Deploying a Content Delivery Service Function Chain on an SDN-NFV Operator Infrastructure

Nicolas Herbaut, Daniel Negru, Damien Magoni
Univ. Bordeaux, LaBRI, UMR 5800
F-33400 Talence, France
{nherbaut,negru,magoni}@labri.fr

Pantelis A. Frangoudis
IRISA/University of Rennes 1
Rennes, France
pantelis.frangoudis@irisa.fr

***Abstract*—Increasing over-the-top video consumption makes the delivery of content unsustainable in the current Internet. The Internet Service Providers (ISP) have a hard time competing on value-added services with content providers and content delivery networks (CDN). We propose a new model of collaboration between content delivery actors where CDNs can deploy their software in Internet Service Providers' infrastructures as NFV modules. As the ISP network state has to be kept private, a high-level SLA is used to negotiate both computing resources and connectivity, leaving it up to the ISP to optimize server placement and routes. The contribution of this paper is to formalize this problem as a Service Function Chaining Embedding problem, and to propose an algorithm to optimize the Service Function Chain to improve embedding probability and decrease consumed bandwidth in the process.**

I. Introduction

Video streaming is the number one service on the current Internet. As the Internet was not originally designed for streaming high quality video, delivering this massive amount of content is a challenging task. Content Delivery Networks (CDN) have been proposed to mitigate those issues. CDNs have been created to improve network performance for end-users while at the same time limiting the need for Content Providers (CP) to own an infrastructure [1]. By deploying servers in strategic locations, CDN Providers assign users to a close-by server, thus reducing hop count and avoiding congestion occurrences. They perform their tasks Over-The-Top, limiting the impact of the other main actor in the content delivery chain: the Internet Service Provider (ISP). ISPs are therefore being left behind from a business point of view [2], only being used as a transport medium, contracting nothing but connectivity.

Considering the important advantage of the ISPs in term of network management, i.e. being able to have a complete control over two connected hosts, we envision a solution for a collaboration between those two actors around a Network Function Virtualization (NFV) architecture [3]. We propose a solution where ISPs can manage a distributed NFV infrastructure where CDNs will run their caching software as vNFs through a virtual CDN (vCDN) Network Function, as exposed in [4] and [5]. It is a win-win approach since CDNs can expand their coverage dynamically without buying new servers and ISPs can increase their revenue by billing them while reducing inter-Autonomous Systems (AS) traffic. Relying on these considerations, we propose a NFV/SDN platform to be deployed within the ISP premises dedicated to virtually instantiate a content delivery service (vCDN). This way, ISPs will be capable of establishing contracts with CDN operators, through Service Level Agreements (SLA). Our contribution addresses the issue of technically implementing the related SLAs into vCDN services deployment. We hence propose a Service Function Chain (SFC) [6] model, to be embedded into the ISP topology.

In this paper, we highlight the feasibility of such platform by outlining its overall concept in Section II, focusing on how the vCDN service can be implemented through the instantiation of the SLA, proposing strategies for embedding as well as SFC transformations in Section III, and showing out results on SLA acceptance capabilities from simulations on real topologies in Section IV. We conclude the paper and present future work in Section V.

II. A NFV/SDN platform for virtual CDN deployment in the ISP's network

Today, the collaboration between CDNs and ISPs is not direct, due to the specificities of each. Many challenges result in willing to make those actors collaborate. We propose a NFV/SDN platform concept instantiated within the ISP network in order to overcome those challenges.

The objective of the platform is to host virtual CDN (vCDN) services, in conjunction with other services. Figure 1 depicts an instantiation of such platform, by providing the view of the ISP on one side and the view of the CDN on the other. The physical equipments are deployed within the ISP network, the CDN having only an abstraction of the underlying topology.

