OPAC: An Optimal Placement Algorithm for Virtual CDN

Hatem Ibn-Khedher[1], Emad Abd-Elrahman[1*], Ahmed E. Kamal[2], and Hossam Afifi[1]

[1]*Institute Mines-Telecom (IMT), RST Department, Telecom SudParis, Saclay, France.*
[*]*Computer and Systems Department, National Telecommunication Institute (NTI), Cairo, Egypt.*
[2]*Department of Electrical Computer Engineering, Iowa State University, Ames, IA 50011-3060, USA.*
hatem.ibn_khedher, emad.abd_elrahman, hossam.afifi@telecom-sudparis.eu, kamal@iastate.edu

Abstract

Cloud computing and Network Function Virtualization (NFV) use cases proposed by ETSI can improve network caching when an adequate optimization algorithm is used. A cache placement algorithm is proposed based on these virtualization tools. Moreover, a cache placement protocol is presented to support the cache migration. Optimal cache and Virtual Machine (VM) placement and migration, and optimal routing based on several new constraints such as system load, network infrastructure, user satisfaction and migration cost is designed. Extensive evaluations and comparison to state of the art techniques are carried out. The results show that our cache techniques outperform other solutions and give better satisfaction to the three constituents: users, network operators and content providers.

Keywords: Virtual Content Delivery Network, Cache Placement, Optimization, Network Function Virtualization, Software Defined Networks

1. Introduction

Content Delivery Network (CDN) targets the deployment of multimedia content (images, files and videos) over multiple data centers through Internet Service Providers (ISPs). The main goal of a CDN is to serve content to the end users with better availability and higher performance [1] and [2]. It is also proposed to ISPs as an offloading solution for intra and inter data centers updating [3] .

Fig. 1 depicts the main components of the typical CDN network. It includes: i) a content replication plane: it is the system responsible for content caching, streaming and distribution. ii) a request routing plane: it is the system responsible for redirecting end-user requests to the optimal surrogate server, and iii) a CDN management plane: it the centralized system responsible for cache management processes such as content eviction, placement, etc.. and it does the accounting and billing tasks. Moreover, this network can perform many file management actions like hosting/caching/fetching/swapping in order to satisfy the client's Quality of Experience (QoE) and enhance the overall Quality of Service (QoS) of the network [4].

In video streaming systems either live-streaming or on-demand services, the CDN placement is a key point in delivery optimization. In a previous work, the video adaptation was considered in terms of users perception (user experience) [5]. In [6], authors focused on the contextual QoE metrics and their effects on data retrieval or caching costs. The overall performance especially in video services cannot be only based on user perception models. In fact, major deployed data centers over the Internet such as Akamai [7], Amazon [8] or Google [9] have to consider other important parameters. Besides financial issues that are out of our scope, we claim that system and core network parameters are key issues in making content movements over the networks. To the best of our knowledge and despite the existence of placement techniques deployed by the above mentioned providers, the literature lacks optimization algorithms taking into consideration actual system and network parameters that meet the multimedia delivery requirements.

Architectures for virtualizing network functions such as Network Functions Virtualization (NFV), Software Defined Networking (SDN) and OpenStack can add flexibility to the design of network applications as in-

Email address:
`hatem.ibn_khedher@telecom-sudparis.eu` (Hatem Ibn-Khedher[1], Emad Abd-Elrahman[1*], Ahmed E. Kamal[2], and Hossam Afifi[1])

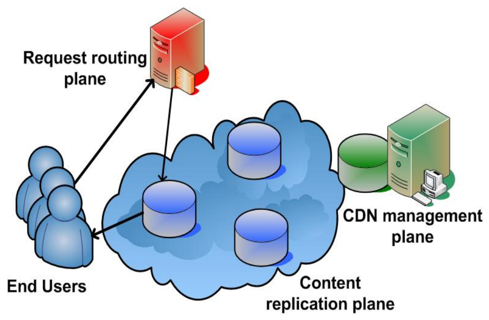


Figure 1: A typical CDN network architecture

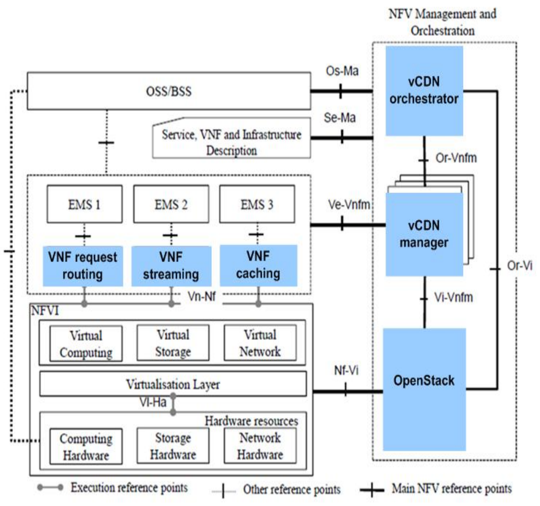


Figure 2: vCDN architecture according to ETSI-NFV [20]

vestigated in [10], [11], [12], [13], [14], [15], [16], and [17]. They were recently adapted to the CDN context for video content migration. The goal was to decrease the load on the origin content providers (like YouTube, Netflix[1]) and to let the operators/ISPs host the content on their network to decrease as much as possible the load on the external links. Yet, an additional mechanism should be proposed to further improve the overall caching performance combining optimization and migration within the operator/ISP network.

The overall virtualization challenges for CDN transition to vCDN are addressed in the research project DVD2C [18]. In a previous work, we have investigated the main network issues in virtual machine migration and especially for vCDN use case [19] and in an SDN/NFV context. Fig. 2 depicts the proposed vCDN architecture based upon the ETSI-NFV reference architecture [20]. Note here that different vCDN implementations could be exist. Indeed, as depicted in Fig. 1, a CDN network is composed of different systems such as caching, distribution, request routing, and management. Therefore, by making the transition from classical CDN to the NFV/SDN-based vCDN context, all theses system components can be virtualized and deployed as VNF instances, but given the growth of video delivery, virtualization has to target mainly the caching and streaming components (i.e., VNF caching, VNF streaming, etc.). Further, a VNF request routing system (vRR) is added to the vCDN proposed architecture in order to orchestrate the VNF caching/streaming nodes.

Currently, despite the importance of optimization tasks in the novel virtualization context of SDN and NFV, such functions are missing in the global network architecture. We therefore, try in this paper to contribute with OPAC (protocol and optimization model) as a potential optimization technique for vCDN placement and explain how it is integrated in the network operator. Although video on demand (VoD), assisted by cascaded caching, enhances the QoE and decreases the load, live video on the other hand can still benefit from our optimization algorithm.

In this paper, we introduce the concept of virtual CDN as a Service (vCDNaaS) using SDN tools. Through our proposal, we focus on the framework architecture, design and orchestrations of vCDN use case based on SDN controller, e.g, OpenDaylight [21]. Then, an optimization placement algorithm for vCDN including content caching and request redirecting is introduced with operating system, network, and quality of experience constraints.

The rest of this paper is organized as follows: Section 2 highlights the state-of-the-art. Then, Section 3 investigates the related work. Section 4 proposes the vCDN concept in terms of system requirements and design methodology. Section 5 details the steps of the proposed caching protocol. The optimization model is introduced in Section 6. The final optimization evaluation is presented in Section 7. Section 8 clarifies the algorithm integration. The update performance if the cache prediction is slightly imprecise is evaluated through a comparison in Section 9 and then the work is concluded in the last section.

[1]It proposes an open CDN using NGINX web proxy server and other open source software: https://openconnect.netflix.com/software/

2. State-of-the-art

The main architectures that can meet the virtualization and the softwarization [2]of the network are NFV/OpenStack and SDN/OpenFlow. Thus, we detail in the next subsections the main taxonomies for virtual caching based on those alternatives and their relevant work.

2.1. Virtualization Taxonomies Context

Virtual caching is the process of cache service abstraction from the underlying hardware (H/W independence). It can be characterized as follows:

2.1.1. Content Moving

In content moving, only the service is replicated or cloned to another location (NFV Infrastructure: NFVI enabled). This case is used when the Internet connection of the end user is deteriorated or when the load on the physical streaming server exceeds a specific threshold.

2.1.2. Server Moving

In server moving, the instantiation of a new service is made in another location (the edge network) by migrating the complete virtual machine that runs the virtual content delivery network function (vCDN). ETSI has highlighted vCDN among the major NFV use cases [22]. Further, different network functions in the mobile core network are also investigated to be virtualized and ease their moving in [23], [24], and [25].

2.1.3. Session Moving

In session moving, the two previous techniques (content and server moving) of the virtualized cache are merged (hybrid technique). Indeed, the end point of the session (server side) that was already established between the streaming server and the client is moved to another video streaming server once the video content was completely replicated to the new location. SDN paradigm is used as a tool to achieve this goal and to build such an application assisted by additional modules carrying out video adaptation, end-to-end video delivery [26], video traffic engineering and content management as explained in [27], [28] and [29].

[2]Softwarization means the execution of software in a virtualized environment.

2.2. Virtualization Taxonomies Relevant Work

In this section we introduce some relevant work addressing the issues described above:

For content moving, Bogdan et al.[30] introduced a cache replacement algorithm that answers the question: "which videos to evict from the cache?". They then describe a content moving algorithm based on decision making after a cache miss. This decision consists of either keeping a copy of the item and transfer the video, or in streaming it without caching. It is based on a penalty calculated based on statistics. This technique enhances CDNs scalability but still suffers from achieving low quality of service (including delay and bandwidth) due to the penalty computation. It requires high computational overhead and extra bandwidth to calculate the instantaneous value of the content popularity. In [31] authors propose another content moving technique for fetching requested items and moving them to end users. The results showed that this approach improves network utilization and user QoS. However, it requires a predefined architecture in order to deal with NFV and SDN technologies.

