Performance Analysis of Content-Centric and Content-Delivery Networks with Evolving Object Popularity*

Michele Mangili, *Student Member, IEEE,* Fabio Martignon, *Member, IEEE,* and Antonio Capone, *Senior Member, IEEE*

***Abstract*—The Internet is currently mostly exploited as a means to perform massive digital content distribution. Such a usage profile was not specifically taken into account while initially designing the architecture of the network: as a matter of fact, the Internet was instead conceived around the concept of host-to-host communications between two remote machines.**

To solve this problem, Content-Delivery Networks (CDNs) are currently used as a well-established technology to serve content-driven demands through an infrastructure that is not tailored for that purpose. On the other hand, the novel paradigm of Content-Centric Networking (CCN) aims at filling the gap of this misalignment by changing the network-layer protocols, solving the content-distribution problem at its root.

In this paper, we formulate novel optimization models to analyze the performance gains that CDN and CCN can achieve, by reducing the total amount of traffic exchanged through the network. We tackle this problem by adopting a time-varying content popularity evolution model that accurately represents the dynamic behavior of users.

We discover that, in most of the cases, CDN reduces the overall traffic exchanged between network nodes, leading to better performance, whereas CCN should instead be preferred in those scenarios where CDN cannot quickly react to popularity evolution. On top of that, we show that very limited benefits can be obtained by changing the cache replacement algorithms. Finally, all our key findings are confirmed by extensive simulation campaigns that further complement this work.

***Index Terms*—Content-Centric Networks, Content-Delivery Networks, Performance Analysis, Optimization.**

I. Introduction

NOWADAYS, the way people exploit the services provided by the Internet is radically changed with respect to the years when the Network was initially designed. As a matter of fact, the evolution of online services, as well as the success of digital multimedia diffusion, both demand for new technologies to turn the Internet into an efficient *content distribution infrastructure* [2], [3].

However, this fundamental change clashes with the original design principles that guided the engineering process of the worldwide network. In particular, it is in conflict with the well known end-to-end principle, according to which intermediate network nodes should be specialized to accommodate basic functionalities such as packet forwarding, whereas application-specific needs should instead be implemented only in the end hosts [4]. Among the direct consequences of this choice, IP routers can nowadays reach the impressive (and still improving) capacity of 921 Tbps [5]; however, this same choice has limited the content-distribution capabilities of Internet, since implementing content-caching functionalities at the network layer is quite demanding [6].

To overcome this limitation, by moving content replicas nearer to the actual consumers location, Content-Delivery Networks (CDNs), such as Akamai [7], are currently used to efficiently serve the content requests of worldwide Internet users, in today's TCP/IP Internet. CDNs are also used to effectively support sudden popularity changes, known as *flash crowds*, which can mine the reliability of the system by overwhelming the servers with a huge number of requests.

Rather than working at the application layer, keeping IP as network protocol, a different approach is instead supported by innovative designs known under the name of *"Content-Centric Networks"* (CCNs), which are recently gaining momentum in the research community [8]. These designs propose to unleash the content-distribution potentials of the Internet by using novel network protocols, making the network capable to quickly react to flash crowds. In particular, one of the advantages obtained switching to these designs is that they can be used to easily provide distributed *in-network caching* at the network level: any router can store (and serve) local copies of given data, thus making the content be replicated closer to the locations where most of the users are actually requesting it, without requiring application-layer solutions.

Moved by the desire (and necessity) to understand whether the migration towards CCN can provide significant benefits to network providers, in this paper we present both theoretical and simulated results that analyze and compare the performance of the CCN and CDN architectures. We consider a scenario where *time-varying* content popularity demands are generated by the consumers, in such a way that we can assess the network capability to react to the dynamic content popularity evolution. We formulate novel optimization models to represent relevant features for the CCN and CDN architectures, and we use them to analyze the best performance bounds that these networks can achieve.

*Preliminary results of this work have been presented in [1].

M. Mangili is with the Laboratoire de Recherche en Informatique (LRI), Université Paris-Sud, Paris, France, and the Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano, Milan, Italy, (email: michele.mangili@lri.fr).

F. Martignon is with the Laboratoire de Recherche en Informatique (LRI), Université Paris-Sud, Paris, France, and Institut Universitaire de France (IUF), (email: fabio.martignon@lri.fr).

A. Capone is with the Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano, Milan, Italy, (email: capone@elet.polimi.it).

Our key findings suggest that the theoretical bounds for CDN architectures lead to better performance than those obtained with CCN solutions, unless we assume that the CDN architecture is slow at re-acting to content popularity evolution. Another clear take-home message of our work is that we further confirm a result already suggested in other papers (i.e., [9]) on cache provisioning for CCN: it is better to deploy caching storage on a limited number of nodes rather than distributing it uniformly throughout the network. All the results that we obtained are confirmed by a thorough simulation campaign that we performed on different network topologies.

Our main contributions can be summarized as follows:

1) We formulate a novel optimization model to study the performance bounds of a Content-Centric Network, solving the joint object placement and routing problem, under a realistic, *time-varying* object popularity evolution scheme and with the most notable cache replacement policies [10].
2) We formulate an optimization model to represent a similar scenario in a Content-Delivery Network (CDN), where replica servers are distributed according to the *k-median* model [11]. We build the CDN model in a way such that we can control the speed of reaction of the network to content popularity changes.
3) We extend the ndnSIM simulator [12] to closely represent the CCN architecture. We then compare the simulated results and those obtained with the optimization models under the same parameterization, and for different cache replacement policies.
4) We extensively discuss the obtained results, and show that in most of the cases CCN does not provide significant benefits. In the considered scenarios, with the Netrail topology, CCN reduces the traffic more consistently than CDN only if this latter is at least 5 times slower to re-act to popularity changes. On the other hand, in the larger Géant topology, we observed that CDN reached 11% better performance than CCN even in the adverse case when we force CDN to be 15 times slower than CCN.

This paper is structured as follows: Sec. II describes the CCN and CDN paradigms and motivates the choices we made to evaluate the content distribution performance of the network. Sec. III describes the content popularity evolution model. Sec. IV illustrates the proposed optimization models used to study the performance bounds. Numerical results obtained solving these models are presented and discussed in Sec. V. In Sec. VI we discuss related works. Finally, concluding remarks are presented in Sec. VII.

II. Evaluating Content Distribution Performance

In this section, we describe relevant characteristics of Content-Centric (Sec. II-A) and Content-Delivery Networks (Sec. II-B) to evaluate and compare their content distribution performance. Finally, in Sec. II-C we illustrate and motivate the methodology we used to perform such comparison.

A. Content-Centric Networks

We first introduce the features provided by Content-Centric Networks (CCN) that are relevant for our study. A comprehensive description of CCN can be found in [8].

In the literature, several network designs such as [13]–[15] are presented as "Information-Centric Networks" (ICNs); however, in this paper we focus our attention on the proposal known under the name of "Content-Centric Networking" (CCN) [8], since it is, to the best of our knowledge, the architecture that has so far received most of the attention from the community.

The communication model proposed by CCN replaces IP host addresses with content names: rather than stating the location *where* the data can be found, in CCN nodes declare *what* information they would like to retrieve. CCN has two distinguished packet types: (1) *Interest* and (2) *Data* packets. The former does not contain the actual data, but it only declares that a node is willing to access a given object whose name is known.

The structure of a CCN router is characterized by three tables: (1) the *Pending Interests Table* (PIT); (2) the *Content Store* (CS) and (3) the *Forwarding Interest Table* (FIB).

The PIT is responsible for memorizing the list of Interests previously forwarded, but not yet answered. Interests might arrive from physical hardware interfaces as well as logical applications running on the node itself and called "*faces*". The PIT stores the faces from which Interests were originally received, in order to implement *reverse path forwarding*: as soon as a router receives a Data packet, it checks the PIT and forwards the packet on the same faces from which Interests for that object arrived. The CS is the data structure used to implement *universal in-network caching*. When an Interest arrives, the router will initially query the CS and, in case of a cache hit, it will be able to directly serve the data. The FIB comes into play when a cache miss happens: it contains the next-hop information for prefix names. In this context, a *Consumer* is a node that requests some content to the network, while a *Producer* is a node that can reply to an Interest by directly providing the associated Data packet.

An example showing the behavior of a CCN router is depicted in Fig. 1. In State 1, the router receives two Interests and one Data packet. As shown in State 2, the Interest for object `/prefix/obj1` is served by the router since it is available in its CS. The Interest for `/prefix/obj2` will be forwarded to Face 3 since it is the destination available in the FIB. Lastly, the Data packet for `/prefix/obj3` will be forwarded to Face 0, as written in the PIT.

In a more realistic scenario, a CCN node may cache only the subset of contents that traversed the node at some point in time. Moreover, we assume that the owner of a given CCN node will not be remunerated for caching specific objects, but it will experience economic savings through bandwidth offloading, by forwarding upstream only the subset of incoming requests that lead to a cache miss. Lastly, it is reasonable to support the idea that CCN nodes belonging to different autonomous systems will be run by different owners, a feature that makes in-network caching be fully decentralized.

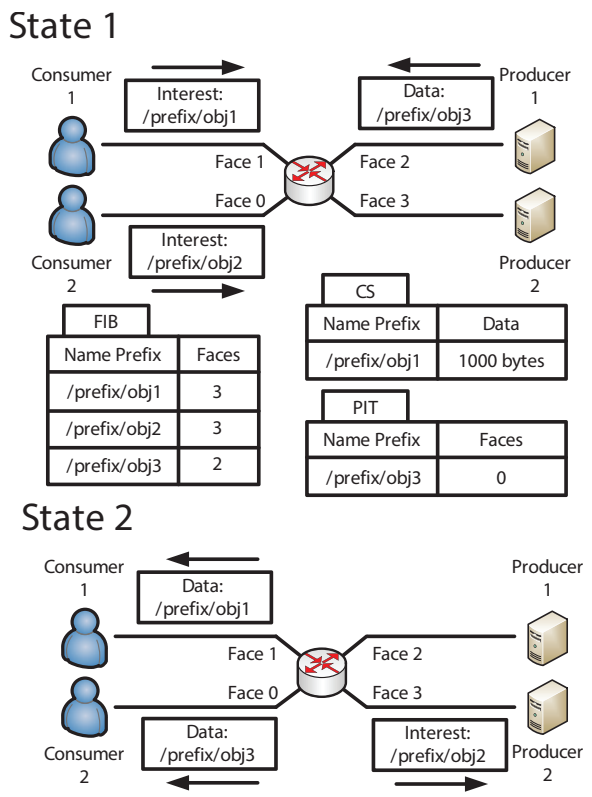


Figure 1. Example illustrating the behavior of a CCN router. Two Interests and one Data packet are received by the router in *State 1*. Given the information contained in the Pending Interests Table (PIT), the Content Store (CS) and the Forwarding Information Base (FIB), in *State 2* the router forwards two Data packets and one Interest.

