Designing a private CDN with an off-sourced network infrastructure: model and case study

Claudia Canali, Andrea Corbelli, and Riccardo Lancellotti

University of Modena and Reggio Emilia,
Department of engineering "Enzo Ferrari",
{claudia.canali, andrea.corbelli, riccardo.lancellotti}@unimore.it

Abstract. Content Delivery Networks for multimedia contents are typically managed by a dedicated company. However, there are cases where an enterprise already investing for a dedicated network infrastructure wants to deploy its own private CDN. This scenario is quite different from traditional CDNs for a twofold reason: first, the workload characteristics; second, the impact on the available choices for the CDN design of having the management of the network infrastructure off-sourced to a third party. The contribution of this paper is to introduce and discuss the optimization models used to design the private CDN and to validate our models using a case study.

Keywords: Content Delivery Network, Multimedia contents, Case study

1 Introduction

Delivery of multimedia streaming content is a challenging task that is typically carried out using dedicated infrastructures, namely *Content Delivery Networks* (CDNs). A CDN is basically a network infrastructure, composed of links and servers (namely *edge servers*) that forward the media stream and are placed as close as possible to the clients, aiming at improving the performance of media contents delivery. The benefits of such infrastructure are intuitive because the edge servers act as demultiplexer, splitting one media stream into one separate stream for each client, thus reducing the risk of network congestion leading to poor performance. Furthermore, each edge server can perform *transcoding*, adapting the quality of the stream to the specific characteristics of a client (in terms of bandwidth and resolution).

Several studies exist in literature proposing strategies and models for edge server placement in traditional highly distributed CDN infrastructures, such as those owned by providers like Akamai [1] and Limelight [2]: these studies aim at achieving the best possible placement of the edge servers in order to satisfy the end-users quality of service, the requirements about bandwidth and latency, as well as the need for the CDN provider to minimize the infrastructure costs [3–6]. While the design of a traditional CDN as previously described and the models to place the edge servers in such infrastructure are well understood, little effort has been devoted to the case of a *Private CDN (P-CDN)* where a company that

already has an access to a geographic network infrastructure (typically leased from a network operator that remains the maintainer of the infrastructure) wants to deliver contents to its branches placed in geographically distinct locations.

In this paper, we focus on a P-CDN scenario owned by a private company where the networking infrastructure is outsourced to a third party. This scenario presents two main differences with respect to the design of a typical CDN. First, we have just one content provider that is the only customer of the network infrastructure: the resulting workload is quite different from the typical case where we have many clients accessing contents from multiple providers. Second, as the network infrastructure is off-sourced to a network provider, the design of the P-CDN has major constraints in the possibility to place the edge servers that will support the delivery of the component.

The main contribution of this paper is the proposal of the mathematical models used to design the P-CDN and their validation through a case study. Our findings suggest that the introduction of a P-CDN can increase by more than eight times the number of clients that can access the multimedia data at the highest quality. On the other hand, we also show that there is a major trade-off between the performance of the P-CDN in terms of media quality for the users and the number of edge servers deployed. As the cost of edge servers may be high for a single content provider, the parameter for deciding if a company branch is large enough to be chosen to host an edge server remains a critical decision for the tuning of the model.

The remainder of this paper is organized as follows. Section 2 discusses the related work and positions our contribution with respect to the state of the art. Section 3 describes our model for placing the edge servers and proposes some heuristics to solve the underlying optimization problem. Section 4 describes the case study considered to validate our model. Finally, Section 5 concludes the paper with some final remarks.

2 Related Work

Content delivery networks (CDNs) have gained immense popularity over the years, attracting the attention of many researchers. An analysis of the main trends in content delivery networks (CDNs) and user access paradigms is presented in [3], while [4] explores the issues of content delivery through CDNs with a special focus on the delivery of multimedia content.

