# Joint Optimization for the Delivery of Multiple Video Channels in Telco-CDN

Fen Zhou
CERI-LIA
University of Avignon
fen.zhou@univ-avignon.fr

Jiayi Liu Gwendal Simon
Network Department
Telecom Bretagne, Institut Mines-Telecom
{firstname.lastname}@telecom-bretagne.eu

Raouf Boutaba
D. Cheriton School of Computer Science
University of Waterloo
rboutaba@uwaterloo.ca

***Abstract*—A Telco-CDN can be regarded as an intra-domain overlay network with tight resources and critical deployment constraints. This paper addresses two problems in this context: (1) the construction of the overlays used to deliver the video channels from the entrypoints of the Telco-CDN to the appropriate edge servers; and (2) the allocation of the required resources to these overlays. Our ultimate goal is to maximize the number of delivered channels while preserving network resources. Two classes of heuristic algorithms, namely two-step optimization and joint-optimization, are proposed to solve these problems. The conducted evaluations confirm the efficiency of the joint-optimization approach.**

***Index Terms*—Content Delivery Networks (CDN), Video Delivery, Joint Optimization, Heuristic Algorithms**

## I. Introduction

It has become clear over the past couple of years that Internet Service Providers (ISP) have to deploy a Content Delivery Network (CDN) *within* their network infrastructure in order to cope with the sharp increase of video traffic. The recent OpenConnect [1] proposal from Netflix illustrates this major shift: content providers now develop peering agreements with ISPs, and rely on co-controlled delivery platforms to serve end-users. As a matter of fact, the management of video streams in *Telco-CDNs* has become a critical topic for both network operators and content providers.

Telco-CDNs differ from traditional CDNs on several aspects. First, Telco-CDNs can be fully controlled, since they own the network and the entire "last mile" from edge servers to end users. This enables to engineer Telco-CDNs through centralized optimization techniques so that the Quality of Service (QoS) can be guaranteed. Second, a Telco-CDN matches the topology of an underlying network. Consider the case of a French network operator (see Fig. 1). The "*entrypoints*" for the content providers are the Internet Exchange Points (IXP) where peering occurs (here the rectangular nodes). As far as we know, *edge servers* in Telco-CDN are deployed near the routers that connect the ISP backbone core network to the metropolitan access networks of the different regions (the circular nodes). Thus each edge server is in charge of a regional area. However, the congestion of the core network is a serious concern for network operators, so content delivery within the Telco-CDN—between entrypoints and edge servers — is an essential operational and management concern.

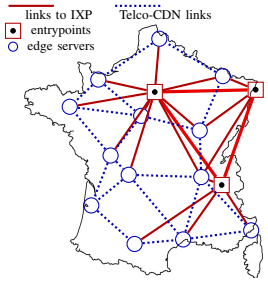



Fig. 1. Topology of a French Telco-CDN

A key challenge for Telco-CDNs is the management of the traffic generated by *live video channels* from Over-The-Top (OTT) video services. OTT content providers include user-generated video platforms like justin.tv or ustream, and also second-screen video services affiliated with traditional TV broadcasters. Traffic related to these services is exploding. Moreover, the adoption of rate-adaptive streaming technologies (*e.g.* DASH) negatively affects the CDN infrastructure because the bit-rate of a video channel corresponds to the accumulated rate of all the video representations, which is frequently greater than 20 Mbps [2].

As it has been shown in many previous works [3]–[5], the delivery of a live channel stream over a CDN can benefit from constructing an *overlay*. Edge servers contribute to the delivery of flows within the CDN using spare upload capacities. Several techniques have been developed for the construction of an overlay for one live channel over CDNs. In particular, *multi-tree* overlays that leverage *rateless codes* (e.g., Raptor codes [6]) enable video stream delivery with optimal bandwidth utilization [5, 7, 8]. This solution however performs well for *one* channel while the traffic that a CDN should carry for a platform like justin.tv consists of a large number of simultaneous channels. Typically, agreements between justin.tv and CDN stipulate the delivery of around fifty channels—the so-called "featured" channels.