From an architectural point of view, two NFVI-POPs (Network Function Virtualization Infrastructure Points of Presence) are deployed at the edges of the network in order to provide computing resources, in terms of CPU, RAM or storage, for running virtual Network Functions (vNFs). The network is configured so that end-users are connected through switches supporting Software Defined Networking (SDN). Thanks to this capability, virtual routes can be deployed considering the input given by the CDN, through the provision of streaming vNFs instances. The CDNs contracting with the ISPs to deploy their services only have access to the information of their deployed vCDNs and the end-users they virtually connect. This way, the

TABLE I: ISP-CDN Collaboration example - SLA negotiation

Name	Description	unit	example value	Evaluation Hypothesis
Bit rate	Video target bit rate	kbps	1500 kbps	Random value uniformly chosen between 500kbps, 750kbps, 1000kbps and 2000kbps
Count	Video download count	#	1500	Random Values from from a normal distribution with mean 1000 and std dev 100
Content Duration	Video average duration	time	3600s	Randomly chosen from a normal distribution with mean 3600s and std dev 360 s
SLA duration	Start and stop of the SLA	time	Start: 10/10/2017 13:30 Stop: 11/10/2017 8:30	Random values from a normal distribution with mean 600 min and std dev 60 min
End-User location	an identifier describe where the users connect from	Ip prefix, IXP id,...	Bordeaux	uniformally chosen between substrate nodes
Master CDN Location	an identifier describing where vCDN cache miss should be directed	host name	paris.akamai-cdn.com	U)niformally chosen between substrate nodes

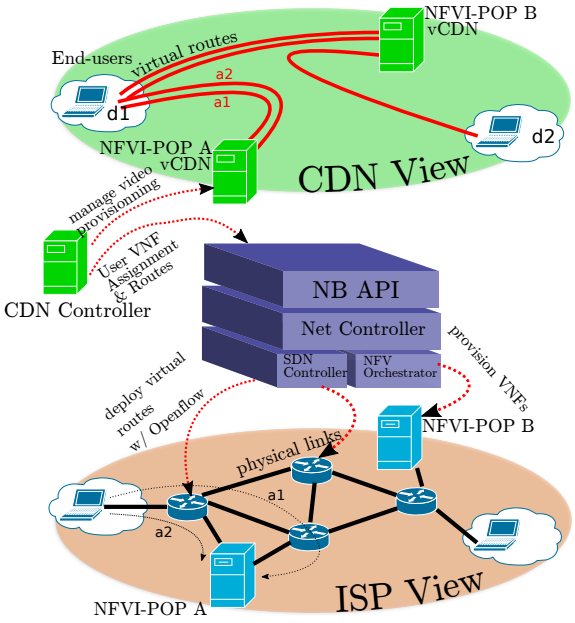


Fig. 1: vCDN deployment - The CDN and ISP Views

underlying network topology and other sensitive features are kept hidden to the CDNs.

Today's interest of the ISPs is more and more focused on technologies to ease network deployment, scalability and management, such as NFV and SDN. Besides, a new type of revenues is hence foreseen for the ISPs, providing them the possibility to explore new types of business opportunities on the cloud market. For instance, thanks to such a proposed platform, ISPs would be capable of contracting Service Level Agreements (SLAs) with actors such as CDNs for deploying and running their vCDN functions. The important advantage for the ISPs stands in the fact that they have full end-to-end control of the network.

In order to set-up such collaboration between the ISP and CDN, a Service Level Agreement (SLA) will be negotiated between them. An example of such a possible SLA can be found in Table I. The CDN operator demands the creation of virtual CDN (vCDN) instances so that their video service will be delivered effectively (i.e., bandwidth and delay requirements) to a certain number of end-users. If the ISP can supply this demand, it will validate the negotiated contract and deploy the instances, wherever it is necessary and without the need to inform the CDN about all the sensitive information.

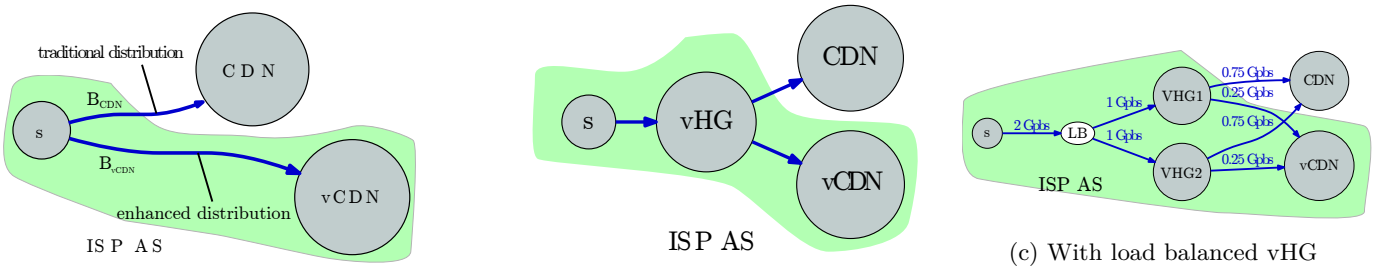
III. The vCDN service implementation

A. From the SLA to the Service Function Chain

The first step for the ISP, once the SLA with the CDN is formulated, is to instantiate the network. Such translation from the SLA to the (Content Delivery) Service Function Chain is composed of 4 steps:

Step 1: the canonical Model: It consists of the raw technical translation of the CDN demand, as shown in Figure 2a. It specifies the connectivity between the End-User S (which represents the geographical area toward End-Users) and two content servers, the CDN and the vCDN. The CDN stands outside the ISP AS and will be reached from S through a certain point in the ISP network, where ISP and CDN have direct or indirect peering. On the opposite, the vCDN is logically located within the ISP AS. BwCDN is the bandwidth required by the CDN and BwvCDN consists of the bandwidth required by the vCDN. The packets of the traditional distribution path ($s \leftrightarrow CDN$) pass through the ISP AS and reach the separate CDN AS. We propose an enhanced distribution path $s \leftrightarrow vCDN$, where the SLA will be applied. This path stays in the ISP AS.

Step2: Leveraging Virtual Home Gateways: CDNs use DNS redirection techniques [1] to specify to which server to route the request. This application-layer method works with the granularity of domain names. However, this technique is too coarse grained to work at the *content* level.



(a) Canonical Representation

(b) Implementation using vHG

(c) With load balanced vHG

Fig. 2: Three representations of the same Service Chain

Indeed, redirecting the whole traffic towards a domain name to the caching servers (as it is the case with the DNS approach) regardless of the actual presence of the content causes a lot of cache misses. A cache miss causes the content to be first pulled from the original server, and then be served from the CDN. This causes a lot of undesired traffic that could have been better served by routing to the original server in the first place.

To mitigate this issue, Virtual Home Gateways can be used to enhance content distribution [7], [8]. vHG are virtual appliances that outsource traditional physical Home Gateway's Network Functions to the cloud, leaving a simple OSI Layer 2 bridge on the user premises. The vHG regroups several network functions, but in the present use case we focus only on the routing function. In a content delivery scenario, the vHG can be seen as a transit network function able to alter data flows according to OSI Layer 5-7 rules, like HTTP URL matching. On the forward path, the vHG can therefore segregate traffic targeted toward the vCDN from the one targeted to the CDN using a mapping table. It enables packets to follow distinct routes with distinct network characteristics. On the return path, the vHG monitors traffic from the POPs to further optimize content delivery, potentially choosing a different one if needed. For example, Figure 3 shows 2 users connected to the same vHG requesting 2 content items from the same domain. User 1's content is not present in any POP, so it is routed normally according to the original domain name. User 2's content however is cached in a POP. Based on the mapping table, the vHG detects that the URL points toward a cached content and routes the request to the POPa where the vCDN is deployed. At the network level, routing is done on a reserved path following a virtual route, with guaranteed bandwidth and delay.

ISPs have full control of their network and can leverage an SDN approach to have more flexibility. SDN relies on southbound technologies to deploy configuration to network devices which are often limited in features. For instance, Open Flow [9] is SDN's most used southbound protocol. To this day, it can operate only on levels 1 to 4 of the OSI layer model. Using the vHG approach, we can mitigate those issues by analyzing user requests at the application level and then configure the data flow to be steered along an SDN-defined virtual route with guaranteed quality of service.

Step 3 : Service Function Chaining embedding: With the addition of the vHG, we can now rewrite the canonical model as a Service Function Chain as shown in Figure 2b. The next step is for the ISP to embed this chain generated from the CDN's SLA into its substrate network. This problem is known as Service Function Chain Embedding [6] where we need to map a service graph $G^S = (N^S, E^S)$ to a substrate graph $G = (N, E)$ representing the real ISP network. This approach is similar to existing overlay networks ones, but applied to NFV. Figure 4 shows such a mapping. We can see that the service nodes are mapped to physical nodes. The edges on the service nodes are mapped to virtual routes. The challenge for the ISP is to embed the chain: this problem is formalized as an optimization problem, following the existing literature on Virtual Network Embedding (VNE) [10], [11], or SFC embedding [12].

