Computer Networks 110 (2016) 104–117

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Optimal multipath congestion control and request forwarding in information-centric networks: Protocol design and experimentation

Giovanna Carofiglio[a], Massimo Gallo[b], Luca Muscariello[a,*]

[a] Cisco Systems, 11 rue Camilles Desmoulins, 92130, Issy Les Moulineaux, France
[b] Bell Labs, Nokia, 7 Route de Villejust, 91620 Nozay, France

ARTICLE INFO

Article history:
Received 26 May 2015
Revised 24 January 2016
Accepted 18 September 2016
Available online 21 September 2016

Keywords:
Information-Centric networking
Multipath congestion control
Request forwarding

ABSTRACT

In this paper we consider the problem of joint congestion control and request forwarding in Information-Centric Networks, namely the named-data networking architecture (NDN). The network architecture we consider is based on information retrieval natively pull-based, driven by user requests, point-to-multipoint and intrinsically coupled with in-network caching. We formalize the problem as global optimization with non-linear objectives and linear constraints with the twofold objective of maximizing user throughput and minimizing overall network cost. We solve it via decomposition and derive a family of optimal congestion control strategies at the receiver and of distributed algorithms for dynamic request forwarding at network nodes. An experimental evaluation of our proposal is carried out in different network scenarios using realistic workloads, to assess the performance of our design and to highlight the benefits of an ICN approach. The experimentation is carried out using the NDN software router implementation on a large grid infrastructure deployed to enable experimental research.

1. Introduction

The TCP/IP core of the Internet architecture has remained stable over the last decades, while accommodating unexpected innovation at lower and upper layers. Recent advances in wireless and optical networking technologies have empowered heterogeneous mobile connectivity and boosted access network capacity. The diffusion of web services, cloud and social networks has in turn driven Internet usage towards information delivery and imposed a new communication model based on Information exchange, caching and real time processing.

Current information delivery solutions are deployed as overlays on top of the existing IP network infrastructure leveraging CDNs, transparent caching etc. However, the inefficiency of the overlay approach in terms of performance guarantees/ resource utilization has been pointed out in the research community, together with question of the sustainability of such evolution model [19,22,27,30]. Some evidence has been provided that "architectural anchors" like IP addressing, host-centric point-to-point communication, inherently curb the Information-driven network evolution.

A mismatch exists between the location addressing of IP and the location-agnostic content addressing by name realized by Information services (e.g. via HTTP). It calls for a paradigm shift of the Internet communication model as proposed by Information-Centric Networking (ICN) architectures [19,22,26,30,41]. The common objective is to embed content awareness into the network layer to enable efficient, mobile and connectionless information-oriented communication. The twist associated to ICN communication model lies on a small set of principles. ICN advocates a *name-based* communication, controlled by the receiver (*pull-based*), realized via name-based routing and forwarding of user requests in a *point-to-multipoint* fashion and supported by the presence of a pervasive *network-embedded caching* infrastructure.

The novel ICN's transport paradigm holds considerable promises for enhanced flexibility in terms of mobility management, improved end-user performance and network resources utilization. This is due to the coupling between (i) a 'connectionless' communication with unknown sender(s), (ii) a unique endpoint at the receiver and (iii) dynamic request routing, decided by network nodes aiming at localizing temporary copies of the content, in a hop-by-hop fashion.

The definition of multipath transport control mechanisms adapted to ICN is at an early stage in the literature [8,23] as well as that of ICN dynamic forwarding mechanisms [13,39]. A comprehensive analysis of the joint problem of optimal congestion control and request forwarding still lacks.

* Corresponding author.
E-mail addresses: gcarofig@cisco.com (G. Carofiglio), massimo.gallo@nokia.com (M. Gallo), lumuscar@cisco.com, luca.muscariello@gmail.com (L. Muscariello).

http://dx.doi.org/10.1016/j.comnet.2016.09.012
1389-1286/

In this paper, we tackle the problem of a joint multipath congestion control and request forwarding for ICN. The key contributions of the paper are the following:

- We formulate a global optimization problem with the twofold objective of maximizing user throughput and minimizing overall network cost. We decompose it into two subproblems for congestion control and request forwarding and solve them separately.
- We derive a family of optimal multipath congestion control strategies to be performed at the receiver and prove the optimality of the proposal.
- We derive a set of optimal dynamic request forwarding strategies and design a distributed algorithm to be implemented by ICN nodes in conjunction with the proposed receiver-driven congestion control.
- We design a mechanism managing fine grained forwarding to high popular content while keeping aggregate based forwarding to less popular content. The mechanisms allows to achieve optimal performance while keeping the system scalable.
- Finally, we implement the proposed joint congestion control and request forwarding in NDN[41] and set up a testbed for large scale experimentation on the Grid'5000 facility ([1]) The performance are assessed through experiments under realistic traffic demand, synthesized by a large set of traffic measurements, and in different network scenarios to appreciate the convergence to the optimal equilibrium, the role of in-path caching and the benefits of ICN multipath. The protocols designed in this paper have been developed and released in open source for the NDN code base and made available at http://systemx.enst.fr/lurch.

In this paper we do not consider several research problems that still requires solutions namely:

- the set of routed faces for a given name prefix are precomputed by a routing algorithm. In this paper routing is part of the assumptions.
- The set of data names that the client issues into the network requires a protocol to manage that. For instance by mean of a content manifest or by using a discovery protocol. This is part of the assumptions made in this paper.

The remainder of the paper is organized as follows. Problem statement, design goals and choices are summarized in Section 2. The formulation of the global optimization problem is presented in Section 3, while in Section 4.1 we solve it by decomposing it into two separate sub-problems related to multipath congestion control and to request forwarding. From the theoretical solution, the optimal rate and congestion controller and the forwarding strategy are derived. Section 4.2 presents the details of the transport protocol and of the design of the distributed forwarding algorithm. The performance assessment by means of experiments is shown in Section 6 and Section 7. Finally, Section 8 surveys related work, while conclusions are drawn in Section 9.

2. Problem statement

ICN transport model fundamentally differs from the current TCP/IP model. To comment on the main differences, let us summarize the basic ICN communication principles.

2.1. System description

Our work is primarily based on CCN/NDN proposal [22,41], though the defined mechanisms have broader applicability in the context of ICN (http://tools.ietf.org/group/irtf/trac/wiki/icnrg) and more generally of content delivery networks employing a similar transport model (e.g. some HTTP-based CDNs). In ICN, *Information objects* are split into Data packets, uniquely identified by a name, and permanently stored in one (or more) repository(ies). Users express packet requests (*Interests*) to trigger Data packets delivery on the reverse path followed by the routed requests.

Interests are *routed by name* towards one or multiple repositories, following one or multiple paths. Rate and congestion control is performed at the end user. Intermediate nodes keep track of outstanding Interests in data structures called PIT (Pending Interest Table), to deliver the requested data back to the receiver on the reverse path. Each PIT entry has an associated timer, so that all requests for the same Data, during such time interval are not forwarded upstream as long as the first query is outstanding. In addition, nodes temporarily store Data packets in a local cache, called *Content Store*.

Upon reception of an Interest packet from an input interface, intermediate nodes perform the following operations: (i) a *Content Store lookup*, to check if the requested Data is locally stored. In case of cache hit, the Data is sent through the interface the Interest is coming from. Otherwise, (ii) a *PIT lookup*, to verify the existence of an entry for the same content name. In this case, the Interest is discarded since a pending request is already outstanding. If not, a new PIT entry is created and (iii) a *FIB lookup* via Longest Prefix Matching returns the interface where to forward the Interest (selected among the possible ones). FIB entries are associated to name prefixes (see [22]). As a consequence, Data may come from the repository, or from any intermediate cache along the path with a temporary copy of the Data packet.

2.2. Potential and challenges of ICN transport model

As a result of the addressing-by-name principle, ICN transport model overcomes the static binding between an object and a location identifier: the receiver issues name-based packet requests over possibly multiple network interfaces with *no a priori knowledge of the content source* (hitting cache or repository). The content-awareness provided by names to network nodes enables a different use of buffers, not only to absorb input/output rate unbalance but for temporary caching of in-transit Data packets. Even without additional storage capabilities in routers, the information access by name of ICN allows two new uses of in-network wire speed storage: (i) *Reuse*: subsequent requests for the same Data can be served locally with no need to fetch data from the original server/repository; (ii) *Repair*: packet losses can be recovered in the network, with no need for the sender to identify and retransmit the lost packet.

Under such assumptions, the notion of connection between two endpoints, as in the TCP/IP model, is no longer required, worse, connections would prevent (i) the early reply of the request at the first unknown hitting cache on the path or (ii) the dynamic path switching due to end-user mobility. If a first step toward a multiparty content-oriented communication has been made by Swift (http://libswift.org/), ICN directly introduces a *connectionless, yet stateful* transport model where just the receiver keeps a flow state associated to an ongoing content retrieval. As a consequence, ICN simplifies mobility/connectivity disruption management, not requiring any connection state migration in case of end-user mobility or path failure.

ICN transport model brings new challenges that TCP, even in its multipath versions, can not address and, instead, motivates the definition of novel transport control and forwarding mechanisms. Indeed, TCP is connection-based and multipath solutions (e.g. [37]) perform load-balancing over static paths, precomputed by a routing protocol. Instead, in ICN, not only the receiver, but also in-path nodes to the repository may perform dynamic packet-by-packet re-

quest scheduling in a purely dynamic fashion by taking on-the-fly decisions about forwarding interfaces (*in-network request scheduling*). The second challenge is the *accrued delay variability* due to in-network caching in ICN: a temporary copy of the content may be retrieved from caches along the path, so impacting the variability of path length and associated delivery time. As a third challenge, in ICN there is a *lack of knowledge of available source(s)*, which prevents from establishing paths at the beginning of the content retrieval, based on routing information like for multipath TCP.

2.3. Design goals and choices

The flexibility brought by the absence of predefined connections, asks for a continuous probing of paths/interfaces at the receiver and at nodes along the paths to drive rate/congestion control at the receiver, as well as forwarding decisions all over the network.

