available in the group, delays will happen when pulling them from outside the server group. Therefore, requests are better to be completed within the group, which shall improve degree of collaboration of cache servers and optimize deployment globally.
EPP-CPS takes expiration time (ET), potential acceleration effect (PAE) and pull gate (PG) as parameters of cache update deployment so as to be more suitable to the optimized goal in the cache collaboration group.

This article firstly introduced concepts of expiration time, potential acceleration effect and pull gate, then explains cache update methods. The expiration time is divided into two concepts including data object’s expiration time and cache expiration time.

\[
\text{Det}(D, C) = T_R - T_l
\]

(1)

A data object’s expiration time is defined as the difference between \(T_R\) (time that a data object is being removed from the cache) and \(T_l\) (time of the last hit). The cache expiration time refers to average numbers of expiration time of data objects that are being removed from the cache over a period of time. The data objects that are being removed from cache \(C\) within \(T\) is graded as a set of \(S\), then the cache expiration time \(Cet\) is shown as the formula below. \(D_{num}\) is the number of data objects in the set \(S\).

\[
Cet(T, C) = \frac{\sum_{\text{data}} \text{Det}(D, C)}{D_{num}}
\]

(2)

The cache expiration time is one of parameters to measure cache competition. If the cache expiration time is large, then the degree of competition is relatively small; vise versa. If one content is being stored in a cache server with relatively small competition, then it will have a longer life cycle in the cache group.

The CDN environment is characterized by particularity. Thus, some special factors in CDN needs to be considered. First of all, the acceleration effect (AE) is defined as a kind of evaluation on increase of access speed (decrease of delays) after a content object is being stored in the cache. If a content object is being cached from \(S\) (content source node) to \(D\) (destination node). The delay from \(S\) to \(D\) is defined as \(\text{Delay}_{SD}\). If the access time of content visitors that are very close to \(D\) is \(l_d\), then the acceleration effect of delivering contents to \(D\) is shown as formula below.

\[
\text{AE}_{SD} = \text{Delay}_{SD} \times l_d
\]

(3)

\[
\text{AE}_{SD} = \text{Delay}_{SD} \times l_d
\]

(4)

The potential acceleration effect (PAE) between two points is defined on the basis of acceleration effects. There are two points including \(C\) and \(A\) in the network. \(C\) stores contents requested by clients, while \(A\) is an access point of Client nodes that are close to server \(A\). If Client requests content object \(C_r\) from server \(A\), but it is not in that server. In this case, server \(A\) can not satisfy Client’s request, but server \(C\) can do that. The delay between \(C\) and \(D\) is taken as \(\text{Delay}_{CA}\). If the forwarding time of content request from \(A\) to \(C\) is \(\text{Rd}_{ac}\), then the potential acceleration effect is defined as formula below.

\[
\text{PAE}_{ca} = \text{Delay}_{ca} \times \text{Rd}_{ac}
\]

(5)

The potential acceleration effect is also used to evaluate content cache values. However, different from the acceleration effect, the potential acceleration effect is used to evaluate decrease of acceleration effect due to the fact that the content object is not available in the local cache.

Every server will record content objects that are being accessed lately. Those records are a kind of structural organization in FIFO queue. When a request is delivered, please check FIFO queue firstly. If there are same contents in the recent records, then it is regard as up to the pull gate. The cache update will only be started once up to the pull gate.

The server does not need to pull missed contents to the local space in the collaborative cache server group, because it will increase the pull cost. Only pulling missed contents when necessary by determining certain rules via intragroup methods has two advantages: first of all, decrease the pull cost; secondly, utilize “group space” in the distributed cache group so as to store more contents in the limited space to improve the group hit rate effectively.

Cache update deployment strategy is shown in Figure 1.
Figure 1. Cache update deployment strategy

**Step 1:** Accept requests outside cache servers; if other servers do not have the requested content, then route the request to the located server via local routing system; the located server will accept the request accepted from other servers;

**Step 2:** Judge whether it is up to the pull gate; if yes, then execute the next step, or else “end”;

**Step 3:** Find out the set $C$ with ET numbers larger than the local cache server;

**Step 4:** Find out server $S$ with the biggest PAE value from set $C$;

**Step 5:** Pull contents from server $S$, end;

The threshold judgement module, PAE module, ET module, update deployment decision module, local content update module and cache replacement algorithm module are involved in cache update deployment.

