Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2017) 25: 3397 – 3408

doi:10.3906/elk-1604-172

A stable marriage-based request routing framework for interconnection CDNs

Bo HUANG[1], **Lijian ZOU**[2], **Xiaowen TONG**[3,*], **Xiaoqun YUAN**[3]

[1]College of Telecommunications and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing, P.R. China

[2]NetEase Inc., Hangzhou, P.R. China

[3]School of Information Management, Wuhan University, Wuhan, P.R. China

Received: 13.04.2016 • **Accepted/Published Online:** 16.01.2017 • **Final Version:** 30.07.2017

Abstract: Content delivery network (CDN) interconnection is a promising solution to addressing the limited service scale of CDNs. It scales the CDN's service footprint through the cooperation of CDNs without significantly changing the existing network architecture. However, in a CDN interconnection system, CDNs are independent of each other and each pursues its own goals, which means that cooperation is hard to establish and easy to break. In our paper, we propose a stable marriage-based routing framework to establish a strong cooperation service for CDNs and to select the 'optimal' server for each request among the cooperative CDNs. To this end, we first investigate the relationship between the service cost and the service profit of CDN interconnection, and we design a price determination strategy to ensure the economic interests of each cooperative CDN, which is helpful in establishing a stable CDN cooperation service. Then we propose a dynamic request routing strategy to select the 'optimal' server for each end user request by applying the 'stable match' theory. This strategy is helpful in scaling the CDN service footprints with guaranteed service quality and in gaining more profit with lower service costs. The simulation results show that our frameworks can scale the CDN's service footprints with guaranteed service quality and gain more services without increasing service costs. Furthermore, our frameworks are win-win request routing frameworks because they help the upstream CDN of the CDN interconnection to increase service profit without increasing its cost. Moreover, they help the downstream CDNs of the CDN interconnection to gain extra revenue by using their idle resources.

Key words: Content delivery networks, interconnection, stable marriage theory, request routing

1. Introduction

Internet-related problems, such as network congestion, packet loss, jitter, and delay, have grown increasingly urgent since the emergence of content delivery networks (CDNs) [1,2]. By deploying servers in the network edge and using the 'optimal' server-to-service end users, a CDN reduces service response time by simultaneously ameliorating packet loss and network congestion. Alongside the booming popularity of high-resolution videos, social media, and mobile applications, Internet traffic has rapidly expanded, which forces some network operators (e.g., AT&T and China Telecom) and content providers (e.g., Netflix [3]) to deploy CDNs to accommodate traffic growth. Unfortunately, although CDNs can improve content service quality by reducing response time to some extent, they do not keep up with the pace of growing demand. A reason is that many CDNs, especially regional CDNs, have limited service coverage and do not provide guaranteed service quality to users outside their coverage. In addition, competition among CDNs forces them to reduce capital and operating costs, which

*Correspondence: tongxiaowen0418@sina.com

leads to a further degradation of their service quality [4]. To address these problems, several solutions, such as content delivery network interconnection (CDNI) [5] and virtual content delivery network (VCDN) [6–8], have been proposed to expand the CDN footprint. A VCDN expands CDN service coverage by virtualizing CDN services on top of the novel layer, whereas CDNI scales CDN footprints through cooperation between CDNs. Since CDNI allows CDNs to expand their service coverage without requiring excessive alterations to the existing network architecture, it has attracted worldwide attention from academia and industry.

Similar to a CDN [9], a CDNI system applies request-routing techniques to select optimal servers to provide users with guaranteed high-quality service [10,11]. Thus, the performance of CDNI request-routing strategies affects CDNI service quality significantly. However, few studies have focused on this issue. Adhikari et al. proposed a measurement-based adaptive CDN selection strategy and a multiple-CDN-based video delivery strategy to improve service bandwidth efficiency [12]. Shin et al. proposed a CDNI request-routing solution by extending the BGP protocol [13]. Though these strategies improve CDNI service performance, they are not suited to existing CDNI systems because they only take service performance into account. In fact, each CDN of a CDNI system is independent of the others and is motivated by the interests of its owner. Thus, it is necessary to take economic factors, such as service cost and profit, into account when designing a CDNI request routing strategy.

In this paper, we design a stable request routing framework for CDNs to scale their service footprint in a CDNI environment. To this end, we investigate the relationship between the service cost and the service profit of a typical CDNI and formulate the service profit as an optimization problem. Then we propose an optimal request-routing strategy by applying the 'stable match' method. This takes the service performance and the service cost of the CDNI into account and creates a win-win situation for all CDNs in the CDNI. The remainder of this paper is organized as follows. Section 2 discusses the framework of the associated solutions. Section 3 investigates the CDNI service profit construction and proposes a suitable request routing strategy for CDNIs with a corresponding dynamic algorithm. Section 4 evaluates the performance of the proposed algorithm via simulation experiments, and Section 5 provides a summary and a conclusion.

2. CDNI request-routing framework, interface, and process

2.1. CDNI request-routing framework

Our proposed CDNI routing framework is shown in Figure 1. The framework includes a content provider, end users, and multiple interconnected CDNs. Each CDN has a mapping system through which it selects the optimal server for each user request. All mapping systems are connected.

For example, a user request is directed to CDN 1 and then sent to CDN 1's mapping center. Based on the Internet conditions and the CDN server operation, the mapping center identifies the edge server of CDN 3 as the optimal provider by matching the request and service with the interconnected CDN request-routing strategy. After the request is redirected to CDN 3's mapping center, CDN 3 matches it in detail and redirects it to the correct edge server to provide service.