Step 4: Service Function Chaining Transformation: SLAs vary in the amount of physical resources needed and can be very demanding sometimes. In this context, it can be quite challenging for the ISPs to map a vNF to a single POP or to route the resulting traffic though a single route with enough bandwidth. Path splitting [13] can be used to spread the traffic on multiple routes in the substrate, however this causes problems in multimedia delivery over HTTP as TCP packet re-ordering can degrade performance.

To solve this issue, we propose splitting the nodes instead by introducing a load balancer that can be used to separate the flows coming from the source node while keeping each flow on a single virtual route. For example, end-users can be dispatched on different vHGs based on their home device's MAC address, assigning 50% of the total traffic to vHG1 and 50% to vHG2 as shown in Figure 2c. This reduces the stress on the substrate nodes by allowing several POPs to host the same vNF. New virtual routes are created to support the split, each one requiring less bandwidth (but the same minimal delay to support the throughput required by the SLA).

For link mapping, as nothing prevents virtual routes

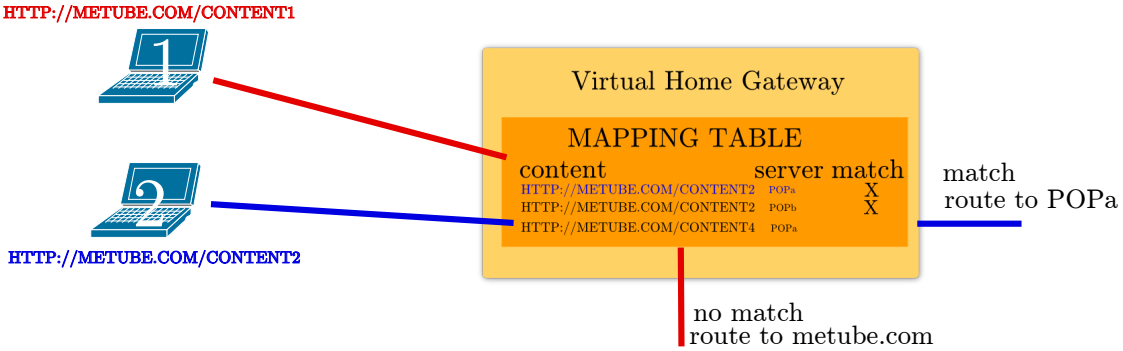


Fig. 3: Routing function of the VHG

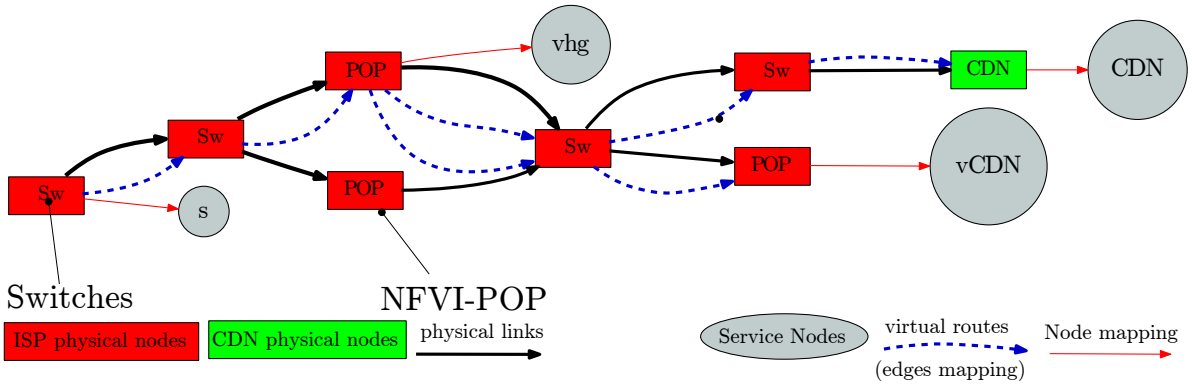


Fig. 4: Service Function Chain Mapping

to be mapped to the same set of edges, the probability of successful embedding for the transformed model is at least equal to the canonical model. However for node mapping, the required system resources of the VNF cannot be split as easily. Indeed, these resources are not strictly linear, meaning that the vHG or vCDN needed to handle half the workload would need more than half the footprint.