Our proposal focuses on an optimization model to minimize the number of edge servers used for a live-streaming scenario for multimedia contents, with the possibility for the edge servers to perform online transcoding [7, 8]. A comprehensive survey on the specific issues of edge server placement algorithms in traditional and emerging paradigm-based CDNs, such as the need to satisfy several requirements about bandwidth, latency, and quality of service for end-users as well as the need for the provider to minimize the infrastructure costs, is presented in [5]. Another study that deserves to be mentioned in this field is [6], that provides design principles for a highly distributed CDN while fo-

cusing on four key aspects: the optimal location of edge servers with caching functionalities, mechanisms for request routing, content replica placement, and content outsourcing and retrieval. It is worth to note that the majority of the state-of-the-art focuses on the scenario of highly distributed commercial CDN infrastructures, owned by CDN owners such as Akamai [1] and Limelight Networks [2]. On the other hand, the scenario considered in our experience paper focuses on a P-CDN, owned and used by a company to serve its own contents, where the networking is outsourced to a third party. This scenario is characterized by different challenges with respect to a typical CDN. Indeed, being the content provider also the only customer of the network infrastructure, the resulting workload is quite different from the typical case where many clients access contents from multiple providers. Moreover, the possibility to place the edge servers of the infrastructure is subject to major constraints due to the fact that the network infrastructure is off-sourced to an external network provider.

Some studies in literature focus instead on Telco-CDNs, where the content distribution services are managed by telecommunications service providers (TSPs) that began to launch their own content delivery networks as a means to lessen the demands on the network backbone and to reduce infrastructure investments. In this case, since the network operator controls both the infrastructure and the content delivery overlay, it is in a position to manage the complete system so that networking resources are optimally utilized. Among these, we cite the study in [9], that proposes an algorithm for the placement of video chunks on the edge servers, and [10], where an architecture for on-demand service deployment over a Telco-CDN is presented. Also in the specific case considered in our study the owner of the CDN is a telecommunication company, however the network infrastructure is outsourced to an external network provider, with the consequence that the edge servers cannot be arbitrarily located but should be placed in determined spots corresponding to the company branches.

In the field of multimedia and video distribution, some research focuses on strategies to improve the performance of content distribution on mobile networks, due to the high percentage of people using mobile devices for streaming video and online applications [11, 12]. Our scenario may include mobile users, but they typically access contents by mobile devices connected through a WiFi network and located in the company branches, that are placed in geographically defined locations (local clients on company LAN locations), with the company being the only provider delivering contents through the network infrastructure. Hence, we do not have to face the issues of the highly transient properties of mobile devices and locations of users accessing contents from multiple providers, but we can focus on the main issue of optimizing the performance of the P-CDN by identifying the best locations to place the edge servers among the company branches geographical positions.

3 P-CDN Model

We now discuss the general scenario of a P-CDN, we introduce the mathematical model used to describe the problem of designing the CDN and we outline some heuristic algorithms for solving the problem.

3.1 General scenario definition

In the considered scenario we assume to have multiple *locations*, corresponding to the branches of the company deploying the P-CDN, where clients are hosted. We denote these locations as $l \in \mathbf{L}$. Each location is characterized by download and upload bandwidths (BWd_l and BWu_l). We also define BWo as the bandwidth available at the origin server for feeding the P-CDN. It is worth to note that, as the network may have additional background traffic, the considered bandwidth is not the nominal available bandwidth of the links but is an estimate of the actually unused bandwidth. Clients can consume multiple streams, each characterized by a type of encoding $t \in \mathbf{T}$ and by a bandwidth requirement BW_t. We denote as $\mathbf{C}_{t,l}$ the set of clients in location l that require a stream with encoding t. When considering the clients, we must take into account that just a fraction of the total clients will actually watch the video stream. The set $\mathbf{C}_{t,l}$ considered in the model consists of an estimate set of clients that will likely concurrently access the live stream.

The delivery of multimedia contents occurs, in the absence of a CDN, between the origin server and the clients. While a multicast-enabled network could make the delivery quite straightforward, the outsourced nature of the underlying network makes the adoption of IP multicasting not viable. It is then necessary to introduce a group of *edge servers* that can be located in the company branches to act as content distributors for the nearby clients. We denote as $e \in \mathbf{L_e}$ the locations that host an edge server. We assume that edge servers may also implement transcoding functions, so that a single, high quality stream can be used to generate lower-quality streams.