B. Content-Delivery Networks

A Content-Delivery Network (CDN) is a communication infrastructure composed by a set of machines, known as *surrogate* (or *replica*) servers, geographically distributed in many *Points-of-Presence* (PoPs), to efficiently serve copies of given contents to nearby users [7], [16].

Despite the fact that the CDN infrastructure is massively distributed on a global scale, all the machines belong to a single owner that runs the network with the precise aim to sell the distribution services it offers. Therefore, since the CDN owner controls the whole infrastructure, he is capable to push any content he wants to any node, as well as choose which replica server should satisfy a given content request.

However, at the same time, pushing a content to a given location in the CDN infrastructure involves a computational and transmission overhead that instead has no counterpart in the CCN network. Therefore, while it is theoretically possible to frequently update the content cached on a CCN node, to perform a *fair* comparison, we must take into accurate account the fact that a surrogate server in a CDN will refresh its content catalog at a *much slower rate*. We further assume that the CDN owner has already deployed replica servers by centrally optimizing their placement with standard optimization techniques.

C. Methodology to Evaluate the Content Distribution Capabilities of CCN and CDN Architectures

In order to compare the performance of these two types of networks, we have chosen to leverage *offline optimization* techniques and to further confirm the results by performing extensive *simulation campaigns*.

The main focus of our study is to evaluate the performance of CCN and CDN, considering a fair scenario that does not introduce biases neither in favor nor against one specific network paradigm. We assume that the most realistic case is the one where the object popularity evolves in time, making the traffic demand profile be expressed as an input parameter that is *time-variant*. In our models, time is a discrete quantity expressed as a finite set of time-slots, denoted with $\mathcal{T}$ and such that each $t \in \mathcal{T}$ has a fixed duration. Moreover, since we would like to perform such evaluation under different conditions of content popularity evolution (slow/medium/fast evolution speed), we should adopt a popularity evolution model that depends on few parameters (ideally only one), and that well represents the burstiness of object traffic demands.

The performance metric that we take into account is the *total traffic* exchanged in the network, which should be minimized. The models then perform optimal *object placement* and *routing* choices for both the CCN and CDN architectures.

A given amount of caching storage, denoted with S, is uniformly distributed on all the caching routers $r \in \mathcal{R}$ in the CCN model. On the other hand, in the CDN model, only CDN nodes have caching capabilities. We denote with $\mathcal{D}$ the set of CDN nodes, whose cardinality is restricted to be $N = |\mathcal{D}|$, where $N \leq |\mathcal{R}|$. The same total caching storage is distributed in the two networks, but, due to their lower cardinality, each CDN node will usually store more objects than those persisted in a CCN node.

CDN nodes are pre-allocated optimally using a well-known *replica server placement* model. This assumption realistically models a CDN, since its owner will choose to deploy the machines only in the locations where they are mostly useful.

Another clear design requirement that our models meet is that there must by a tunable parameter that lets us change the speed at which the CDN can adapt to sudden popularity changes. We call this parameter "*Relocation Time*" ($\mathcal{T}_r$). In our CDN model, the relocation time is a subset of consecutive time-slots $\mathcal{T}_r \subseteq \mathcal{T}$, and it is used to represent a time window under which CDN nodes cannot change the objects they persist in their storage. By setting $|\mathcal{T}_r| = 1$, we are forcing object relocation in CDN to have the same dynamics of the popularity evolution model. Since we can say that CCN can always promptly react to popularity changes, with $|\mathcal{T}_r| = 1$ CDN is "*as reactive as*" CCN; instead, when we set $|\mathcal{T}_r| = 10$, for instance, it means that CDN is 10 times slower than CCN.

Another relevant feature that our models take into account is the fact that a CDN is run by a single owner in a centralized manner: at a given point in time the owner can choose to create a replica of a given object on any node in the network, by pushing it towards that destination, for instance because it performed popularity forecasting and chose to pre-fetch an object on a given server. The same condition does not apply to the CCN model, where instead we restrict the objects that

a given router can cache to the subset of those it forwarded in the past.

Due to their inner differences, we strongly support the idea that CCN and CDN should complement each other rather than being perceived as two antagonist models. In particular, our numerical results give evidence that CCN by itself does not lead to better performance than the one that can be obtained using a CDN. At the same time, we recognize that the simplicity of CCN should reduce the management costs with respect to CDN, but such type of analysis is out of the scope of our work.

III. Content Popularity Evolution Model

This section discusses the content popularity evolution model we used to generate synthetic traffic demand traces. Many research papers (e.g., [17]–[19]) agree in supporting the idea that the content popularity dynamics is highly non-stationary, and characterized by a bursty and oscillatory behavior, mostly governed by exogenous events. In this subsection we will accurately describe the synthetic traffic model, based on the proposal of Ratkiewicz et al. [19], that we use to generate the input traffic demand to our proposed optimization and simulation models. Our focus is on finding ways to generate demands that well represent bursty behavior, in order to mimic non-linear popularity shifts for a fixed-size object catalog denoted with $\mathcal{O}$. Despite the fact that content churn[1] may increase the realism, it is a common assumption made in the CDN and CCN literature to consider a fixed-size content catalog [20], [21].

Time is discretized into a finite set of time-slots, denoted with $\mathcal{T}$. Each object $o \in \mathcal{O}$ has a time-dependent rank parameter r_o^t which describes how likely that object will be requested during $t \in \mathcal{T}$. The lower the rank of an object, the higher the likelihood that such content will receive requests in that time-slot. We assume all the time slots $t \in \mathcal{T}$ last for the same (and constant) amount of time and such that the popularity rank of every object in t does not change. In other terms, we assume that the popularity of every object can evolve only from one time slot to another.

[1]The churn is a measure of the number of individuals moving in or out a given collective over a specific period of time.

Algorithm 1: Popularity Evolution, Rank-shift Model

Input : $r_o^t, \mathcal{O}, \rho$
Output: r_o^{t+1}
1 **for** $o \in \mathcal{O}$ **do**
2 **if** *UniformRandom*$(0,1) \leq \rho$ **then**
3 $r' \Leftarrow$ UniformDiscreteRandom$(1, r_o^t)$;
4 **for** $o' \in \mathcal{O}$ **do**
 if $r' \leq r_{o'}^t \leq r_o^t$ **then**
 $r_{o'}^{t+1} \Leftarrow r_{o'}^t + 1$;
 end
  end
  $r_o^{t+1} \Leftarrow r'$;
  end
end

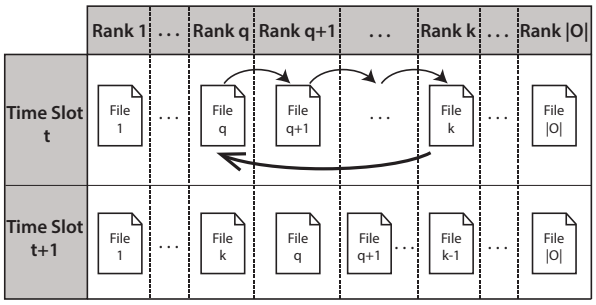


Figure 2. *The Rank-Shift Model*. In this example, at time slot $t+1$, the popularity of the k-th ranked file is updated to the value of q (randomly chosen). As a result of this popularity update, all the files from q to $k-1$ are shifted to a new popularity value of one rank less popular than the previous.

Given the object rank r_o^t, the Zipf discrete distribution is often used in the literature to represent the popularity of Internet contents, since it was shown that it is an adequate model for it [9], [21], [22]. The Zipf Probability Mass Function (PMF) and Cumulative Density Function (CDF) for the r_o^t popularity rank are defined as follows:

$$P(X = r_o^t) = \frac{\left(1/r_o^t\right)^\alpha}{\sum_{i=1}^{|\mathcal{O}|} \left(1/i^\alpha\right)}, \tag{1}$$

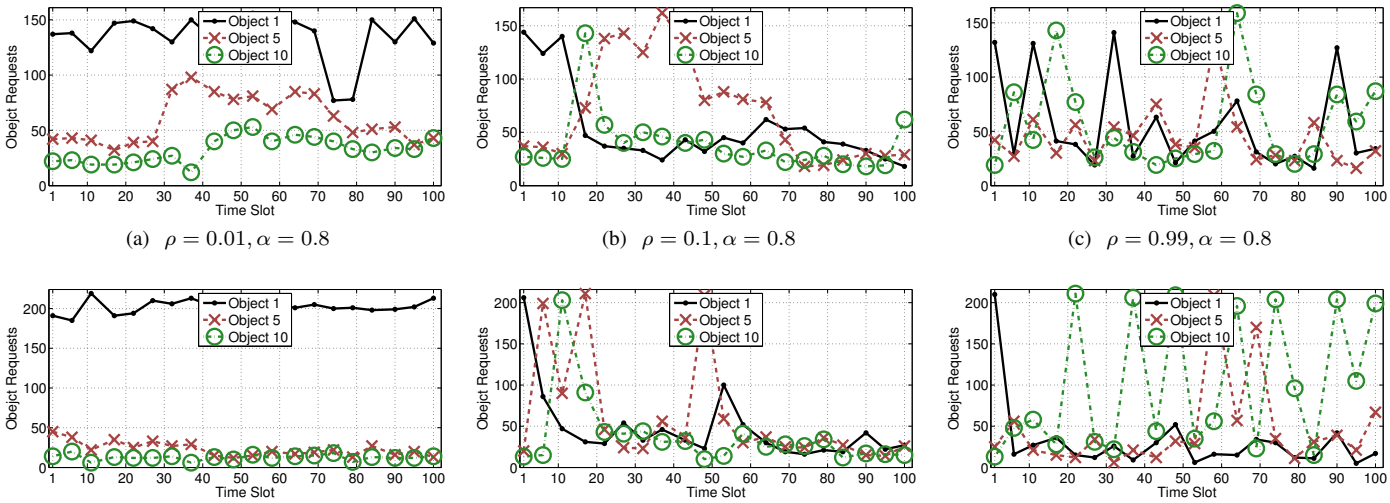
$$F(r_o^t) = \sum_{i=1}^{r_o^t} P(r_o^t) = \frac{\sum_{i=1}^{r_o^t} \left(1/i^\alpha\right)}{\sum_{i=1}^{|\mathcal{O}|} \left(1/i^\alpha\right)}. \tag{2}$$

The distribution is characterized by the popularity exponent α: the higher the α, the more skewed the requests are.

The time-dependent popularity dynamics is governed by the rank evolution parameter ρ in the 0 to 1 range. At each time slot, the ranking of a given object might evolve to become more popular in the immediate future. We control the speed at which this change happens through the usage of ρ: the higher the ρ value, the faster the popularity evolves. In the extreme cases, by setting $\rho = 0$ we neglect popularity evolution, whereas $\rho = 1$ removes temporal correlation of the requests. When the rank r_o^t of an object is updated, it makes the other objects' rank be shifted to a new (less popular) value.

Algorithm 1 illustrates the pseudocode used to compute the future object rank, given its current value. For each object (Step 1), with probability ρ (Step 2), the algorithm randomly selects a lower popularity class r' (Step 3), making the object suddenly become more popular. In Step. 4, the rank of the other objects is instead shifted to a reduced popularity level, in order to make sure that the popularity rank will be unique among all the objects. An example of object popularity evolution is shown in Fig. 2, where at time slot $t+1$, the popularity of the k-th ranked file is updated to the new value of q, randomly chosen. As a consequence of this choice, the popularity of the other objects evolves to a lower rank.