Our main design goal is to effectively address ICN point-to-multipoint communication with no established paths, so to realize (i) efficient content delivery and network resource sharing (bandwidth/storage) via (ii) fully distributed network protocols, and (iii) minimum state to be kept at nodes with no additional signalization w.r.t. what already available in the current legacy NDN data plane. In the following section, we formulate the corresponding optimization problem for joint multipath congestion control and request forwarding problem under the following assumptions: (i) finite link bandwidth: bandwidth sharing modeled by utility-based fairness criteria between users; (ii) in-network caching: a flow can be terminated at one or multiple repositories or at intermediate network caches; (iii) name-based RIBs with name prefix entries already computed per name prefix and stored in the FIB.

3. Problem formalization

1) Network model and notation. The network is modeled as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ with bi-directional arcs: if Interests of flow n traverse link (i, j), corresponding Data are pulled down over link (j, i) according to ICN symmetric routing principle, aka a bread-crumb routing principle which routes Interests only and sent back Data over the reverse path of Interests by keeping a soft state of Pending Interests in a dedicated Table (PIT). For more details, the reader is referred to [22]. Links have finite capacities $C_{ij} > 0$, $\forall (i, j) \in \mathcal{A}$. A content retrieval (also denoted as *flow*) $n \in \mathcal{N}$ is univocally associated to a user node $u^n \in \mathcal{U} \subset \mathcal{V}$ and to one or multiple repositories. With y^n ($\tilde{y}^n$) we denote the Data (Interest) rate associated to flow n at the original user node u^n. Through the network such flow and, hence, its rate, can be split over multiple paths. We will denote with y_i^n ($\tilde{y}_i^n$) the portion of Data (Interest) rate of flow n measured at node i, with $i \neq u^n$. Hence, for a flow n associated to user node $u^n \in \mathcal{U}$, y^n will denote the rate entering u^n, $y_i^n \leq y^n$ will denote the rate of the same flow entering node $i \neq u^n$.

The variable x_{ij}^n ($\tilde{x}_{ji}^n$) denotes the Data (Interest) rate of flow n over link (i, j) (j, i), in packet/s. Given the ICN link flow balance between Interest and Data packets on the reverse link, x_{ij}^n and $\tilde{x}_{ji}^n$ coincide in the fluid representation of flow rates, which can be interpreted as a long term average of the instantaneous rates. Thus, we will omit the indication of Interest rate in the formulation of the optimization problem and only considers for each flow n the Data packet direction.

For each node $i \in \mathcal{V}$ and for a given flow n, we can then define $\Gamma^{+,n}(i)$ and $\Gamma^{-,n}(i)$ respectively the set of egress and ingress nodes for flow n.

Each network node $i \in \mathcal{V}$ has a finite size cache to store in-transit Data. In the current deterministic fluid representation of system dynamics, $h^n(i) \in \{0, 1\}$ defines the binary function equal to 1 when node i may serve Data of flow n and 0 otherwise. As a result, we can denote by $\mathcal{S}(n) = \{i : h^n(i) = 1\}$ the set of available sources for flow n. Notation is summarized in Tab. 1.

Table 1
Notation.

Parameter	Definition
$\mathcal{V}, \mathcal{A}, \mathcal{U}$	Vertices, Arcs, Users sets
$\mathcal{N}$	Flows (content retrievals) set
C_{ij}	Capacity of links $(i, j) \in \mathcal{A}$
$h^n(i) \in \{0, 1\}$	Binary cache function (flow n, node i)
$\mathcal{S}(n) = \{i : h^n(i) = 1\}$,	Set of sources for flow n
x_{ij}^n ($\tilde{x}_{ji}^n$)	Link Data (Interest) rate
$y^n(\tilde{y}^n), y_i^n$	Rate of flow n at receiver or node i
Ψ_{ij}^n	Fraction of flow n at link (i, j)
$\phi(r, t)$	Fraction of flow n on route r at time t
ρ_{ij}	Link load over $(i, j) \in \mathcal{A}$
$\Gamma^{+,n}(i) = \{j \in \mathcal{V} : (i, j) \in \mathcal{A}\}$	Set of egress nodes for $i \in \mathcal{V}, \setminus \in \mathcal{N}$
$\Gamma^{-,n}(i) = \{\ell \in \mathcal{V} : (\ell, i) \in \mathcal{A}\}$	Set of ingress nodes for $i \in \mathcal{V}, \setminus \in \mathcal{N}$

Observation 3.1. Remark that, from a macroscopic viewpoint, the system evolves driven by a stochastic user's demand affecting link bandwidth and node's storage (as analyzed in [6,7]), while here we focus on a microscopic flow-level description of network resource sharing, more suitable to protocol design and optimization. From a macroscopic point of view the $h^n(i)$ become the cache hit probabilities.

2) Optimization problem. The global objective of the optimization problem is a joint user performance maximization and network cost minimization. We consider a convex user's utility function $\boldsymbol{U}^n$ and a concave link cost function $\boldsymbol{C}_{ij}$, and we compute the global objective as the difference between the total user's utility and total network cost (1). The problem is formulated as a multi-commodity flow problem with linear constraints and convex objective:

$$\begin{cases} \max_{\mathbf{y}} \sum_n \boldsymbol{U}^n(y^n) - \sum_{i,j\in\mathcal{A}} \boldsymbol{C}_{ij}(\rho_{ij}) & (1) \\ \sum_{n\in\mathcal{N}} x_{ij}^n = \rho_{ij} \qquad \forall (i, j) \in \mathcal{A} & (2) \\ \sum_{\ell\in\Gamma^{-,n}(i)} x_{\ell i}^n = y_i^n, \quad \forall i, n & (3) \\ \sum_{j\in\Gamma^{+,n}(i)} x_{ij}^n (1 - h^n(i)) = y_i^n \qquad \forall i \neq u^n, n & (4) \\ \rho_{ij} \leq C_{ij} \qquad \forall (i, j) \in \mathcal{A} & (5) \\ x_{ij}^n \geq 0 \qquad \forall i, j, n & (6) \end{cases}$$

With ρ_{ij} we denote the link load as the total flow rate flowing through link (i, j) (2), while y_i^n denoting flow n's rate arriving at node i is defined as the sum of the ingress rates $x_{\ell i}^n$, $\ell \in \Gamma^{-,n}(i)$ (3). y_i^n is also equal to the sum of egress rates x_{ij}^n, $j \in \Gamma^{+,n}(i)$, unless $h^n(i) = 1$ (hence, Interest are locally satisfied and not forwarded upstream). This holds for all nodes i except the user node for flow n, u^n, for whom $x_{ij}^n = 0$, $\forall j$. Data rates are subject to link capacities constraints (5) and must be non negative (6).

3) Problem decomposition. The problem can be decomposed according to the two objectives: utility maximization of end-users throughput (at the receiver) and network cost minimization (at in-path nodes). Let us detail the decomposition in such two problems. A primal decomposition of the global optimization problem is possible when fixing the fraction Ψ_{ij}^n of the total Data (Interest) rate of each flow on a given link, defined by $x_{ij}^n = \Psi_{ij}^n y^n$, while keeping the total rate at the receiver, y^n, as a variable. The sub-problem assuming fixed Ψ_{ij}^n corresponds to the *utility maximization of end-users throughput*:

$$\begin{cases} \max_{\mathbf{y}} \sum_n \boldsymbol{U}^n(y^n) & (7) \\ \sum_{n\in\mathcal{N}} \Psi_{ij}^n y^n \leq C_{ij} \qquad \forall (i, j) \in \mathcal{A} & (8) \\ y^n \geq 0 \qquad \forall n \in \mathcal{N} & (9) \end{cases}$$

The utility maximization in (7) is Kelly's formulation of distributed congestion control [25,33]. We will show in Section 4.1, how to derive an optimal congestion controller for ICN, based on Interests transmission control at the receiver.

The master problem is the overall *network cost minimization* where, instead, the rate at the receiver, y^n, is given and the variables are the split ratios Psi_{ij}^n or equivalently the rates x_{ij}^n determining the forwarding strategy.

$$\begin{cases} \min_{\boldsymbol{x}} \sum_{i,j\in\mathcal{A}} \boldsymbol{C}_{ij}\left(\sum_{n\in\mathcal{N}} x_{ij}^n\right) & (10)\\ \sum_{\ell\in\Gamma^{-,n}(i)} x_{\ell i}^n = y_i^n, \quad \forall i,n & (11)\\ \sum_{j\in\Gamma^{+,n}(i)} x_{ij}^n(1-h^n(i)) = y_i^n \qquad \forall i\notin\mathcal{U},n & (12)\\ x_{ij}^n \geq 0 \qquad \forall (i,j)\in\mathcal{A} \qquad \forall n\in\mathcal{N} & (13)\end{cases}$$

Network cost minimization is a multi-commodity flow problem, whose flows may have multiple sources (caches or Data repositories) and one unique receiver $u^n \in \mathcal{U}$. In the following we will consider the binary variables h as fixed. The coupling with time variant caching dynamics is left for future work.