3.2 Live-streaming scenario

We can model the live streaming scenario as an optimization problem where we simply aim to minimize the number of edge servers used. We model the optimization problem using a boolean variable to decide if a location should host or not an edge server. Specifically, E_l ha value 1 if in location l we have an edge node, 0 otherwise (implicitly we also define $\overline{E_l}$ that is the boolean negation of variable E_l). In this first model we assume that an edge server can support only the clients in the same location as the server. More complex scenarios, where a single edge server can serve clients in more than one location will be analyzed in the following.

$$\min \sum_{l \in \mathbf{L}} E_l \tag{1.1}$$

subject to:

$$BWd_l \geq \overline{E_l} \sum_{t \in \mathbf{T}} BW_t \cdot |\mathbf{C}_{t,l}| + E_l \cdot \max BW_t \quad \forall l \in \mathbf{L}, \tag{1.2}$$

$$BWo \geq \sum_{l \in \mathbf{L}} \overline{E_l} \sum_{t \in \mathbf{T}} BW_t \cdot |\mathbf{C}_{t,l}| + \sum_{l \in \mathbf{L}} E_l \cdot \max BW_t \tag{1.3}$$

The optimization problem can be summarized as follows: we aim to minimize the number of edge servers (Objective function 1.1) under the double condition that on each location the bandwidth of the clients must not exceed the available bandwidth available for media download (constraint 1.2) and that the outbound bandwidth on the origin server must be able to satisfy the clients that are not served by an edge server and must not exceed the bandwidth necessary for the edge server themselves (constraint 1.3).

A solution for this optimization problem may be reached in a straightforward way in two steps:

- we check the constraint 1.2 for every location and, if the requirement is not met, we place an edge server in that location;
- afterwards, we check constraint 1.3 and, if the condition is not met, we add edge server in the most bandwidth-requiring locations until the constraint is satisfied.

3.3 Heuristic solution

While this model is a viable option to design the P-CDN, there are a few critical issues that may hinder a straightforward application of the model to the actual case study. The main issues can be summarized as:

- It is hard to clearly identify client requirements in terms of encoding. For most client devices there is the possibility to use multiple encodings, depending on the actual available bandwidth, so the approach of asking for the highest possible quality level and accept a lower quality stream seems the most viable option.
- For some company branches the number of clients is very low and the bandwidth is rather limited as well. An edge server would be required to guarantee their access to streaming media, but the cost of installing such device is not acceptable for these small branches. We will therefore consider a threshold to limit our study to the medium/large branches.

Concerning the first problem, we define a simple algorithm for assigning the available bandwidth to the clients within the same location (Algorithm 1).

Algorithm 1 allocates bandwidth to the clients starting with the lowest possible quality and up to the highest quality (corresponding to the highest bandwidth utilization). We define $\mathbf{C}_l$ as the set of clients in location l. The outer loop

Algorithm 1 Client bandwidth allocation

Require: $\mathbf{C}_l$
Ensure: $t_c \quad \forall c \in \mathbf{C}_l$
  for $t \in \texttt{sort}(\mathbf{T} \cup \{na\} \uparrow)$ **do**
    for $c \in \mathbf{C}_l$ **do**
      $BWd_l \leftarrow BWd_l + BW_{t_c} - BW_t$
      if $BWd_l \geq 0$ **then**
        $t_c \leftarrow t$
      else
        break
      end if
    end for
  end for

iterates over the possible stream encodings in ascending order of bandwidth consumption. For each client, if there is enough bandwidth, the algorithm assigns that bandwidth to the client and updates the remaining bandwidth in the location. The special value na for the stream quality means that the client cannot access any stream at any quality; a stream quality na has a null bandwidth consumption. The inner loop just assigns the stream quality to the clients and updates the available bandwidth until it is exhausted.