Based on our CDNI request-routing framework, the CDNI mapping system is designed as shown in Figure 2, consisting of an edge server monitoring system, a data storage module, a request service-matching module, and a CDNI interface. The monitoring system monitors the Internet conditions, e.g., network bandwidth utilization, and the CDN operation stations, such as the condition of the load on the server. In addition, it stores history data in its data center, locates end users, and collects Internet information such as the response delay. The data storage module stores data from the monitoring system, stages data from the matching center, and caches

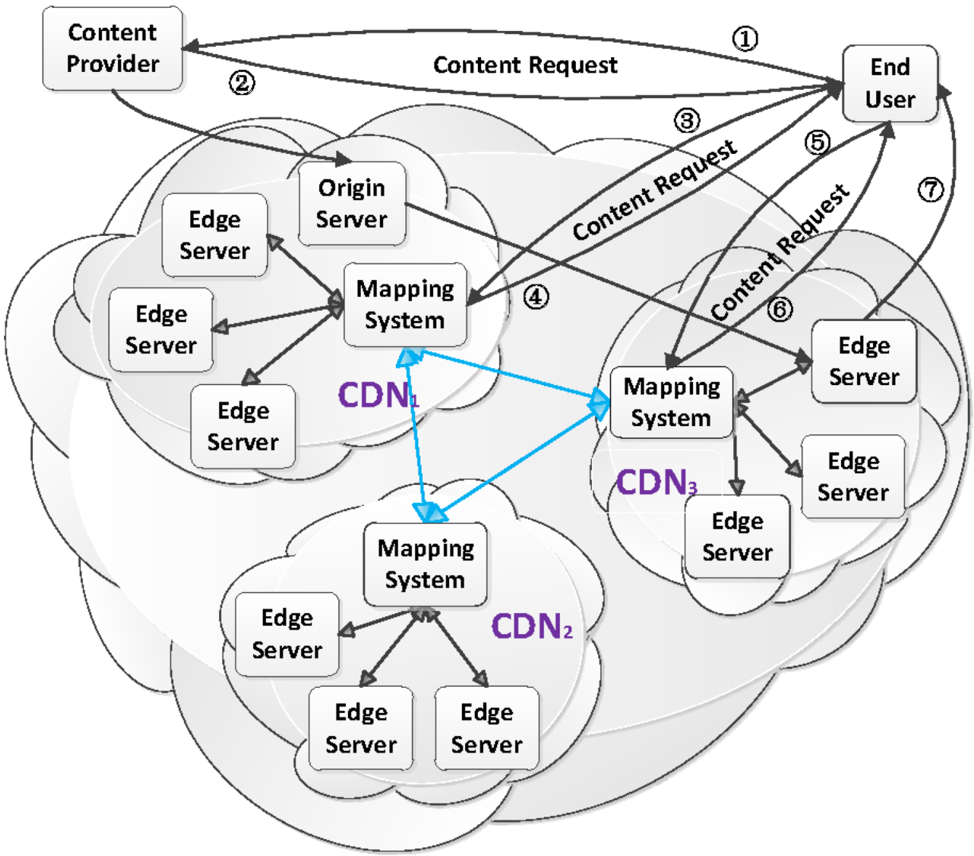

Figure 1. CDNI request routing framework.

data from the CDNI interface. The request service matching module redirects each request to the optimal edge server by applying a matching algorithm. It also controls the monitoring system and interface module to collect necessary Internet information for the matching algorithm. The Internet interface module has two main functions. As a client interface, it requests relevant information from its upstream CDN (uCDN), sends feedback to the request-matching center, and updates the corresponding information. As a server interface, it delegates client requests to its request service-matching module and sends feedback to the Internet interface.

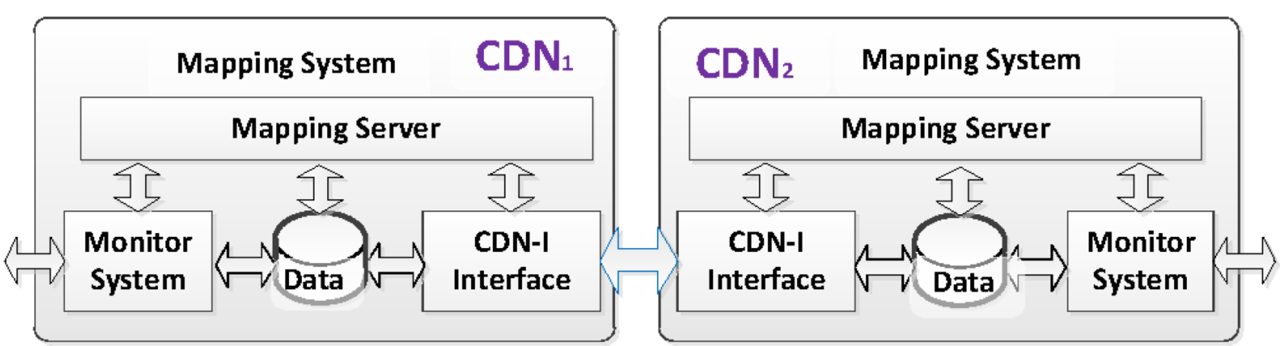

Figure 2. Mapping system of interconnection CDNs.

2.2. Interface with CDNI process

The CDNI's mapping system connects different CDNs without requiring excessive alterations to existing CDN architectures. In our framework, CDNs cooperate through the Internet interface in two ways: a client-side interface, running in the uCDN, and a server-side interface, running in the downstream CDN (dCDN or CDN/d). The server-side interface module includes a Session Manage and Mapping Module, whereas the client-side module includes a Session Manage Module and Request Module, as shown in Figure 3.