For server moving or replication, authors in [32] proposed a replication algorithm called Replication Algorithm Load Balancing (RALB) to enhance videos adaptation to the available bandwidth. This approach is based on content replication in P2P systems for video on demand scenarios. Peers store some movies in order to increase network bandwidth and reduce server's load. Authors use random replication assisted with load balancing. Firstly, they introduced a centralized approach where video replication is static. Then, they added a reactive and distributed algorithm in which peers decide, based on video popularity, to either store or discard the movie. Hence, the replication is amended after any peer views the video. Therefore, this mechanism adapts to the dynamicity of nodes and peer churn. However, authors did not specify neither the used video popularity function nor the cache update mechanism which are very important in video caching. All the previous contributions did not address the virtualization issue.

For session moving, we proposed in a previous work [19], a solution using NFV concepts and SDN-Open Virtual Switch (OVS) communications, in which we live-migrate the streaming point using Video-LAN media player (VLC-server) to another location while keeping the session opened and with service continuity. This contribution does not address any optimization issue and is restricted to a global architecture design.

3. Related Work

This section highlights the relevant NFV/SDN optimization algorithms in the virtualization context. Thus, facing the difficult combinatorial optimization problem in the sense of the theory of algorithmic complexity, there are two main solutions for the Virtual Machine (VM)/Virtual Network Function (VNF) Placement Problem (PP): 1) Exact approaches (optimal algorithms) and 2) Heuristic approaches (e.g., best-fit decreasing, first-fit decreasing, genetic, and meta-heuristic algorithms). For many reasons inherent in our constraints; viz., VNF size, system and network parameters, the first solution (exact approach) is the most used in the VNF-PP's state of the art. Although one can find several interesting solutions to VM placement using heuristic approaches, very few studies addressed the aforementioned problem using an exact algorithm. We try to give an overview of these recent works.

Niels et al. [33] provide a framework for multimedia delivery CDN-based on NFV. They gave a structure of Point of Presence (PoP) deployment within the network. They used the three-layer structure (core network, aggregation layer, access layer) of the network to seek the optimal locations of cache nodes. Their work missed an optimization algorithm to measure the QoS/QoE. Further, they don't take into consideration the cooperation between CDN edge servers to serve user request. Also, the optimization model lacked many constraints such as the maximum server's storage capacity and the migration of content from one server to the optimal one.

Michele et al. [34] after presenting the state of the art of NFV, CDN, and stochastic optimization techniques, use a mixed architecture where real and virtual CDN nodes coexist. Although they present an interesting hybrid solution, their architecture assumes that the distribution of the future traffic demand is known which is not a realistic assumption.

Hendrick et al. [35] gave a model for resource allocation of VNFs within a virtualized environment (NFV Infrastructure). Cloud scaling technologies for VNF instantiation are criticized in order to prepare for NFV use cases inside the Network Service Provider (NSP). They affirm that the NFV paradigm was not designed to be used only in Data Centers (DCs). We agree with their affirmation since in data centers we don't have link or storage capacity shortage. Indeed, cloud administrators provision their deployed clouds with enough link speed, storage, and computation resources. However, it is not the case in NSP since we face dynamic constraints on network, storage, bandwidth and latency. Authors propose another hybrid solution inside the operator network that mixes between the legacy networks based dedicated hardware for each network function and the virtualized software/instance running on basic and standard hardware in NFVI environment.

The NFV resource allocation proposed by the authors gives different approaches for task and service deployment in hybrid NFV networks such as VM deployment on physical servers, and service deployment either on physical server or on VM. That model named (VNF-P) for virtual network function placement aims to find an optimal way for deploying a service chain characterized by a chain of VNFs taking into account some optimization features to minimize the overall cost of the deployment. VNF-P is a good abstraction of the deployment of NFV in an operator's network. But still, it is too general and does not address CDN specificity. Further, among the shortcomings of the proposed placement model for VNF, authors didn't consider real virtualization constraints such as storage, virtual CPU, virtual streaming costs. Moreover we think that they did not use an adequate distribution of user's requests and that they did not consider any QoE parameter or constraints to deal with user's demands variation.

Bernardetta et al. [36] highlight three aspects in the general NFV placement problem which are: 1) network modeling, 2) generic VNF placement algorithm, and 3) VNF routing problems. They raise an important network flow problem related to compression/decompression processes applied on through-traffic passing by VNF instances. This problem of VNF chaining requires high resource allocation. Further, they propose a multi-level objective function for VNF placement optimization. In fact, they present the problem of minimizing the allocated computing resources in a first stage and in a second stage, they perform a second optimization: the minimization of the maximum link utilization. The limitations of their approach are:

They used a prioritization method to solve the multi-objective optimization problem which leads to sub-optimal results especially when the cost functions are orthogonal as in their case. Moreover, authors didn't take into consideration the virtual RAM which is most important virtualization parameter especially in service migration among virtualized launched instances.

Authors have not properly configured the latency constraint in the core layer (it should be lower than the remaining network layers). Furthermore, the problem was resolved through a math-heuristic algorithm assisted by a prioritization method which has several limitations such as non-optimal cost analysis.

Mathieu et al. [37] propose a placement problem optimization for the Deep Packet Inspection (DPI) networks through designing a vDPI (virtualized DPI) solution. Moreover, they consider the strong relationship between NFV and ISPs in order to host dedicated cloud systems in ISP's Point of Presence (PoP). Their work was concentrated on where to instantiate vDPI in the network to reduce the cost and overcome challenges in the vDPI deployment process. They then gave their general placement model for vDPI cost based placement constrained by OPEX, CAPEX, and cybersecurity costs (since DPI is a security network function). Authors aimed to propose a model capable of minimizing network load and total number of deployed vDPI engines. This contribution is useful for our CDN placement problem. There are however major differences between the deployment of DPIs and CDNs. Moreover, authors assume the knowledge of the traffic flow distribution. This static parameter has limited their solution.

Literature lacks an optimization algorithm that introduces at the same time system, network, and user quality parameters. Moreover, none of the above work introduced a placement and/or migration protocol for virtual content delivery network functions. Hence, we contribute by OPAC protocol beside an optimization model.

4. OPAC: System Requirements and Design Concepts

We present an efficient solution for optimizing video streaming delivery according to the proposed use cases requirements detailed in the first subsection. Then, we design the vCDN cache algorithm.

4.1. Requirements of OPAC

We quote the following requirements that should be satisfied:

1. vCDN as a Service: It happens either from a Video on Demand (VoD) server or from end users. Video content should be cached in a vCDN deployed either in a Point of Presence (PoP), in a Home Network Gateway (HNG), in a Set Top Box (STB), or in the end user device.
2. Video Identification: After that a user requests a desired video, we should identify if the requested video is cacheable (located in a near vCDN) or not.
3. Simple Integration: Our infrastructure is based on well known NFV and SDN architecture.
4. Cooperation: Through SDN paradigm, the cache instances should be able to cooperate so as to satisfy and serve the end user's request.

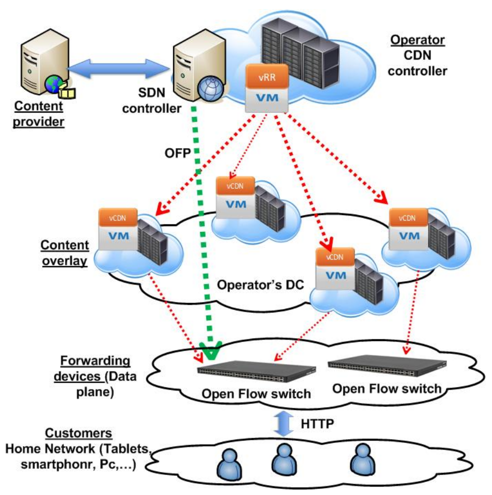


Figure 3: OPAC-based vCDN deployment architecture

5. Programmability: Cache instances should be easily Created, Retrieved, Updated and Deleted (CRUD functions).
6. Minimize Network Utilization: Caching techniques should take into account video popularity (videos that are frequently requested by end users).
7. Users' QoE: The QoE of the end users is an important parameter in the cache management.
8. Load Balancing: Video services should be placed on the best vCDN. Otherwise, the service/the vCDN should migrate elsewhere.

4.2. System design of OPAC

We build our system based on the SDN paradigm assisted with traffic engineering rules using Network Function Virtualization Infrastructure (NFVI presented in ETSI standard [20]). We apply virtualization on content delivery networks. NFV concept is used here to virtualize the CDN components (cache/stream nodes and a request router node) while SDN paradigm is used for controlling the behavior of these virtualized instances through open switch-based configurations. Beside vCDN traffic engineering, vCDN programmability, and flexibility, SDN acts as the request routing system of OPAC protocol that communicates with the CDN controller and the virtual instances created with respect to the hypervisor-based virtualization of NFV. We have implemented OPAC algorithm and we have used both paradigms SDN (with OpenFlow protocol) and NFV (via the OpenStack platform enriched by our vCDN services) [3] . Our proposed system model is shown in Fig.

[3]A first version of the software is available at: https://github.com/TelecomSudparis-RST/vIOS

3 and consists of two new entities (with respect to the SDN basic architecture [38]):

1. vCDN node: It is the software that virtualizes the CDN caching and streaming services on a standard physical server. It runs on a simple VM. vCDN nodes are distributed inside the operator's network. They act as a broker between content provider and end users.
2. Virtual Request Router (vRR): It constitutes the controlling element for a CDN network. We propose an overlay network of virtual nodes (vCDNs) instantiated by the virtual request router which acts as the vCDN controller. The vRR is hence the virtual function in the vCDN controller that redirects end users requests to the nearest (suitable) vCDN node. The request routing process is based on the OpenFlow Protocol [39] (OFP). It uses a Time To Live (TTL) interval as a cache update strategy.

As shown in Fig. 3, OPAC is used for orchestrating and creating vCDN as a service. Further, the vRR proposed here is able to redirect users requests to the *optimal* vCDN streaming node. Hence, once user requests are directed to the CDN server (VNF), the latter redirects the request to the optimal vCDN relative to the user location and other algorithm parameters.