In this paper, we study the problem of delivering multiple channels over a Telco-CDN. Our goal is to maximize the number of channels that are delivered to the edge servers, given that the Telco-CDN is constrained by the capacity of the edge servers. This optimization problem involves two distinct

subproblems: the construction of one overlay per channel, and the allocation of resources among the competing overlays. To tackle this problem, we envision two approaches:

- **Two-Step optimization**– First, we construct all overlays, then we allocate the bandwidth among the different overlays so that the most important channels are delivered, until resource exhaustion.
- **Joint optimization**– We jointly construct the overlays and reserve the required resources on the fly. This approach leads to better solutions.

The contribution of this paper is two-fold. First, it provides a step into the study of joint optimization problems for the provisioning of multiple overlays. Joint optimization approach achieve better performance at the cost of more intensive computations. Our study provides a comprehensive evaluation of this trade-off. Second, this paper introduces some key management needs for scarce resources in Telco-CDNs.

## II. Related Work

Optimization problems related to the construction of tree overlays have been widely studied in the context of IP multicast [9]–[13]. Researchers have identified two main ways to build a multicast tree. The first one consists in building a Steiner tree using the heuristic algorithm given in [10]. The second one, namely Minimum Path heuristic algorithm [9], tries to construct a cost-optimal tree. In this paper, we use a more contemporary video delivery technique based on rateless codes [6], where each delivery is ensured by multiple trees.

The use of rateless coding in content delivery overlays was introduced in [7]. Its key advantages include: $(i)$ rateless codes have very low computational cost; $(ii)$ they minimize delivery redundancy when a server receives data concurrently from multiple other servers; and $(iii)$ they are adaptive to varying channel conditions since the encoder can generate on the fly as many encoded symbols as needed. Following this seminal work, several performance improvements were proposed in the literature. End-to-end delivery delay is reduced by the method described in [14]. Shorter start-up delay and more stable service are among the objectives addressed in [15]. Previous works consider the delivery in only one overlay. The objective of this paper is different since it aims at constructing *several* rateless coding based overlays while addressing the critical problem of bandwidth reservation for these overlays. Our approach does not require any specific overlay construction method, hence any of the existing overlay tree construction algorithms can be used in conjunction with our approach.

## III. System Model

The Telco-CDN network is modeled as a connected symmetric digraph $G(V, E)$. See an example in Fig. 1. The set $V$ ($|V| = n$) contains the *edge servers* (here the access points to the regional areas) and a set of *entrypoints* $S \subseteq V$, which are located near the peering points (here the three main French peering IXPs). A content provider usually delivers its most popular channels to a given ISP through only one peering point. But a Telco-CDN serves *several* content providers, so the set of channels $\mathcal{I}$ comes from several entrypoints. Each entrypoint $s$ is associated with the set of channels $\mathcal{I}(s) \subseteq \mathcal{I}$ that it actually receives. The sets $\mathcal{I}(s)$ for all $s \in S$ form a partition of $\mathcal{I}$. Each channel $i \in \mathcal{I}$ is associated with two key parameters. First, the *targets* are the set of edge servers that must receive the channel $i$. This subset of edge servers is denoted $V_i \subseteq V \setminus S$. Second, the *importance* of channel $i$, denoted by $f_i$, sets the priority to the delivery of channels. The channel importance is generally related to the popularity of the content in the area. The monitoring of the CDN and the algorithms that allow to set both parameters are out of the scope of this paper. For major services, the content provider provides these parameters to the CDN provider.

We leverage previous works on multi-tree (forest) overlay networks based on rateless codes [5, 8]. Each edge server $v$ reserves some *upload capacity* $c_v$ to assist the entrypoints. To simplify, we assume there exists a unit of stream, and we express upload capacities as the number of units that the server can transmit over a period of time. The system introduced in [8] suggests to build, for each channel $i$, a forest overlay $F_i$ where each overlay tree supports one unit of encoded video stream. An edge server can re-build the stream if it is spanned in at least $K$ trees.