The outcome of the algorithm can be one of the following conditions:

- $BWd_l < \min(BW_t)$ (resulting in $t_c = na \forall c$): the location cannot support any stream, even at the lowest quality. without a network upgrade, the company branch cannot benefit from the CDN and must be excluded from the analysis;
- $\exists c : t_c = na$: not every client can access the stream even at the lowest quality. In the absence of an edge server, the consumption of live streams will be possible only for a subset of the considered client population;
- $\exists c : t_c < \max(t_c)$: every client can access to the stream but some of them (or even all) can only consume a lower quality stream in order to avoid congestion on the location link. An edge server can fix this issue.
- $t_c = \max(t_c) \forall c$: every client can access to the stream at the highest possible quality.

As previously mentioned, an additional problem in the considered scenario is the presence of badly connected locations where installing an edge server is not advisable for economic reasons. To this aim, we introduce a threshold parameter thr to control the number of clients in a given location and we define the following conditions to manage this trade-off:

- $|\mathbf{C}_l| \geq thr$: the location may host an edge server;
- $|\mathbf{C}_l| < thr$: the location is too small to host an edge server, hence we force $E_l = 0$.

Finally, we consider also an evolution of the basic CDN delivery model discussed up to now. Specifically, we consider that an edge server in a given location

$e \in \mathbf{L_e}$ may provide the streaming content not just to the clients in e, but also to clients placed in nearby locations. To this purpose, we introduce the concept of *Satellite Location*, that is the case where a location relies on the edge server of a different location for its clients. $\mathbf{L_s}_e$ is the set of satellite locations depending on the edge server in e. The edge server in e will consume the upload bandwidth of the location to this aim. The main constraint in this case is that the download bandwidth of the clients in the satellite locations must not exceed the upload bandwidth of the edge server:

$$BWu_e > \sum_{s \in \mathbf{L_s}_e} BWd_s$$

Algorithm 2 Management of Satellite locations

Require: $\mathbf{L}$, $\mathbf{L_e}$, $niter$, $minBW$
Ensure: $\mathbf{L_s}_e, \forall e \in \mathbf{L_e}$
  $\mathbf{L_s} \leftarrow \mathbf{L} - \mathbf{L_e}$
  $\mathbf{L_s}_e \leftarrow \emptyset, \forall e \in \mathbf{L_e}$
  for $i \in [1, niter]$ **do**
    for $s \in \mathbf{L_s}$ **do**
      $e \leftarrow$ `nearestEdge`(s, $\mathbf{L_e}$)
      Add s to $\mathbf{L_s}'_e$
    end for
    for $e \in \mathbf{L_e}$ **do**
      for $s \in$`sort`($\mathbf{L_s}'_e$, from nearest) **do**
        if $BWu_e \geq BWd_s$ **then**
          Add s to $\mathbf{L_s}_e$
          $BWu_e \leftarrow BWu_e - BWd_s$
          Remove s from $\mathbf{L_s}$
        end if
      end for
      if $BWu_e < minBW$ **then**
        Remove e from $\mathbf{L_e}$
      end if
    end for
  end for

In Algorithm 2, we consider a set location where we have edge servers ($\mathbf{L_e}$) and the overall set of locations $\mathbf{L}$ that may host clients. The location that do not host an edge server ($s \in \mathbf{L_s}$) is the difference between the two sets and is the list of location that may be selected as satellite locations. Furthermore, $\mathbf{L_s}_e$ is the list of satellite location for each edge server e.

The algorithm starts by dividing the set of possible satellite locations, assigning each $s \in \mathbf{L_s}$ to the nearest edge server. The set $\mathbf{L_s}'_e$ is then the list of *potential* satellite locations for that edge (first nested loop). Next, for each edge server we analyze the potential satellites from the nearest one and, if the upload

bandwidth is enough, we insert the potential satellite in the list of satellite location for that edge, updating the available upload bandwidth. If at the end of the cycle the bandwidth for the edge server is exhausted (that is, less than a given threshold $minBW$), we remove the edge server from the analysis. This process is iterated $niter$ times to take into account the re-organization in the list of edge servers as their upload bandwidth is exhausted.