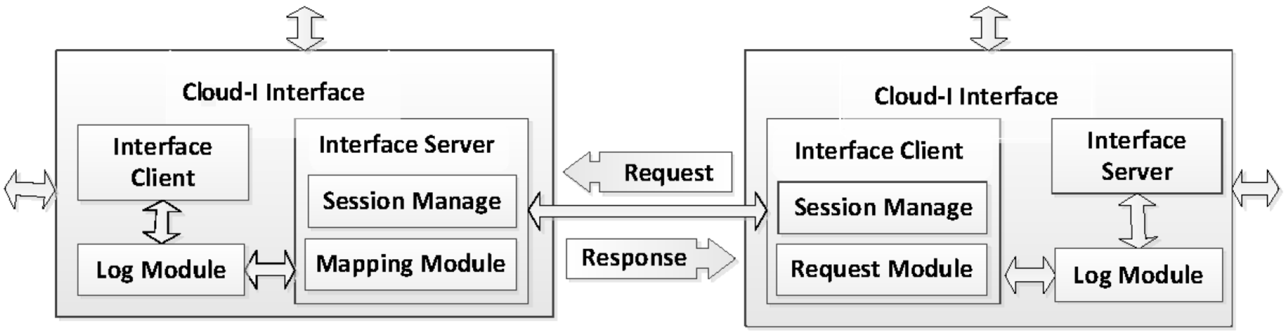

Figure 3. Module of interconnection interface.

The Session Manage Module of the client-side interface maintains the connection and sends query requests to other CDNI interfaces. The Session Manage Module of the server-side interface transfers requests to the Mapping Module and sends the resulting feedback to the other session interfaces. The Mapping Module then matches this request by request-routing strategies based on the data stored in the Log Module, which includes basic information, key information, and other information. Basic information describes the request, which is the input for CDNs to make optimal matches. Key information includes the reference time delay served by the CDN (R-RTT), price, and service level, which are helpful in determining the optimal matching request-routing.

3. Request-routing decisions

3.1. Problem formulation

In our CDNI request-routing framework, $CDN = \{CDN_1, ..., CDN_k\}$, $C = \{c_{ij} | i = 1, ..., M; j = 1, ..., N\}$ is the content set and $\{d_{ij}\}$ is the file size of $\{c_{ij}\}$. M is the business type and N is the number of content items. Figure 4a shows the relationship between the CDN service prices of two service types: the CDN directive service (Class A) and the cooperative CDN service (Class B). Let P_A and P_B be the service prices of Class A and Class B, respectively, and let Ct denote the cost of the service provided by CDN$_i$. To simplify our model, we assume the relationship between service cost and user request as shown in Figure 4b, where α is the CDN ratio and P is the service price of the adjacent CDN.

Let $\{q_{ij}\}$ denote the request for the content c_{ij}, where Class A is $\{q_{ij}^A\}$ and Class B is $\{q_{ij}^B\}$. Then the profit RT for the CDN is:

$$RT = \sum_{i=1}^{M} \sum_{j=1}^{N} (p_A q_{ij}^A - (1-r) p_B q_{ij}^B) d_{ij} - \sum_{i=1}^{M} \sum_{j=1}^{N} (Ct_i(1-a) + (1-r) p_i a) q_{ij} d_{ij}. \qquad (1)$$

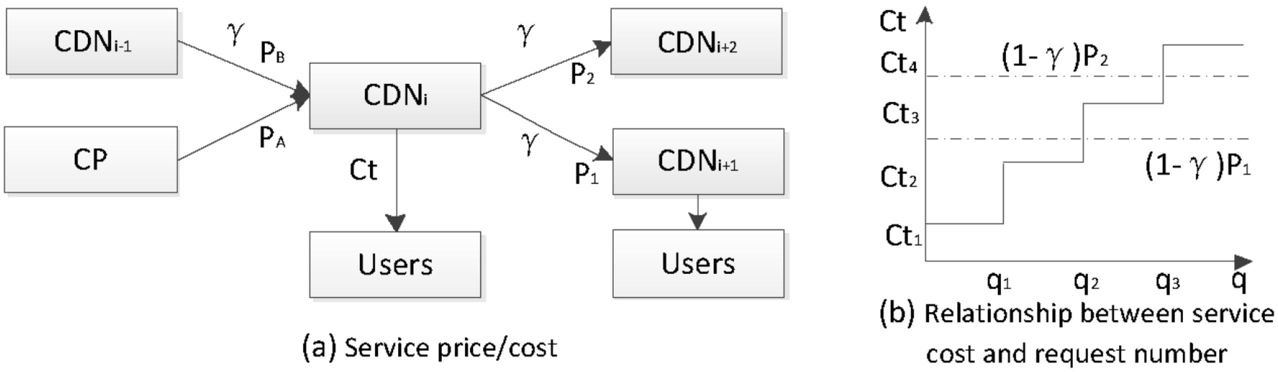

Figure 4. Relationship of price.