### *Studied Optimization Problem*

The objective is twofold: to deliver the maximum number of channels (with regard to their importance) and to reduce the traffic load on the CDN infrastructure, *i.e.*, to reduce the number of edge servers that are involved in the delivery of a channel if they are not a target for this channel. The former objective prevails over the latter. Therefore we formulate the optimization problem as first to deliver the maximum number of channels, then to find the delivery scheme that imposes the least traffic load on the network. Let $R_i$ be a binary variable equal to one if the channel $i$ is delivered to all edge servers in $V_i$, and zero otherwise. Our first objective is to maximize $\sum_{i \in \mathcal{I}} f_i \times R_i$, and our second objective is to minimize the number of overlay links that are used to deliver the channels to the edge servers. In our rateless code based multiple-channel video streaming system, three main constraints are considered:

- **Edge servers capacity constraint**. The contribution of edge servers to stream delivery should not overcome their capacities. Let $deg_T^+(v)$ be the out degree of node $v$ in a tree $T$. Clearly the sum of its out-degree in the trees of all the overlays it is involved in cannot be greater than its capacity:

$$\sum_{i \in \mathcal{I}} \sum_{T \in F_i} deg_T^+(v) \leq c_v, \forall v \in V \tag{1}$$

- **Video content constraint related to rateless codes**. Each edge server subscribing to channel $i$ should be spanned in at least $K$ trees in the forest $F_i$ dedicated to channel $i$, so that the edge server is able to receive $K$ units of stream and decode the video. Let $F_i(v)$ be the set of trees in which the edge server $v$ is included in $F_i$. We have

$$\forall i \in \mathcal{I}, \forall v \in V_i, |F_i(v)| \geq K \tag{2}$$

- **Hop-count constraint**, which may be used to bound the delay and transmission reliability from the entrypoint to each subscribing edge server. For each tree $T \in F_i$, the number of hop-counts from $s$ to any node of $T$ should be smaller than a given threshold $D_h$. Let $SP_T(v)$ be the shortest path (list of arcs) from $s$ to $v$ in $T$, we have

$$\forall i \in \mathcal{I}, \forall T \in F_i, \forall v \in T, \sum_{e \in SP_T(v)} 1 \leq D_h \qquad (3)$$

## IV. Joint-Optimization and Two-Step Optimization Heuristic Algorithms

We propose two classes of heuristic algorithms for solving the video delivery of multiple channels in practical Telco-CDNs, i.e., two-step optimization and joint-optimization. Both heuristics are greedy algorithms, with one main loop. At each step, we select one channel, then we compute the forest overlay for this channel, and, if it is possible to support the overlay using the residual capacities of edge servers, then we reserve the needed resources for this overlay. We highlight the main differences below.

### A. Two-Step Optimization Approach

In the two-step optimization approach, we first construct all overlays, then we allocate the bandwidth among the different overlays so that the most important channels are delivered, until resource exhaustion. Following this idea, the *SOP-heu* algorithm is proposed, where all overlays are computed before the resources are allocated. Pseudocode is given in Algorithm 1. Here we emphasize: $(i)$ in line 1, all the overlays are pre-computed; and $(ii)$ in line 3, the channels are selected iteratively according to a *utility score*.

**Algorithm 1:** Heuristic Main Loop for *SOP-heu*

**Input** : $\mathcal{I}, \{(f_i, V_i) : \forall i \in \mathcal{I}\}, G, \{c_v : \forall v \in V\}$
**Output**: $F_i$ for all $i \in \mathcal{I}$
1 precompute $F_i$ for all $i \in \mathcal{I}$;
2 **while** *not all channels have been processed* **do**
3  $i \leftarrow$ the remaining channel with max. utility;
4  **if** $F_i$ *not null* **then**
5  update capacities for every edge server in $F_i$

The simplest implementation of the utility score, denoted $\mathcal{U}_i, \forall i \in \mathcal{I}$, is to set $\mathcal{U}(i) = f_i$, *i.e.*, the channels are processed according to their importance. This method is named *SOP1-heu*. It is however well-known from knapsack and bin packing literature that better performance can be obtained using utility scores that take into account both the gain (here $f_i$) and the *penalty cost* that this overlay produces on the infrastructure. This approach is denoted by *SOP2-heu*. In our implementation, we utilize the following parameters to set the penalty cost: $(i)$ *minimum remaining nodal bandwidth*: we try to prevent edge servers to be exhausted, with the risk of network disconnection, and $(ii)$ *saving on critical edge servers*: we analyse the channels that have not been processed yet, and we estimate the edge servers that are the most demanded.