4) Solution. To solve both problems via distributed algorithms: (i) we compute the respective Lagrangians $L_{\boldsymbol{U}}$ and $L_{\boldsymbol{C}}$; (ii) we decompose $L_{\boldsymbol{U}}$ and $L_{\boldsymbol{C}}$ with respect to user's and in-path nodes respectively; (iii) we identify local Lagrangians at users and in-path nodes to be respectively maximized/minimized. Let us consider $L_{\boldsymbol{U}}$ and $L_{\boldsymbol{C}}$ separately.

$$\begin{aligned} L_{\boldsymbol{U}}(\boldsymbol{y},\boldsymbol{\lambda}) &= \sum_n \boldsymbol{U}^n(y^n) - \sum_{ij}\lambda_{ij}\left(\sum_{n\in\mathcal{N}}\Psi_{ij}^n y^n - C_{ij}\right)\\ &= \sum_n \boldsymbol{U}^n(y^n) - \sum_n\sum_{(i,j)}\lambda_{ij}\Psi_{ij}^n y^n + \sum_{ij}\lambda_{ij}C_{ij}\\ &= \sum_n (\boldsymbol{U}^n(y^n) - \lambda^n y^n) + \sum_{(i,j)}\lambda_{ij}C_{ij}\end{aligned}$$

where $\lambda^n \equiv \sum_{(i,j)}\Psi_{ij}^n\lambda_{ij}$. Hence, every user maximizes the local Lagrangian $L_{\boldsymbol{U}}^n$:

$$L_{\boldsymbol{U}}^n = \boldsymbol{U}^n(y^n) - \lambda^n y^n \quad (14)$$

which requires to compute the Lagrange multipliers λ_{ij} associated to the capacity constraints, whereas multipliers associated to constraints (9) are always null as we assume fixed positive $y^n > 0\ \forall n$. The utility maximization is responsible for adjusting the rates y^n, which are, instead, assumed to be constant in the network cost problem minimization. Let us then compute $L_{\boldsymbol{C}}$ and decompose it as a function of in-path nodes which are coupled by the flow balance constraint (4). Hence, $L_{\boldsymbol{C}}(\boldsymbol{x},\boldsymbol{\mu}) =$

$$\begin{aligned} &\sum_{i,j}\boldsymbol{C}_{ij}\left(\sum_{n\in\mathcal{N}}x_{ij}^n\right) - \sum_n\sum_i\mu_i^n\left(\sum_{l\in\Gamma^{-,n}(i)}x_{li}^n - \sum_{j\in\Gamma^{+,n}(i)}x_{ij}^n\right)\\ &= \sum_{i,j}\boldsymbol{C}_{ij}(\rho_{ij}) - \sum_n\sum_{i,l}\mu_i^n x_{li}^n + \sum_n\sum_{i,j}\mu_i^n x_{ij}^n\\ &= \sum_i\sum_{l\in\Gamma^{-,n}(i)}[\boldsymbol{C}_{li}(\rho_{li}) - \sum_n(\mu_i^n-\mu_l^n)x_{li}^n]\end{aligned}$$

Every node i minimizes network cost by reducing a local Lagrangian $L_{\boldsymbol{C}i}$ given by

$$L_{\boldsymbol{C}i} = \sum_{l\in\Gamma^{-,n}(i)}[\boldsymbol{C}_{li}(\rho_{li}) - \sum_n(\mu_i^n-\mu_l^n)x_{li}^n] \quad (15)$$

which requires the computation of multipliers μ_i^n. Multipliers associated to constraints (13) are always null as we assume $x_{li}^n > 0$. This is due to the assumption that some probing traffic must be present even along non optimal paths to adapt to network congestion variations.

In the following sections, we derive (i) an optimal Interest transmission control protocol to be performed at the receiver from (14) (Section 4.1) and (ii) an Interest interface selection protocol to be performed at network nodes from (15) (Section 4.2).

4. Optimal solution

4.1. Receiver-driven congestion control

In ICN, Data packets are pulled down by Interest packets along Interests reverse path. Hence, one can regulate Interest transmission rate at the receiver, $\tilde{y}_t^n$, to realize optimal Data delivery rate y_t^n. To derive a family of optimal controllers, let us apply a gradient algorithm to solve the utility maximization sub-problem in (14) (See [4,31]). It results that:

$$\dot{\lambda}_{ij}(t) = \kappa_{ij}(t)\left(\sum_{n\in\mathcal{N}}\Psi_{ij}^n y^n(t) - C_{ij}\right) \quad (16)$$

$$\dot{y}^n(t) = \gamma^n(t)\left(\boldsymbol{U}^{n'}(y^n(t)) - \lambda^n(t)\right) \quad (17)$$

where $\kappa_{ij}(t)$ is a function of link (i,j). Notice that for compact notation, we represent the first derivative over time of a function $f(t)$ as $\dot{f}(t) = \frac{d}{dt}f(t)$. Let us take for instance $\kappa_{ij}(t) = \frac{1}{C_{ij}}$, then $\lambda_{ij}(t)$ can be interpreted as (i,j)'s link delay, as its evolution (16) is determined by a fluid queue evolution equation with input rate $\sum_{n\in\mathcal{N}}\Psi_{ij}^n y^n(t)$ and service rate C_{ij}. Thus, $\lambda^n(t)$ accounts by for the total network delay experienced by flow n.

Observation 4.1. *Link, flow and route delays.* The request/reply nature of ICN communication model suggests a natural way to measure $\lambda^n(t)$ at the receiver via the Interest/Data response time, while $\lambda_{ij}(t)$ are not known and hard to measure at the receiver.

However, as mentioned in Section 2.2, the accrued delay variability of ICN multipath communication due to the lack of a priori knowledge of the sources and to the coupling with in-network caching makes it inefficient to control Interest rate by simply monitoring the total flow delay $\lambda^n(t)$ (also noticed in [8]).

Therefore, we decompose $\lambda^n(t)$ into the sum of route delays, $\lambda^n(r,t)$, where a route $r \in \mathcal{R}^n$ identifies a unique sequence of nodes from any source $s \in \mathcal{S}(n)$ to the receiver node, for flow n, $\lambda^n(t) = \sum_{r\in\mathcal{R}^n}\lambda^n(r,t)\phi(r,t)$, where $\phi(r,t)$ is the fraction of flow routed over route r at time t (as decided in a distributed fashion by the request forwarding algorithm). Also,

$$\lambda^n(r,t) = \sum_{(i,j)\in r: r\in\mathcal{R}^n}\lambda_{ij}(t)$$

The receiver has clearly no knowledge over time of the cache serving a given Data packet. Still, it is desirable to determine available fast and slow routes, because such information can locally be used to better adapt the Interest transmission strategies. Different routes can be distinguished by explicit labeling, as proposed in [8]. Eq. 17 becomes:

$$\dot{y}^n(t) = \gamma^n(t)\left(\boldsymbol{U}^{n'}(\tilde{y}^n(t)) - \sum_{r\in\mathcal{R}^n(s)}\lambda^n(r,t)\phi(r,t)\right) \quad (18)$$

Note that, if the rate decrease is proportional to a linear combination of route delays, the rate increase is unique per flow as it depends on the overall flow rate maximization. The choice of the utility function $\boldsymbol{U}(y)$ specifies the form of the controller. In light of TCP/IP literature (cfr. [33]), we can for instance choose an AIMD (Additive Increase Multiplicative Decrease) controller, by selecting $\gamma^n(t) = K_1(t)(y^n(t))^2$ and $\boldsymbol{U}(y) = K_2(t)/(\lambda^n(t)y^n(t))$. Hence, for all flows, we derive the following Interest transmission rate control:

$$\dot{y}^n(t) = \frac{K_1K_2}{\lambda^n(t)^2} - K_1y^n(t)\sum_{r\in\mathcal{R}}\lambda^n(r,t)\phi(r,t)y^n(t) \quad (19)$$

The transport protocol derived from (19) is described in Section 5. Let us focus here on the second sub-problem related to optimal request forwarding.

4.2. Dynamic request forwarding

As observed in Section 2.2, in ICN, not only the receiver, but also in-network nodes may perform dynamic packet-by-packet request scheduling by taking on-the-fly decisions on the selection of forwarding interfaces (*in-network request scheduling*). Under our problem formulation in Section 3, the distributed nature of request forwarding is enabled by the decomposition of $L_{\mathcal{C}}$ w.r.t. in-path nodes and at each node the local objective to minimize is given by (15). To derive a family of optimal distributed forwarding algorithms let us first prove that

Proposition 4.2. *The local minimization of (15) at every network node $i \in \mathcal{V}$ is equivalent to the minimization of μ_i^n, for all $n \in \mathcal{N}$.*

Proof. Let us start by imposing the Karush-Kuhn-Tucker conditions to Eq. (15), that is:

$$\frac{\partial \boldsymbol{C}_{ji}}{\partial x_{ji}^n} = \frac{\partial \boldsymbol{C}_{ji}}{\partial \rho_{ji}} \frac{\partial \rho_{ji}}{\partial x_{ji}^n} = \frac{\partial \boldsymbol{C}_{ji}}{\partial \rho_{ji}} = \mu_i^n - \mu_j^n \tag{20}$$

for all ingress interfaces $j \in \Gamma^{-,n}(i)$ such that $x_{ji}^n > 0$ and

$$\frac{\partial \boldsymbol{C}_{ji}}{\partial x_{ji}^n} \leq \frac{\partial \boldsymbol{C}_{li}}{\partial x_{li}^n}, \qquad \forall l \in \Gamma^{-,n}(i) \tag{21}$$

Eq. (21) states that the interfaces selected for Interest forwarding at node i are those minimizing the cost derivatives. This implies that there can be one or multiple interfaces with equal and minimum cost derivative. For every of such interfaces, let us iterate the reasoning and apply (20) at all subsequent hops $j, j+1, j+2, \ldots, r$ up to one of the sources $s \in \mathcal{S}(n)$. At the last hop minimization of the cost derivative corresponds directly to the minimization of μ_r^n, while at previous hops the variable to be minimized is the difference $\mu_k^n - \mu_{k+1}^n$ for $k = j+1, \ldots, r$ and $\mu_i^n - \mu_j^n$ at the first hop. Clearly, starting from the source, once minimized μ_r^n and denoted by $\mu_r^{n,*}$ its value, the minimization of $(\mu_{r-1}^n - \mu_r^{n,*})$ is equivalent to the minimization of μ_{r-1}^n and so on until μ_i^n.