The event sequence is that a Content Provider (CP) asks the network operator (virtualized CDN provider) to perform a cache process for its videos (referenced by URLs). URLs are classified as *cacheable* or not in the network (see Fig. 4). Moreover, each content should be cached (the most popular and requested by end user) into an optimal vCDN node (content placement problem). Therefore, we use a cache function to track the migration of videos as follows:

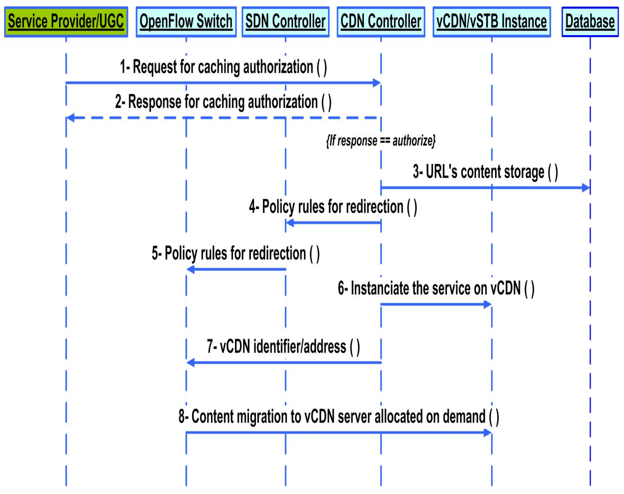
$$Cache(v, t) = vCDN_{ID} \tag{1}$$

Where $vCDN_{ID}$ is the vCDN identifier, v is the video identifier (it is often a key file) and t is the time when we place/allocate the video identified by v.

We can hence program, deploy our caching algorithm proposing better optimization to enhance the QoE/QoS of end users, decrease the load on the origin content provider (e.g., Youtube, Netflix, etc...) and let the operator host the content on its content network ($vCDN_1$, $vCDN_2$,... $vCDN_n$).

OPAC uses a Distributed Hash Tables (DHT) to index the content URLs over the vCDNs. vCDNs communicate periodically with vRR to maintain their connectivity and location.

An additional fast redirection happens as follows: if the video is cache-able then it should be declared so.



Service Provider/UGC
OpenFlow Switch
SDN Controller
CDN Controller
vCDN/vSTB Instance
Database
1- Request for caching authorization ( )
2- Response for caching authorization ( )
{If response == authorize}
3- URL's content storage ( )
4- Policy rules for redirection ( )
5- Policy rules for redirection ( )
6- Instanciate the service on vCDN ( )
7- vCDN identifier/address ( )
8- Content migration to vCDN server allocated on demand ( )

Figure 4: vCDN cache as a service

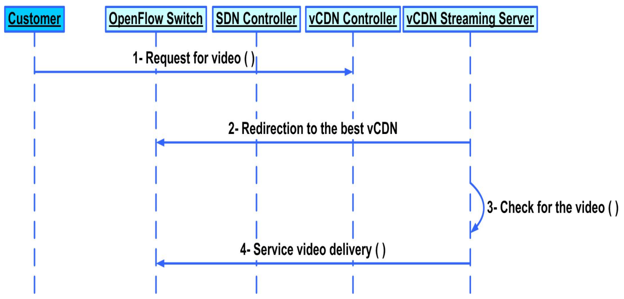


Figure 5: vCDN hit cache scenario

And then, the request is redirected based on OpenFlow rules and inserted in OpenFlow switches when the request is intercepted.

We use a greedy algorithm: a user requests a video content. If video content is cached. Then, redirect the request to an optimal vCDN node based on OPAC protocol as detailed in the next section.

Algorithm 1 summarizes the pseudo code of vCDN cache algorithm. Hereafter, we describe these main stages.

Algorithm 1 Greedy algorithm for vCDN caching

```
1: Input: $video, user, vCDN$,
2: user-request (user u, video v, vCDN $vCDN_{ID}$);
3: if $video$ $isCached$() then
4:     OPAC Protocol ( );
5: end if
```

5. OPAC Protocol

The main steps of the proposed protocol are detailed hereafter:

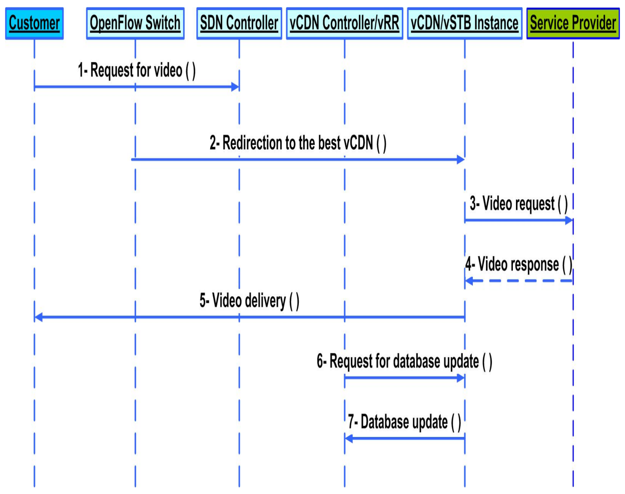


Figure 6: vCDN miss cache scenario

1. The end user i requests a video j , $video_{i,j}$.
2. Network Service Provider (NSP) assisted with SDN network, management layer and NFV deployment (see Fig. 4) redirects customer's request to the optimal vCDN cache node (the optimal placement is derived with the help of an exact optimization algorithm, formulated and detailed later).
3. If $video_{i,j}$ is located in the optimal vCDN, then deliver($video_{i,j}$) (see Fig. 5).
4. Else VoD origin server delivers the requested video to the user, and optimal vCDN node caches the video (see Fig. 6).
5. Notification of the caching controller about the mapping between the new vCDN address and video name.

For the sake of simplicity, we assume that the video service must be cached through the vCDN (we name it a vCDN cache as a service). Once the content is cached, requests from clients are dynamically redirected to the migrated vCDN (the deployment location has been proposed after executing OPAC optimization algorithm). Hence, We can enumerate the steps of the OPAC protocol as the following:

1. vCDN cache as a service (Fig. 4):
   - Authorization: Either the Service Provider (SP) or the User Generated Content (UGC) must request authorization (step 1) in order to cache a popular video item (e.g., "/video.mp4").
   - SP/UGC takes the authorization (step 2) from the Caching Controller (CC) which is part of the network operator.
   - CC initiates policy rules (PRs) for redirection (step 3) by communicating with the database server (DB), then it interacts with SDN controller (step 4) to notify it by the PRs. The latter injects those PRs in OpenFlow switch device (step 5). Then, CC instantiates (if needed regarding to the available resource) the video service on vCDN. This virtual node can be a set-top box (STB) device that supports the virtualization technique (step 6).
   - CC sends the vCDN address (@vCDN) to the requester (CP or UGC) (step 7).
   - SP/UGC migrates/pushes the desired video content to the vCDN allocated on demand (step 8).
2. Video delivery (Fig. 5 and Fig. 6):
   - Customer requests for a desired video (step 1)
   - OpenFlow physical/virtual switch redirects under the previous PRs customer's request to the optimal vCDN streaming server (step 2)
   - vCDN streaming server checks for the video (Fig. 5, step 3).
   - If desired video is located in the local cache of vCDN (hit cache scenario) then it executes service video delivery process (Fig. 5, step 4).
   - Else (miss cache scenario) it retrieves the video from the origin VoD server (Fig.4, step 3 and 4).
   - vCDN streaming server delivers the video to the end user ((Fig. 6, step 5).
   - vCDN streaming server notifies the CC in order to register the mapping between vCDN node identifier and video item (Fig.4, step 6).
   - CC updates/refreshes the database (Fig. 6, step 7).

6. OPAC Optimization Model

We propose an exact optimization algorithm that takes as an input the topology of the underlying network. It aims then to optimally place, and migrate the virtual content delivery functions (vCDN nodes) upon the virtualized infrastructure. Those virtual nodes move from one location to the optimal point in order to increase user satisfaction and decrease server and network loads. The optimization algorithm for placement, and migration is modeled, implemented, and evaluated in the next subsections.

6.1. Problem statement, constraints and main objectives

The problem considers delivering videos on demand or services, e.g., television programs, the movies, etc to a large group of users located in distant zones and served by an operator network. This operator may or may not be the service provider. Moreover, the problem supports users with different resolution requirements.

The statement is as follows: determine where to locate the VoD streaming headend, and how to migrate the virtual delivery node from one location the optimal one.

We consider different types of constraints in our virtualization process: system type constraints such as RAM, CPU, and storage, and network type constraints such as flow balance (conservation), QoE, link capacity, and server streaming constraints: maximum number of simultaneous connections per server). Recall that none of the previously described contributions considered the QoE parameter. The latter aims to link context information such as the operator delivery network to video streaming. OPAC optimization will maximize the Mean Opinion Score (MOS) of end users having different devices and connected over different technologies (ADSL, WiFi, and Ethernet). Video delivery adaptation mechanisms using utility functions [28] and MDASH (MOS Dynamic Adaptive Streaming over HTTP) can hence be supported by our work.

The objective is to minimize the cost of resources (i.e. bandwidth and migration).

The main goal of our study is to propose an algorithm and the underlying mechanisms to optimize video delivery networks. These mechanisms will guarantee the desired user's satisfaction level and on demand service optimization as:

1. Allocation mechanism and virtual delivery node migration: we introduce an optimization algorithm for vCDN allocation based on the maximum number of connections related to each file and its viewing rate (according to information calculated by counters such as DailyMotion/YouTube [4]). Our algorithm enables simple file movement between different domain vCDNs considering minimum cost of content access.
2. Service optimization mechanism (service on demand): The second step is to analyze the virtualized delivery node and migrate them to the optimal PoP headend in order to achieve a better quality of experience (QoE) and better savings for network operator resources.