In this paper, we also consider only simple SFC transformation strategies:

- **Strategy 1**: adding vHGs, and equally splitting the traffic coming from the source between each of them, as illustrated in Figure 2c.
- **Strategy 2** adding vCDNs, and equally splitting the traffic coming from the vHGs between each of them.
- **Strategy 3** adding vCDNs and vHGs each in their turn.

More advanced strategies, using substrate information, will be assessed in future work.

SFC embedding is an optimization problem that the ISP solves. We implemented an Integer Linear program that exactly solves this problem. For the SFC transformation, we implemented the three strategies described in the previous section.

Data: CDN's provided SLA
Result: A Service Mapping $\mathscr{M}$ for SLA or rejection
$s \leftarrow$ canonical_service(SLA);
do
  $\mathscr{M} \leftarrow solve(s)$;
  if $\mathscr{M} = \emptyset$ **then**
    $s \leftarrow$ transform(s);
    if $s = \emptyset$ **then**
      throw rejection;
  else
    return $\mathscr{M}$;
while *true*;

Algorithm 1: Service Mapping algorithm

B. SFC Embedding Algorithm

The Algorithm 1 describes the general algorithm used in our approach. The *canonical_service* function computes the simplest SFC from the SLA. The *solve* function produces a valid mapping for the given service and the substrate according to the optimization problem presented in III-C. If the problem is declared unfeasible, mapping is null. In this case, we transform the service according to the chosen transformation strategy with the *transform* function, and try to solve the mapping problem again. If the transformation is not possible, for example if we reach the maximum number of vHG or vCDN in the service, the

embedding fails and the SLA is rejected. In case of success, this algorithm returns a mapping $\mathscr{M}$.

C. SFC Embedding Optimization problem

TABLE II: Notations

Symbol	Domain	Description
$\mathbb{M}$		The domain of all possible mappings of $G^S = (N^S, E^S)$ in $G = (N, E)$.
$\mathscr{M}$	$\mathbb{M}$	mapping of service graph $G^S = (N^S, E^S)$ to a substrate graph $G = (N, E)$.
$i \xrightarrow{\mathscr{M}} u$		indicate that for a mapping $\mathscr{M}$ service node i is mapped to substrate node u
$\mathscr{P}^S(i,j)$	$N^S \times N^S \mapsto (E^S)^n$	the set of edges in E^S that links node $i \in N^S$ to node $j \in N^S$
$\mathscr{P}^S_{\mathscr{M}}(i,j)$	$N^S \times N^S \mapsto (E)^n$	the set of edges in E that links service nodes i and j for a mapping $\mathscr{M}$
$\mathscr{D}_{\mathscr{M}}(i,j)$	$N^S \times N^S \mapsto \mathbb{N}$	the total substrate delay for mapping $\mathscr{M}$ for $(i,j) \in E^S$
s	N^S	the starting node of the service chain.
$\mathscr{T}$	$(N^S)^n$	the set of terminal nodes of the service chain.
$y^{i,j}_{u,v}$	$[0,1]$	$\begin{cases} 1, & \text{if link } (i,j) \in E^S \text{ is mapped on substrate link } (u,v) \in E \\ 0, & \text{otherwise} \end{cases}$
x^i_u	$[0,1]$	$\begin{cases} 1, & \text{if } i \xrightarrow{\mathscr{M}} u \\ 0, & \text{otherwise} \end{cases}$
$d_{a,b}$	$\mathbb{N}$	the delay for edge $(a,b) \in E$
$d^S_{i,j}$	$\mathbb{N}$	the maximum delay on $(i,j) \in E^S$ admissible for chain G^S.
$b^S_{i,j}$	$\mathbb{N}$	the required bandwidth on $(i,j) \in E^S$ for chain G^S's.
$b_{u,v}$	$\mathbb{N}$	the available bandwidth on edge $(u,v) \in E$ of the substrate
c^S_i	$\mathbb{N}$	the required computing resources on node $(i) \in N^S$ for chain G^S's.
c_u	$\mathbb{N}$	the available computing resources on node $u \in E$ on the substrate

We formulate the optimization problem using standard network flows literature notation [14]. Table II summarizes the variables, functions and sets used in this section. We used the SCIP toolkit[15] to produce mappings[1]. SCIP employs a branch-and-bound approach, that is complemented by linear programming relaxation and cutting plane separators, by constraint specific domain propagation algorithms and by conflict analysis [16]. As we try to takle nodes and edges assignement as the same time, the problem is know to be $\mathcal{NP}$-hard [10].