Thus, we can conclude that a family of optimal distributed algorithms for request forwarding can be derived by simply minimizing μ_i^n at each node i and $\forall n \in \mathcal{N}$. □

To compute μ_i^n we apply a gradient algorithm on the Lagrangian $L_{\boldsymbol{C}}(\boldsymbol{x}, \boldsymbol{\mu})$ and obtain (See [4]):

$$\dot{\mu_i^n} = \eta_i^n \left(\sum_{j \in \Gamma^{+,n}(i)} x_{ij}^n - \sum_{l \in \Gamma^{-,n}(i)} x_{li}^n \right) \tag{22}$$

Observe that, by invoking the flow balance between Interest and Data over a given interface, $x_{ij}^n = \tilde{x}_{ji}^n$. Hence, $\mu_i^n(t)$ turns out to be a measure of the total flow rate unbalance at node i, which is available at an ICN node and equal to the number of pending Interests in the PIT. Observe that the PIT increase rate in (22) corresponds, in practice, to the Interest rates $\tilde{x}_{ji}^n$, while the decrease term to Data rates (consuming PIT entries). This implies that the optimal strategy consists in *minimizing the total number of pending Interests* at node i. The intuition behind is that the number of pending Interests reflects (i) content proximity: path length and response time associated to a given content; (ii) congestion status, i.e. the quality of residual path towards repository/closest copy.

To derive an optimal interface selection algorithm, The PIT size evolution is driven by two components: the rate of interests sent towards the upstream network, causing a size increase, and the rate of the data forwarded downstream causing a size decrease. See [10] for a recent work that evaluate the PIT size distribution under various scenarios.

Let us now distinguish the Interests per output interface and apply an equivalent of Little's law to couple number of Interests and associated response time. Let us omit the indices n and i, as we focus on flow n and node i. $\mu = \sum_{j \in \Gamma^{-,n}(i)} \mu_j = \sum_{j \in \Gamma^{-,n}(i)} x_j \mathrm{VRTT}_j(x_j)$, where $\mathrm{VRTT}_j(x_j)$ is the average response time at output interface j. Hence, at each output interface, this requires to solve the following problem:

$$\begin{cases} \min \sum_{j \in \Gamma^{-,n}(i)} x_j \mathrm{VRTT}_j(x_j) & (23) \\ \sum_{j \in \Gamma^{-,n}(i)} x_j = y & (24) \\ x_j \geq 0 & (25) \end{cases}$$

Assuming that: 1) $\mathrm{VRTT}_j(x_j)$ are monotonically increasing with x_j, 2) $\mathrm{VRTT}_j(0) = \mathrm{VRTT}_{\min}$, and 3) $\mathrm{VRTT}_j(x_j)$ is differentiable, the optimal allocation can be easily proved by imposing KKT conditions, which give: $x_j = \mu_j'^{-1}(\theta)$, with θ unique solution of the equation $\sum_j \mu_j'^{-1}(\theta) = y$.

Observation 4.3. Observe that this algorithm has in practice two major drawbacks: (i) it requires explicit knowledge of VRTT, (ii) it just exploits a subset of paths depending on the value of y. This latter results from the fact that $\mu_j'(x_j) = \mathrm{VRTT}_{\min} + x_j \mathrm{VRTT}'(x_j)$. As $\mathrm{VRTT}_j(0) = \mathrm{VRTT}_{\min}$. Thus $\exists \theta^* > 0 : \mu_j'^{-1}(\theta) = 0, \forall \theta \leq \theta^*$. This means that slower interfaces could never be probed for some y.

As we observed in Section 2.2, two important challenges ICN needs to cope with are the unknown sources and the accrued delay variability due to in-network caching.

To guarantee optimal interface selection over time through continuous monitoring of the interfaces, one needs a slightly different objective at each output interface of node i, which minimizes the number of pending Interests associated to the most loaded output interface, i.e.

$$\begin{cases} \min \max_j \mu_j(x_j) & (26) \\ \sum_j x_j = y & (27) \\ x_j \geq 0 & (28) \end{cases}$$

The objective can be attained very easily, by observing that the optimal allocations are such that $\mu_j = \mu$ for all interfaces $j \in \Gamma^{+,n}(i)$. Consider now an interval of time T, such that every interface j has been probed and selected, i.e. $x_j(t) = y(t)$, for a fraction of time ΔT_j over $T = \sum_j \Delta T_j$ and zero elsewhere. Thus,

$$\begin{aligned} \mu &= \lim_{T \to \infty} \frac{1}{T} \int_0^T \mu_j(t) dt = \lim_{T \to \infty} \frac{\int_0^T x_j(t) \mathrm{VRTT}_j(x_j(t)) dt}{\sum_l \Delta T_l} \\ &= \frac{\Delta \bar{T}_j}{\sum_l \Delta \bar{T}_l} y \mathrm{VRTT}_j(y) = \phi_j \bar{\mu}_j \end{aligned} \tag{29}$$

where $\bar{\mu}_j$ is the average measured number of pending Interests at interface j, $\phi_j = \mu / \bar{\mu}_j$ represents the probability to select interface j, and $1/\mu = \sum_j \mu_j$ is a normalization constant. This algorithm does not require the explicit knowledge of the VRTT_j and probes slow interfaces as well, with rate $x_j = \mu_j / \mathrm{VRTT}_j$. The algorithm only requires to locally measure the average number of interests per used interface, which is an available information in NDN by design.

5. Design and analysis of optimal congestion control and forwarding strategies

5.1. Congestion control protocol description

Let us define more in detail a receiver-driven congestion control mechanism derived from (19). Data are requested via Interests (one Interest per Data packet) in the order decided by the application, according to a window-based *AIMD* mechanism. The Interest rate $\tilde{y}^n(t)$ of flow $n \in \mathcal{N}$ is controlled via a congestion window of Interests, $W^n(t)$, kept at the receiver which defines the maximum number of outstanding Interests the receiver is allowed to send.

Congestion window dynamics can be obtained from Eq. (19) by assuming the relation $W^n(t) = \tilde{y}^n(t) \cdot R^n(t) = y^n(t) \cdot R^n(t)$, with $R^n(t) \equiv \lambda^n(t)$ denoting the overall round trip delay experienced by flow n as averaged over all its routes $r \in \mathcal{R}^n$. This holds in virtue of Little's law and it is commonly found in TCP literature (e.g. [3]).

Window Increase. $W^n(t)$ is increased by $\eta/W^n(t)$ upon each Data packet reception. This corresponds to an increment of η (= 1 by default) each time a complete window of Interests is acknowledged by Data reception.

Window Decrease. According to (19), Interest rate/window decrease is proportional to route delays $\lambda^n(r, t)$. The rationale behind is that by triggering the variations of such delay, mainly due to bottleneck(s) queuing delay, the receiver can effectively adapt Interest window/rate to anticipate network congestion status. In this way, the receiver realizes a *Remote Active Queue Management*, similar to [2], based on the per-route round trip delays estimate. The bottleneck(s) queuing delay of route r is inferred by the measured round trip delay over time $R_r(t)$, and determines a per-packet decrease probability function, $p_r(t)$. In case of window decrease, $W^n(t)$ is multiplied by a decrease factor $\beta < 1$.

Route Delay Monitoring. When an Interest is sent out, the receiver measures the instantaneous round trip delay as the time elapsed between the send of the request and the reception of the corresponding Data packet. To ease the notation let us omit in the following the index n. Estimates of the minimum and maximum round trip delay associated to a given route r ($R_{\min,r}$ and $R_{\max,r}$) are maintained via an history of samples (a sliding measurement window of 30 samples is kept by default in our implementation), excluding retransmitted packets. Note that the samples can be distinguished per route via the route label introduced in ICN Data packets as proposed in [8,9]. Also, consider that the number of monitored routes per flow results to be limited to a few by a minimum threshold of utilization set to 20 packets (cfr.[8]). This prevents a large flow state to be kept at the receiver. A complete window of samples is required before triggering a Interest window decrease.

The measured instantaneous round trip delay $R_r(t)$ and the estimates of $R_{\min,r}$ and $R_{\max,r}$ concur to determine over time the probability $p_r(t)$ of a window decrease. $p_r(t)$ is assumed to be a monotonically increasing function of $R_r(t)$ (e.g. linear as in [8]), which we consider in our experiments: $p_r(t)$ goes from a minimum value $p_{\min}$ (by default set to 10^{-5}) up to a maximum value $p_{\max} \leq 1$ when above $R_{\max,r}(t)$,

$$p_r(t) = p_{\min} + \Delta p_{\max} \frac{R_r(t) - R_{\min,r}(t)}{\Delta R_{\max}} \tag{30}$$

with $\Delta p_{\max} = p_{\max} - p_{\min}$ and $\Delta R_{\max} = R_{\max,r}(t) - R_{\min,r}(t)$. $R_{\min,r}(t)$, $R_{\max,r}(t)$ indicate estimates at time t. Whenever $R_{\min,r}(t) = R_{\max,r}(t)$ in a window of samples, we take a decrease probability equal to $p_{\min}$. The resulting evolution of $W(t)$ is:

$$\dot{W}(t) = \frac{\eta}{R(t)} - \beta W(t) \sum_{r \in \mathcal{R}} p_r(t) \phi(r,t) \frac{W(t)}{R_r(t)}, \tag{31}$$

By taking $y(t) = W(t)/R(t)$, $K_1 = \beta$, $K_2 = \eta/\beta$ one finds (19).

Interest scheduling. The Interest controller at the receiver may perform request scheduling to decide what to request and when, according to application constraints and user preferences. Like in BitTorrent or Swift, but directly within transport layer, the client exploits local information to prioritize Data requests in the communication swarm. Smart Interest scheduling in the transmission window is a powerful differentiator w.r.t. TCP-like content-agnostic transmission. All its possibilities are beyond the scope of this paper. However, in our work we exploit it to perform Interest retransmission for applications with loss recovery time constraints. This is made by introducing as a rule that Interests to be retransmitted are always put in front of the transmission window after PIT timer expiration.