1) **Advantages**: The most costly channels are not processed first if they are not very important. This strategy is expected to allow the distribution of more channels. Moreover, in case of churn of video channels, only the bandwidth allocation part should be re-computed.
2) **Drawbacks**: Algorithm 1 uses pre-computed forest overlays, therefore it may be unable to utilize the last remaining resources at the end of the loop.

### B. Joint-Optimization Approach

Different form the previous approach, the joint-optimization approach proposes to jointly construct the overlays and reserve the required resources on the fly. This approach generally leads to better solutions. The *JOP-heu* heuristic algorithm adopts this idea. Its pseudocode is given in Algorithm 2. We emphasize the following main aspects: $(i)$ in line 2, the channel to process is selected according to the channel importance; and $(ii)$ in line 3, the overlay computation takes into account the remaining capacities of edge servers.

**Algorithm 2:** Heuristic Main Loop for *JOP-heu*

**Input** : $\mathcal{I}, \{(f_i, V_i) : \forall i \in \mathcal{I}\}, G, \{c_v : \forall v \in V\}$
**Output**: $F_i$ for all $i \in \mathcal{I}$
1 **while** *not all channels have been processed* **do**
2  $i \leftarrow$ the most important remaining channel;
3  compute $F_i$ with the remaining capacities;
4  **if** $F_i$ *not null* **then**
5  update capacities for every edge server in $F_i$

1) **Advantages**: Since the forest overlays are computed with the remaining capacity, this algorithm is able to construct overlays even when most capacities have been exhausted. It should result in no waste of resources.
2) **Drawbacks**: Algorithm 2 is oblivious to the amount of capacities that are utilized by the channels. That is, an important but very costly channel can be processed before some channels that are just slightly less important, but far cheaper in terms of resources. Moreover, if a new channel has to be delivered, or if a channel should not be delivered anymore, the algorithm should be executed from scratch.

### C. Overlay Construction

The construction of multi-tree overlays under capacity constraint has been addressed in several previous works, including [8]. Here, we propose to use a tree construction algorithm for general graphs that aims at minimizing the used capacity by linking every edge server iteratively according to their distance to the entrypoint. For a channel $i$, which should be rooted at an entrypoint $s$, the idea of this greedy algorithm is as follows. We first determine the edge server in $V_i$ that is the closest to $s$ in terms of number of intermediate edge servers. If several edge servers are tied, we pick the one with the largest capacity.

Then, we iteratively pick the non-spanned edge server that is the closest to the set of spanned edge servers with spare resources. We always choose the one with maximum capacity if there is a tie. The spanning process continues until all edge servers in $V_i$ are spanned $K$ times in the resultant forest $F_i$.