As illustrated in Figures 3a-3f, by changing the ρ and α parameters, we can easily mimic very different behaviors: a range of different content requests can be generated with such model, thus allowing us to well represent almost any type of request generation process. In particular, by comparing the absolute value of object requests in Fig. 3a-3c with the one of Fig. 3d-3f it is clear that by decreasing the α value, content



(a) $\rho = 0.01, \alpha = 0.8$ (b) $\rho = 0.1, \alpha = 0.8$ (c) $\rho = 0.99, \alpha = 0.8$

(d) $\rho = 0.01, \alpha = 1.2$ (e) $\rho = 0.1, \alpha = 1.2$ (f) $\rho = 0.99, \alpha = 1.2$

Figure 3. *Time-Dependent Object Request Evolution*. The figures plot the time-dependent object request evolution as a function of the Zipf α exponent and the rank evolution probability ρ, considering 100 time slots and 10 objects. The figures show the evolution of requests for the objects that in the first time slot have rank 1, 5 and 10, being 1 and 10 the most and least popular ranks, respectively.

requests are less polarized towards most popular objects. Furthermore, higher ρ values make the content popularity dynamics evolve faster.

IV. Network Models for Content Distribution

This section discusses the proposed optimization models that we use to study the performance of the CCN and CDN paradigms. Sec. IV-A presents the *Object Routing model* (OR) with *time-varying* demands. In Sec. IV-B and Sec. IV-C, we tailor the OR model to better represent relevant characteristics of CCN and CDN, respectively.

A. Object Routing Model with Time-Varying Demands

In this subsection we describe our proposed *Object Routing model* (OR) with *time-varying* demands. Such model well describes a TCP/IP network that does not have any caching functionality. In the following subsections we will further extend the OR model, taking into account relevant features that characterize CCN and CDN architectures. As summarized in Table I, three extensions of the OR model will be presented below:

1) *Object Allocation and Routing (OAR)*, a model tailored for CCN, that finds the optimal solution on the overall time horizon;
2) *OAR - Single Time Slot Heuristic (OAR-TS)*, tailored for CCN, which chooses the optimal solution for single time slots;
3) *OAR - Single Relocation Time CDN Heuristic (OAR-TR-CDN)*, which is the equivalent for CDN of OAR-TS.

We model the network as an undirected graph $G(V, E)$, where V is the set of nodes and E the set of edges. The set of nodes V is partitioned into three disjoint sub-sets: *consumers* (denoted by $\mathcal{C}$), *producers* (denoted by $\mathcal{P}$) and *routers* (denoted by $\mathcal{R}$), such that $V = \mathcal{C} \cup \mathcal{P} \cup \mathcal{R}$. Furthermore, we denote with $\mathcal{T}$ the set of time slots, whereas $\mathcal{O}$ represents the set of objects that can be retrieved from the network, also known as the *catalog*.

Consumers are nodes that express demands for objects in each time slot. We denote with $d^t_{c,o}$ the demand of consumer $c \in \mathcal{C}$ for object $o \in \mathcal{O}$ at time $t \in \mathcal{T}$. The demand is expressed in data size units (e.g., Mbytes); moreover, we assume that traffic demands are *inelastic*, meaning that they cannot be postponed to subsequent time slots.

In order to satisfy the demands, producers can serve the subset of objects they own. For each producer $p \in \mathcal{P}$ and object $o \in \mathcal{O}$, we denote with a^o_p the producer-object allocation, where:

$$a^o_p = \begin{cases} 1, & \text{if object } o \text{ is available at producer } p \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

All the routers in the network perform routing and forwarding. Considering the producer-router pairs, $\forall (p, r) \in \mathcal{P} \times \mathcal{R}$ we denote the corresponding link capacity with $b_{p,r}$. Similarly, $b_{r,c}$ denotes the router-consumer bandwidth $\forall (r, c) \in \mathcal{R} \times \mathcal{C}$, while b_{r_1,r_2} represents the router-router capacity $\forall (r_1, r_2) \in \mathcal{R} \times \mathcal{R}$.

We denote with $y^{t,o}_{p,r}$ the producer-router time-dependent flow variable $\forall (p, r, o, t) \in \mathcal{P} \times \mathcal{R} \times \mathcal{O} \times \mathcal{T}$; it represents the amount of data that producer $p \in \mathcal{P}$ sends to router $r \in \mathcal{R}$ for object $o \in \mathcal{O}$ in time slot $t \in \mathcal{T}$. Similarly, we define the router-consumer flow variable $\forall (r, c, o, t) \in \mathcal{R} \times \mathcal{C} \times \mathcal{O} \times \mathcal{T}$ as $y^{t,o}_{r,c}$, and the router-router flow variable $\forall (r_1, r_2, o, t) \in \mathcal{R} \times \mathcal{R} \times \mathcal{O} \times \mathcal{T}$ with $y^{t,o}_{r_1,r_2}$. For the sake of clarity, Table II summarizes the notation we will use in our optimization models.

We begin by describing the *Object Routing* model we use to compute the optimal packet routing to minimize the overall network traffic. By solving this model, we determine the performance bound of a network that does not support any

Table I
SUMMARY OF THE NETWORK OPTIMIZATION MODELS WE PROPOSE IN THIS PAPER.

Model Name	Optimal Routing	Supports Caching	Caching Strategy	Time Horizon	Network Type	Section
Object Routing (OR)	✓	✗	-	Many Time Slots	TCP/IP Network	IV-A
Object Allocation and Routing (OAR)	✓	✓	Optimal	Many Time Slots	CCN	IV-B
OAR - Single Time Slot Heuristic (OAR-TS)	✓	✓	Optimal LFU Random	Single Time Slot	CCN	IV-B
OAR - Single Relocation Time CDN Heuristic (OAR-TR-CDN)	✓	✓	Optimal	Single Time Slot	CDN	IV-C

type of caching functionality. In the following sections (viz., Sec. IV-B and IV-C) we will extend the OR model in order to describe the behavior of a CCN and a CDN architecture, respectively. Given the above definitions and assumptions, we formulate the optimal *Object Routing* model (OR) with *time-varying* demands as follows:

$$\min \sum_{\substack{\forall o\in\mathcal{O}\\ \forall t\in\mathcal{T}}} \left(\sum_{\substack{\forall r_1\in\mathcal{R}\\ \forall r_2\in\mathcal{R}}} y_{r_1r_2}^{o,t} + \sum_{\substack{\forall p\in\mathcal{P}\\ \forall r\in\mathcal{R}}} y_{pr}^{o,t} + \sum_{\substack{\forall r\in\mathcal{R}\\ \forall c\in\mathcal{C}}} y_{rc}^{o,t} \right) \tag{4}$$

subject to:

$$\sum_{\forall c\in\mathcal{C}} y_{r_1,c}^{o,t} + \sum_{\forall r_2\in\mathcal{R}} y_{r_1,r_2}^{o,t} = \sum_{\forall p\in\mathcal{P}} y_{p,r_1}^{o,t} + \sum_{\forall r_2\in\mathcal{R}} y_{r_2,r_1}^{o,t} \quad \forall (r_1,o,t)\in\mathcal{R}\times\mathcal{O}\times\mathcal{T} \tag{5}$$

$$\sum_{\forall o\in\mathcal{O}} y_{r_1r_2}^{o,t} \le b_{r_1,r_2} \quad \forall (r_1,r_2,t)\in\mathcal{R}\times\mathcal{R}\times\mathcal{T} \tag{6}$$

$$\sum_{\forall o\in\mathcal{O}} y_{p,r}^{o,t} \le b_{p,r} \quad \forall (p,r,t)\in\mathcal{P}\times\mathcal{R}\times\mathcal{T} \tag{7}$$

$$\sum_{\forall o\in\mathcal{O}} y_{r,c}^{o,t} \le b_{r,c} \quad \forall (c,r,t)\in\mathcal{C}\times\mathcal{R}\times\mathcal{T} \tag{8}$$

$$\sum_{\forall r\in\mathcal{R}} y_{r,c}^{o,t} = d_c^{t,o} \quad \forall (c,o,t)\in\mathcal{C}\times\mathcal{O}\times\mathcal{T} \tag{9}$$

$$y_{p,r}^{o,t} \le b_{p,r}\cdot a_{p,o} \quad \forall (p,r,o,t)\in\mathcal{P}\times\mathcal{R}\times\mathcal{O}\times\mathcal{T} \tag{10}$$

$$y_{r_1r_2}^{o,t} \ge 0 \quad \forall (r_1,r_2,o,t)\in\mathcal{R}\times\mathcal{R}\times\mathcal{O}\times\mathcal{T} \tag{11}$$

$$y_{pr}^{o,t} \ge 0 \quad \forall (p,r,o,t)\in\mathcal{P}\times\mathcal{R}\times\mathcal{O}\times\mathcal{T} \tag{12}$$

$$y_{rc}^{o,t} \ge 0 \quad \forall (r,c,o,t)\in\mathcal{R}\times\mathcal{C}\times\mathcal{O}\times\mathcal{T}. \tag{13}$$

The objective function (4) minimizes the total traffic transferred across all network links.

The set of constraints (5) imposes the flow balance condition at each router. Constraints (6), (7) and (8) bound the total link bandwidth that can be used on router-router, producer-router and router-consumer links, respectively. The set of constraints (9) makes sure that the network satisfies, in each time slot, all consumers' demand. Producers can only serve the subset of objects they possess, and this condition is expressed by the set of constraints (10). Lastly, constraints (11), (12) and (13) impose that router-router, producer-router and router-consumer flows are non-negative.

As represented in the OR model, the behavior of the network in each time slot is such that it does not depend on the solution obtained in other time slots. Therefore, it is possible to speed-up the model resolution by solving the routing problem independently in each time slot, and then aggregating the solution to compute the final objective function value.

Table II
SUMMARY OF THE NOTATION USED IN OUR OPTIMIZATION MODELS

Parameters	
$\mathcal{C}$	Set of Consumers
$\mathcal{P}$	Set of Producers
$\mathcal{R}$	Set of Routers
$\mathcal{O}$	Set of Objects
$\mathcal{T}$	Set of Time Slots
d_c^{to}	Traffic demand generated by consumer c for object o in time slot t
b_{rc}	Link capacity between router r and consumer c
b_{pr}	Link capacity between producer p and router r
$b_{r_1r_2}$	Link capacity between router r_1 and router r_2
a_p^o	0-1 parameter to indicate if producer p is publishing object o
S	Maximum number of objects that each router can cache
Q	A large number
N	Maximum number of CDN nodes that can be deployed
m	The maximum memory that a CDN node can use for caching
M	The total memory shared by all CDN nodes for caching

Decision Variables	
x_r^{to}	0-1 variable indicating if router r is caching object o at time t
$y_{r_1r_2}^{to}$	Data flow of object o from router r_1 to the neighbor router r_2, during time slot t
y_{pr}^{to}	Data flow of object o from producer p to router r during time slot t
y_{rc}^{to}	Data flow related to object o from router r to consumer c during time slot t
l_r	0-1 variable indicating if a CDN node is installed in router r
$z_r^{o,t}$	0-1 variable indicating if CDN node at router r caches object o during time slot t

B. Model for Content-Centric Network

In this section, we present the *Object Allocation and Routing* (OAR) model, an extension of the OR model tailored for a CCN. Due to the high computational complexity to find the optimal solution on the overall time history and the assumption that future demands should be known, we then formulate a *single time-slot heuristic* for the OAR model (OAR-TS) that solves both these two issues.