We aim at maximizing the following objective function function (1) which drives the mapping function to use the links with the higher available bandwidth for virtual routes.

$$\sum_{(i,j) \in \mathscr{P}^S(s,x)} \sum_{(u,v) \in \mathscr{P}^S_{\mathscr{M}}(i,j)} \frac{b_{u,v} - (y^{i,j}_{u,v} \times b^S_{i,j})}{b_{u,v}} \quad (1)$$

Network function location unicity constraint: Equation (2) ensures that each network function is allocated to only 1 node u in the substrate.

$$\sum_{u \in N} x^i_u = 1, \forall i \in N^S \quad (2)$$

Substrate node constraint: Equation (3) ensures that each network function is allocated to a node that matches the required computing resource c^S_i. Without loss of generality, we considered evaluating computing resources as an integer, instead of a vector of server features like [CPU, RAM, storage].

$$\sum_{i \in N^S} x^i_j \times c^S_i \leq c_u, \forall u \in N \quad (3)$$

Substrate edge capacity constraint: Equation (4) ensures that for each segment used to map the service edge (i,j) has enough bandwidth.

$$\sum_{(i,j) \in E^S} y^{i,j}_{u,v} \times b^S_{i,j} \leq b_{u,v}, \forall (u,v) \in E \quad (4)$$

Substrate edge delay constraint: Equation (5) ensures that each segment used to map the service edge (i,j) has a smaller delay that the one needed by the service.

$$\sum y^{i,j}_{u,v} \times d_{u,v\,(u,v) \in E} \leq d^S_{i,j} \forall (i,j) \in E^S \quad (5)$$

Flow conservation constraint: Equation (6) implements classical flow conservation constraints.

$$\sum_{v \in N} y^{i,j}_{u,v} - y^{i,j}_{v,u} = x^i_u - x^j_u, \forall (i,j) \in E^S, \forall u \in N / \{u \in \mathscr{T}\} \quad (6)$$

Loop Constraints: Equation (7) ensures that there's no loop in the mapped service.

$$y^{i,j}_{u,v} + y^{i,j}_{v,u} \leq 1, \forall (u,v) \in E, \forall (i,j) \in E^S \quad (7)$$

Domain Constraints: Equations (8) and (9) represent domain constraints.

$$x^u_i \in \{0,1\} \quad (8)$$

$$y^{i,j}_{u,v} \in \{0,1\} \quad (9)$$

IV. Results

To evaluate our solution, we have performed simulations on a real topology retrieved from an open database [17]. The Geant[2] topology has been selected as it contains the most complete set of data (Bandwidth, GPS coordinates, etc.). We were able to use the real bandwidth and we estimated delay based on the distance (according to

[1]Complete source code and data can be found online at http://www.labri.fr/perso/nherbaut

[2]data can be retrieved here: http://www.topology-zoo.org/files/Geant2001.graphml

TABLE III: Comparison of computing time for each strategy

Strategy	Average Embedding time for 1 SLA
S0 (no transformation)	0.146 s
S1 (adding VHG)	0.343 s
S2 (adding VCDN)	0.283 s
S3 (adding VHG and VCDN)	0.673 s

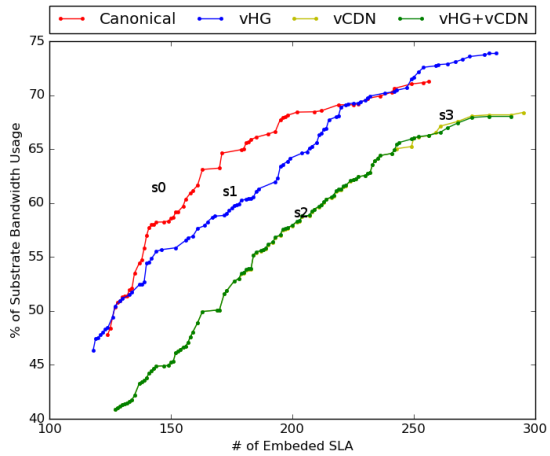


Fig. 5: Impact on SFC transformations on total bandwidth consumed.