It follows that the routing selection strategy for the CDN can be formulated as follows:

$$\begin{aligned}
&\max\left\{\sum_{i=1}^{M}\sum_{j=1}^{N}(p_A q_{ij}^A-(1-r)p_B q_{ij}^B)d_{ij}-\sum_{i=1}^{M}\sum_{j=1}^{N}(Ct_i(1-a)+(1-r)p_i a)q_{ij}d_{ij}\right\}\\
&s.t. Qos>Qos_{\min} \qquad . \qquad (2)\\
&q_{ij}^A+q_{ij}^B=q_{ij}
\end{aligned}$$

From Eq. (2), we know that the profits and costs are independent. Therefore, the routing strategy can be designed in two separate parts: service cost control and service revenue improvement. On the other hand, our previous study [14] showed that a CDN provides service to all its user requests by itself when $Ct_i<(1-r)p_i$, whereas it will delegate service to a dCDN when $Ct_i>(1-r)p_i$. It also showed that both Ct_i and p_i fluctuate with CDN operation, making the cost control strategy fluctuate dynamically, as shown in Figure 4b. In the initial stage, CDN_i receives few service requests $(q<q_1)$ with service costs of $Ct_1\,(Ct_1<(1-r)p_1)$. When $q_1<q<q_2$, the service cost $Ct_2\,(Ct_2<(1-r)p_1)$ is still below the threshold at which CDN_i should delegate its service to dCDNs. Therefore, CDN_i still services its users by itself. As user requests increase, namely when $q>q_2$, the service cost $Ct_3\,(Ct_3>(1-r)p_1)$ goes beyond the threshold, which means that CDN_i cannot serve all the users; thus, it delegates new requests to dCDNs. Similarly, with the users' increase of one dCDN service, the service cost of this dCDN also rises, which means it will raise its service price p_2. Thus, the price of the dCDN is limited by $(1-r)p_1<Ct_3<(1-r)p_2$, as shown in Figure 5. Consequently, when $q_2<q<q_3$, CDN_i does not delegate new user requests to its dCDN. Instead, it should delegate new users to other dCDNs. If there are no other available dCDNs, CDN_i should serve these new users with service cost Ct_3. Finally, when $q>q_3$, CDN_i must provide service at price p_3.

Eq. (2) shows that the entire revenue includes the incomes of both Class A and Class B services. Generally, the income of Class A service is stable, whereas that of Class B is influenced by q_{ij}^B and $p_B(p_B<p_A)$. Additionally, Eq. (2) shows where the service cost is influenced by $p_B<p_A$; the more users the uCDN delegates to dCDNs, the more users the dCDNs can serve. Thus, it is a suitable strategy for Class B to maximize its business by providing a lower service price.

Figure 5 shows the service price curve of Class B service at different stages of the CDN function. From Figure 5, we can deduce that there is a minimum profit RP_0 between the service price and cost for CDN_i and that if $q<q_1$ and $q>q_2$, the service cost of the dCDNs increases when the number of requests increases, which causes the increase of the dCDN service price. Thus, when the service cost does not change, the dCDNs will serve these requests and service price p_B does not change. Otherwise, the dCDNs increase their service price p_B with their increasing service cost.

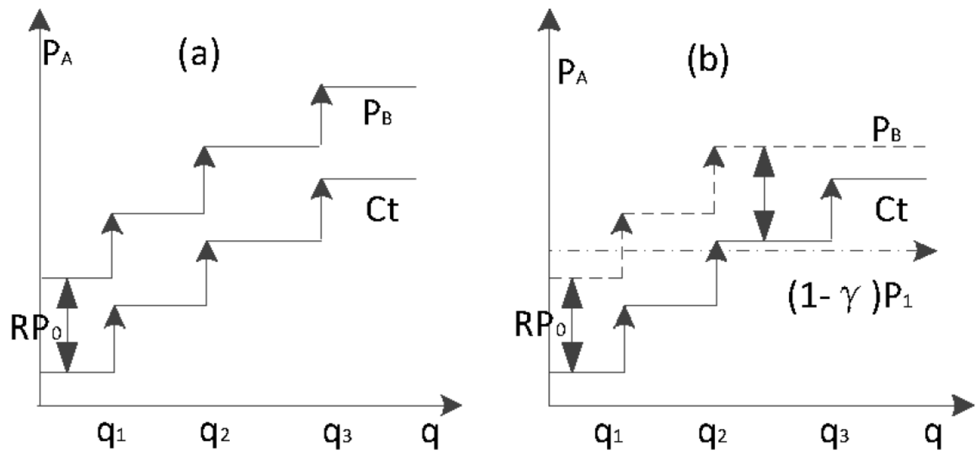

Figure 5. Service price of Class B.

As mentioned earlier, maximizing $\{RT\}$ means that this type of CDN request-routing strategy aims to pursue service profit maximization and disregards the service quality of the CDN. However, most users choose a minimum-delay service and reject low-quality service with less concern for cost. Obviously, CDN providers and end users evaluate CDN service quite differently. To bridge this gap, we propose a request-routing algorithm that selects the optimal server for each user by applying a stable match theory.

3.2. Request-routing strategy

In our paper, the CDNI request-routing framework includes a request query response, a two-layer request-routing strategy, and a single request-routing strategy.

3.2.1. Request-routing query

The request-routing query was designed for dCDNs to respond to uCDN queries. After receiving the request-routing query from a uCDN, the dCDN reports feedback to the uCDN with operation information such as time delay and user request service price. The uCDN then selects its dCDNs and delegates requests to them. The operation information can be obtained from the monitoring system, and the service price is determined based on operation conditions, service cost, and service profit requirements. Obviously, the key factor of the request-routing query is the dCDNs' service prices, which are determined by Algorithm 1.

3.3. Single request-routing decisions

In the request-routing query, if the feedback information of the dCDN has detailed service node information (such as the topology of available service nodes and their loads), then the uCDN may determine the request-routing strategy by integrating this node information into its routing strategy and redirecting the request to the optimal server, as shown in Algorithm 2.