The PAE module calculates the potential acceleration effect by accessing historical data. There are $n$ servers in the network, including $S_1, \ldots, S_n$. The server $S_i$ records the times $Rdnum$ aiming at the request to $C_i$ forwarded from other servers $S_j$. The potential acceleration effect (PAE) from $C_i$ to various nodes is obtained based on parameters of delay between the request time and nodes. The pseudo-code of PAE module function is shown in figure 2.
The ET module is used to calculate ET numbers and exchange for ET information with other nodes. The ET module calculates ET numbers of local cache servers via formula. It not only needs to calculate the local ET numbers, but stores ET numbers of other servers. The ET numbers shall also be forwarded to the target server at the time when a server forwards content requests to another one. In this way, ET information of other server will also be obtained.

The pull gate module records information of content objects that are being accessed lately. When a new request is delivered to the server, the content information will be recorded at the tail of the queue. Meanwhile, the record at the head of the queue will be deleted. This access record queue is used to judge whether it is up to the update deployment threshold: when a content request is delivered, it shall be judged whether this is being stored in the recent access records. If yes, then start the cache update decision process.

The update deployment decision module selects alternative nodes of content update deployment through parameters in PAE and ET modules. Then the content object is being updated and deployed in nodes. The local content update module will execute its update process after accepting relevant information.

3. Analysis and verification
In order to verify EPP-CPS, a simulation framework of an edge cache server group is established in Opnet network software. There are 9 servers in every cache group. Every server has their own user request nodes that will access the cache group according to Poisson's distribution, which includes 1,000 document types. The group hit rate between EA method and EPP-CPS is further compared. T/S refers to the ratio of storage space in every node to single file space. Five testing scenes are established with their T/S values of 10, 30, 50, 70 and 100. The experimental results are shown in the figure 3.

It can be seen from figure 3 that EPP-CPS update strategy has higher group hit rate than EA strategy. The average GHR of EPP-CPS is 43.8, while that of EA is 30.4, which is improved by 31%. EPP-CPS optimizes the update strategy based on EA so as to spread contents in the cache group effectively. Therefore, it can store more contents in a limited storage space so as to improve the group hit rate.

Figure 2. Pseudo-code of PAE module function

```c
PAE_arrary(Content) // PAE
{
    For every server i
    {
        Delay=SelectDelay(server i);
        Rdnum=SelectRdnum(content);
        PAE_arrary[i]=Delay*Rdnum;
    }
    Return PAE_arrary;
}
```
Figure 3. Comparison of EPP-CPS and EA

References

[1] Sanaa Sharafeddine, Karim Jahed, Omar Farhat, et al. Failure recovery in wireless content distribution networks with device-to-device cooperation[J]. Computer Networks, 2017, 128(9): 108-122.

[2] Rami Halloush, Hang Liu, Lijun Dong, et al. Hop-by-hop Content Distribution with Network Coding in Multihop Wireless Networks[J]. Digital Communications and Networks, 2017, 3(1): 47-54.

[3] Xiaoying Zheng, Ye Xia. Optimizing network objectives in collaborative content distribution [J]. Computer Networks, 2015, 91(14): 244-261.

[4] Uttam Mandal, Pulak Chowdhury, et al. Energy-efficient networking for content distribution over telecom network infrastructure[J]. Optical Switching and Networking, 2013, 10(4): 393-405.

[5] Lazaros Gkatzikis, Vasilis Sourlas, Carlo Fischione, et al. Low complexity content replication through clustering in Content-Delivery Networks[J]. Computer Networks, 2017, 121(5): 137-151.

[6] Bernd Klasen. Efficient content distribution in social-aware hybrid networks [J]. Journal of Computational Science, Pages 209-218, 2013, 4(4): 209-218.

[7] Tom Bostoen, Sape Mullender, Yolande Berbers. LofoSwitch: An online policy for concerted server and disk power control in content distribution networks[J]. Ad Hoc Networks, 2015, 25(B).