This implies three levels of optimization:

1. Multi-criteria optimization: this first level aims to satisfy as maximum as possible the involved actors in the delivery process: the content provider that wants to push the maximum content to the CDN provider with minimum cost, the network operator which wants to decrease its server's load and save its resource (storage, bandwidth, ...) as possible, and the client who requests high quality of experience and better satisfaction with minimum price.
2. Network optimization: we optimize the load on the main network links and increase the operators gain (delay reduction in content streaming).
3. QoE/QoS optimization: we focus on users QoE by adapting the video streaming resolution to its context. We assume that vCDN users may require different video resolutions (Standard Definition (SD), High Definition (HD) and Ultra High Definition (UHD). The quality parameter d_v^f corresponds to the resolution requirement per each cluster of end users is defined in Table 1. The model can be easily extended to accommodate future ultra higher definition TV services like 8K.

We will detail the optimization problem constraints for vCDN migration and the objective function of our work in the next section.

6.2. Mathematical formulation

In this section we specify the parameters and the constraints that are defined and proposed in formulating the optimization/analytic model. This formulation determines the migration of vCDN nodes to the optimal location.

- OPAC Parameters: OPAC deals with system, network, and QoS/QoE parameters. We represent each by some parameters although it would be better to have them all detailed. For complexity issues, vCDN resource's size (f_{size}) (beside the other system/network parameters) is a normalized function of some relevant values that describes the size of the instance (system). Moreover, we introduce the QoS/QoE metric (d_v^f) in our optimization problem. This parameter represents the consumer demands matrix. Periodically, the network operator feeds OPAC database with the existing demands: requests for vCDN services. A demand comes from a client group ($v \in V$) and targets a vCDN

[4]YouTube MyTop100Videos:
https://www.youtube.com/user/MyTop10Videos

($f \in F$). This is to link end-users need (i.e., requesting vCDNs) with a specific throughput that is mentioned in their requests as a matching parameter between the requested quality (SD, HD, or UHD in terms of throughput) from the client group (v) and the streaming throughput from the vCDN serve (f). The algorithm tries to satisfy all the user quality of streaming requests. Including all parameters would be too complex to solve in a reasonable time.

- Decision Variables: We quote in Table 1 the main system parameters and decision variables.

  1. The binary variable x indicates the placement of the streaming headend, and its migration from one server s to another (best/optimal) location s'. It is defined as:

  $$x_{s,s',f} = \begin{cases} 1 & \text{if } f \text{ migrates to } s' \\ 0 & \text{Otherwise} \end{cases} \tag{2}$$

  2. The binary variable y indicates client request need for a vCDN service located on the optimal server. It is defined as :

  $$y^{s'}_{v,f} = \begin{cases} 1 & \text{if } v \text{ needs } f \text{ and } s' \text{ caches } f \\ 0 & \text{Otherwise} \end{cases} \tag{3}$$

  3. The binary variable $z^{v,f}_{i,j}$ indicates whether a link (i, j) is used (from i to j) to stream from a server s' replicating f (the one for which $y^{s'}_{v,f} = 1$) to client v.

Table 1: Mathematical Notation

Parameters	***Definition***
V	The set of client group nodes
S	The set of server nodes
S'	The set of optimal server nodes
$D^{s'}$	Maximum throughput of the streaming server $s' \in S'$
F	The set of vCDN nodes
f_{size}	vCDN resource's size ($vRAM, vCPU, and\ vDISK$) ($f \in F$)
$C^{s'}$	Maximum storage capacity of the server $s' \in S'$
$C_{i,j}$	Link capacity between two nodes i and j (from i to j)
d^f_v	Throughput used for streaming the vCDN functionality $f \in F$ to the client group $v \in V$. It represents the user's demand requirements ($SDTV, HDTV, UHDTV$)
$c_{s,s',f}$	The migration cost of the functionality f from s to s'
Decision variables	***Definition***
$x_{s,s',f}$	Placement and migration binary variable which indicates that f should move from from s to s'
$y^{s'}_{v,f}$	Binary variable which indicates the video hit from node v of f in server s
$z^{v,f}_{i,j}$	Binary variable indicating whether the link (i, j) is used to stream f to v

- Constraints:

  1. The binary variable y should be less than or equal to x. In fact, y equals 1 if and only if v needs f from the optimal server s' which has the functionality. This means that x should be equal to 1.

  $$\forall s' \in S', v \in V, s \in S : y^{s'}_{v,f} \leq x_{s,s',f} \tag{4}$$

  2. Only one optimal server should serve the attached node that requests the functionality f:

  $$\forall v \in V \mid d^f_v \neq 0 : \sum_{s' \in S'} y^{s'}_{v,f} = 1 \tag{5}$$

  3. The cost of streaming f by the optimal server should be less or equal to the maximum server capacity:

  $$\forall s' \in S' : \sum_{v \in V} \sum_{f \in F} y^{s'}_{v,f} \times d^f_v \leq D_{s'} \tag{6}$$

  4. The storage of the optimal server should not exceed its maximum size:

  $$\forall s' \in S' : \sum_{s \in S} \sum_{f \in F} x_{s,s',f} \times f_{size} \leq C^{s'}_{max} \tag{7}$$

  5. Flow balance or conservation constraint (which is the laws of Kirchhoff) between the optimal server s' and the client node v should be as the following:

  $$\sum_j z^{v,f}_{i,j} - \sum_j z^{v,f}_{j,i} = \begin{cases} 0 & \text{if } i \neq v, i \neq s' \\ y^{s'}_{v,f} & \text{if } i = s' \\ -1 & \text{if } i = v \end{cases} \tag{8}$$

  This is the network flow constraint. The sum of incoming flows must be equal to the outgoing ones.

  6. Link capacity between source i and sink j should not exceed its maximum:

  $$\forall i, j \in V \cup S' : \sum_{v \in V} \sum_{f \in F} z^{v,f}_{i,j} \times d^f_v \leq C_{i,j} \tag{9}$$

6.3. Mono objective resolution

The objective function: the migration cost (transit) as formulated in equation (10).

- Objective function:

$$\min \sum_{s' \in S'} \sum_{s \in S, f \in F} x_{s,s',f} \times c_{s,s',f} \tag{10}$$

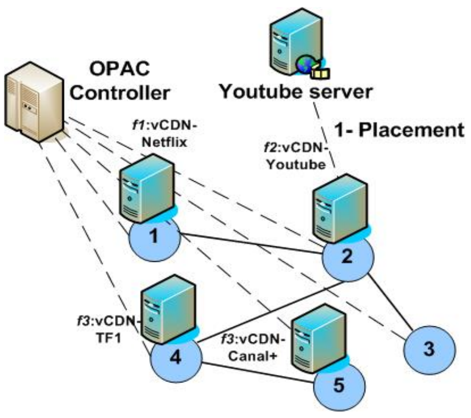
where $c_{s,s',f}$ is the migration cost:

$$c_{s,s',f} = \alpha \times f_{size} \tag{11}$$

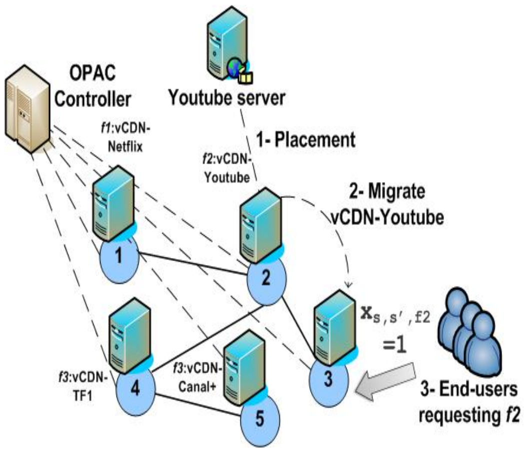
Where α is a parameter depending on the position of s' and the position of s (the server initially containing f before migration). α depends also on the operator policy (e.g., the internal policy of the operator like configuration policy).

- Optimization technique: The OPAC-based optimization domain targets a set of large size vCDN nodes to be placed in network operator snapshot (with a relatively low number of NFV servers client group nodes). The goal of OPAC is to intelligently (and dynamically) place and migrate the set of vCDN nodes in response to client group demands in order to increase the in-network caching performance (and hence QoE) while minimizing the total migration cost function as formulated in Eq. 10. The migration problem of vCDN to the optimal placements is an Integer Linear Problem (ILP) that can be solved within a few seconds. It (the migration) is modeled through branch and bound optimization technique (method) that minimizes the network virtualization cost (total migration cost) among a set of candidate solutions that maximize the end user quality of experience. Starting from the previous system, network and content quality constraints, OPAC searches the optimal data center locations, where the binary migration vector $x_{s,s',f}$(defined above) equals to 1.

- Analytical procedure (OPAC-based decision example): For the sake of clarity, we propose an example based on OPAC procedure for vCDN placement/migration context as shown in Fig. 7. The algorithm is as follows: 1) As an input, OPAC controller gathers SDN/NFV architectural information such as the initial placement of vCDNs ($f_1 \in s_1$, $f_2 \in s_2$, etc.) and all necessary dynamic parameters (system capacity, network capacity, throughput, location, etc.) as shown in Fig. 7a representing the network state before migration, 2) Then, it needs also a prediction of end-user demands to execute OPAC and decide about the potential points of deployment of the requested vCDNs, and 3) as a result, end-users requesting vCDN YouTube®, to watch a popular video in a live event for example, will be redirected to the videos hosted in the new optimal calculated location as depicted in Fig.7b representing the network state after migration. Recall that the network operator hosting the vCDN of YouTube content provider migrates the vCDN cache/stream node to an optimal PoP where resources are available and content quality streamed are satisfied.