4 Experimental results

We now introduce the reference scenario used in our case study to validate our model for the design of the P-CDN and we provide a first analysis on the parameters of the model. We recall that the main output of our model consists in an indication of:

- placement of the edge server;
- definition of satellite locations.

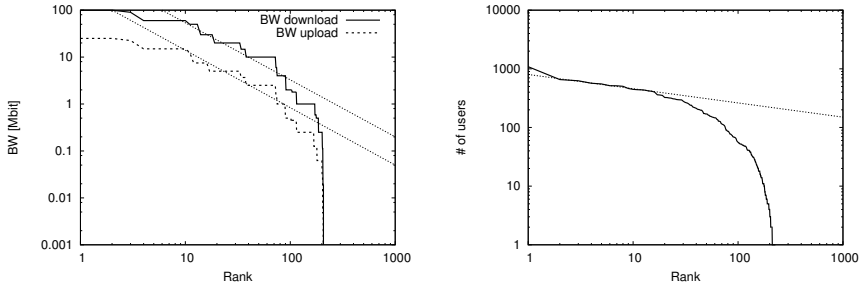
Throughout this section, we define the reference scenario of our experiments, provide some results on the effect of adopting a P-CDN infrastructure, and discuss the impact of the threshold parameter thr.

4.1 Reference scenario

The reference scenario is based on a real scenario supplied by 47deck company; some of the most significant parameters are outlined in Figure 1. For privacy reason, we omit some detail on the real dataset and we limit our analysis on a synthetic reconstruction of the initial network. Specifically, we consider a set of more than 200 locations with download and upload bandwidth distributions modeled according to a truncated Zipf distribution, as shown in Figure 1a. The thin lines represent the Zipf distribution, while the thick lines are the actual bandwidth considered in our case study. To obtain these values we started from the original Zipf distribution, we truncated it and we rounded the values to available bandwidths of cable connections. Download and upload bandwidths show a similar behavior, with the upload bandwidth being typically 25% of the download one. In a similar way, also the clients per location follow a truncated distribution, as shown in Figure 1b.

For our scenario, we consider the following scenario. We consider three types on encodings that can be consumed by clients. Specifically, we distinguish between:

- Low definition (LD), characterized by a bandwidth consumption of 512 Kb/s
- Standard definition (SD), characterized by a bandwidth consumption 1024 Kb/s
- High definition (HD), characterized by a bandwidth consumption of 1536 Kb/s



(a) Bandwidth distribution (b) Clients distribution

Fig. 1: Bandwidth and Clients distributions

For each location, we consider that just 60% of the upload/download bandwidth is available for the streaming, while the remaining 40% is used by the normal operation carried out on the network. Furthermore, we explicitly discard any location with an available bandwidth below 512 Kb/s as it would not be able to support any type of streaming even in the presence of an edge server.

4.2 Live streaming results

Given the setup described previously, we now evaluate the results of the proposed model when applied to the above-described scenario. Specifically, we evaluate how the considered scenario can be used to improve the user experience in terms of streaming video quality and we provide a first design of the P-CDN infrastructure. For this first analysis we consider that the main parameter of the model, that is the threshold *thr*, to decide if a location has enough clients to be eligible for hosting an edge server is set to 50.

Table 1: Impact of P-CDN

	P-CDN	**no P-CDN**
# of Locations		
Too small	99	0
Edge	77	0
Satellite	19	0
Origin	18	213
# of Clients		
LD	1577	1986
SD	94	94
HD	466	57