The OAR model extends the OR model adding *caching capabilities* to the routers. Given router $r\in\mathcal{R}$, object $o\in\mathcal{O}$ and time slot $t\in\mathcal{T}$, the router cache state is denoted by the binary variable $x_r^{t,o}$, which is such that:

$$x_r^{t,o} = \begin{cases} 1, & \text{if object } o \text{ is cached at router } r \text{ at time slot } t \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

In particular, we model a network where the cache is uniformly spread among all the nodes. Our model solves the *joint* optimal Object Allocation and Routing (OAR) problem, where the consumers express a *time-varying demand*. We can therefore formulate the mixed integer linear programming (MILP) model for OAR as follows:

$$\min \sum_{\substack{\forall o\in\mathcal{O}\\ \forall t\in\mathcal{T}}} \left(\sum_{\substack{\forall r_1\in\mathcal{R}\\ \forall r_2\in\mathcal{R}}} y^{o,t}_{r_1r_2} + \sum_{\substack{\forall p\in\mathcal{P}\\ \forall r\in\mathcal{R}}} y^{o,t}_{pr} + \sum_{\substack{\forall r\in\mathcal{R}\\ \forall c\in\mathcal{C}}} y^{o,t}_{rc} \right) \quad (15)$$

subject to constraints (6)-(13), and:

$$\sum_{c\in\mathcal{C}} y^{o,t}_{r_1c} + \sum_{r_2\in\mathcal{R}} y^{o,t}_{r_1r_2} \le Q\cdot x^{o,t}_{r_1} + \sum_{r_2\in\mathcal{R}} y^{o,t}_{r_2r_1} + \sum_{p\in\mathcal{P}} y^{o,t}_{pr_1} \quad \forall (r_1,o,t)\in\mathcal{R}\times\mathcal{O}\times\mathcal{T} \quad (16)$$

$$x^{o,t}_r \le x^{o,t-1}_r + \sum_{\forall r_2\in\mathcal{R}} y^{o,t-1}_{r_2,r} + \sum_{\forall p\in\mathcal{P}} y^{o,t-1}_{pr} \quad \forall (r,o,t)\in\mathcal{R}\times\mathcal{O}\times(\mathcal{T}\setminus\{0\}) \quad (17)$$

$$\sum_{\forall o\in\mathcal{O}} x^{o,t}_r \le S \qquad \forall (r,t)\in\mathcal{R}\times\mathcal{T} \quad (18)$$

$$x^{o,0}_r = 0 \qquad \forall (r,o)\in\mathcal{R}\times\mathcal{O} \quad (19)$$

$$x^{o,t}_r \in \{0,1\} \qquad \forall (r,o,t)\in\mathcal{R}\times\mathcal{O}\times\mathcal{T}. \quad (20)$$

The objective function (15) is the same as the one proposed for the OR model. The set of constraints (16) imposes flow-balance at each router, by also taking into account its caching capabilities. In particular, Q is a large number such that, when router r_1 is caching object o at time t (that is, $x^{o,t}_{r_1}=1$), r_1 can directly serve all the incoming requests for that object.

In CCN, each node acts independently from all the others by choosing which content it should store according to the *local* information available. This means that each node can choose to store a content in its local cache if and only if in the previous time slot it has forwarded such object, or if it was already caching such data. This constraint is imposed by (17) on all the caching routers in the network.

In (18), we make sure that each caching router stores at most S objects in its local cache. In (19), we make sure that the caches are empty at the initial time slot (in a sort of initialization step); finally, the set of constraints (20) forces the variable $x^{t,o}_r$ to be binary. It is important to note that the OAR model supports multipath packet routing, a native feature provided by CCN, and it also supports *off-path* caching. In fact, each router chooses the face on which packets should be forwarded knowing also the availability of content replicas in the caching storage of all the other nodes. Supporting off-path caching is in line with our goal of computing the performance bound of the CCN network.

Due to the fact that CCN routers can cache only the subset of objects they have seen in the immediate past, current network behavior strongly influences possible future choices, making every time slot be linked to all the others, as modeled by constraints (17). However, finding the optimal solution over the complete set of time slots is computationally very expensive, since the model has to consider all the demands on a global scale. Moreover, this optimization strategy assumes that the model can perform the optimal choice by knowing also *"future"* demands.

As shown by numerical results presented in Sec. V-A, knowing the future (as OAR does) seems to provide only small improvements of the objective function when compared to the alternative case where the solver knows the current demands and the previous state of the system. We call this latter model OAR *Single Time Slot Heuristic* (OAR-TS). While OAR can only be solved by setting a small catalog size and considering few time-slots, OAR-TS lets us consider much larger scenarios. The OAR-TS model is illustrated hereafter:

$$\min \sum_{\forall o\in\mathcal{O}} \left(\sum_{\substack{\forall r_1\in\mathcal{R}\\ \forall r_2\in\mathcal{R}}} y^{o}_{r_1r_2} + \sum_{\substack{\forall p\in\mathcal{P}\\ \forall r\in\mathcal{R}}} y^{o}_{pr} + \sum_{\substack{\forall r\in\mathcal{R}\\ \forall c\in\mathcal{C}}} y^{o}_{rc} \right) \quad (21)$$

subject to:

$$\sum_{c\in\mathcal{C}} y^{o}_{r_1c} + \sum_{r_2\in\mathcal{R}} y^{o}_{r_1r_2} \le Q\cdot x^{o}_{r_1} + \sum_{r_2\in\mathcal{R}} y^{o}_{r_2r_1} + \sum_{p\in\mathcal{P}} y^{o}_{pr_1} \quad \forall (r_1,o)\in\mathcal{R}\times\mathcal{O} \quad (22)$$

$$\sum_{\forall o\in\mathcal{O}} y^{o}_{r_1r_2} \le b_{r_1,r_2} \qquad \forall (r_1,r_2)\in\mathcal{R}\times\mathcal{R} \quad (23)$$

$$\sum_{\forall o\in\mathcal{O}} y^{o}_{p,r} \le b_{p,r} \qquad \forall (p,r)\in\mathcal{P}\times\mathcal{R} \quad (24)$$

$$\sum_{\forall o\in\mathcal{O}} y^{o}_{r,c} \le b_{r,c} \qquad \forall (c,r)\in\mathcal{C}\times\mathcal{R} \quad (25)$$

$$\sum_{\forall r\in\mathcal{R}} y^{o}_{r,c} = d_{c,o} \qquad \forall (c,o)\in\mathcal{C}\times\mathcal{O} \quad (26)$$

$$y^{o}_{p,r} \le b_{p,r}\cdot a_{p,o} \qquad \forall (p,r,o)\in\mathcal{P}\times\mathcal{R}\times\mathcal{O} \quad (27)$$

$$\sum_{\forall o\in\mathcal{O}} x^{o}_r \le S \qquad \forall r\in\mathcal{R} \quad (28)$$

$$x^{o}_r \le K^{o}_r \qquad \forall (o,r)\in\mathcal{O}\times\mathcal{R} \quad (29)$$

$$y^{o}_{r_1r_2} \ge 0 \qquad \forall (r_1,r_2,o)\in\mathcal{R}\times\mathcal{R}\times\mathcal{O} \quad (30)$$

$$y^{o}_{pr} \ge 0 \qquad \forall (p,r,o)\in\mathcal{P}\times\mathcal{R}\times\mathcal{O} \quad (31)$$

$$y^{o}_{rc} \ge 0 \qquad \forall (r,c,o)\in\mathcal{R}\times\mathcal{C}\times\mathcal{O} \quad (32)$$

$$x^{o}_r \in \{0,1\} \qquad \forall (r,o)\in\mathcal{R}\times\mathcal{O}. \quad (33)$$

The objective function (21) as well as constraints (22)-(28) and (30)-(33) can easily be derived from the corresponding constraints of the OAR formulation, by removing the time-slot index.

We model time-slots relations in constraints (29), where in each time slot $t\in\mathcal{T}$ we use the binary parameter K^{o}_r to impose restrictions on the subset of objects that a given router $r\in\mathcal{R}$ can cache. In particular, we emulate the original behavior studied in the OAR formulation by setting $K^{o}_r=0$ for the initial slot $t=1$, whereas for the other time slots $t>1$, we set $K^{o}_r=1$ if, in $t-1$, $x^{o}_r=1 \vee \sum_{\forall r1\in\mathcal{R}} y^{o}_{r1,r}>1$, otherwise we set $K^{o}_r=0$. Such condition checks whether in the previous time slot the router was already caching object o, or it forwarded at least one traffic unit for it.

By using other strategies to set the K^{o}_r parameter we can find tighter performance bounds for *real caching policies*; in particular we can emulate the "*Least Frequently Used*" (LFU) policy, forcing caches to store the objects that are requested

more frequently, as well as the "Random" caching policy, making caches store the objects randomly [23].

More in depth, we emulate the LFU policy by computing Ω_t^{ro}, the cumulative traffic that router $r \in \mathcal{R}$ has forwarded for a given object $o \in \mathcal{O}$ up to time slot $t \in \mathcal{T}$, which is defined as: $\Omega_t^{ro} = \Omega_{t-1}^{ro} + \sum_{r_2 \in \mathcal{R}} y_{r,r_2}^o + \sum_{c \in \mathcal{C}} y_{r,c}^o$. For each router, we sort the Ω_t^{ro} values in decreasing order; we then remove all those values referred to objects $o \in \mathcal{O}$ that the router has not seen in the previous time slot $t-1$, i.e. all those objects such that the condition $x_r^o = 1 \vee \sum_{\forall r1 \in \mathcal{R}} y_{r1,r}^o > 1$ does not hold. Let $\Omega_{S,t}^r$ be the S-th element of such sorted list. We set $K_r^o = 1$ if $\Omega_t^{ro} \geq \Omega_{S,t}^r$, otherwise it is set to zero. As a result, in $t+1$ the S most frequently requested objects are stored in the cache of the r-th router, according to its local cumulative traffic data.

The random cache policy can be implemented using even an easier algorithm: we randomly sample S objects such that for each of them, $o \in \mathcal{O}$, the following condition holds: $x_r^o = 1 \vee \sum_{\forall r1 \in \mathcal{R}} y_{r1,r}^o > 1$.

Computing instead a better optimization model for the LRU policy is quite involving: since we aggregate all the traffic in the entire time slot we lose all information regarding the sequence of requests received by the routers, and for such motivation we do not attempt to model this caching policy, which will instead be thoroughly studied in our simulation analysis.