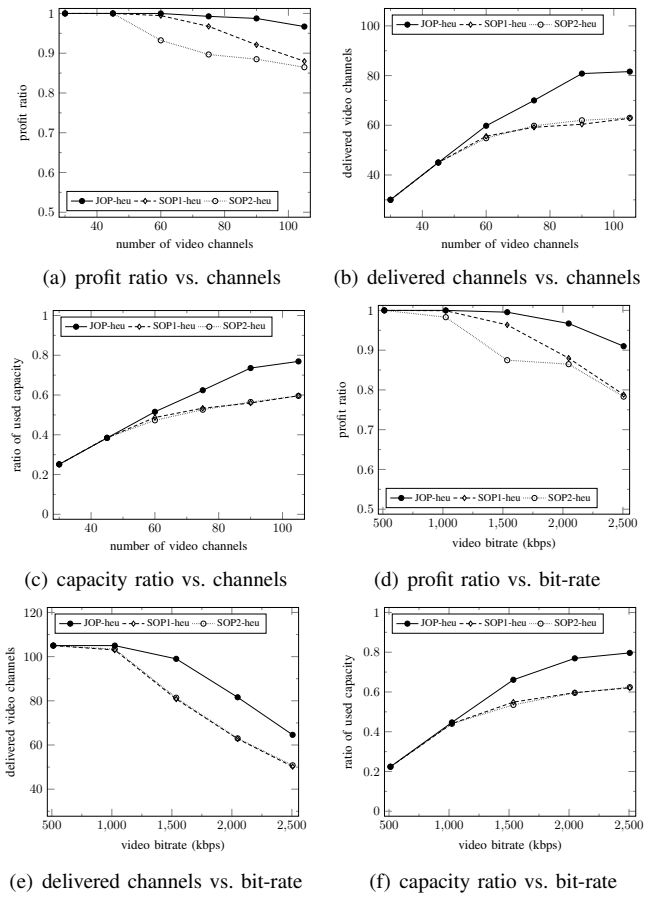
## V. Evaluation

Extensive simulations are conducted to evaluate the performance of the proposed solutions in this paper: *JOP-heu*, *SOP1-heu* and *SOP2-heu*. The French CDN network (Fig. 1) is used as the network topology. The following three metrics are used in our comparisons:

1) **profit ratio**, i.e., $\frac{\sum_{i\in\mathcal{I}} R_i \times f_i}{\sum_{i\in\mathcal{I}} f_i}$, which indicates the satisfaction of the service provider since it measures if channels are well delivered, according to their importance
2) **number of delivered video channels**, i.e., $\sum_{i\in\mathcal{I}} R_i$
3) **ratio of used capacity**, i.e., $\frac{Cap}{\sum_{v\in V} c_v}$

Here, we study the performance of the proposed heuristic algorithms: *JOP-heu*, *SOP1-heu* and *SOP2-heu*. Node bandwidth follows a lognormal distribution with an average of 96 Mbps. The channel importance follows the Zipf$(0.5, 105)$ distribution. The number of subscribers $|V_i|$ for a channel $i$ ranges from 3 to 9. Thus, there are some configurations where the network is over-provisioned (*e.g.* when $|\mathcal{I}| \in \{30, 45\}$), and some others where not all channels can be delivered.

In Figs. 2(a)-2(c), the video bit-rate is 2,048 kbps, and we evaluate the performance of our algorithms with respect to the number of channels from 30 to 105. It is found that the proposed three heuristic algorithms serve well the most important channels, so the overall profit is almost the same for every algorithm in the over-provisioned network configuration. In the under-provisioned scenario, i.e., when the number of channels is bigger than 45, all algorithms obtain an achievable profit ratio no less than 87%. The joint optimization *JOP-heu* is always able to obtain up to 12% higher profit ratio and deliver 20 more video channels than the two-step heuristic algorithms. Thus, it clearly results in high network bandwidth usage, i.e., 9% higher. Moreover, the simple importance-first heuristic *SOP1-heu* performs better than *SOP2-heu* in term of achievable profit ratio, while the latter one tends to accept more video channels with smaller bandwidth requirement. This may be explained by the fact that a video channel with smaller importance requires less bandwidth due to the Zipf distribution of importance value. The *SOP1-heu* packs the most important videos better than *SOP2-heu*, meaning that the more sophisticated utility score of *SOP2-heu* is counter-productive. In Figs. 2(d)-2(f), the number of channels is fixed to $|\mathcal{I}| = 105$ and we study the performance of our algorithms when the video bit rate varies from 512 kbps to 2560 kbps. We can find that algorithm *JOP-heu* is always superior to the others. As the video bit rate increases, the achievable profit ratio degrades slightly while the number of delivered videos diminishes quickly. This is due to the lack of bandwidth in the video entrypoints or the edge servers around the videos entrypoints for accommodating high quality videos. It also




(a) profit ratio vs. channels (b) delivered channels vs. channels

(c) capacity ratio vs. channels (d) profit ratio vs. bit-rate

(e) delivered channels vs. bit-rate (f) capacity ratio vs. bit-rate

Fig. 2. Simulation Results in the French Telco-CDN (Fig. 1)

should be noted that the network bandwidth usage is up to 80%, which is not high. This is because, we employ a mesh topology (c.f. Fig. 1), where there are only three entrypoints. Since an entrypoint node does not have a link to all the edge servers, the bandwidth of an edge server far away from the entrypoint node may be wasted when the neighbors of the entrypoint exhaust their bandwidth.