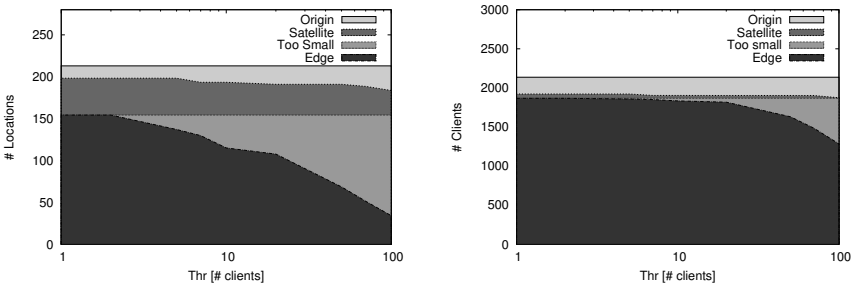
Table 1 provides a first demonstration of the impact of a P-CDN architecture for the delivery of live media. Without the P-CDN the scenario, is fairly simple

as the origin server must cope with serving all the 213 locations considered in this study. As we introduce the P-CDN infrastructure we have 77 locations that are selected to host an edge server. These edge servers can further provide streaming support for 19 additional locations. 99 locations should host an edge server, but having just a few clients are discarded (we recall that in this analysis we consider $thr = 50$). The origin serve is thus responsible for 18 locations together with the 99 small company branches previously mentioned.

If we consider the user experience, the adoption of the P-CDN has a clear impact. Without edge server just a small fraction of clients can access the stream with something better than a low definition. Specifically only 94 (4%) and 57 (3%) clients can access to SD and HD video quality, respectively. If we consider the adoption of the P-CDN, we observe that the number of clients accessing the HD stream increases by more than eight times. The number of clients with a low quality feed remains high, but this is mainly due to the presence of a non-negligible number of clients that fall in the *Too small* company branches that are discarded as potential locations for edge servers.

4.3 Sensitivity to threshold

The high impact of locations and, consequently, clients that are removed from the potential list of edge servers due to their size being below the threshold *thr* motivates our analysis on the sensitivity to the impact of the *thr* parameter. Figure 2 provides an analysis of how the edge servers are influenced by this threshold.



(a) Number of locations (b) Number of clients

Fig. 2: Locations and clients as a function of *thr*

Figure 2a presents an analysis on the locations hosting edge servers of the P-CDN and on the corresponding satellite locations. Throughout this analysis we change our threshold *thr* to decide if a location is too small from 5 to 100 clients. The results are quite intuitive: as *thr* grows, the number of locations hosting an edge server is reduced and the corresponding number of locations considered *Too small* to host an edge server grows. The change in the number

of edge servers has also an impact on the number of satellite locations: as the number of edge server is reduced, the number of satellite locations that can be supported decreases as well. Clearly, locations that do not host neither an edge server nor a satellite location are serviced directly by the origin servers.

Figure 2b refers on the number of clients. Specifically, we consider the number of clients corresponding to the previously defined four categories (that is, clients in locations *Too small*, clients service by edge servers, clients in satellite locations and clients serviced by the origin server). Again we run the tests for a threshold value $thr \in [5, 100]$. The main messages are comparable with the findings shown in Fig. 2a. The main difference is the, considering the number of clients rather than the number of locations, the clients serviced by an edge server remains much larger as the number of locations with an edge server as thr grows. This effect can be explained considering that due to the nature of the threshold, locations in the *Too small* category host less clients than the locations with an edge server.

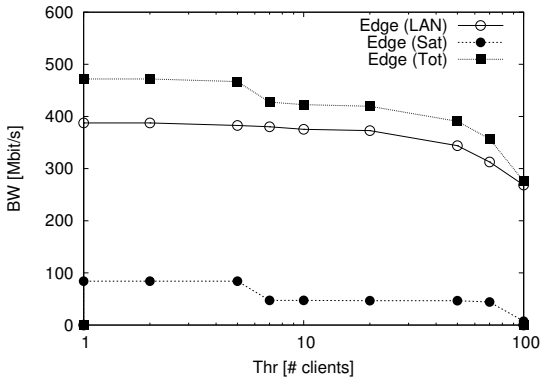


Fig. 3: Bandwidth utilization

The last analysis is provided in Figure 3 and shows the bandwidth for edge server throughout the P-CDN as a function of the threshold thr. We observe that, as the threshold increases, we have a reduction in the global bandwidth served by the edge servers due to their number reduction. We also provide a break-down between the bandwidth provided by the edge servers to support their local clients and the bandwidth used to service the satellite locations. As expected, the aggregate internal bandwidth of edge servers for servicing local clients (on the location LAN) is reduced by the decrease in the number of locations hosting and edge. In a similar way, the drop in the number of satellite locations determines a reduction in the bandwidth devoted by the edge servers to clients outside their location.