5.2. Single path stability analysis

In this section we neglect intermediate caches in the network and focus on the evolution of $N = |\mathcal{N}|$ flows (content retrievals) over a single path in presence of delays. The analysis allows to determine the stability region of the flow controller in a simpler setting as a function of the different parameters, which might be optimized in the different network settings. Each flow is associated to a congestion window $W^n(t)$, $n \in \mathcal{N}$ and a corresponding rate $y^n(t)$. Let us define the queuing delay on the bottlenecked link and the rate evolution for a single path flow as:

$$\dot{Q}(t) = \sum_{n=1}^{N} y^n(t) - C, \quad R(t) = R_{\min} + Q(t)/C, \tag{32}$$

$$\dot{y}^n(t) = \frac{\eta}{R(t)^2} - \beta y^n(t) y^n(t - R(t)) p(t - R(t)), \tag{33}$$

Notice that in Eq. (33) we introduce delayed variable which are not considered in previous Section 3. In particular, the multiplicative decrease $\beta y^n(t)$ occurs with rate $y^n(t - R(t))p(t - R(t))$. This because a Data packet sent in the previous round trip time triggers a drop with probability $p(t - R(t))$, computed based on the round trip delay estimates of the previous round trip time. Similar fluid representation of window dynamics under RED-like AQM can be found in [21].

Following [21] we approximate $R(t)$ with a constant value, $\bar{R}$, equal to the average value. We denote by $\bar{g} = \lim_{t\to\infty} 1/t \int_0^t g(u)du$. The resulting rate evolution law is,

$$\dot{y}^n(t) = \frac{\eta}{\bar{R}^2} - \beta y^n(t) y^n(t - \bar{R}) p(t - \bar{R}) \tag{34}$$

The equilibrium point of Eq. (30–34) is

$$\bar{y} = C/N, \quad \bar{p} = \frac{\eta N^2}{\beta \bar{R}^2 C^2}, \quad \frac{\bar{Q}}{C} = \frac{(\bar{p} - p_{\min})\Delta R_{\max}}{\Delta p_{\max}} \tag{35}$$

The stability analysis of AIMD congestion control with AQM has been studied in the literature [21]. Similar arguments allow to prove the stability of Eq. (30–34), in absence and presence of feedback delays. The main difference is given by the form of the congestion notification and the computation of the stability region in presence of delay.

Proposition 5.1. *Given the system of ODEs in Eq. (30–34) in absence of delays, i.e. where the rate evolution is given by*

$$\dot{y}^n(t) = \frac{\eta}{\bar{R}^2} - \beta y^n(t)^2 p(t) \quad \forall n \in \mathcal{N}, \tag{36}$$

Eq. (35) is a global asymptotically stable equilibrium $\forall \beta > 0$, $\Delta p_{\max} > 0$. *In presence of delays, where the rate evolution is described by Eqs. (34),(35) is a locally asymptotically stable equilibrium. A region of attraction for the equilibrium is* $\left\{(y^n(t), \ldots, y^n(t), p(t)) : y^n < \bar{y}^n\left(1 + \frac{\kappa\bar{R}\sqrt{\zeta}}{p_{min}}\right), \forall n\right\}$

The proof is reported in Appendix A.

5.3. Request forwarding algorithm

Request forwarding decisions are based on the selection of longest prefix matching interfaces in the FIB. Indeed, FIB entries specify name prefixes rather than full object names and clearly appears to be unfeasible to maintain per-object name information. We apply optimal interface selection algorithm per output interface and per prefix rather than per objects. Notice that applying

the same transmission strategy to different objects with the same name-prefix preserves reasonably limited FIB size at the cost of introducing a potential loss of performance w.r.t. the optimal strategy. This may occur when flows sharing the same name-prefix might have a significantly different set of sources (See Section 7 for more details).

Let us now describe the steps of the algorithm summarized in Algorithm 1. At each Data/Interest packet reception, the number of Pending Interests (PI) associated to a given FIB entry (thus to a name prefix) and to a particular output interface is updated (FIB_PI_Incr/FIB_PI_Decr). Based on the instantaneous value of PI, a moving average (with parameter 0.9) is regularly updated and used to recompute interfaces' weights (FIB_Weight_Update). A random weighted algorithm is used to select the output interface for each incoming Interest (RND_Weight). For a given prefix/output interface, weights are normalized w.r.t. the sum of all weights associated to available interfaces for that particular prefix. At the beginning, each interface has the same weight equal to one and the randomized forwarding process is uniform over available output interfaces.

Algorithm 1 Request forwarding algorithm.

```
Preconditions: A router receives a Data or Interest packet from an input face.
At Data_Packet_Reception (name, face_in)
if (!PIT_miss) then
    Forward_Data(face_out);
    FIB_PI_Decr(face_in, prefix);
    FIB_Weight_Update(face_in, prefix);
else Drop_Data;
end if
At Interest_Packet_Reception (name, face_in)
if (CACHE_miss && PIT_miss) then
    (prefix,weight_list)=FIB_Lookup(name);
    f*=RND_Weight(weight_list);
    FWD_I(f*);
    PIT_I_Update(name,f*);
    FIB_PI_Incr(f*, prefix);
    FIB_Weight_Update(f*, prefix);
else NDN_standard_processing;
end if
At PIT_Timeout(name, prefix, face)
   FIB_PI_Decr(face, prefix);
   FIB_Weight_Update(face, prefix);
procedure FIB_Weight_Update((face, prefix))
avg_PI(face, prefix) ← α · avg_PI + (1 − α)I(face, prefix);
w(face, prefix) ← 1/avg_PI(face, prefix);
end procedure
```

5.4. Hybrid object-based and prefix-based forwarding (OBF/PBF)

As already observed, the request forwarding solution based on per-object state shows scalability issues in presence of very large catalogs. In this section, we design a hybrid Object-Based Forwarding (OBF) and Prefix-Based Forwarding (PBF) solution, providing a good trade-off respectively between optimality and forwarding scalability. The compromise we propose is based on the differentiation of the most popular objects whose forwarding state is kept on a per-object basis, from the rest of the catalog items grouped according to their name prefixes in order to reduce the forwarding state. As a result of the different granularity used for the more popular versus the less popular items, the forwarding metric may be significantly different from objects under the same name prefix when there is an important diversity in terms of popularity.

The solution we propose, consists in the following four steps:

- (i) Regular monitoring of per content object name popularity via a fixed size vector that probabilistically collect data names, reflecting the data popularity;
- (ii) Periodical insertion/deletion of a heavy hitter entry in the FIB;
- (ii) Forwarding based on name/prefix entry according to longest prefix matching algorithms;
- (iv) Regular update of the metrics.

While steps (iii) and (iv) remain similar to Algorithm 1, steps (i) (traffic monitoring) and (ii) (insertion/deletion of popular content names in the FIB) require some modifications to the basic per prefix forwarding strategy proposed in Section 5.3.

The pseudo-code of the proposed hybrid approach is represented in Algorithm 2. It basically consists in probabilistically inserting the name of the received DATA in a fixed size FIFO (or LRU) vector of *k elements*. In this way, the above mentioned vector will contain the k most popular content object names that are periodically (i.e. every T seconds) inserted (and removed) in the FIB. Notice that per content object name FIB entries will start with forwarding metrics set to the default value (i.e. average number of pending interest equal to 0).

Algorithm 2 Hybrid per prefix and per popular content object's name request forwarding.

```
At Data_Packet_Reception (name, face_in)
if (!PIT_miss) then
    With probability p;
        Insert_popularity_vector(name);
    Forward_Data(face_out);
    FIB_PI_Decr(face_in, prefix);
    FIB_Weight_Update(face_in, prefix);
else Drop_Data;
end if
Every T seconds
    FIB_Remove_popular_name(old_popularity_vector);
    FIB_Insert_popular_name(popularity_vector);
    old_popularity_vector = popularity_vector;
```

5.5. Complexity analysis

The solution presented in Section 3 includes a receiver driven congestion control protocol and a load balancer deployed on end-hosts (content receivers) and network nodes respectively. Both algorithms are completely distributed meaning that congestion control and forwarding decisions are taken independently at each node (end-host or network node). The congestion controller at the receiver is AIMD (additive increase multiplicative decrease). The proposed protocol has the same complexity of TCP in the Internet. The advantage of having congestion control at the receiver is significant on the server side where no socket state is kept, see also [28] for some analysis of receiver driven congestion control in the Internet. Regarding the load balancer, both Algorithm 1 and 2 are new and their complexity need to be analyzed. Algorithm 1 and 2 are distributed and applied on a per-Interest/Data basis using local statistics. Hence, the computational complexity of Algorithm 1 and 2 is given by the complexity needed to update the local statistics which is O(1). Notice that the additional operations needed to maintain local forwarding statistics for both algorithms, do not require any additional (and expensive) lookup operations on the FIB table where such informations are stored. These operations can hence be made at wire speed.

6. Experimental evaluation

6.1. Implementation in the NDN prototype

We implemented the above-described protocols in the NDN prototype [41] by developing additional modules that can run up to NDN 0.3 (the library ndn-cxx-0.2 and the forwarder NFD 0.2.0 released in November 2014). The additional modules include a standalone application, `ndn-icp-download` running at the user with congestion control functionality; a path labeling mechanism in the NFD daemon; the implementation of the request forwarding strategy in the NFD daemon and a novel cache manager, in NFD for the content store. The implementations are briefly summarized below, while a detailed description is out of the scope of this paper and can be found at http://systemx.enst.fr/lurch.

(i) *Congestion control*: the protocol presented in Section 5.1, is implemented at the user, together with an Interest scheduler, which decides the Interests to insert in the transmission window. Path labeling has been implemented via a sequence of unique node identifiers collected by the data packet in the downstream and hashed in an additional packet TLV field `path_label`.

(ii) *Request forwarding*: The NFD implementation requires a new data structure, to track the evolution of the number of pending Interests on a given face, for a name-prefix. (iii) *Caching*: The NFD daemon implements FIFO caching with replacement based on a periodic clean up of the content store. We have implemented a framework to manage the content store including LRU, used in the experiments presented below.

(iv) *Virtual content repository*: We have developed the `ndn-virtual-repo` application, based on the older CCNx implementation developed at Washington University [40], which significantly simplifies tests using very large repositories.

6.2. Experimental setting

The experimental environment used to perform our evaluation is a large scale grid of general purpose servers, Grid'5000 (see [1] for more details on the available network equipment, server's hardware and software). The infrastructure allows to boot custom Linux kernel images (Debian wheezy 3.2.35-2 x86_64 GNULinux in our tests), including the NDN software described in Section 6.1. Servers' reservation and remote kernel boot services allow to manage large scale experiments (few hundreds in this paper). Our tests rely on five blocks:

(i) *Network topology*: We use virtual interfaces based on tap IP tunnels as a link abstraction layer, and kernel token bucket shapers to create our substrate overlay network topology with customized link rates, in the grid.