POP GPS coordinates) and the speed of light, neglecting other types of delay. We had to neglect other types of delay (transit etc.), but we considered that they would impact each link uniformly, without changing the trend of the results.

Geant is a small topology composed of 28 nodes (POP) and 35 edges. Most of the execution time is spent optimizing the embedding for each SLA. For strategies supporting transformations (s1, s2 and s3), several optimization loops occur for the same SLA, as the algorithm try to find a service that can fit the subtrate. Table III shows the mean execution time for 1 embedding, broken down by strategies.

We have only considered the 3 transformations presented in section III-A and have compared them to an embedding with exactly 1 vHG and 1 vCDN. Algorithm 2 details the procedure to compute the metrics. The algorithm repeats the evaluation for each transformation type using the same random seed to submit the same SLAs. The hypotheses to generate the SLAs are presented in Table I. A topology is loaded and nodes are assigned random capacity values of mean 100 with a standard deviation of 20, drawn from a normal distribution. Each vHG consumes 1 unit of node capacity and each vCDN consumes 3 units. The simulator picks up an SLA from the list, and tries to map it to the ISP network, possibly transforming it in case some constraints (node capacity, edge bandwidth or delay) cannot be fulfilled. Once the SLA rejection threshold is reached, the algorithm stops and reports results.

Figure 5 shows that the simple implementation using

Data: ISP Topology Data, random seed
Result: Number of successfully embedded SLAs, Total Bandwidth usages, Total CPU Usage
foreach $transformation\_type$ **do**
  $rs \leftarrow initialize\_random\_generator(seed)$;
  $SLA\_tab[] \leftarrow generate\_slas(seed)$;
  $substrate \leftarrow load\_topology()$;
  $rejection \leftarrow 0$;
  do
    $sla \leftarrow SLA\_tab.pop()$;
    $service \leftarrow get\_service\_from\_SLA(sla)$;
    do
      $\mathscr{M} \leftarrow solve(service, su)$;
      if $\mathscr{M} = \emptyset$ **then**
        $service \leftarrow transform(service)$;
        if $service = \emptyset$ **then**
          $rejection \leftarrow rejection + 1$;
          break;
        else
          $consume\_service(substrate, service)$;
    while $\mathscr{M} = \emptyset$;
    $store\_metrics(\mathscr{M}, substrate, service)$
  while $rejection < rejection\_threshold$;

Algorithm 2: Transformation comparison algorithm

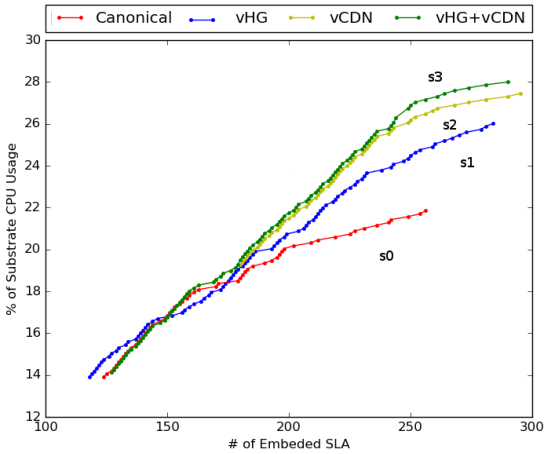


Fig. 6: Impact on SFC transformations on the substrate node capacity consumed.

the canonical model (strategy 0) accepts 161 SLAs while strategy 1 accepts 186 and strategies 2 accept 177 and 3 accept 190, representing a 15% improvement. Interestingly, dominant strategies also show a lower utilization of the substrate bandwidth. Strategies 2 and 3 save 10% of bandwidth over strategy 0. These results show that by modifying the SFC, one can increase the embedding rate and save bandwidth in the process. The tradeoff of using vHG and vCDN is the amount of node resources that needs to be allocated to the VNF. Figure 6 shows that the amount of computing resources increases when using transformations. This is logical as we are deploying more vNFs, consuming some resources.

Choosing the right strategy is a matter of computing the total cost of ownership of the computing resources and bandwidth, and is specific to each ISP. However, when analyzing the structure cost of clouds, we see the trend of networking costs surpassing computing resources cost. We believe that trading network resources for computing resources can be a good perspective in ISPs' efforts to limit their global costs.