(a) Before migration



(b) After migration

Figure 7: Example based OPAC-analytical procedure for vCDN placement

Note that in a micro cache deployment (large scale) scenario, OPAC is still acceptable for vCDN placement/migration application. This is because the algorithm run time is in term of seconds. Therefore there is no need to develop a heuristic algorithm. OPAC algorithm is proposed to be executed in a virtual CDN controller/manager entity (vRR). It has to control/manage/orchestrate the VNFs running virtual CDN nodes. This placement is executed after three main triggers:

- System constraints (through f_{size} and $C^{s'}$ parameters).
- Network constraint (through $C_{i,j}$, $D^{s'}$, and d_v^f parameters).
- Service Level Agreements (SLA): It is QoS/QoE constraint through a contract between the content provider (the customer of the vCDN solution) and the network operator (the server or the provider of the NFV servers) (d_v^f resolution requirement per each cluster of end users). It (SLA) is considered through a valid range of streaming qualities: SD, HD, and UHD of each couple (client group–vCDN). The optimization process uses this requirement and the result satisfies the SLA.
- Storage/bandwidth prediction: OPAC makes the migration and assumes that the network and system parameters are available (through OpenStack dashboard). We mean by prediction that those parameters may change and cannot be known in advance. Recall that data centers are highly dynamic and shared, so live metrics and parameters are used from the virtual infrastructure manager (e.g., OpenStack data-centers) and update at the same time the OPAC database.

In the current implementation version of OPAC, triggers are not used to execute the optimization. However, as we use a database for our real-time parameters, all the dynamic values can be annotated with a triggering threshold that re-launches the process. The user demands, system parameters and network load are the potential triggers. Moreover, in this version OPACv1, we dont consider individual user demands, only groups (represented by the d_v^f matrix). Second we envision the execution of the algorithm in a predictive scenario. Till now, these demands are a forecast of client groups ($v \in V$). The consumer demands are fictional and have been initialized in the OPACv1 database and have the corresponding vCDN request list, PoP request list, etc. New demands are treated in the next optimization round. The optimization operation should be executed in accordance with the data aggregation frequency

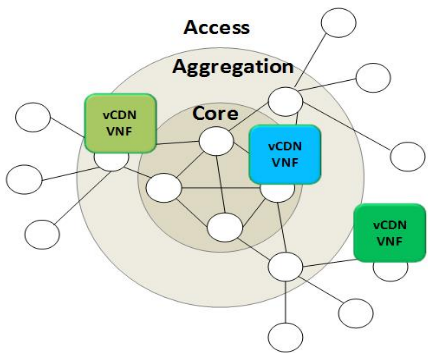


Figure 8: Network topology used for CDNaaS

of the demands and the system/network load based on OpenStack.

7. OPAC: Optimization Evaluation

We evaluate our work in a network that obeys the 3-tier layers architecture (access, aggregation and core layer)[5]. Fig. 8 shows the network topology used for the evaluation. Further, for the interest of assessing the efficiency of OPAC algorithm, we used CPLEX [40] as an optimization tool.

We note that servers may be deployed in any network layer while client nodes are deployed only in the access network. We mention also that vCDN cache nodes are deployed on top of the physical servers that among NFVI and migrate from the source s to the optimal server s'.

Further, we define the NFV cost and the placement time as follows:

- The NFV cost is defined as follows:

$$NFV\ cost = \sum_{s \in S, s' \in S'} \sum_{f \in F} x_{s,s',f} \times c_{s,s',f} \tag{12}$$

- The migration time is defined as follows:

$$Placement\ time = \sum_{s \in S, s' \in S'} \sum_{f \in F} x_{s,s',f} \times f_{size} \times \max_{(i,j) \in P_{s,s'}} \frac{1}{C_{i,j}} \tag{13}$$

Where $P_{s,s'}$ is a given path from s to s'. In fact we are here assuming that placement is done in sequential way. If placement is performed in parallel, then the placement time would be given by

[5]https://tools.ietf.org/pdf/draft-bagnulo-nfvrg-topology-00.pdf.

$\max_{s' \in S', s \in S, f \in F} \left(x_{s,s',f} \times f_{size} \times \max_{(i,j) \in P_{s,s'}} \frac{1}{C_{i,j}} \right)$. However, we will only consider (13).

Furthermore, for the sake of assessing the efficiency of OPAC algorithm, three main cost factors are defined as follows:

$$vCache\ cost = \frac{\sum_{s \in S} \sum_{s' \in S', f \in F} x_{s,s',f} \times f_{size}}{\sum_{s' \in S'} C^{s'}} \tag{14}$$

$$vStream\ cost = \frac{\sum_{s' \in S'} \sum_{v \in V} \sum_{f \in F} y^{s'}_{v,f} \times d^f_v}{\sum_{s' \in S'} D^{s'}} \tag{15}$$

$$Link\ utilization\ cost = \frac{\sum_{v \in V} \sum_{f \in F} z^{v,f}_{i,j} \times d^f_v}{\sum_{i,j \in V \cup S'} C_{i,j}} \tag{16}$$

Where the cost UNIT represents the amount of system resources (vCache) or network resources (vStream and Link utilization cost) consumed when migrating a vCDN from a server to another one.

Hereafter, we evaluate the OPAC under different key performance indicators introduced above to assess impact of increasing virtual content delivery number, client group node number. Besides, we evaluate the virtual content delivery resolution impact, delivery capacity impact and delivery storage impact.

7.1. Virtual content delivery number impact

Firstly, we evaluate in this subsection the impact of increasing the vCDN number. For this, we choose for the three-tier network the following input parameters: N = 6 for client (as aggregation requests) node, L = 21 for links (number of aggregated edges), S = 10 for severs (as data center zone).

Fig. 9 shows the migration time needed for placing vCDN in the optimal place taking into consideration the previous NFVI constraints (like computing, storage, and streaming resources) that are strongly related to CDN caching service and virtualization cost. In Fig. 10, we measure the overall NFV cost which means the overall cost of the migration including all the aforementioned constraints as a function of the vCDN number. We remark that the total NFV cost is linearly increasing. The vCDN placement time has an increasing slope for a vCDN number ranging from 3 to 9. This is due to the client node distribution which is not uniform.

In Fig. 11, cost unit [6] is plotted against the vCDN number. The storage cost for vCDNs in the network is constant, the virtual streaming cost and the link capacity cost are variable due to the demand variation of the end user.

[6] Cost unit is defined by the utilization percentage of the available resource.

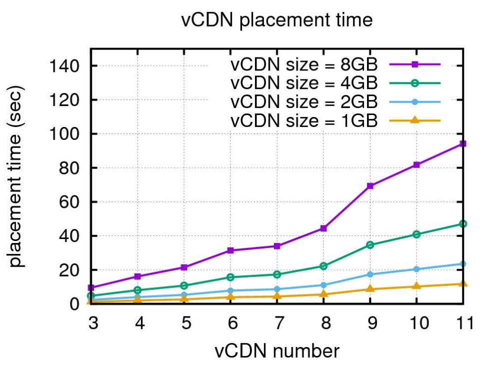


Figure 9: vCDN placement cost

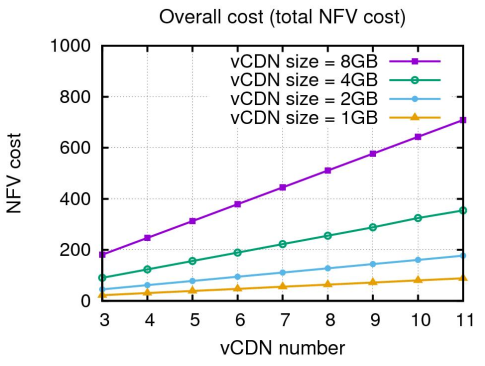


Figure 10: Overall NFV cost of vCDN migration

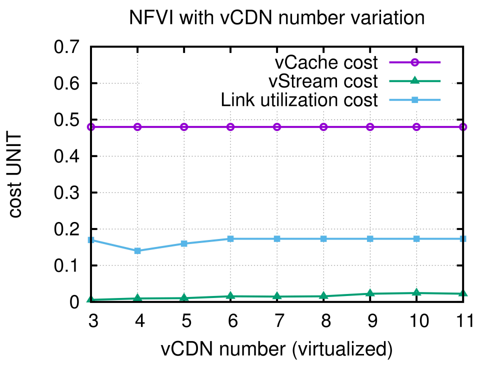


Figure 11: NFVI cost with vCDN number variation

7.2. Client node number impact

Secondly, we evaluate the impact of the client number on the overall cost taking into account the storage, network and streaming features. Therefore, we have fixed the total number of the virtualized content delivery networks that should be placed within the network (VNF deployed number = 10), and the same previous three-tier network.

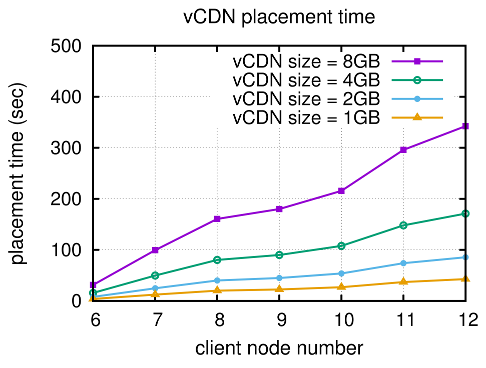


Figure 12: vCDN placement cost

In Fig. 12, we plot the placement time against client node number. In Fig. 13, we represent the NFV cost needed for this placement.

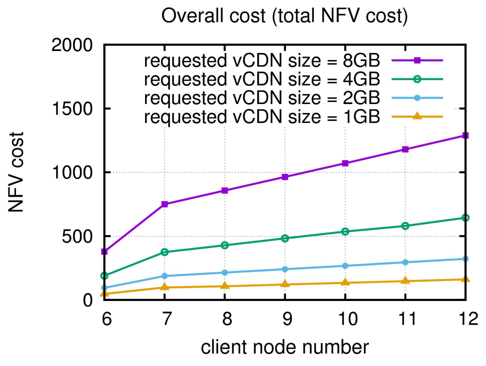


Figure 13: Overall NFV cost of vCDN migration

In Fig. 14, we measure the virtual caching cost. It is in increasing till it reaches a constant value. We represent also link utilization cost, and we conclude that the streaming cost is still increasing because it is affected by the number of location from where clients receive the video service.