(ii) *NDN network*: An NDN overlay network is built on top of the IP overlay topology and name based routing is configured according to pre-computed FIBs to make content reachable to clients.

(iii) *Catalogs and users*: objects are made available to users through the virtual repositories described in 6.1. The application `ndn-icp-download` (http://systemx.enst.fr/lurch) retrieves a fixed amount of named content from the network before closing.

(iv) *Experiment launching*: The experiment is configured and monitored using a centralized application that orchestrates the test issuing ssh exec passwordless commands on remote servers. Tests are run by launching parallel clients' workloads.

(v) The *dynamic workload* used in the experimentation is derived from data traffic traces collected at the radio mobile gateway of the Orange operational mobile network in France (2G/3G/4G radio access). From such measurement point we extract a peak hour of HTTP traffic for an area serving about 50 000 different observed users, 200 active at the same time on average. From this sample

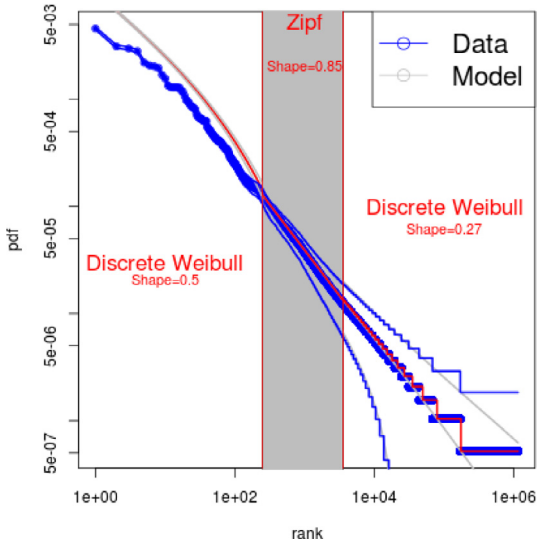

Fig. 1. Empirical web content popularity and the model fitting the sample with corresponding confidence bands.

we obtain the amount of transferred content, its popularity and the amount of shareable data (percentage of bytes requested more than once). In one hour we measure 60GB of data, 35% shareable. From the analysis carried out over a week, we extract the results for a given day, leaving a mored detailed analysis to a future dedicated study. We test empirical popularity against various models and found the discrete Weibull to be the best fit with shape around 0.27 (long tailed). In Fig. 1, we report the empirical popularity distribution for web traffic (HTTP only), with corresponding 95% confidence bands in blue (see [17] for similar analysis). We also report in red the model fit to the available sample and 95% confidence bands. While for the tail a simple discrete Weibull distribution passes a χ^2 goodness of fit test [14], with p-values exceeding 5% significance level, the good model for the entire distribution turns out to be trimodal with three components: a discrete Weibull for the head of the distribution, a Zipf for the waist and a discrete Weibull for the tail, i.e.

$$f(k) = \begin{cases} \phi_1 d \frac{\beta_1}{\lambda_1} \left(\frac{k}{\lambda_1}\right)^{\beta_1 - 1} e^{-(k/\lambda_1)^{\beta_1}} & k < k_1 \\ \frac{\phi_2}{k^{\alpha_2}} & k \in [k_1, k_2] \\ \phi_3 \frac{\beta_3}{\lambda_3} \left(\frac{k}{\lambda_3}\right)^{\beta_3 - 1} e^{-(k/\lambda_3)^{\beta_3}} & k > k_2 \end{cases}$$

with parameters $\lambda_1, \beta_1, \alpha_2, \lambda_3, \beta_3, \phi_1, \phi_2, \phi_3 \in \mathbb{R}^+$; $k_1, k_2 \in \mathbb{N}$ that can be estimated using standard techniques. Linear regression on a log log plane must be avoided to estimate parameters for being a considerable source of mistakes. For the sample reported in Fig. 1 $\beta_1 = 0.5$, $\alpha_2 = 0.85$, $\beta_3 = 0.27$.

Observation 6.1. We remark here the importance of a correct identification of the three components of the distribution. For instance, assuming the waist Zipf is prolonged to the tail would imply a finite support for the distribution. This is due to the fact that the shape parameters of the Zipf in the waist $\alpha_2 < 1$ (as in [12,15,16]), hence only summable over a finite support: requiring the estimation the content catalog size. Such estimation is however very difficult because of the very large confidence bands around the tail (see also [17] for model identification issues of web content size). In Fig. 1, we also report in red $\lceil f(k)\mathcal{S} \rceil / \mathcal{S}$, being $\mathcal{S}$ the total number of requests in the sample. This staircase shaped curve shows that the model well fits a finite size sample.

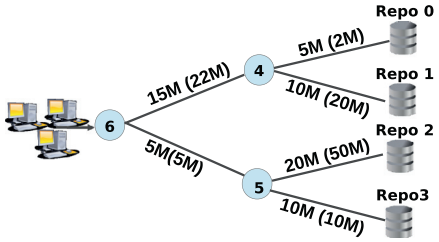

Fig. 2. Scenario 1: Topology Case I (Case II).

	Case I	Case II
Link	$x_{i,j}$ [Mbps] Exp/Optim	$x_{i,j}$ [Mbps] Exp/Optim
(0,4)	4.74 / 5	1.79 / 2
(1,4)	9.18 / 10	18.35 / 20
(2,5)	2.44 / 2.5	2.53 / 2.5
(3,5)	2.40 / 2.5	2.46 / 2.5
(4,6)	13.91 / 15	20.12 / 22
(5,6)	4.82 / 5	4.85 / 5

Fig. 3. Scenario 1: Average link rate.

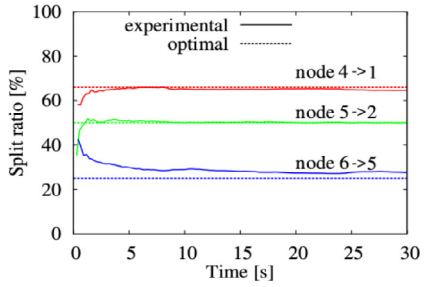

Fig. 4. Scenario 1: Split ratio convergence.

Such workload model is adopted in the tests presented in this section employing a dynamic workload, down-scaling the amount of available content and number of active users, due to NDN throughput limitations.

6.3. Scenario 1: Efficiency in multipath flow control

We first focus on a simple multipath network scenario (see Fig. 2), where users are connected to four repositories via four non disjoint paths. Users in the unique access node #6 retrieve twenty different named objects with a common prefix (e.g. ndn://amazon.com/{object1,...,object20}). Two sets of link capacities are separately considered, with small and large range of values (Case I and II), to assess the efficiency of the per-route remote delay control in presence of highly different delays (up to 1:25 ratio). In this scenario, we neglect the impact of caches studied later on. Fig. 3 reports the average link utilization in the two cases in comparison with the expected optimal usage. Notice that, in both cases, links (0, 4), (1, 4), (5, 6) are bottlenecked and achieve full utilization. Instead, links (2, 5), (3, 5) are not bottlenecked and the request forwarding algorithm load balances requests arriving at node 5 in a nearly 1:1 ratio. Fig. 4 shows the split ratios, Ψ, over time at nodes #4, #5 and #6 towards nodes #1, #2 and #5 respectively (the remaining traffic is routed towards #0, #3, #4). As expected, split ratios converge to the optimal values. However, slower convergence time can be observed at the client node #6, due to a globally higher response time, compared to in-path nodes, which slows down pending Interest number updates.

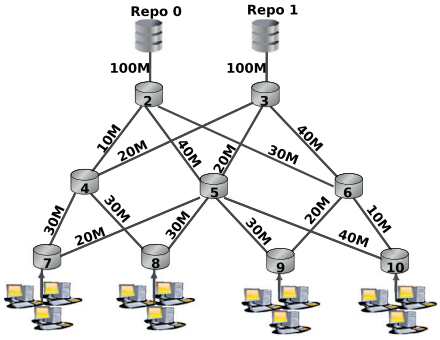

Fig. 5. Scenario 2 topology.

	Optimal	Experimental		
Link	0 Cache	0	$1.4GB$	$2.8GB$
(4,2)	33.33	36.20	37.51	37.98
(5,2)	66.66	62.14	59.12	58.01
(6,2)	50	49.47	49.58	49.65
(7,4)	50	52.61	54.37	55.19
(8,4)	50	46.90	47.33	47.67
(9,5)	50	52.93	56.13	56.97
(10,5)	66.66	68.11	70.88	71.85

Fig. 6. Scenario 2: average split ratios.

6.4. Scenario 2: impact of in-network caching

We consider a typical CDN scenario to evaluate the impact of in-network caching on the proposed congestion control and forwarding mechanisms. The topology in Fig. 5 is composed by fifteen nodes and eleven routes, available to each user to retrieve content from intermediate caches (if hitting) or repositories. Content catalog, cache sizes, and popularity are synthesized from the traffic measurements described above: each repository stores the entire content catalog composed by 1000 objects divided in 1 000 pkts of 4kB. Object requests arrive from leaf nodes (users) according to a Poisson process of rate 0.5 objects/s and a Weibull tailed popularity distribution. Nodes are equipped with an LRU cache whose size vary in the different experiments. In particular, leaf nodes (#7 to #10) are equipped with a cache of size ranging from 50MB to 200MB, while other nodes's cache size (#2 to #6) vary from 100MB to 400MB in the different presented tests. Each experiment lasts more than twelve hours (more than 20 000 object requests per experiment) and the output is averaged over multiple runs. Realized split ratios for different cache sizes and expected optimal values with no caches are reported in the table in Fig. 6. Fig. 7 reports the load reduction on links (2, 5), (3, 5) and, total and server (accounting only the links toward the servers) bandwidth savings with increasing cache sizes. Bandwidth saving is defined as $\frac{\sum_{i,j} x_{i,j}(S=0) - x_{i,j}(S)}{\sum_{i,j} c_{i,j}}$ where S is the total cache size in the system, expressed in pkts, $x_{i,j}(S=0)$, $x_{i,j}(S)$ are link rates in absence and presence of cache, respectively. Of course, bandwidth savings increase with caches sizes and link load decreases accordingly, but most of the advantages are achieved with 100MB (nodes #7 to #10) and 200MB (nodes #2 to #6) cache sizes (10% and 20% of total catalog). Notice that, we considered naïve LRU caching in this experiment. Hence, there is space for improvements in caching performances using smarter content eviction policies or cache coordination techniques (i.e. probabilistic caching , See [29,32]).