Step 1: CDN_i collects the response delay of (parts of) the service cluster $\{S_j\}$ using its monitoring system, which calculates and constructs the service delay table ($Delay\_List\_L$) through which each S_j serves

Algorithm 1 Request query algorithm

Input:

The selected service node load Ld_i ;
Requirement bandwidth BW_i;
User request number Q_i;
Acceptation minimum profits RP_0;
Service price P_1 from downstream CDN

Output:

Service price P_B of downstream CDNs:

```
Ct_i = f(Ld_i, BW_i, Q_i);
if Ct_i < P_i then
    P_B = Ct_i + RP_0;
else
    P_B = P_1 + RP_0;
end if
RETURN P_B;
```

user U. Then this table is resorted based on the extension of the service delay. The preference number PS_j is determined according to the index of the $Delay\_List$.

Step 2: The CDN_i collects the service cost of each service cluster $\{S_j\}$ and constructs the cost list $Cost\_List\_L$. In our paper, we let the service cost $Ct_j = f\,(Load_j, BW_j, ...)$ be the function of the server load ($Load_j$) and bandwidth availability BW_j. CDN_j acquires the dCDN service price P from the query interface, which ensures $(1-r)p$, and constructs the price table. Like Step 1, these two tables are merged and resorted to obtain the corresponding preference PC_j of the service cluster S_j.

Step 3: Based on these two preference tables, we satisfy user preferences and service cluster preferences. Then a stable match algorithm is applied to determine the 'optimal' match between users and server clusters. In fact, for each request, the CDNI redirects it to the 'optimal' server, which means that we can apply M: 1 stable matching to simplify the complexity of our algorithm. Moreover, to avoid the problem that users' time delay is not guaranteed due to CDN priority, we introduce two preferences: a weighted method to the stable matching algorithm to design our routing matching algorithm, the details of which are described as follows.

First, the weighted-factor score of the CDN's preference PC_i and user preference PS_j is calculated based on $Score_i = \beta.PS_i + (1-\beta)PC_j$, where β is defined by the actual demand of the system. Then the minimum value of the CDN preference PC_k should be found. If the corresponding weighted score $Score_k$ is no more than the corresponding weighted value $Score_m$, which is the second smallest preference PC_m of the CDN, that is, if $Score_k \leq Score_m$, then $Score_k$ will be the optimal matching. If $Score_k > Score_m$, then the weight values of the CDN's third-smallest preference should be compared with $Score_m$ to determine whether there is an optimal matching or not. This process will continue until the optimal matching is found.

3.3.1. Two-layer request-routing decision

Each CDN of the CDNI has a request-routing strategy and uses it to independently select its 'optimal' server for requests, which means that the single-request routing decisions algorithm does not choose the 'optimal' server of the CDNI for each request. Therefore, we design the two-layer request-routing decision algorithm, shown in

Algorithm 2 Single request-routing algorithm

Input:

Load rate Ld_i of all selected service node in CDN $S_j, i = 1, N$;

Bandwidth BW_i; User request number Q_i;

Time delay list DL_i of all selected service node S_i:

Service price of downstream CDN P_1 ;

Time delay list $DD_j, j = 1, M$ downstream CDN offered to users

Output:

The optimal service node for user matching S:

for i in N;
  $Ct_i = f(Ld_i, BW_i, Q_i)$;
$\{C_k, k = 1, 2, ..., N + M\} = \{Ct_i\} \cup \{P_1\}$;
$\{D_k, k = 1, 2, ..., N + M\} = \{Dl_i\} \cup \{DD_j\}$;
for k in $N + M$;
  $PS_k = C_k \ Ranked \ in \ \{C_k\}$;
  $PC_k = D_k \ Ranked \ in \ \{D_k\}$;
  $Score_k = \beta * PS_k + (1 - \beta) PC_k$;
$\forall k \in N + M, PS_k \rightarrow PS_j$;
for j in $N + M$;
if $Scoreps_j < Scoreps_{j+1}$ **then**
  $S = PS_j$;
  break;
end if
RETURN S;

Algorithm 3. In this algorithm, the CDN first determines which CDN should respond to a new request, which is based on service costs and the price of the downstream CDN. Then the mapping system redirects this request to the selected CDN. Finally, the selected CDN selects the 'optimal' server for this request and redirects the request to the selected server by applying the single-layer routing decision algorithm.

4. Experiment and performance analysis

To evaluate the feasibility and availability of our request-routing strategy, we design a simulation experimental framework and develop a simulator based on our designed simulation framework with the C programming language to simulate the interconnection of CDNs, as shown in Figure 6. Our simulator consists of interconnecting CDNs and a Request Simulation Module. Each CDN has functions such as cost evaluation, price assessment, routing decision, and logging module. The Request Simulation Module generates random content requests and sends them to the CDNs. In our experiment, we simulate four months of interconnection of two CDNs (a uCDN and a dCDN) with PCs (Inter Core Quad Q8300 2.5 GHz and 2 GB RAM) with our developed simulator. The service content for each request is limited to a size between 10 M and 1000 M, and the bandwidth is limited to a speed of 0–5 M/s. In addition, we assume that there are 15 requests for the uCDN and 5 requests for the dCDNs. The CDN service cost had a linear relationship with system bandwidth, i.e. $CT = 0.01BW + CT_0$.