7.3. *Virtual content delivery resolution impact*

Thirdly, we evaluate the impact of virtual content delivery resolution. In Fig. 15, we plot the cost of streaming against vCDN number. The vCDN caching/streaming nodes may serve SD, HD, or UHD content resolution to the clients. Recall that vCDN nodes serve the clients according to their initial requirements. Virtual streaming cost is increasing in each category when vCDN number increase. Moreover, the additional cost needed for streaming high resolution video is significant compared to lower definition ones. There is

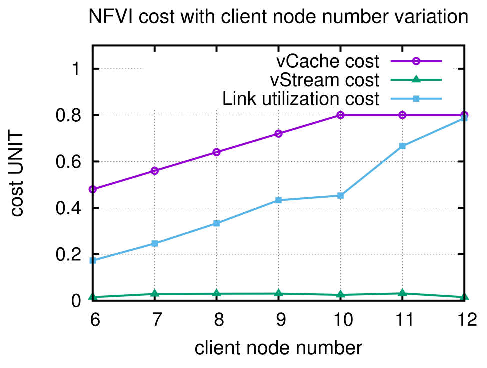


Figure 14: NFVI cost with client node number variation

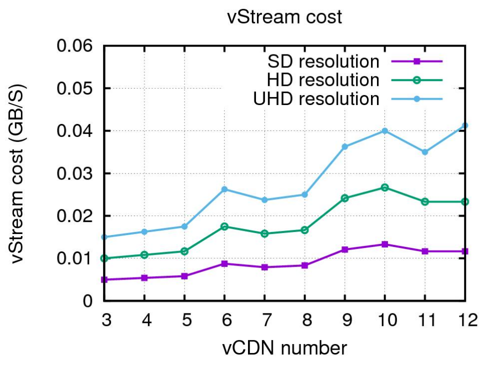


Figure 15: Stream resolution

about 100 % increase in cost from SD to HD and from HD to UHD.

In Fig. 16, we plot the cost of link utilization against vCDN number. Moreover, this link utilization cost increases with each content resolution when the vCDN number increases. As in virtual streaming cost (Fig. 15), the additional link utilization cost needed for streaming high resolution video is significant. But, the main observation is that cost remains constant when increasing the vCDN number.

7.4. *Delivery capacity impact*

Fourthly, in this subsection we show the impact of increasing the server's throughput on the cost of caching, streaming vCDNs and on the link utilization. Fig. 17 includes 3 curves. These curves plot the cost unit against the server's throughput. We assume that clients may request either SD, HDV, or UHDV video quality. Regardless of the quality requested by clients, the caching cost is constant. There is no additional cost to deliver either SD, HD, or UHD to the client in terms of a caching cost in (GB). However, the streaming cost decreases when

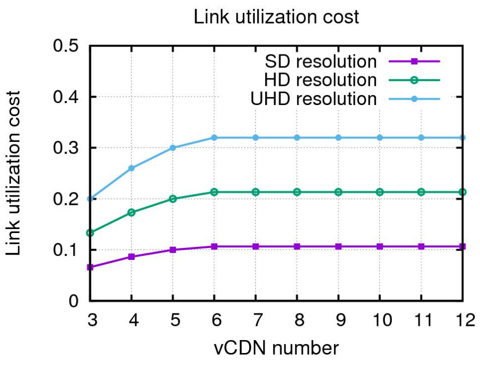


Figure 16: Link utilization cost with different vCDN resolution

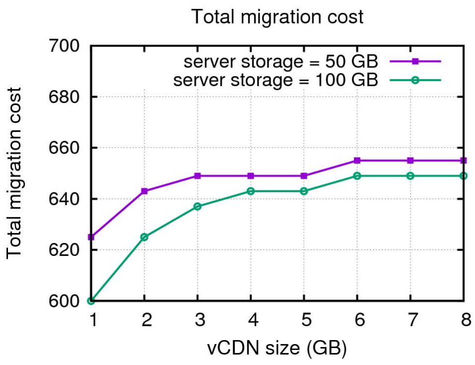
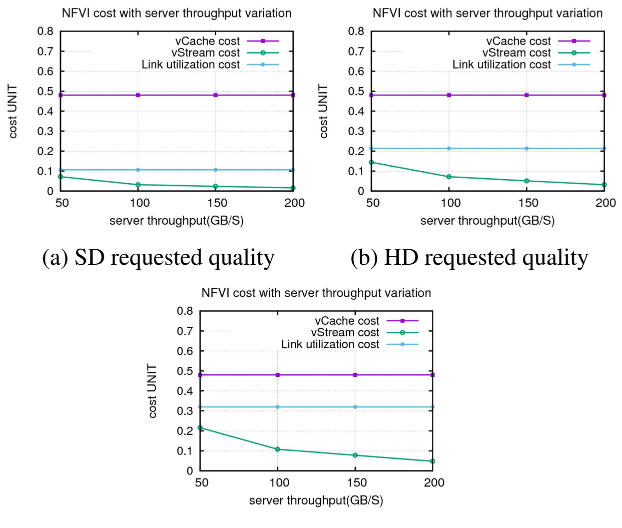


Figure 18: Total migration cost



(a) SD requested quality (b) HD requested quality

(c) UHD requested quality

Figure 17: NFVI costs with server's throughput variation

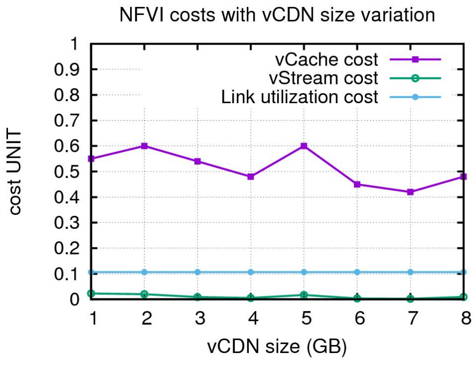


Figure 19: NFVI costs with vCDN size variation

the server throughput increases. This leads to more efficient streaming and better QoE. This streaming cost increases when clients request high vCDN quality. Link utilization cost on the other hand is constant but it increases with high requested quality (from SD through UHD). In conclusion, in both cost factors (caching and streaming), the increment to get a better quality is not significant. It is only about 10 % .

7.5. Delivery storage impact

Fifthly, the impact of delivery storage is evaluated. In Fig. 18, we plot the total migration cost needed for migrating a vCDN with random demand vector per client. The additional cost needed for vCDN migration decreases when we increase the server storage. This is an encouraging result since storage is an important service. Storage could be extended up the end user terminal. Fig. 19 plots our three proposed costs: vCache, vStream, and Link utilization against vCDN size. Although, the vCDN size increased, there are no important variations in the three cost factors.

8. OPAC Integration

Practically, the proposed optimization algorithm has to be integrated in a vCDN controller (as seen Fig. 7). It interacts with the vCDN manager software in a legacy NFV-MANO framework [41] and under the context of DVD2C project [18]. Further, as the migration problem deals with different heteroclite parameters and variables, a clear view about how they are extracted and reflected on an SDN/NFV framework is necessary.

For the sake of simplicity, the two main use cases needed for integrating OPAC algorithm by the network operator are quoted as the following:

- Use case 1: The network operator checks the current placement of vCDNs. To do this, he queries a Database using SQL structured language and an NFV/SDN controllers (e.g., OpenStack

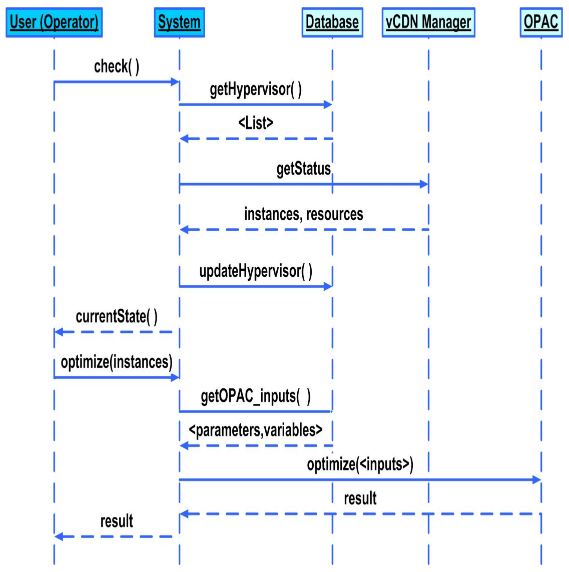


Figure 20: Sequence diagram for OPAC integration

and/or Opendaylight [21]) using API interfaces. OpenStack horizon and Opendaylight DLUX providing respectively the system and network resource information needed to complete the operator database.

- Use case 2: The network operator requests for the vCDN optimization placement/migration result using OPAC according to the actual network scale.

From an operator perspective, integrating the proposed algorithms may follow the sequence diagram represented by the Fig. 20. Indeed, the main stakeholders are:

- User (Operator): Operator of the infrastructure, deciding the migration or not of a vCDN instance according to the current state of the environment and the result given by the optimization algorithm OPAC.
- System: This system, offering the Users an interaction with the OPAC algorithm inputs and results. It still maintains the operator database updated by the necessary information about the operator's servers (*getHypervisor()*) and the provider's vCDNs (*getStatus()*).
- Database: It contains the information about the SDN/NFV infrastructure status and values. In this case, it is an SQLite database.
- vCDN Manager: It takes care of the interaction with the software managers running in the architecture, to poll from them the required information for the OPAC algorithm. It will use internally API calls to the different software managers (OpenStack Horizon, Opendaylight, etc.).
- OPAC Algorithm: Implementation of the optimization algorithm that optimizes the vCDN placement and migration based on exact/heuristic optimization approaches. A file is given as an input and a file is returned as an output of the selected algorithm. The algorithm selection depends mainly on the network scale (e.g., OPAC versions).