C. Model for Content-Delivery Network

In this section, we illustrate the model used to find the performance bound of a CDN architecture. Since we are mostly interested in studying the performance of the network *after* the physical deployment of the nodes, we assume that the replica server placement problem has already been solved *a-priori*. As discussed in the related work section (VI-A), many proposals to solve the placement problem have already been formulated in the literature, and among all of them we use the *k-median* model [11] since it is a simple strategy that only depends on the nodes distribution in the network.

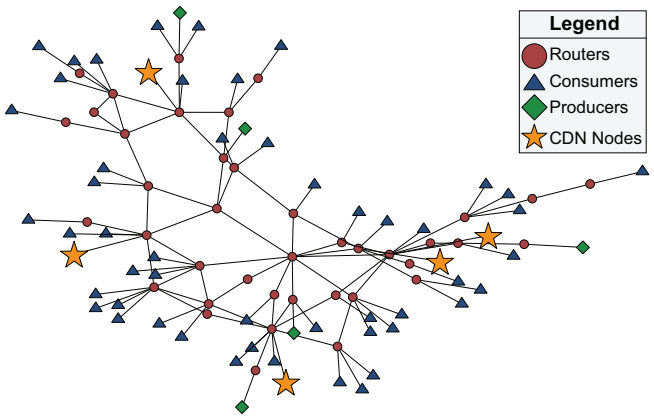


Figure 4. *K-median model*. Example illustrating the placement of 5 CDN replica servers according to the solution of the *k-median model*, in the Géant network topology, with uniform distribution of 50 consumers and 10 producers.

An example showing how the *k-median* model distributes the CDN nodes in the Géant network topology [24] is shown in Fig. 4. In such example, 50 consumers and 5 producers are spread uniformly in the network that is composed of 40 routers, while the *k-median* solution distributes 5 CDN nodes placing them in positions close to the consumers' locations.

Once that we have found the optimal CDN replica server allocation, we can then solve the *joint object placement and request routing problem*, as we did in the previous section with the OAR and the OAR-TS model. Two important features that our optimization model accurately takes into account are:

(1) Each CDN node can cache whatever content available. In fact, the CDN owner can optimize such allocation pushing whatever content on whatever replica server.
(2) Since this content migration is quite costly, the objects stored in a CDN node evolve with slower frequency than the one that we can get for CCN.

It is straightforward to address requirement (1), since we can simply relax a constraint in the OAR-TS model formulation. On the other hand, in order to enforce requirement (2), we introduce another concept that we call "*Relocation Time*", $\mathcal{T}_r$. The relocation time is a subset of consecutive time-slots $\mathcal{T}_r \subseteq \mathcal{T}$ during which the content cached in each CDN node is "frozen" and cannot be changed.

The set of CDN nodes is denoted with $\mathcal{D}$. We then formulate the *Optimal Allocation and Routing, Single Relocation Time Heuristic* model for CDN (OAR-TR-CDN) as follows:

$$\min \sum_{\forall o \in \mathcal{O}} \left(\sum_{\substack{\forall r_1 \in \mathcal{R} \\ \forall r_2 \in \mathcal{R}}} y_{r_1 r_2}^o + \sum_{\substack{\forall p \in \mathcal{P} \\ \forall r \in \mathcal{R}}} y_{pr}^o + \sum_{\substack{\forall r \in \mathcal{R} \\ \forall c \in \mathcal{C}}} y_{rc}^o \right) \tag{34}$$

subject to:

$$\sum_{\forall c \in \mathcal{C}} y_{r_1,c}^o + \sum_{\forall r_2 \in \mathcal{R}} y_{r_1,r_2}^o = \sum_{\forall d \in \mathcal{D}} y_{d,r_1}^o + \sum_{\forall p \in \mathcal{P}} y_{p,r_1}^o + \sum_{\forall r_2 \in \mathcal{R}} y_{r_2,r_1}^o \quad \forall (r_1, o) \in \mathcal{R} \times \mathcal{O} \tag{35}$$

$$\sum_{\forall o \in \mathcal{O}} y_{r_1 r_2}^o \leq b_{r_1,r_2} \cdot |\mathcal{T}_r| \qquad \forall (r_1, r_2) \in \mathcal{R} \times \mathcal{R} \tag{36}$$

$$\sum_{\forall o \in \mathcal{O}} y_{p,r}^o \leq b_{p,r} \cdot |\mathcal{T}_r| \qquad \forall (p, r) \in \mathcal{P} \times \mathcal{R} \tag{37}$$

$$\sum_{\forall o \in \mathcal{O}} y_{r,c}^o \leq b_{r,c} \cdot |\mathcal{T}_r| \qquad \forall (c, r) \in \mathcal{C} \times \mathcal{R} \tag{38}$$

$$\sum_{\forall o \in \mathcal{O}} y_{d,r}^o \leq b_{d,r} \cdot B_d \qquad \forall (d, r) \in \mathcal{D} \times \mathcal{R} \tag{39}$$

$$\sum_{\forall r \in \mathcal{R}} y_{r,c}^o = \sum_{\forall t \in \mathcal{T}_r} d_c^{t,o} \qquad \forall (c, o) \in \mathcal{C} \times \mathcal{O} \tag{40}$$

$$y_{p,r}^o \leq a_{p,o} \cdot b_{p,r} \cdot |\mathcal{T}_r| \qquad \forall (p, r, o) \in \mathcal{P} \times \mathcal{R} \times \mathcal{O} \tag{41}$$

$$\sum_{\forall o \in \mathcal{O}} x_d^o \leq S' \qquad \forall d \in \mathcal{D} \tag{42}$$

$$y_{d,r}^o \leq B_d \cdot x_d^o \qquad \forall (d, r, o) \in \mathcal{D} \times \mathcal{R} \times \mathcal{O} \tag{43}$$

$$y_{r_1 r_2}^o \geq 0 \qquad \forall (r_1, r_2, o) \in \mathcal{R} \times \mathcal{R} \times \mathcal{O} \tag{44}$$

$$y_{pr}^o \geq 0 \qquad \forall (p, r, o) \in \mathcal{P} \times \mathcal{R} \times \mathcal{O} \tag{45}$$

$$y_{rc}^o \geq 0 \qquad \forall (r, c, o) \in \mathcal{R} \times \mathcal{C} \times \mathcal{O} \tag{46}$$

$$y_{d,r}^o \geq 0 \qquad \forall (r, d, o) \in \mathcal{R} \times \mathcal{D} \times \mathcal{O} \tag{47}$$

$$x_r^o \in \{0, 1\} \qquad \forall (r, o) \in \mathcal{R} \times \mathcal{O}. \tag{48}$$

The OAR-TR-CDN objective function (34) is similar to the one proposed for the other models, since we want to minimize the overall network traffic.

In (35) we set the flow balance constraints, while in (36), (37), (38) and (39) we set the bandwidth constraints on router-router, producer-router, router-consumer and CDN-router flows, respectively. The overall consumer demand should be satisfied, as enforced by constraints (40), where we aggregate consumers' demands on all the time-slots in the relocation time $t \in \mathcal{T}_r$. In (41) we force producers to offer the subset of objects they own.

The set of constraints (42) limits the available caching space at each CDN router, while in (43) we limit the subset of objects that a CDN node can serve to those that it is caching. Finally, non-negativity constraints for flow variables are imposed in (44)-(47), while in (48), we make sure that the x_r^o variable is binary.

The relocation time is a subset of consecutive time slots, and it is used in the OAR-TR-CDN model to aggregate the entire demand as if the model represented one single time-slot. For this reason we multiply in constraints (36)-(38) and (41) the bandwidth by $|\mathcal{T}_r|$, a non-dimensional quantity that is a multiple of time-slots included in the given relocation time. Such choice permits to reduce the computational time required to solve this problem.

Comments

The OAR-TS and OAR-TR-CDN models capture relevant characteristics of the CCN and CDN architectures, respectively. In particular, while in OAR-TS all the routers are equipped with caching storage, in OAR-TR-CDN only the subset of CDN nodes implement caching functionalities. Moreover, OAR-TS sets a constraint on the objects that each router can cache, restricting such subset to all the data that the router has forwarded in the previous time slot. On the other hand, CDN nodes can cache whatever content they want, but by setting a different relocation time we can control the speed at which the CDN reacts to popularity evolution. For the sake of simplicity, and due to their modest impact on the objective function, we make the assumption that adding an object replica to a CDN node does not consume available network resources.

Our analysis focuses on the performance bounds of CCN and CDN: in particular, we are not considering management cost savings that the CCN architecture can obtain with respect to CDN.

V. Numerical Results

This section presents the results obtained formulating the proposed optimization models in OPL and solving them using the CPLEX solver [25]. In Sec. V-A we compare the objective function obtained using OAR with the heuristic OAR-TS solution. Sec. V-B extensively presents and discusses the numerical results we recorded while comparing the performance bounds of CCN and CDN.

A. Comparison of OAR and OAR-TS

In this subsection we compare the results obtained using the OAR and OAR-TS models and show that, in the considered scenario, the bounds of the two models are very close to each other, even though OAR is computationally more demanding than OAR-TS. Unless stated otherwise, Table III summarizes the parameters used to perform the analysis.

The topology we consider for this analysis is the Netrail topology, but we reduced the catalog size to 10 objects, and we also restricted our analysis to 10 time slots only. These parameters let us compare the solution of OAR and OAR-TS in a reasonable amount of time. On the other hand, the OAR model cannot scale to larger scenarios due to the high computation time needed to solve it.

As portrayed in Fig. 5, the OAR and OAR-TS curves not only are very close to each other, but they also confirm that knowing future demands (as OAR does), can lead to better values of the objective function. However, in such scenario, the gap between the two curves does not exceed 4%, which supports our choice to use the OAR-TS heuristic to derive performance bounds in larger topologies.

B. Performance Comparison of CCN and CDN Using Optimization Models

In this section we present the results obtained performing an extensive evaluation of the models we designed. We considered 6 different network topologies, as shown in Table III. Hereafter, we present the results obtained with Netrail and Géant, which are the smallest and largest topologies considered in our analysis, respectively. The other trends observed for Abilene, Sprint, Claranet and Airtel are in between the extreme values obtained for Netrail and Géant.

As shown in Table III, in all the scenarios we consider 10 consumers and 5 producers spread uniformly in the network. The content catalog we take into account is composed of 10^7 objects partitioned into 100 popularity classes, where the average object size is 1 Mbyte, as done in [20]. Each consumer generates 40 Mbit/s of demand in each time slot. Object requests are distributed according to the rank-shift model discussed in Sec. III, and we consider the ρ exponent in the 10^{-4} to 0.99 range, while the alpha exponent of the Zipf distribution can either be $\alpha = 0.8$ or $\alpha = 1.2$, as done in [26]. We consider different cache sizes, such that they can store from

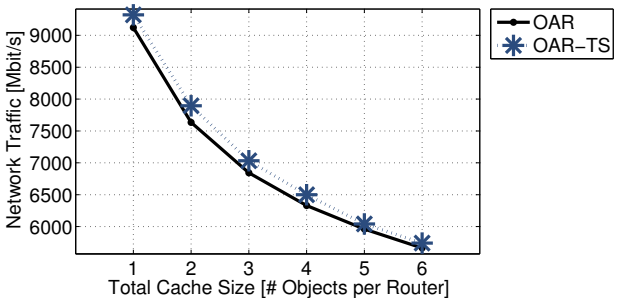


Figure 5. *Comparison of OAR and OAR-TS.* We solve the Netrail topology with a reduced catalog $|O| = 10$ and time-slot size $|\mathcal{T}| = 10$.