In Fig. 6, we also report the average split ratios at the different network nodes when varying cache sizes. Face selection adapts (e.g. at nodes #5, #9, #10) to the availability of network caching by reducing balancing the load on links (2, 5) and (3, 5), (see Fig. 7).

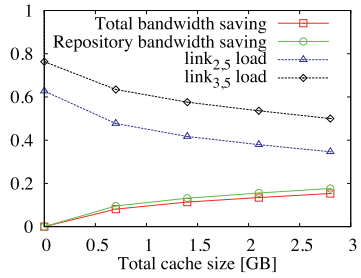

Fig. 7. Scenario 2: bandwidth/Storage tradeoff.

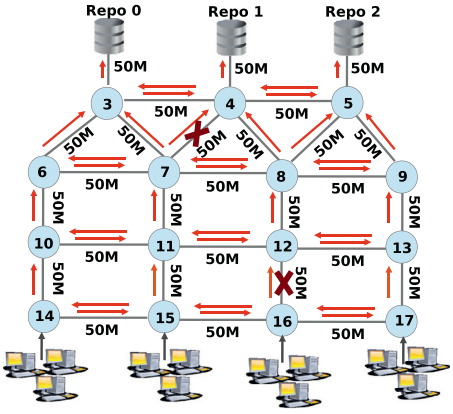

Fig. 8. Scenario 3: topology.

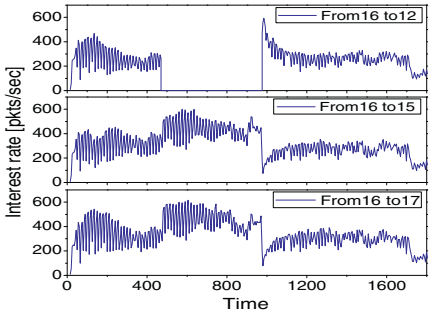

Fig. 9. Scenario 3: time evolution.

The bandwidth bottlenecks are moved down for the most popular objects (cached downstream), affecting optimal bandwidth sharing and split ratios as a result (similar considerations are made in [6]).

6.5. Scenario 3: reaction to link failure in a mobile back-haul

Traffic back-hauling for LTE mobile networks is usually based on over-provisioned links and highly redundant topologies. The objective is to support traffic growth in a reliable way, over multiple access technologies (e.g. HSPA, LTE). We select this network scenario to test protocols' responsiveness to link failure. In Fig. 8, we report a representation of a small part of such network segment with down-sized link rates (from Gbps to Mbps). The repositories are located at the place of regional/national PoPs. A failure event occurs at time $t = 500$s as in Fig. 8: two links, (16-12) and (7-4) are dropped by removing the associated NDN FIB entry at nodes #16 and #7. The workload is the same as in scenario #2 with the same average load. Let us focus on node #16. Rate evolution over time (in Fig. 9) shows how congestion control and forwarding protocols rapidly adapt to the different network conditions. Before the failure, request from node #16 are equally splitted among nodes #12,#15,#17, i.e. split ratios are equal to 1/3. During the failure,

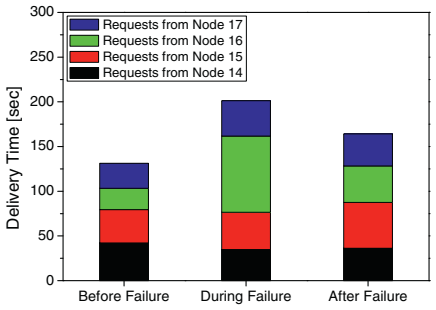

Fig. 10. Scenario 3: delivery time.

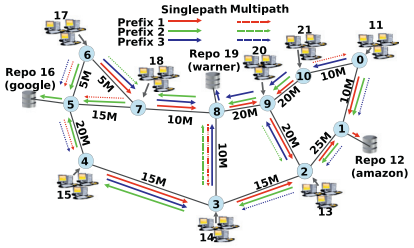

Fig. 11. Abilene network with available paths from clients to content repositories.

they grow up to 1/2 for nodes #15,#17, to compensate for link failure over (16, 12). Once (16, 12) becomes available, the request forwarding protocol sends most of the Interests across such link to probe the response time, until the number of pending Interests on (16, 12) reaches the same level attained by the other two faces and the split ratios stabilize to the optimal values.

In Fig. 10, we report the average content delivery time estimated at the receiver at the different access nodes in the three test phases (requests for different nodes are plotted one on top of the other). Users' performance are determined by the maximum throughput of the topology (an aggregate of 150 Mbps at the repositories) that is fairly shared. However user's at node #16 are forced to use longer paths during the failure and pay for more network congestion.

7. Request forwarding scalability

We consider the performance of the presented protocols, in a backbone network topology, The Abilene topology in Fig. 11. There are three repositories at nodes #12, #16, #19, each storing content under a given name prefix (respectively, ndn:/amazon.com, ndn:/google.com, ndn:/warner.com). Each catalog has the same characteristics as that in the previous scenario. Users' requests follow a Poisson process with different rates: for nodes #11, #13, #14, #15, #17, #18, #20, #21 rates are respectively 45, 48, 36, 42, 30, 24, 30, 33 objects per 100 seconds. In order to have a stationary number of data transfers in the network, rates are chosen to keep the total offered load below network service capacity.

Single vs multiple path comparison. While aggregate performance always improve using multiple paths, we report the user performance for the different name prefixes. Fig. 12 shows, as slices one on the of the other, the average delivery times per name prefix and per user's access node in the two cases: single and multiple paths. The multipath solution reduces significantly delivery times, except at nodes #14,#15, which experience a little higher latency due to the increased number of flows, exploiting links (3, 4), (4, 5), which fairly share available bandwidth. Still, the overall gain using multipath is significant.

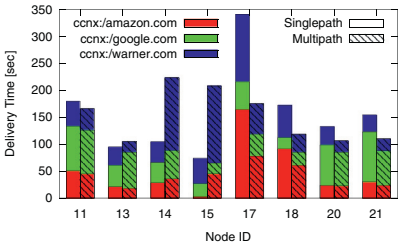

Fig. 12. Average Delivery time at each node, for the three name prefixes and with or without multipath enabled.

Object name vs prefix request forwarding. Bandwidth sharing is clearly affected by in-network caching. Indeed, content can be retrieved from different locations according to their popularity and in principle, nodes should route requests on a per-object name basis to capture such dependency.However, the approach suffers from scalability issues in terms of FIB size and results unfeasible for typical catalog sizes. This consideration motivates our design choice to route requests based on prefixes rather than names. In this scenario, we quantify the impact of forwarding granularity and observe that the number of object names that require separate FIB state is much smaller than expected. We give here an intuition. When content popularity under a given prefix is uniformly distributed, caching becomes irrelevant: all flows are routed up to the repositories with a single prefix FIB entry. When content popularity is long-tailed, the small portion of popular cacheable objects would benefit from per-object name forwarding. Thus, the amount of useful additional FIB state is of the same order of the most popular objects stored in a local cache.

Here, we consider a single name prefix (ccnx:/google), a single repository (repo #16, storing 1000 objects of size 1000 pkts of 4KB) and a single access node (#3) with different bidirectional link capacities for links (3,4), (4,5)=25 Mbps , (5,7)=5 Mbps and (7,8), (8,3)=50 Mbps. We also insert two caches at nodes #7-#8 of sizes 100MB (25 objects). Requests arrive to node #7 according to a Poisson process of rate 1 object/s according to a zipf popularity with parameter $\alpha = 1.3$. Under this setting, the most popular objects (small rank) are retrieved from the cache of node #8, twice as fast as the alternative paths to the repositories (#3, #4, #5). We run two sets of experiments: one with per name prefix based forwarding (PBF) and the other with a FIB entry for each content object name (OBF) in the catalog.

Hybrid object name - Prefix request forwarding. In this experiment we show the benefits of the hybrid approach defined in Section 5.4, combining OBF for the most popular objects with PBF for all the others. Clearly an OBF approach applied to every object would not be scalable while PBF is suboptimal by definition. As expected, Fig. 13 confirms that the hybrid approach achieves the performance of the optimal OBF mechanism in terms of content delivery time, while reducing considerably the overall forwarding state.

8. Related work

A large body of work addresses joint multipath routing and congestion control in TCP/IP networks. In [20,24,36] authors study the stability of multipath controllers and derive a family of optimal controllers to be coupled with TCP. Various transport protocol implementations based on TCP are proposed (e.g. mTCP [42], MPTCP [37]), either loss-based or delay-based ([5]). Multipath TCP [37], as proposed by the IETF working group *mptcp*, improves previous designs in terms of efficient network usage utilization and of RTT unfairness compensation. All proposals require to establish separate per-path connections (subflows) with a given number of known

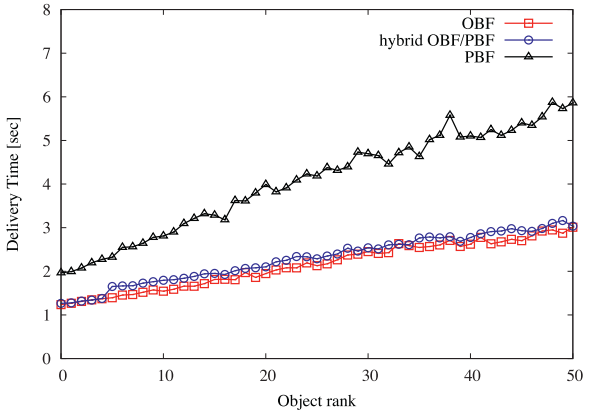

Fig. 13. Average content delivery time by using, (i) full object based forwarding (OBF) (ii), hybrid forwarding (OBF/PBF) and (iii) prefix based forwarding (PBF).

sources that do not vary over time. Subflow may be uncoupled, i.e. independently managed, like in [42], or coupled as in MPTCP. Such approaches are not suitable to ICN, where the sources are a priori unknown and may vary over time, subject to in-network cache dynamics and on-the-fly request forwarding performed at network nodes. Also, unlike our work, previous multipath transport efforts assume a sender-based windowed congestion control. However, there exists a fair amount of work on *receiver-driven congestion control* that the interested reader can find summarized in [28].