Algorithm 3 Two-layer request-routing algorithm

Input:

Load rate Ld_i of all selected service node in CDN $S_j, i = 1,, N$;
Bandwidth BW_i; User request number Q_i;
Time delay list DL_i of all selected service node S_i:
Service price of downstream CDN P_1 ;
Time delay list $DD_j, j = 1,, M$ downstream CDN offered to users

Output:

The optimal service node for user matching S:

for i in N;
$Ct_i = f(Ld_i, BW_i, Q_i)$;
if $DD_{\max} \leq DL_{\max}$ and $(1-\gamma)P_1 < \min\{Ct_i\}$ **then**
  $S =$ The entry of d-CDN;
else
  for k in N;
    $PS_k = Ct_k$ *Ranked in* $\{Ct_k\}$;
    $PC_k = DL_k$ *Ranked in* $\{DL_k\}$;
    $Score_k = \beta * PS_k + (1-\beta)\, PC_k$;
  $\forall k \in N + M$, $PS_k \rightarrow PS_j$;
  for j in N;
  if $Scoreps_j < Scoreps_{j+1}$ **then**
    $S = PS_j$;
    break;
  end if
end if
RETURN S;

The service price of Class B is determined by $P_B = CT + RP_0$. The uCDN chooses whether to provide service by itself or send it to a dCDN freely.

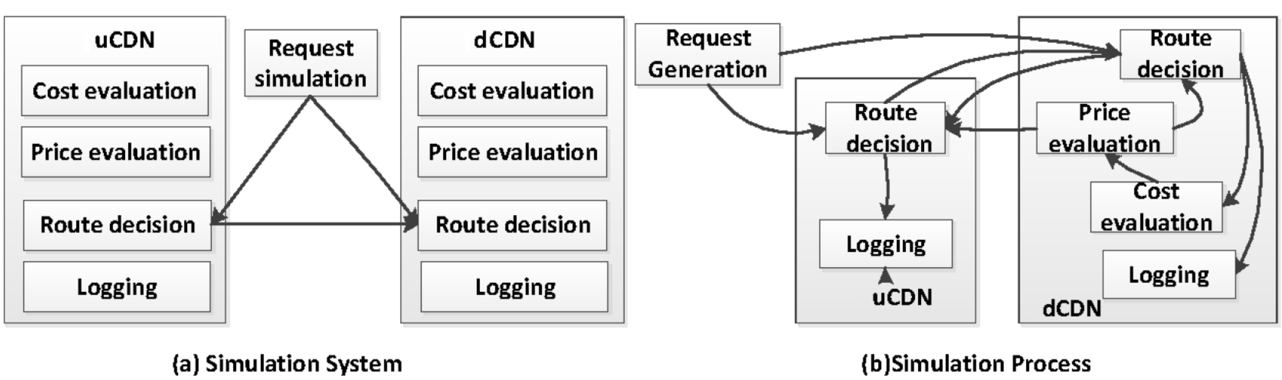

Figure 6. Simulation frameworks.

As is well known, Internet bandwidth is an important criterion for CDN service revenue as well as for the core component of the operation and maintenance cost of a CDN service. Thus, in our experiment, we use bandwidth as the primary CDN service profit criterion to analyze the cost and profit performance of our CDNI request-routing strategy.

Figure 7 shows the Internet bandwidth utilization of different CDNs, where no CDNi means there is no interconnection among CDNs, whereas CDNi means there is interconnection among CDNs. Figure 7a plots the bandwidth utilization of the uCDN and Figure 7b shows the bandwidth utilization of the dCDN. From Figure 7, we find that the bandwidth utilization of the uCDN dropped by approximately 20%, where a portion of the requests were delegated to the dCDN at a lower price. This is because some requests served by the uCDN are also served by the dCDN. As mentioned above, the CDN service cost has a positive relationship with bandwidth utilization, and we can conclude that the service cost of the uCDN decreases. Then, by applying the equation of *service profits = total income – total cost*, we can find that service profits increase dramatically. As for the dCDN, the service bandwidth increased by nearly 50%, as shown in Figure 7b, which means that the dCDN may use its idle bandwidth to serve additional users. As is well known, the dCDN serves these users with its determined service price. Therefore, the dCDN profits more from the CDNI service. In short, the proposed CDNI routing strategy increases profits for both the uCDN and dCDN.