Table III
PARAMETERS OF THE NUMERICAL RESULTS

Numerical Results: Common Parameters	
Topologies	Netrail (7 nodes), Abilene (11 nodes) Sprint (11 nodes), Claranet (15 nodes) Airtel (16 nodes), Géant (40 nodes)
Number of Consumers	5
Number of Producers	10
Zipf α Exponent	$\{0.8, 1.2\}$
Content Popularity Evolution ρ	In the range from 10^{-4} to 0.99
Link Rate	500 Mbps
Consumer Demand	40 Mbps
Cache Size per Router	In the range 1-5% of the total catalog
Catalog Size	10^7 objects - 100 popularity classes
Object Size	1 Mbyte per object

Numerical Results: Optimization Models	
Number of Time Slots	100
CDN Relocation Time	In the range 1-15, default value: 3
Number of CDN nodes	In the range 1-10, default value: 3

Numerical Results: Simulations	
Cache Replacement Policies	Random (RND), Least Frequently Used (LFU), Least Recently Used (LRU)
Time Slot Size	1 second
Trace Length	1 hour

1 to 5% of the total number of objects available, as considered in other studies (e.g., [10], [20]). The number of CDN nodes we deploy is up to 10, whereas we consider a relocation time between 1 and 15, meaning that in our analysis the CDN might be "as fast as" a CCN or up to 15 times slower than that. If not stated otherwise, 3 CDN nodes will be deployed and they will have a relocation time equal to 3. We believe that these values can be used to perform a *fair* comparison of CCN and CDN: by deploying such a small number of CDN nodes we do not bias the analysis in favor of this latter architecture. The performance metric we consider is the total network traffic, since it is the objective function we took into account for the optimization models.

The behavior of the Netrail topology as a function of the cache size is shown in Fig. 6a (for $\alpha = 0.8, \rho = 0.99$), Fig. 6b (for $\alpha = 1.2, \rho = 0.99$) and Fig. 6c (for $\alpha = 1.2, \rho = 10^{-4}$). Considering the *No-Cache* curves for different combinations of α and ρ values (as in Fig. 6a-6c), we observe that a network that does not have caching functionalities leads to the same overall traffic (about $1.35 \cdot 10^5$ Mbit/s for Netrail), even for different popularity evolution parameters. By introducing caching functionalities, the total network traffic can be reduced significantly: in the Netrail topology, caching reduces traffic of at least 7% (Fig. 6c, CCN-Random caching policy, 1% cache size) and of at most 49% (Fig. 6c, CDN-Optimal caching policy, 5% cache size).

By deploying in Netrail 3 CDN nodes that have a relocation time set to 3, CCN and CDN reach almost the same traffic values, as depicted in Fig. 6a-6c. When caching storage increases, all the models lead to better performance; however, as shown in Fig. 6b, CDN seems to gain more than CCN as the amount of available caching storage increases.

An interesting observation on the random caching policy is that it leads to a total traffic that is rather independent with respect to the popularity evolution parameters α and ρ: the total traffic for CCN-Random is on average 10% lower than for No-Cache. On the contrary, the LFU caching policy is very sensitive to the value of ρ: the lower the speed at which the popularity evolves, the better the objective function is, as shown in Fig. 6b and 6c. In particular, while in Fig. 6b LFU scores on average 12% traffic reduction compared to No-Cache, in Fig. 6c the same performance gain raises to 39%.

The topology size has a strong impact on the results, in particular, caching benefits more larger topologies. In fact, if we compare the CDN curves for Géant (Fig. 6d-6f), with the corresponding values for Netrail (Fig. 6a-6c), we observe that they are very close to each other: Géant requires, on average, less than 6% more traffic than Netrail.

Another remarkable result regards the efficiency of the CDN architecture in different topologies. As a matter of fact, while in the small Netrail topology CCN and CDN almost exhibit the same performance, in Géant there is a large gap between the two solutions: on average, CDN reduces the total network traffic 25% more than what CCN does, leading to an overall 49% performance gain (on average), with respect to the total traffic transmitted in the No-Cache scenario.

The sensitivity of the objective function to the relocation time is depicted in Fig. 6g and 6j, for the Netrail and Géant topology, respectively. In both cases we deploy 3 CDN nodes. In the Netrail topology (Fig. 6g), if we assume that CDN nodes are 5 times slower than the CCN counterpart, the objective functions are almost the same in the two scenarios. On the other hand, for the Géant topology (Fig. 6j), even when we force CDN to be 15 times slower than CCN, the former architecture still leads to 11% better performance.

The ρ parameter that drives the speed of the content popularity evolution only marginally affects the objective function, as depicted in Fig. 6h for Netrail and Fig. 6k for Géant. The LFU policy is the one that is subject to the largest variation: by increasing the ρ value we reduce the locality of reference of the requests, and this, in turn, penalizes the LFU strategy.

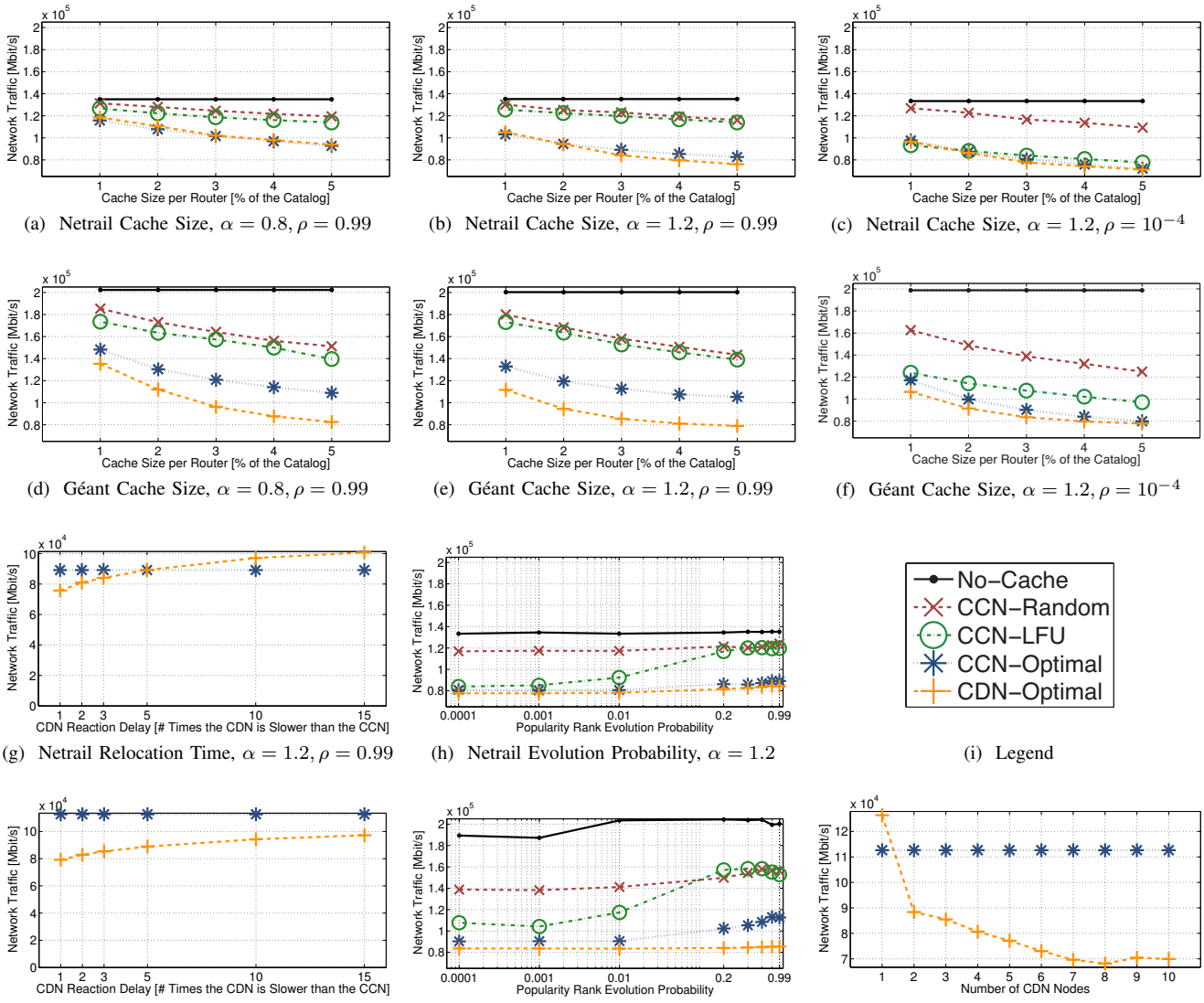
Finally, in Fig. 6l we observe that few CDN nodes (only 2 in the considered scenario) are sufficient to make CDN reach better performance values than those obtained with CCN.

C. Simulations Numerical Results

In this section, we present the methodology and the numerical results we obtained performing extensive simulation campaigns to further complement our analysis.

To this purpose, we extended the ndnSIM simulator [12] in order to accurately capture the behavior of a CCN architecture under the rank-shift model discussed in Sec. III. Compared to the analysis carried out using optimization models, simulations let us study real traffic performance values, rather than their theoretical bounds. In addition to that, through the usage of simulations we can assess the performance of the Least Recently Used (LRU) cache policy that otherwise would be neglected in our analysis.

Table III summarizes the simulation parameters used to perform our analysis. In particular, whenever possible, we used the same values used to solve the optimization models; in



(a) Netrail Cache Size, $\alpha = 0.8, \rho = 0.99$ (b) Netrail Cache Size, $\alpha = 1.2, \rho = 0.99$ (c) Netrail Cache Size, $\alpha = 1.2, \rho = 10^{-4}$

(d) Géant Cache Size, $\alpha = 0.8, \rho = 0.99$ (e) Géant Cache Size, $\alpha = 1.2, \rho = 0.99$ (f) Géant Cache Size, $\alpha = 1.2, \rho = 10^{-4}$

(g) Netrail Relocation Time, $\alpha = 1.2, \rho = 0.99$ (h) Netrail Evolution Probability, $\alpha = 1.2$ (i) Legend

(j) Géant Relocation Time, $\alpha = 1.2, \rho = 0.99$ (k) Géant Evolution Probability, $\alpha = 1.2$ (l) Géant CDN Nodes, $\alpha = 1.2, \rho = 0.99$

Figure 6. *Numerical Results of the Optimization Models*. The legend of Fig. 6i is common to all the plots. The figures show the obtained values for the Netrail and Géant topologies, considering a content catalog of 10^7 objects partitioned in 100 popularity classes, 100 time slots, 10 consumers and 5 producers.

such way, we can compare the results obtained using both methodologies.