5 Conclusions

Throughout this paper we presented the problem of designing a P-CDN for streaming media, that is a private CDN where a company aims at defining an infrastructure for the delivery of multimedia streams. Such scenario is peculiar for multiple reasons: first, content providers and content consumers are part of the same company; second, the content underlying infrastructure is considered to be outsourced to a third party that owns and operates the network. Even if the main goal of designing a P-CDN is similar to that of a traditional CDN, the above mentioned characteristics have a major impact on the expected workload and on the available choices for edge server placement.

Our contribution is the definition of models and algorithms to cope with these problems. We validate our solution using a case study based on a real-world problem were a P-CDN must be deployed over hundreds of locations. Our experiments demonstrate the benefit of the proposed methodology for designing a P-CDN and outline the trade-off between the need to place edge servers on locations with just a few clients and the desire to reduce the number of used edge server.

Acknowledgements

The authors acknowledge:

- the support of 47Deck (`www.47deck.it`) for the definition of the reference scenario used in the model and in the experiments presented in the paper.
- the support of the University of Modena and Reggio Emilia through the project *S^2C: Secure, software-defined Cloud*.

References

1. Akamai: The world's largest and most trusted cloud delivery platform (2017) https://www.akamai.com.
2. Limelight: Limelight networks (2017) https://www.limelight.com.
3. Kaafar, M.A., Berkovsky, S., Donnet, B.: On the potential of recommendation technologies for efficient content delivery networks. SIGCOMM Comput. Commun. Rev. **43**(3) (July 2013) 74–77
4. Canali, C., Cardellini, V., Colajanni, M., Lancellotti, R.: Content delivery and management. In Buyya, R., Pathan, M., Vakali, A., eds.: Content Delivery Networks: Principles and Paradigms. Springer (2008)
5. Sahoo, J., Salahuddin, M.A., Glitho, R., Elbiaze, H., Ajib, W.: A survey on replica server placement algorithms for content delivery networks. IEEE Communications Surveys Tutorials **19**(2) (2017) 1002–1026
6. Spagna, S., Liebsch, M., Baldessari, R., Niccolini, S., Schmid, S., Garroppo, R., Ozawa, K., Awano, J.: Design principles of an operator-owned highly distributed content delivery network. IEEE Communications Magazine **51**(4) (April 2013) 132–140

7. Krishnappa, D.K., Zink, M., Sitaraman, R.K.: Optimizing the video transcoding workflow in content delivery networks. In: Proc. of the 6th ACM Multimedia Systems Conference. MMSys'15, Portland, Oregon (2015)
8. Canali, C., Cardellini, V., Lancellotti, R.: Content adaptation architectures based on squid proxy server. World Wide Web Journal **9**(1) (2006) 63–92
9. Li, Z., Simon, G.: In a telco-cdn, pushing content makes sense. IEEE Transactions on Network and Service Management **10**(3) (September 2013) 300–311
10. Frangoudis, P.A., Yala, L., Ksentini, A., Taleb, T.: An architecture for on-demand service deployment over a telco cdn. In: IEEE International Conference on Communications (ICC). (May 2016)
11. Akamai: A content centric approach to delivering high quality mobile experiences (2017) Akamai White paper Library– https://content.akamai.com/PG2376-Mobile-Network-Solutions.html.
12. Almashor, M., Khalil, I., Tari, Z., Zomaya, A.Y., Sahni, S.: Enhancing availability in content delivery networks for mobile platforms. IEEE Transactions on Parallel and Distributed Systems **26**(8) (Aug 2015) 2247–2257