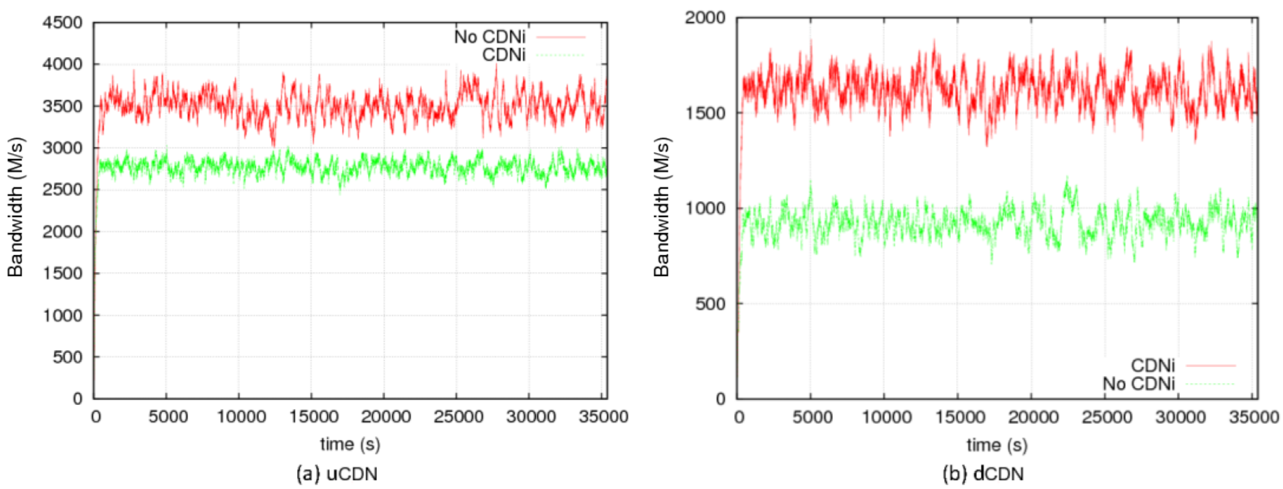

Figure 7. Internet bandwidth utilization of different CDNs.

In contrast to Figure 7, Figure 8 plots the service bandwidth curves of the uCDN and dCDN under different RP_0 values, where RP_0 denotes the excepted minimum service profits of Class B for CDNs. Figure 8a plots the entire service bandwidth curves of the uCDN and dCDN under different RP_0 values, and Figure 8b shows these curves at the initial stage of the CDN service. The uCDN delegated more requests to the dCDN at $RP_0 = 15$ than at $RP_0 = 20$. Interestingly, at $RP_0 = 15$, the uCDN started to delegate requests to the dCDN once service bandwidth exceeded 1600 M/s; at $RP_0 = 20$, however, delegation began at a bandwidth of over 2400 M/s. This was because the uCDN appropriately selected its dCDN based on the service price and operation cost. Clearly, the service price algorithm that we propose in this study guarantees service profits for both the uCDN and dCDN by successfully applying the request-routing strategy.

In addition to securing profits for the dCDNs and reducing uCDN service cost, our CDNI request-routing strategy may facilitate slow-starting or flocking control, which is helpful for avoiding quick server overload, as shown in Figure 9. From Figure 9a, we find that when the service bandwidth increased to the diversion point, the service bandwidth under the proposed CDNI request-routing strategy was significantly lower than

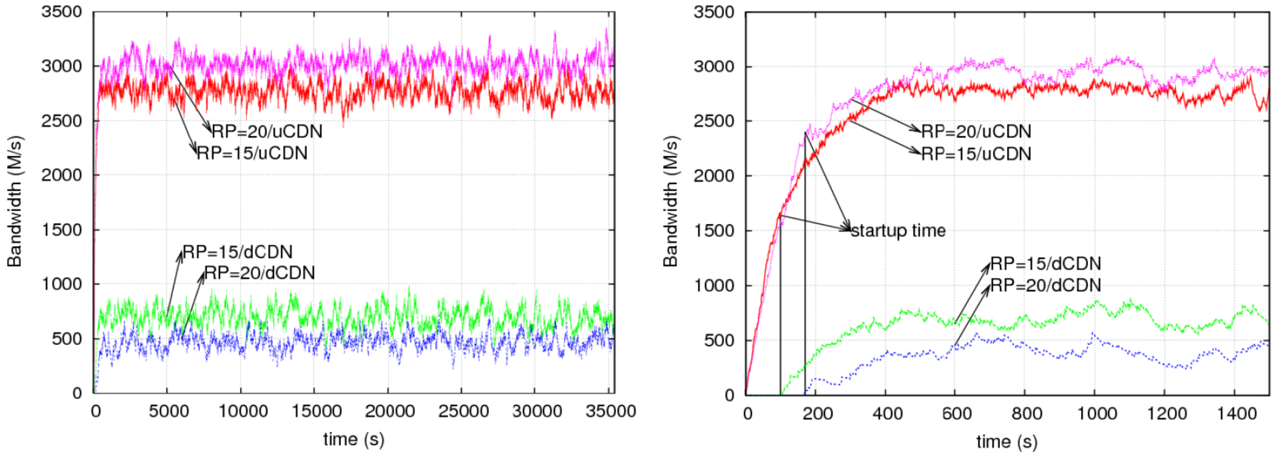

Figure 8. Service bandwidth curves of uCDN and dCDN with time under different RP_0 values.

without request routing. Figure 9b shows that when users flock to the server, the pressure on the CDN is significantly reduced. Current CDNs ease the server's load pressure by deploying more servers, which means that CDN providers need to increase the investment cost for these servers. Different from this solution, our CDNI request-routing strategy eases the server's load pressure by redirecting these extra users to the dCDN, which means that more CDNs share their resources to serve users. Moreover, as the extent of network traffic, video size, and high-definition demands continues to dramatically expand, unpredictable user-flocking will place sizable pressure on service bandwidth. Therefore, controlled flocking prevents resource waste caused by spikes in the demand for CDN service.