We generate traffic traces of 1 hour. In order to eliminate any transient behavior, we remove the first 10 minutes from the traces, then we aggregate the traffic in intervals of 5 minutes each, and plot the average traffic value for all of them, computing the very narrow 95% confidence intervals shown for all the curves in Figures 7.

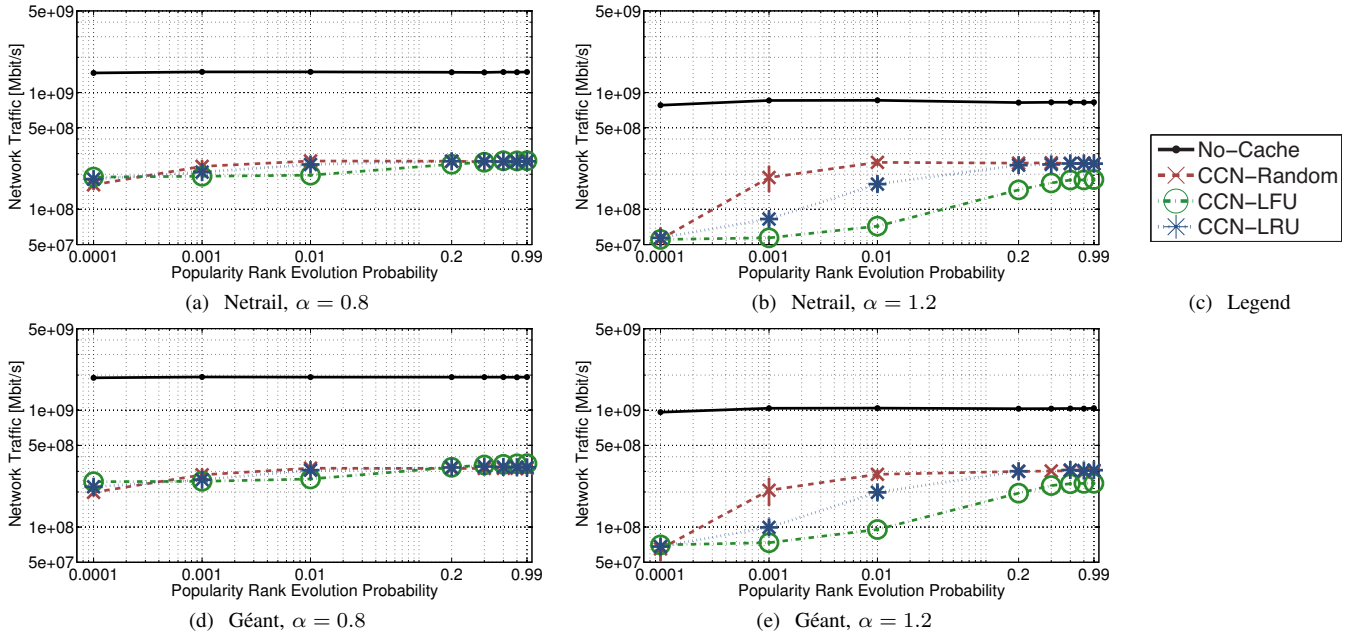
Figures 7a and 7b refer to the Netrail topology, whereas corresponding results for the Géant topology are depicted in Fig. 7d and 7e, respectively. First of all, the simulations confirm the same trends for the caching policies presented in the previous section. The total traffic for Géant reaches higher values than that obtained for Netrail. This behavior is due to the fact that the Géant network has a larger network diameter than Netrail, which raises the value of the traffic metric.

Simulations further confirm that the popularity evolution parameter (ρ) does not have any impact on a network that is not implementing caching functionalities, as shown by the No-Cache curves. We also observe that by using higher Zipf α exponents, caching can further lower the total traffic exchanged in the network, since content requests will be mostly concentrated on a small subset of the available objects that will likely be replicated in the distributed caches.

An interesting finding (in line with the results presented in [27]), is that different cache replacement policies do not lead to significantly distant performance results, especially with smaller values for the α exponent, (as shown in Fig. 7a and 7b). In addition to that, we also observe that the Least Frequently Used (LFU) policy performs usually better than the others, saving up to $5 \cdot 10^7$ Mbit/s more than the Random caching policy, when $\alpha = 1.2$.

VI. Related Work

This section provides a survey on Content-Centric and Content-Delivery Networks, as well as on content popularity



(a) Netrail, $\alpha = 0.8$ (b) Netrail, $\alpha = 1.2$ (c) Legend

(d) Géant, $\alpha = 0.8$ (e) Géant, $\alpha = 1.2$

Figure 7. *Simulation Results*. Simulated average traffic, in a 5 minute interval, as a function of the popularity evolution parameter, under different topologies and α values. Very narrow 95% confidence intervals are plotted for all the curves. The legend 7c is common for all the plots.

evolution models. More specifically, Sec. VI-A deals with CCN and CDN architectures, focusing on their performance analysis. Sec. VI-B reviews the most notable content popularity models, addressing also the problem of content popularity evolution.

A. Performance Analysis of Content-Centric and Content-Delivery Networks

Hereafter we review the most notable research works on CCN and CDN architectures, specifically focusing on the performance analysis of these networks.

Fayazbakhsh et al. study in [9] the performance of a Content-Centric Network specifically comparing the improvements achievable with an *edge-based caching* with respect to a *full-fledged* CCN. In particular, they show that most of the performance benefits can be gained by deploying caching storage only on the edge nodes, whereas by distributing the memory also on the other nodes they can achieve a modest 2% performance gain.

Thorough simulation campaigns are presented in [27], where Rossini and Rossi study the performance of CCN focusing on forwarding strategies and caching policies. They show that limited benefits can be achieved by adopting complex caching policies: randomized caching decisions can perform as well as more complex ones, while significantly reducing the computational overhead that they introduce. Moreover, they also show that multi-path forwarding capabilities may play against the network efficiency.

One of the scenarios where CCN can potentially boost the network performance is video distribution [28]–[30]. Due to the relatively small content catalog, large file sizes and the static nature of content (especially when compared to web pages), video distribution should take advantage of in-network caching capabilities of CCN, moving the content closer to the locations where most of the users are actually requesting it.

In order to satisfy video demands, nowadays other technological infrastructures known as Content-Delivery Networks (CDNs), built as overlays on top of TCP/IP, are exploited to serve this precise scope.

Adhikari et al. study in [31] the Netflix video streaming platform, by analyzing the video delivery performance in a scenario where many CDNs are used for video streaming purposes. In [32], Mansy and Ammar focus on a scenario where the CDN cooperates with P2P to serve *adaptive* video streaming requests. The authors show that such a hybrid scenario has interesting performance properties that can potentially reduce the costs paid by the content provider. Liu et al. have measured in [33] the performance of video distribution when clients leverage the CDN infrastructure to retrieve video content. In order to further improve the users' *quality of experience*, the authors suggest to adopt a control plane that automatically selects the best bit-rate and CDN server according to the network state.

In terms of performance optimization, three classic problems have to be solved in a CDN:

1) *Server Placement*: choose the locations where to place the CDN servers.
2) *Replica Object Placement*: choose the locations and the number of object replicas by distributing them on the available servers.
3) *Surrogate Server Selection*: route object requests by selecting a replica server storing a copy of the given object.

Without pretending to be exhaustive, hereafter we mention relevant works addressing each of the above-mentioned problems. In particular, the *center placement problem* is used to solve both server placement as well as the replica object

placement, by modeling the problem as a graph where a given distance metric should be minimized [11], [34]. The size of the problem, which becomes even larger when studying object placement, forces to formulate heuristic algorithms that find sub-optimal solutions, such as those presented in [35], [36]. Finally, in [37], surrogate server selection strategies used by YouTube are evaluated in order to understand which parameters may influence the CDN server selection process.

B. Content Popularity Models

In this section we review relevant works regarding content popularity evolution in the Internet. Due to its worldwide success, many research works have studied the popularity evolution of video contents [17], [38], [39].

In [38], Cha et al. have studied *Video-on-Demand* (VoD) systems for *User Generated Content* (UGC). The authors provide evidences that the popularity of UGC is ephemeral, and traditional content popularity prediction techniques are ineffective. In fact, while TV-broadcasters deliver the same content to all their users at the same time, VoD services, instead, let each customer choose which video she is going to watch.

Dán and Carlsson in [39], and Cha et al. in [38] observe that an exponential cutoff term should be added to a power-law model to better represent the content popularity. In [17], Figueiredo et al. provide evidences that the popularity evolution of a given UGC video depends also on the type of content it represents. In particular, copyright-protected materials tend to get most of their views very early in the lifetime, and sudden burst of popularity are very common among top-list videos.

Despite being very interesting, all the works reviewed so far cannot be easily used to generate *synthetic workloads*; instead, two models to produce synthetic traces have been presented in [18], [19].

Borghol et al. present in [18] a model that can be used to generate such synthetic workload to represent the popularity evolution of UGC. Their model splits the lifetime of a content in three stages: *before*, *at* or *after* the age at which the content reaches its peak popularity. The main shortcoming of this model is that it necessitates several parameters to be used in practice: 1) the content popularity distribution for each stage of the lifetime; 2) the content movement process across the different stages and 3) the consumer view rate distributions for contents belonging to each group.

Another model, that we adopted in this paper to generate realistic workloads in the presence of bursty popularity evolution, has been presented by Ratkiewicz et al. in [19], and has been be extensively discussed in Sec. III. Despite the fact that it models the popularity evolution with only one parameter, this proposal has many strengths: it is accurate (as shown by the authors themselves [19]) and, at the same time, simple.

VII. Conclusion

In this paper we compared CCN and CDN by formulating novel optimization models to analyze the performance bounds of these network architectures, considering time-varying demands to represent content popularity evolution. The proposed models are such that the relevant characteristics of CCN and CDN are explicitly taken into account.

Our numerical results suggest that the performance bounds for CDN architectures are better in terms of network traffic than those observed for CCN, even when few CDN replica servers are deployed in the network. In our considered scenarios, we observed that in the Netrail topology CCN can reach a better value for the objective function compared to CDN only if this latter is at least 5 times slower to react to popularity changes. On the other hand, while considering the larger Géant topology, we observed that CDN is always to be preferred since it reaches 11% better performance than CCN even in the adverse case when we force CDN to be 15 times slower than CCN.

Our analysis confirms the results presented in [9]: by concentrating available caching storage on few specific nodes, the network performance improves. By extending the ndnSIM simulator, we further complemented this work showing that simulations exhibit the same performance trends.

Our view is that rather than being two antagonist models, CCN and CDN should complement each other; in particular CCN is a very promising architecture to reduce the management overhead that the network introduces, whereas CDN is to be preferred if the main goal is to achieve the highest performance gains.