For simplicity, when the user (network operator) executes the optimize command, the system fetches the operator databases to get the required information (*getOPAC_input*())in order to launch in turn the optimize command. Then, OPAC algorithm decides where to place and migrate the vCDNs and provide the system/operator the result. Finally, the system executes the migration process according to our results and releases the dedicated resources in case of Service Level Agreement (SLA)-expiration/vCDN-delete-request.

Recall that the proposed algorithms involved also the content provider who wanted to rent a cloud of vCDN for its customers. Therefor, from a content provider perspective, the main exchanges between the content provider, the vCDN manager and the user (operator) are depicted:

- The content provider (e.g., YouTube) requests a vCDN creation during a specific time (e.g., 2 hours) and with a specific QoE (e.g., excellent) and to cover a specific region (e.g. Paris).
- The network operator, representing the owner of the NFV infrastructure, checks the current state of its NFV resources and call the proposed optimization algorithms through the provided *system*.
- This system interacts with a operator database server to retrieve the required information about network topology, NFV resources and update another database dedicated to the placement/migration decision.
- The NFV Infrastructure (NFVI) administrator interacts with the decision's database to migrate the resources while keeping the signed SLA between the content provider and the network operator valid and standing.
- Optionally, the NFVI administrator may let the content provider as the manager of the video content.

- The decision database is updated by the operator for security issues.
- In case of SLA expiration (e.g., covering time expiration, vCDN delete-request, etc..), the operator releases the dedicated NFV resources.

9. OPAC: Comparisons

In this section, we introduce two main comparisons. First, we compare between OPAC (our optimal solution) and a non optimal approach. In the second, we compare between OPAC and the related work in two ways: 1) a comparison between OPAC and Bernadetta et al.'s work in term of link utilization cost, and 2) a comparison between OPAC and the state-of-the-art.

9.1. Comparison between OPAC and non optimal migration algorithm

The OPAC algorithm as detailed in Section 5 searches for the optimal placement to cache vCDN nodes according to the network constraints and users' demands. The network and system constraints are deterministic parameters and can be measured precisely over time. However users' demands can only be predicted as it is difficult to determine a priori what content the user will decide to watch next. Hence, some predictions can be erroneous and lead to non optimal placement of content. Thus, the optimal and the non-optimal definitions for our implemented algorithm are defined as follows:

$$x_{s,s',f} = \begin{cases} 1 & \text{if } f \text{ migrates to } s' \text{ (optimal)} \\ 0 & \text{if } f \text{ migrates to } S \setminus \{s'\} \text{ (non optimal)} \end{cases} \tag{17}$$

In Fig. 21 , we made a comparison in terms of the total virtualization cost (accumulated percentage of delay (normalized resource costs used in virtualization of vCDNs))[7] between our optimal algorithm and the non optimal one. It is clear that our exact and optimal allocation/migration algorithm outperforms the non optimal one. The operator's gain which is proportional to the reduction in delay is therefore obvious.

In Fig. 22, we plot the computation/execution time gained after the allocation/migration process. The gain is the reduction in delay if our optimal solution is adopted compared to the non optimal approach. The latter is obtained if the target PoPs that will host the virtual

[7]Total virtualization cost is defined by the total utilization of the virtual resources (vCache, vStream, and Link utilization).

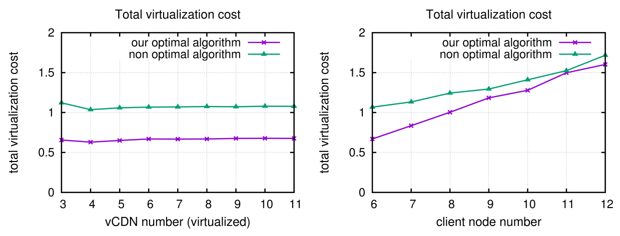


Figure 21: comparison between the optimal and the non optimal algorithm using total virtualization cost parameter and under vCDN/client node variation

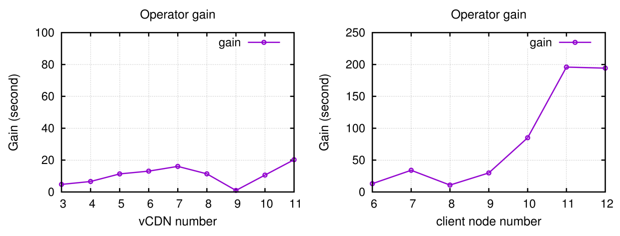


Figure 22: Operator gain after executed the optimal migration algorithm comparing to the non optimal

CDN is non optimal. It shows that the reduction in delay is noticeable and reaches 20 *sec* for $vCDNnumber = 11$. This can lead to a better QoE. However, the reduction delay is very important when client node number is increasing and reaches 3 *minutes* at $v = 12$. Recall that v represents an average demand of a large group of subscribers and a vCDN serves a large number of subscribers as standardized by $(IETF)$[8] .

9.2. Comparison between OPAC and related work

9.2.1. Comparison between OPAC and Bernadetta et al. work

Network link utilization is a relevant metric. Thus, to evaluate the network performance of OPAC from the telecommunication operator perspective, we compared our solution with Bernadetta et al [36], who used the same network and NFV parameters.

At first glance, we quote the difference between our approach and Bernadetta approach as follows:

- First of all, Bernadetta protocol uses three flow balance constraints and imposes a single path flow balance. We believe that this increases the link utilization and causes bottlenecks in the network links (congestion).

[8]https://tools.ietf.org/pdf/draft-bagnulo-nfvrg-topology-00.pdf.

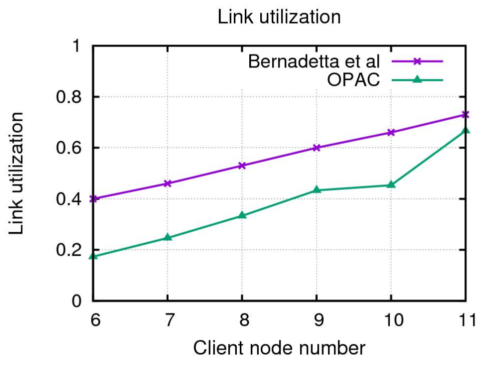


Figure 23: Comparison between OPAC and Bernadetta et al work

- The second difference is that Bernadetta algorithm includes compression which we believe is unnecessary since it can be replaced by the VNF size.

- The third difference is that Bernadetta does not use any quality of experience parameter while our approach uses a dynamic one according to user's demand requirement.

  Moreover, in Fig. 23, we plot the link utilization cost which is clearly greater than our approach. Thus, their proposed algorithm is still efficient for placing but can not deal with user's demand dynamicity ($d_v^f = 10$). In the simulation, we kept the same parameter used by Bernadetta (2 VNF type, node capacities).

9.2.2. Brief comparison between OPAC and the state-of-the-art

We show in Table 2 a comparison between our optimization approach and a few recent work:

- Authors in [33][34][35][36][37] introduced either system or network metrics in order to model the problem of placement of a VNF in general. However, we introduced in OPAC both: system and network metrics.
- Authors also did not introduce any parameter characterizing users' requirement. Therefore, we introduced an additional parameter to do this.
- Authors also tried to solve an initial placement problem of a VNF through static and offline optimization, while our optimization solves the placement problem and adds the migration context in dynamic way and while the vCDN service is running.

Table 2: Comparison between OPAC and state-of-the-art

Work	***Metrics***	***Limitations***
Niels et al. [33]	Bandwidth, deployment cost	No virtual node cooperation, no QoE parameters, missed capacity and RAM constraints
Michele et al. [34]	Usage cost	Known traffic distribution (static), unreliable
Hendrick et al. [35]	Deployment cost, service request	Virtual CPU, virtual storage, virtual capacity, and QoE constraints are missed
Bernardetta et al. [36]	Allocated computing resources, link utilization	Prioritization method with orthogonal cost functions, virtual RAM and QoE constraints are missed
Mathieu et al. [37]	CAPEX, server's load	Offline measurement, and no migration
OPAC algorithm	System, network, and quality metrics	No limitations

- In OPAC, we introduced a virtual cache cost since our VNF (vCDN) is dedicated to streaming/caching services.

10. Conclusion

This paper presents a new optimization technique for cache distribution with underlying virtualization tools. We proposed to integrate new constraints such as QoE, VNF mobility and system requirements. For this purpose, a redirection/cache/update protocol that defines the messages exchanges between end-user, virtual CDN, operator, and service provider was proposed. Furthermore, OPAC, an optimization algorithm was modeled, implemented and evaluated when demand on different content changes. It gave a short execution time (i.e., the running time was in terms of seconds). Different comparisons for our placement solution and the state-of-the-art are conducted. The results show a net improvement in cache update. We also evaluate the update performance if the cache prediction is slightly imprecise. Our Cache placement algorithm still gives satisfactory results. Future work can be conducted on the cache replacement/prediction techniques combined with optimization and learning algorithms.

11. Acknowledgment

This work is supported by the French FUI-18 call for DVD2C project (*Diffusion de Videos sur technologie IP en mode aDaptatif*, *assistee par Cloud de CDN*(*s*) *virtualise*(*s*)) [18].