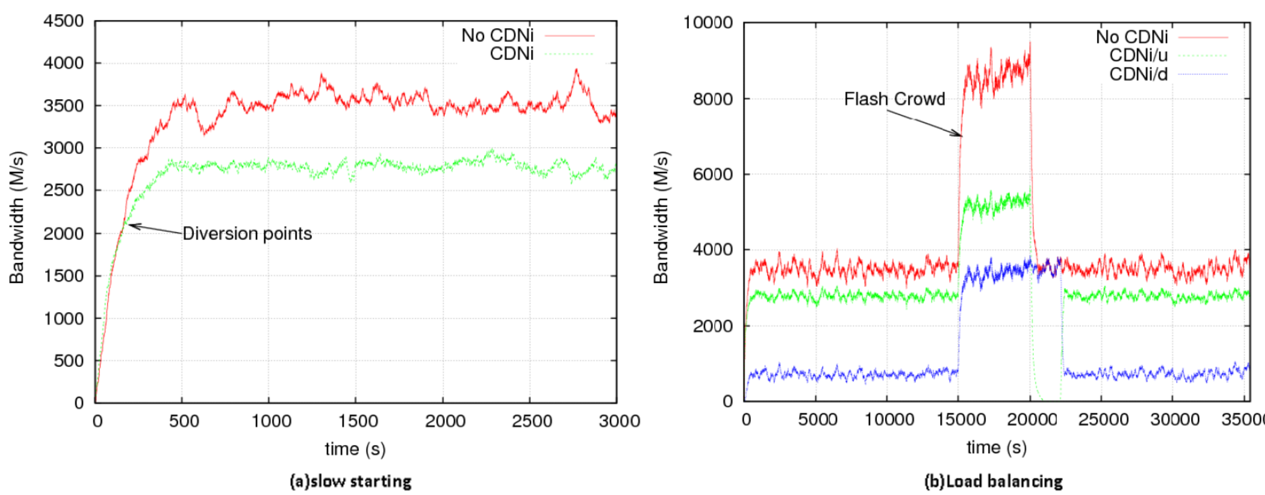

Figure 9. Starting and flocking controlling performance of the CDN.

5. Conclusion

Although the CDNI can scale a single CDN's service footprints by cooperating with other CDNs, the independence of the CDN makes the establishment and maintenance of this cooperation difficult. Additionally, the 'optimal' server is chosen from the cooperating CDNs to provide service for each request, which means that the current route request-routing strategies in one CDN cannot yield their excepted performance; hence, a global request-routing strategy should be designed for CDNI. Thus, we proposed a stable marriage-based routing framework to address these problems. To this end, we first investigated the relationship between the service cost and the service profit of a typical CDNI. Based on this relationship, a service price determination strategy was designed to help CDNs build stable CDNI systems to serve more scale footprint users with the guaranteed quality of CDN services. Then a dynamic stable marriage-based request-routing strategy was proposed to select and redirect each request to the 'optimal' server in the CDNI according to the end user's location, operation state of the CDNIs, and Internet conditions. The results of the simulation experiment show that our request-routing strategy can effectively scale the single CDN's service footprint with the guaranteed quality service. Additionally, it may help the uCDN to achieve more profit without additional service costs, whereas the dCDN can obtain some extra revenue by using the idle resources to serve end users.

Acknowledgment

This work was supported by the Science and Technology Department of Jiangsu Province (Project No. BY2015011-1, NUPTSF under the agreement of NY211036).

References

[1] Pallis G, Vakali A. Insight and perspectives for content delivery networks. Commun ACM 2006; 49: 101-106.

[2] Vakali A, Pallis G. Content delivery networks: status and trends. IEEE Internet Comput 2003; 7: 68-74.

[3] Pacifici V, Dan G. Distributed algorithms for content allocation in interconnected content distribution networks. In: IEEE 2015 Computer Communications and Networks Conference; 26 April–1 May 2015; Kowloon, Hong Kong. New York, NY, USA: IEEE. pp. 2362-2370.

[4] Durillo JJ, Prodan R, Barbosa JG. Pareto tradeoff scheduling of workflows on federated commercial clouds. Simul Model Pract Th 2015; 58: 95-111.

[5] Niven-Jenkins B, Faucheur FL, Bitar N. Content Distribution Network Interconnection (CDNI) Problem Statement. Freemont, CA, USA: Internet Engineering Task Force, 2012.

[6] Gonzalez JL, Perez JC, Sosa-Sosa VJ, Sanchez LM, Bergua B. SkyCDS: A resilient content delivery service based on diversified cloud storage. Simul Model Pract Th 2015; 54: 64-85.

[7] Woo H, Han S, Heo E, Kim J, Shin S. A virtualized, programmable content delivery networks. In: IEEE 2014 Mobile Cloud Computing, Services, and Engineering Conference; 8–11 April 2014; Oxford, UK. New York, NY, USA: IEEE. pp. 159-168.

[8] Heo K, Jung I, Yoon C. A study of enhancement in virtual content distribution network. In: IEEE 2014 Information Science and Applications Conference; 6–9 May 2014; Seoul, South Korea. New York, NY, USA: IEEE. pp. 1-5.

[9] Pathan AMK, Buyya R. A taxonomy and survey of content delivery networks. Lect Notes Electr En 2008; 9: 33-77.

[10] Kiesel S, Previdi S, Stiemerling M, Woundy R, Yang Y. Application-Layer Traffic Optimization (ALTO) Requirements. Freemont, CA, USA: Internet Engineering Task Force, 2012.

[11] Adhikari VK, Jain S, Ranjan G, Zhang ZL. Understanding data-center driven content distribution. In: ACM 2010 CoNEXT Student Workshop; 30 November–3 December 2010; Philadelphia, PA, USA. New York, NY, USA: ACM. pp. 1-4.

[12] Pathan M, Buyya R. Resource discovery and request-redirection for dynamic load sharing in multi-provider peering content delivery networks. J Netw Comput Appl 2009; 32: 976-990.

[13] Seedorf J. CDNI Request Routing with ALTO. Freemont, CA, USA: Internet Engineering Task Force, 2013.

[14] Chen J, Zou L, Yuan X. An economic service strategy for cloud computing collaboration. J Comput Inform Syst 2015; 11: 3891-3902.