References

[1] M. Mangili, F. Martignon, and A. Capone, "A Comparative Study of Content-Centric and Content-Distribution Networks: Performance and Bounds," *Proceedings of IEEE Global Communications Conference (Globecom 2013)*, Atlanta, GA, USA, December 2013.

[2] G. Carofiglio, G. Morabito, L. Muscariello, I. Solis, and M. Varvello, "From content delivery today to information centric networking," *Computer Networks*, vol. 57, no. 16, pp. 3116 – 3127, 2013.

[3] A.-J. Su, D. R. Choffnes, A. Kuzmanovic, and F. E. Bustamante, "Drafting Behind Akamai: Inferring Network Conditions Based on CDN Redirections," *IEEE/ACM Trans. on Networking (TON)*, vol. 17, no. 6, pp. 1752–1765, 2009.

[4] J. H. Saltzer, D. P. Reed, and D. D. Clark, "End-to-end arguments in system design," *ACM Trans. on Computer Systems (TOCS)*, vol. 2, no. 4, pp. 277–288, Nov. 1984.

[5] "Cisco Press Release - Cisco Adds Carrier Routing System X (CRS-X) Core Router to Industry-Leading CRS Family, San Jose, USA, June, 2013," http://newsroom.cisco.com/release/1208192, Last accessed: Nov. 2013.

[6] A. Vakali and G. Pallis, "Content Delivery Networks: Status and Trends," *IEEE Internet Computing*, vol. 7, no. 6, pp. 68–74, 2003.

[7] E. Nygren, R. K. Sitaraman, and J. Sun, "The Akamai network: a platform for high-performance internet applications," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 3, pp. 2–19, July 2010.

[8] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking Named Content," in *Proc. of the 5th Int.l Conf. on Emerging networking experiments and technologies (CoNEXT)*. ACM, Rome, Italy, December 2009, pp. 1–12.

[9] S. K. Fayazbakhsh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. Maggs, K. Ng, V. Sekar, and S. Shenker, "Less Pain, Most of the Gain: Incrementally Deployable ICN," in *Proc. of the ACM SIGCOMM conference*, Hong Kong, China, August 2013, pp. 147–158.

[10] D. Rossi and G. Rossini, "On Sizing CCN Content Stores by Exploiting Topological Information," *IEEE INFOCOM, NOMEN Workshop*, pp. 280 – 285, Orlando, Florida, (USA), March 2012.

[11] L. Qiu, V. N. Padmanabhan, and G. M. Voelker, "On the placement of web server replicas," in *Proc. of IEEE INFOCOM Conference*, Anchorage, Alaska, (USA), April 2001, pp. 1587–1596.

[12] A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM: NDN simulator for NS-3," NDN, Technical Report NDN-0005, October 2012.

[13] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A Data-Oriented (and Beyond) Network Architecture," *SIGCOMM Computer Communication Review*, vol. 37, no. 4, pp. 181–192, Aug. 2007.

[14] N. Fotiou, P. Nikander, D. Trossen, and G. C. Polyzos, "Developing Information Networking Further: From PSIRP to PURSUIT," in *Broadband Communications, Networks, and Systems*, ser. LNCS, Social Informatics and Telecommunications Engineering. Springer, 2012, vol. 66, pp. 1–13.

[15] N. Niebert, S. Baucke, I. El-Khayat, M. Johnsson, B. Ohlman, H. Abramowicz, K. Wuenstel, H. Woesner, J. Quittek, and L. Correia, "The way 4WARD to the Creation of a Future Internet," in *Proc. of IEEE 19th Int.l Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC*, Cannes, France, September 2008, pp. 1–5.

[16] G. Pallis and A. Vakali, "Insight and perspectives for content delivery networks," *Commun. ACM*, vol. 49, no. 1, pp. 101–106, Jan. 2006.

[17] F. Figueiredo, F. Benevenuto, and J. M. Almeida, "The Tube Over Time: Characterizing Popularity Growth of Youtube Videos," in *Proc. of the 4th ACM Int.l Conf. on Web Search and Data Mining (WSDM)*, Hong Kong, February 2011, pp. 745–754.

[18] Y. Borghol, S. Mitra, S. Ardon, N. Carlsson, D. Eager, and A. Mahanti, "Characterizing and Modelling Popularity of User-enerated Videos," *Performance Evaluation*, vol. 68, no. 11, pp. 1037–1055, 2011.

[19] J. Ratkiewicz, F. Menczer, S. Fortunato, A. Flammini, and A. Vespignani, "Traffic in Social Media II: Modeling Bursty Popularity," in *Proc. of the 2nd Int.l IEEE Conf. on Social Computing (SOCIALCOM)*, Minneapolis, MN, USA, August 2010, pp. 393–400.

[20] G. Carofiglio, V. Gehlen, and D. Perino, "Experimental Evaluation of Memory Management in Content-Centric Networking," in *Proc. of the IEEE Int.l Conf. on Communications (ICC)*, Kyoto, Japan, June 2011, pp. 1–6.

[21] G. Pallis, "Improving Content Delivery by Exploiting the Utility of CDN Servers," in *Proc. of the 5th Int.l Conf. on Data Management in Cloud, Grid and P2P Systems (Globe)*. LNCS, Springer, Vienna, Austria, September 5-6, 2012, pp. 88–99.

[22] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: Evidence and implications," in *In Proc. of IEEE INFOCOM*, vol. 1, New York, USA, March '99, pp. 126–134.

[23] D. Applegate, A. Archer, V. Gopalakrishnan, S. Lee, and K. K. Ramakrishnan, "Optimal content placement for a large-scale VoD system," in *Proc. of the 6th Int.l Co-NEXT Conference*. ACM, Philadelphia, Pennsylvania, (USA), December 2010, pp. 4:1–4:12.

[24] "Géant Network Website," http://geant3.archive.geant.net/Network/NetworkTopology/pages/home.aspx, Last accessed: November 2013.

[25] P. Van Hentenryck, *The OPL optimization programming language*. Mit Press, 1999.

[26] C. Fricker, P. Robert, J. Roberts, and N. Sbihi, "Impact of traffic mix on caching performance in a content-centric network," *IEEE INFOCOM, NOMEN Workshop*, pp. 310–315, Orlando, Florida (USA), March, 2012.

[27] Giuseppe Rossini and Dario Rossi, "Evaluating CCN multi-path interest forwarding strategies," *Computer Communications*, vol. 36, no. 7, pp. 771 – 778, 2013.

[28] Z. Li and G. Simon, "Time-Shifted TV in Content Centric Networks: The Case for Cooperative In-Network Caching," in *Proc. of the IEEE International Conf. on Communications (ICC)*, Kyoto, Japan, June 2011.

[29] J. Choi, J. Han, E. Cho, T. Kwon, and Y. Choi, "A Survey on Content-Oriented Networking for Efficient Content Delivery," *IEEE Communications Magazine*, vol. 49, no. 3, pp. 121–127, 2011.

[30] S. Lederer, C. Mueller, B. Rainer, C. Timmerer, and H. Hellwagner, "An experimental analysis of Dynamic Adaptive Streaming over HTTP in Content Centric Networks," in *Proc. of IEEE Conf. on Multimedia and Expo (ICME)*, San Jose, California, (USA), July 2013, pp. 1–6.

[31] 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 *Proc. of IEEE INFOCOM*, Orlando, Florida, (USA), March 2012, pp. 1620–1628.

[32] A. Mansy and M. H. Ammar, "Analysis of adaptive streaming for hybrid CDN/P2P live video systems," in *Proc. of the 19th IEEE Int.l Conf. on Network Protocols ICNP*, 2011, pp. 276–285.

[33] X. Liu, F. Dobrian, H. Milner, J. Jiang, V. Sekar, I. Stoica, and H. Zhang, "A case for a coordinated internet video control plane," in *Proc. of ACM SIGCOMM*, Helsinki, Finland, August 2012, pp. 359–370.

[34] S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang, "On the placement of Internet instrumentation," in *Proc. of IEEE INFOCOM*, vol. 1, Tel-Aviv, Israel, March 2000, pp. 295–304.

[35] P. Krishnan, D. Raz, and Y. Shavitt, "The cache location problem," *IEEE/ACM Trans. on Networking (TON)*, vol. 8, no. 5, pp. 568–582, Oct. 2000. [Online]. Available: http://dx.doi.org/10.1109/90.879344

[36] S. Jamin, C. Jin, A. Kurc, D. Raz, and Y. Shavitt, "Constrained mirror placement on the Internet," in *Proc. of IEEE INFOCOM*, vol. 1, Anchorage, Alaska, (USA), April 2001, pp. 31–40.

[37] R. Torres, A. Finamore, J. R. Kim, M. Mellia, M. M. Munafo, and S. Rao, "Dissecting Video Server Selection Strategies in the YouTube CDN," in *Proc. of IEEE Conf. on Distributed Computing Systems (ICDCS)*, Minneapolis, Minnesota, (USA), June 2011, pp. 248–257.

[38] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "Analyzing the Video Popularity Characteristics of Large-Scale User Generated Content Systems," *IEEE/ACM Trans. on Networking*, vol. 17, no. 5, pp. 1357–1370, Oct. 2009.

[39] G. Dán and N. Carlsson, "Power-law Revisited: A Large Scale Measurement Study of P2P Content Popularity," in *Proc. of the Int.l Workshop on Peer-to-Peer Systems (IPTPS)*, San Jose, CA, USA, April 2010.

Michele Mangili received his M.S. degree in Computer Science and Engineering from the University of Bergamo in 2012. He is currently pursuing an international Ph.D. in Information Engineering at LRI (Laboratory for Computer Science), Paris-Sud University and DEIB, Politecnico di Milano. His current research interests are Content-Centric Networks, Network Modelling, Planning and Optimization. He is a student member of the IEEE Communication Society.

Fabio Martignon received the M.S. and the Ph.D. degrees in telecommunication engineering from the Politecnico di Milano in October 2001 and May 2005, respectively. He has been associate professor at University of Bergamo, and he is now Full Professor in LRI (Laboratory for Computer Science) at Paris-Sud University, and member of Institut Universitaire de France. His current research activities include cognitive radio networks, content-centric networks, network planning and game theory applications to networking problems.

Antonio Capone is Full Professor at the Information, Communication and Bioengineering Technology Department (Dipartimento di Elettronica, Informazione e Bioingegneria) of Politecnico di Milano, where he is the director of the Advanced Network Technologies Laboratory (ANTLab). Dr. Capone is co-founder and CTO of MobiMESH (www.mobimesh.eu), a spin-off company of Politecnico di Milano. His expertise is on networking and his main research activities include protocol design (MAC and routing) and performance evaluation of wireless access and multi-hop networks, traffic management and quality of service issues in IP networks, and network planning and optimization. He received the M.S. and Ph.D. degrees in electrical engineering from the Politecnico di Milano in 1994 and 1998, respectively. He currently serves as editor of IEEE/ACM Trans. on Networking, Wireless Communications and Mobile Computing (Wiley), Computer Networks (Elsevier), and Computer Communications (Elsevier). He is a Senior Member of the IEEE.