In summary, *JOP-heu* performs best among the proposed heuristic algorithms. When the resources are more constrained, it serves one third of the channels more. Such gain in performance is enough to justify the implementation of *JOP-heu* in resources-constrained Telco-CDNs.

## VI. Conclusion

This paper addressed the problem of overlay construction and bandwidth allocation for delivering video channels from the entrypoints of the Telco-CDN to edge servers. The pursued goal is to maximize the total profit of delivered channels while preserving network resources. To this end, two optimization methods have been compared: two-step optimization and joint optimization. We analyzed the relevance of implementing the joint optimization approach, which is theoretically more efficient. Our work provided insights to network operators for making informed resource provisioning decisions in the support of a Telco-CDN.

## REFERENCES


[1] "Netflix Open Connect Content Delivery Network," https://signup.netflix.com/openconnect.

[2] V. K. Adhikari, Y. Guo, F. Hao, M. Varvello, V. Hilt, M. Steiner, and Z.-L. Zhang, "Unreeling netflix: Understanding and improving multi-CDN movie delivery," in *INFOCOM*, 2012, pp. 1620–1628.

[3] M. Adler, R. K. Sitaraman, and H. Venkataramani, "Algorithms for optimizing the bandwidth cost of content delivery," *Computer Networks*, vol. 55, no. 18, pp. 4007–4020, 2011.

[4] J. M. Almeida, D. L. Eager, M. K. Vernon, and S. J. Wright, "Minimizing delivery cost in scalable streaming content distribution systems," *IEEE Trans. on Multimedia*, vol. 6, no. 2, pp. 356–365, 2004.

[5] N. Thomos and P. Frossard, "Network coding of rateless video in streaming overlays," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 20, no. 12, pp. 1834–1847, 2010.

[6] A. Shokrollahi, "Raptor codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2551–2567, 2006.

[7] C. Wu and B. Li, "rstream: Resilient and optimal peer-to-peer streaming with rateless codes," *IEEE Trans. Parallel Distrib. Syst.*, vol. 19, no. 1, pp. 77–92, 2008.

[8] F. Zhou, S. Ahmad, E. Buyukkaya, G. Simon, and R. Hamzaoui, "Minimizing Server Throughput for Low-Delay Live Streaming in Content Delivery Networks," in *ACM NOSSDAV*, 2012.

[9] H. Takahashi and A. Matsuyama, "An approximate solution for the steiner problem in graphs," *Mathematica Japonica*, vol. 24, no. 6, pp. 573–577, 1980.

[10] L. Kou, G. Markowsky, and L. Berman, "A fast algorithm for steiner trees," *Acta Informatica*, vol. 15, no. 2, pp. 141–145, 1981.

[11] C. A. Noronha and F. Tobagi, "Optimum routing of multicast streams," in *Proc. of INFOCOM*, vol. 2, jun 1994, pp. 865–873.

[12] C.-F. Wang, C.-T. Liang, and R.-H. Jan, "Heuristic algorithms for packing of multiple-group multicasting," *Comput. Oper. Res.*, vol. 29, no. 7, pp. 905–924, jun 2002.

[13] C. A. Oliveira, P. M. Pardalos, and M. G. Resende, *Optimization Problems in Multicast Tree Construction*. Springer, 2006, pp. 701–731.

[14] M. Grangetto, R. Gaeta, and M. Sereno, "Rateless codes network coding for simple and efficient p2p video streaming," in *Proc. of ICME*, 2009.

[15] H. R. Oh, D. O. Wu, and H. Song, "An effective mesh-pull-based p2p video streaming system using fountain codes with variable symbol sizes," *Computer Networks*, vol. 55, no. 12, pp. 2746–2759, 2011.