[1] K. Johnson, J. Carr, M. Day, M. Kaashoek, The measured performance of content distribution networks, Comput. Commun. 24 (2) (2001) 202–206. doi:10.1016/S0140-3664(00)00315-7.
URL http://dx.doi.org/10.1016/S0140-3664(00)00315-7

[2] S. Pallickara, G. Fox, Enabling hierarchical dissemination of streams in content distribution networks, Concurrency and Computation: Practice and Experience 24 (14) (2012) 1594–1606. doi:10.1002/cpe.1909.
URL http://dx.doi.org/10.1002/cpe.1909

[3] B. Frank, I. Poese, Y. Lin, G. Smaragdakis, A. Feldmann, B. Maggs, J. Rake, S. Uhlig, R. Weber, Pushing cdn-isp collaboration to the limit, SIGCOMM Comput. Commun. Rev. 43 (3) (2013) 34–44. doi:10.1145/2500098.2500103.
URL http://doi.acm.org/10.1145/2500098.2500103

[4] G. Haílinger, F. Hartleb, Content delivery and caching from a network provider's perspective, Comput. Netw. 55 (18) (2011) 3991–4006. doi:10.1016/j.comnet.2011.07.026.
URL http://dx.doi.org/10.1016/j.comnet.2011.07.026

[5] M. T. Diallo, H. Moustafa, H. Afifi, K. U. R. Laghari, Quality of experience for audio-visual services, in: UP-TO-US '12 Workshop : User-Centric Personalized TV ubiquitOus and secUre Services, Fraunhofer FOKUS, Berlin, Germany, 2012, pp. 299–305.
URL https://hal.archives-ouvertes.fr/hal-00768322

[6] M. T. Diallo, N. Marechal, H. Afifi, A hybrid contextual user perception model for streamed video quality assessment, in: Proceedings of the 2013 IEEE International Symposium on Multimedia, ISM '13, IEEE Computer Society, Washington, DC, USA, 2013, pp. 518–519. doi:10.1109/ISM.2013.104.
URL http://dx.doi.org/10.1109/ISM.2013.104

[7] Akamai, Content delivery network (Jun. 2016).
URL https://www.akamai.com

[8] Amazon, Amazon cloudfront (Jun. 2016).
URL http://www.cdnplanet.com/cdns/cloudfront/

[9] Google, Google global cache-peering and content delivery (Jun. 2016).
URL https://peering.google.com/about/ggc.html

[10] L. M. Contreras, P. Doolan, H. Lønsethagen, D. R. López, Operational, organizational and business challenges for network operators in the context of sdn and nfv, Comput. Netw. 92 (P2) (2015) 211–217. doi:10.1016/j.comnet.2015.07.016.
URL http://dx.doi.org/10.1016/j.comnet.2015.07.016

[11] R. Jain, S. Paul, Network virtualization and software defined networking for cloud computing: a survey, IEEE Communications Magazine 51 (11) (2013) 24–31. doi:10.1109/MCOM.2013.6658648.

[12] W. Ding, W. Qi, J. Wang, B. Chen, Openscaas: an open service chain as a service platform toward the integration of sdn and nfv, IEEE Network 29 (3) (2015) 30–35. doi:10.1109/MNET.2015.7113222.

[13] T. Wood, K. K. Ramakrishnan, J. Hwang, G. Liu, W. Zhang, Toward a software-based network: integrating software defined networking and network function virtualization, IEEE Network 29 (3) (2015) 36–41. doi:10.1109/MNET.2015.7113223.

[14] J. Martins, M. Ahmed, C. Raiciu, V. Olteanu, M. Honda, R. Bifulco, F. Huici, Clickos and the art of network function virtualization, in: Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation, NSDI'14, USENIX Association, Berkeley, CA, USA, 2014, pp. 459–473.
URL http://dl.acm.org/citation.cfm?id=2616448.2616491

[15] J. Matias, J. Garay, N. Toledo, J. Unzilla, E. Jacob, Toward an sdn-enabled nfv architecture, IEEE Communications Magazine 53 (4) (2015) 187–193. doi:10.1109/MCOM.2015.7081093.

[16] H. Farhady, H. Lee, A. Nakao, Software-defined networking: A survey, Computer Networks 81 (2015) 79 – 95. doi:http://dx.doi.org/10.1016/j.comnet.2015.02.014.
URL http://www.sciencedirect.com/science/article/pii/S1389128615000614

[17] M. Garca-Valls, T. Cucinotta, C. Lu, Challenges in real-time virtualization and predictable cloud computing, Journal of Systems Architecture 60 (9) (2014) 726 – 740. doi:http://dx.doi.org/10.1016/j.sysarc.2014.07.004.
URL http://www.sciencedirect.com/science/article/pii/S1383762114001015

[18] Dvd2c project (2016).
URL https://dvd2c.cms.orange-labs.fr/

[19] H. Ibn-Khedher, E. Abd-Elrahman, H. Afifi, J. Forestier, Network issues in virtual machine migration, in: Networks, Computers and Communications (ISNCC), 2015 International Symposium on, 2015, pp. 1–6. doi:10.1109/ISNCC.2015.7238579.

[20] E. G. N. . V1.1.1, Network functions virtualization (nfv); architectural framework (2013).
URL http://www.etsi.org/deliver/etsi_gs/nfv/001_099/002/01.01.01_60/gs_nfv002v010101p.pdf

[21] Opendaylight, Opendaylight platform (Jun. 2016).
URL https://www.opendaylight.org/start

[22] E. G. N. V1.1.1, Network functions virtualization (nfv); use cases (2013).
URL http://www.etsi.org/deliver/etsi_gs/nfv/001_099/001/01.01.01_60/gs_nfv001v010101p.pdf

[23] F. Z. Yousaf, J. Lessmann, P. Loureiro, S. Schmid, Softepc - dynamic instantiation of mobile core network entities for efficient resource utilization, in: 2013 IEEE International Conference on Communications (ICC), 2013, pp. 3602–3606. doi:10.1109/ICC.2013.6655111.

[24] T. Taleb, M. Corici, C. Parada, A. Jamakovic, S. Ruffino, G. Karagiannis, T. Magedanz, Ease: Epc as a service to ease mobile core network deployment over cloud, IEEE Network 29 (2) (2015) 78–88. doi:10.1109/MNET.2015.7064907.

[25] T. Taleb, A. Ksentini, A. Kobbane, Lightweight mobile core networks for machine type communications, IEEE Access 2 (2014) 1128–1137. doi:10.1109/ACCESS.2014.2359649.

[26] V. K. Adhikari, Y. Guo, F. Hao, M. Varvello, V. Hilt, M. Steiner, Z. L. Zhang, Unreeling netflix: Understanding and improving multi-cdn movie delivery, in: INFOCOM, 2012 Proceedings IEEE, 2012, pp. 1620–1628. doi:10.1109/INFCOM.2012.6195531.

[27] M. T. Diallo, F. fieau, E. Abd-Elrahman, H. afifi, Utility-based Approach for Video Service Delivery Optimization, ICSNC 2014: International Conference on Systems and Network Communication (2014) 5–10.
URL https://hal.archives-ouvertes.fr/hal-01077552

[28] M. T. Diallo, Quality of experience and video services adaptation, Ph.D. thesis, thse de doctorat dirige par Afifi, Hossam Informatique et tlcommunications Evry, Institut national des tlcommunications 2015 (2015).
URL http://www.theses.fr/2015TELE0010

[29] P. Georgopoulos, M. Broadbent, A. Farshad, B. Plattner, N. Race, Using software defined networking to enhance the delivery of video-on-demand, Computer Communications 69 (2015) 79 – 87. doi:http://dx.doi.org/10.1016/j.comcom.2015.06.015.
URL http://www.sciencedirect.com/science/article/pii/S0140366415002315

[30] B. Carbunar, M. Pearce, V. Vasudevan, M. Needham, Predictive caching for video on demand cdns, in: Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE, 2011, pp. 1–5. doi:10.1109/GLOCOM.2011.6133574.

[31] P. Georgopoulos, M. Broadbent, B. Plattner, N. Race, Cache as a service: Leveraging sdn to efficiently and transparently support video-on-demand on the last mile, in: 2014 23rd International Conference on Computer Communication and Networks (ICCCN), 2014, pp. 1–9. doi:10.1109/ICCCN.2014.6911775.

[32] Y. Zhou, T. Z. J. Fu, D. M. Chiu, On replication algorithm in p2p vod, IEEE/ACM Trans. Netw. 21 (1) (2013) 233–243. doi:10.1109/TNET.2012.2196444.
URL http://dx.doi.org/10.1109/TNET.2012.2196444

[33] N. Bouten, J. Famaey, R. Mijumbi, B. Naudts, J. Serrat, S. Latr, F. D. Turck, Towards nfv-based multimedia delivery, in: 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), 2015, pp. 738–741. doi:10.1109/INM.2015.7140364.

[34] M. Mangili, F. Martignon, A. Capone, Stochastic planning for content delivery: Unveiling the benefits of network functions virtualization, in: 2014 IEEE 22nd International Conference on Network Protocols, 2014, pp. 344–349. doi:10.1109/ICNP.2014.56.

[35] H. Moens, F. D. Turck, Vnf-p: A model for efficient placement of virtualized network functions, in: 10th International Conference on Network and Service Management (CNSM) and Workshop, 2014, pp. 418–423. doi:10.1109/CNSM.2014.7014205.

[36] B. Addis, D. Belabed, M. Bouet, S. Secci, Virtual network functions placement and routing optimization, in: Cloud Networking (CloudNet), 2015 IEEE 4th International Conference on, 2015, pp. 171–177. doi:10.1109/CloudNet.2015.7335301.

[37] M. Bouet, J. Leguay, V. Conan, Cost-based placement of vdpi functions in nfv infrastructures, in: Network Softwarization (NetSoft), 2015 1st IEEE Conference on, 2015, pp. 1–9. doi:10.1109/NETSOFT.2015.7116121.

[38] O. N. Foundation, Software-defined networking definition (2013).
URL https://www.opennetworking.org/sdn-resources/sdn-definition

[39] O. N. Foundation, Openflow (2013).
URL https://www.opennetworking.org/sdn-resources/openflow

[40] IBM, Ibm ilog cplex optimization studio community edition, version 12.5 (2015).
URL https://www-01.ibm.com/software/websphere/products/optimization/cplex-studio-community-edition/

[41] E. G. N.-M. . V1.1.1, Network functions virtualization (nfv); management and orchestration (2014).
URL http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_nfv-man001v010101p.pdf

Annex A: OPAC Platform [9]

[9] https://github.com/TelecomSudparis-RST/vIOS/tree/master/bin/CPLEX