V. Conclusion and future work

We have proposed a design for an ISP-CDN collaboration platform in which CDNs can run their delivery functions. We have modeled the problem of embedding a Content Delivery Service Function Chain over the ISP network, leveraging innovative network functions.

By applying an algorithm that splits the traffic to several vNFs in the chain, we have increased the embedding acceptance level. We plan to improve this algorithm by extracting topology information to help design a better transformation algorithm. We are in the process of implementing a proof of concept of our approach in collaboration with an ISP, into which we will measure the performance in terms of peering traffic reduction, quality of service for end-users and scalability. Furthermore, deriving an economic model of this new platform will enable us to analytically evaluate how the equilibrium/balance in revenue sharing is reached and a full win-win approach is achieved.

Finally, as this optimization problem is tractable only for small topologies, we will investigate the use of heuristics to reduce computation time.

Acknowledgment

The work performed for this paper has been partially funded by the FP7 IP T-NOVA European Project (Grant Agreement N#619520) and the FUI French National Project DVD2C

References

[1] M. Pathan, "Cloud-based content delivery and streaming," *Advanced Content Delivery, Streaming, and Cloud Services*, pp. 1–31, 2014.

[2] J. Wulf, R. Zarnekow, T. Hau, and W. Brenner, "Carrier activities in the cdn market-an exploratory analysis and strategic implications," in *Intelligence in Next Generation Networks (ICIN), 2010 14th International Conference on.* IEEE, 2010, pp. 1–6.

[3] G. Xilouris, M.-A. Kourtis, M. J. McGrath, V. Riccobene, G. Petralia, E. Markakis, E. Palis, A. Georgios, G. Gardikis, J. F. Riera *et al.*, "T-nova: Network functions as-a-service over virtualised infrastructures," in *Network Function Virtualization and Software Defined Network (NFV-SDN), 2015 IEEE Conference on.* IEEE, 2015, pp. 13–14.

[4] N. Bouten, J. Famaey, R. Mijumbi, B. Naudts, J. Serrat, S. Latré, and F. De Turck, "Towards nfv-based multimedia delivery," in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on.* IEEE, 2015, pp. 738–741.

[5] T.-W. Um, H. Lee, W. Ryu, and J. K. Choi, "Dynamic resource allocation and scheduling for cloud-based virtual content delivery networks," *ETRI Journal*, vol. 36, no. 2, pp. 197–205, 2014.

[6] P. Quinn and T. Nadeau, "Problem statement for service function chaining," *RFC 7498*, 2015.

[7] G. ETSI, "001,"network functions virtualization (nfv): Use cases", 10-2013."

[8] N. Herbaut, D. Negru, G. Xilouris, and Y. Chen, "Migrating to a nfv-based home gateway: introducing a surrogate vnf approach," in *Network of the Future (NOF), 2015 6th International Conference on the.* IEEE, 2015, pp. 1–7.

[9] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[10] M. Chowdhury, M. R. Rahman, and R. Boutaba, "Vineyard: Virtual network embedding algorithms with coordinated node and link mapping," *IEEE/ACM Transactions on Networking (TON)*, vol. 20, no. 1, pp. 206–219, 2012. [Online]. Available: http://dl.acm.org/citation.cfm?id=2182767

[11] A. Fischer, J. Botero, M. Till Beck, H. de Meer, and X. Hesselbach, "Virtual Network Embedding: A Survey," *IEEE Communications Surveys Tutorials*, vol. 15, no. 4, pp. 1888–1906, 2013.

[12] M. Ghaznavi, N. Shahriar, R. Ahmed, and R. Boutaba, "Service function chaining simplified," *arXiv preprint arXiv:1601.00751*, 2016.

[13] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: substrate support for path splitting and migration," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 17–29, 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1355737

[14] R. K. Ahuja, "Network flows," Ph.D. dissertation, Technische Universitat Darmstadt, 1993.

[15] A. Tobias, "Scip: Solving constraint integer programs," *Mathematical Programming Computation*, vol. 1, no. 1, pp. 1–41, 2009, http://mpc.zib.de/index.php/MPC/article/view/4.

[16] T. Achterberg, "Scip: solving constraint integer programs," *Mathematical Programming Computation*, vol. 1, no. 1, pp. 1–41, 2009.

[17] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The internet topology zoo," *Selected Areas in Communications, IEEE Journal on*, vol. 29, no. 9, pp. 1765–1775, 2011.