In the context of ICN, an adaptive forwarding mechanism is sketched out in [39]. Interface ranking is performed at NDN routers according to a three colors scheme. Congestion control in [39] is still under definition according to the authors and envisaged via Interest NACKs (Negative Acknowledgments) sent back to the receiver. Such explicit congestion notification approaches may require message prioritization (opening to security weaknesses) and bring additional overhead. In a similar spirit, TECC [38] looks at the joint problem of routing and caching in NDN within an optimization framework, mostly focusing on bandwidth and cache provisioning at longer timescales.

9. Conclusions

Previous work on ICN mostly magnifies one specific aspect of the paradigm at a time, like in-network caching or adaptive forwarding, while devoting less attention to the novel communication model as a whole. Overall end-user experience and network-wide resource utilization are often left out of the picture. Also, one may argue that the limited benefits of a single building block taken apart, like in-network caching, can be approached by off-the shelf solutions, so missing the bigger potential of the ICN idea head-to-toe.

In this paper, we tackle the problem of a joint multipath congestion control and request forwarding in ICN and design scalable, fully distributed and inter-working mechanisms for efficient point-to-multipoint content delivery. More precisely, we formulate and solve a global optimization problem with the twofold objective of maximizing end-user throughput, while minimizing network cost. Our framework explicitly accounts for the interplay between multipath flow control, in-path caching and in-network request scheduling. As a result, we derive: *(i)* a family of receiver-driven multipath controllers, leveraging a unique congestion window and per-route delay monitoring. *(ii)* A set of optimal dynamic request forwarding strategies based on the number of pending requests which is the base for the design of a fully distributed algorithm run at ICN nodes. Note that, beyond ICN, the problem of multipath congestion control with a priori unknown and variable sources (due to in-path caching) has never been considered before.

To assess the performance of the design, we have implemented the proposed protocols in NDN [41] and set up a testbed for large scale experimentation. The experimental evaluation in different network scenarios confirms efficiency and global fairness, as expected by the optimal solution, and robustness w.r.t. the inherent variability in delay brought by in-path caching and network congestion under realistic workloads. Additional experiments can be found in [11].

However, we have observed important limitations of the NDN forwarding engine preventing to scale beyond 150 Mbps of throughput. As a future work, we plan to investigate the design of name-based forwarding engines on a high-speed ICN data plane [34]. Moreover, we foresee a theoretical analysis of PIT/FIB dynamics and plan to develop realistic models to predict their dynamics.

Several assumptions are made in this work about the existence of a number of protocols that produce an input to what is presented in this paper. Namely a routing protocols is assumed to populate the FIB with name prefixes and output faces to route the request toward the destinations. It is also assumed that the client is able to issue the set of request names composing the whole content object. This can be obtained in several ways, namely by using a pre-fetched manifest or by using a name discovery protocol.

Acknowledgments

This research work has been partially funded by the Technological Research Institute SystemX, within the project "Network Architectures" hosted at LINCS.

We thank the students: Michele Papalini (Cisco Systems) for helping testing flow control modules, Sen Wang (Tsinghua University) for developing the load balancer and the flow controller in NDN strategy layer, Dengyuan Zhou (Eurecom) for helping processing data sets for characterizing the traffic demand. We also thank Claudio Imbrenda (IBM) for the performance improvements of LRU, and fast implementations of the virtual repository originally developed at Washington University in the NDN project. We thank the Grid'5000 team for providing access and support to their infrastructure for large scale experimentation.

Appendix A. Proof of Prop. 5.1

Proof. Let us denote with $(\bar{\mathbf{y}}, \bar{p})$ the equilibrium point of Eq. (30–34), where $\mathbf{y}(t) = (y_1(t), \ldots, y^n(t))$. Let us consider the following Lyapunov function

$$V(\mathbf{y}(t), p) = \frac{1}{2}\sum_{n=1}^{N} \Delta y^n(t)^2 + \frac{1}{2}\zeta\,\Delta p(t)^2 \tag{A.1}$$

where $\Delta y^n(t) = y^n(t) - \bar{y}^n$, $\Delta p(t) = p(t) - \bar{p}_n$ and $\zeta = \frac{\Delta R_{\max}\beta\bar{C}(y^n)^2}{\Delta p_{\max}}$. Clearly, $V(\bar{\mathbf{y}}, \bar{p}) = 0$, $V(\mathbf{y}, p) \geq 0$ and

$$\begin{aligned}
\dot{V} &= \sum_{n=1}^{N} \Delta y^n \dot{y}^n + \zeta\,\Delta p\ \dot{p} = \\
&= \sum_{n=1}^{N} \Delta y^n \left(\frac{\eta}{\bar{R}^2} - \beta y^{n,2} p\right) + \frac{\beta C^2 \Delta p}{N^2}\sum_{n=1}^{N}\Delta y^n \\
&= \sum_{n=1}^{N} \Delta y^n \left(\frac{\eta}{\bar{R}^2} - \beta y^{n,2} p + \frac{\beta C^2 p}{N^2} - \frac{\beta C^2 \eta N^2}{\bar{R}^2 \beta N^2 C^2}\right) \\
&= -\beta p \sum_{n=1}^{N} \Delta y^n (y^{n,2} - y^{\bar{n},2}) \\
&= -\beta p \sum_{n=1}^{N} (\Delta y^n)^2 (y^n + \bar{y}^n) < 0
\end{aligned}$$

where we omitted the indication of the time variable t. Therefore $\dot{V}(\mathbf{y}, p)$ is negatively semi-definite for any ball including the equilibrium point. This proves that $(\bar{\mathbf{y}}, \bar{p})$ is a globally stable equilibrium as per [35]. Let us consider the case with delays and start from the same Lyapunov function.

$$\begin{aligned}
\dot{V} &= \sum_{n=1}^{N} \Delta y^n \dot{y}^n + \zeta\,\Delta p \dot{p} \\
&= -\beta \sum_{n=1}^{N} \Delta y^n (y^{n,2} p(t-R) - y^{\bar{n},2} p(t)) \\
&= -\beta \sum_{n=1}^{N} \Delta y^n \left(y^{n,2} p - y^{\bar{n},2} p - y^{n,2}\int_{t-R}^{t} \dot{p}(u)du\right) \\
&\leq -\beta \sum_{n=1}^{N} \Delta y^n \\
&\left(y^{n,2} p - y^{\bar{n},2} p - \frac{y^{n,2}\Delta p_{\max}}{C\Delta R_{\max}}\int_{t-R}^{t} \Delta y^n(u)du\right)
\end{aligned}$$

We apply the Lyapunov-Razumikhin theorem [18] to compute a stability region for the systems. Hence, by assuming $V(\mathbf{y}(u), p(u)) \leq V(\mathbf{y}(t), p(t))$ as $u \in [t-R, t]$ we need to show that $\dot{V}(\mathbf{y}(t), p(t)) < 0$ in a non empty region including the equilibrium.

$$\begin{aligned}
&\dot{V}(\mathbf{y}(t), p(t)) \leq \\
&-\beta \sum_{n=1}^{N} \Delta y_i \left(p(y^{n,2} - y^{\bar{n},2}) - y^{n,2}\kappa R\sqrt{\zeta + (\Delta y^n)^2}\right)
\end{aligned}$$

with $\kappa = \Delta p_{\max}/(C\Delta R_{\max})$. It is easy to see that $\dot{V}(\mathbf{y}(t), p(t)) < 0$ as long as $y_i < \bar{y}^n$ for all n. Note that by assuming $y^n > \bar{y}^n$ for all n we are computing a smaller stability region, as in general $y^n < \bar{y}^n$ may hold for some n only. To have $\dot{V}_t < 0$, when $y^n > \bar{y}^n$ $\forall n$ it is sufficient to impose $p(1 - y^{\bar{n},2}/y^{n,2})) > \kappa R\sqrt{\zeta + (\Delta y^n)^2})$. The exact y^n range where the inequality holds can be numerically computed. However, since we are interested in a region of attraction including the equilibrium point, we can provide a smaller region of attraction for $y^n \approx \bar{y}^n$. In this case, $\sqrt{\zeta + (\Delta y^n)^2} \approx \sqrt{\zeta}$ and $y^n + \bar{y}^n \approx 2\bar{y}^n$. Thus, a region of attraction for the considered equilibrium point is $\left\{(\mathbf{y}(t), p(t)) : y^n < \bar{y}^n\left(1 + \frac{\kappa\bar{R}\sqrt{\zeta}}{p_{min}}\right), \forall n\right\}$. Remark that $k\sqrt{\zeta} = \bar{y}^n\sqrt{\frac{\Delta p_{max}\beta}{\Delta R_{max}}} \propto \sqrt{(p_{max} - p_{min})\beta}$. □

References

[1] The Grid'5000 Experimentation Test-Bed. http://www.grid5000.fr.
[2] D. Ardelean, E. Blanton, M. Martynov, Remote active queue management, in: Proc. of ACM NOSSDAV, Braunschweig, Germany, 2008.
[3] F. Baccelli, G. Carofiglio, M. Piancino, Stochastic Analysis of Scalable TCP, in: Proc. of IEEE INFOCOM, Rio de Janeirio, Brazil, 2009.
[4] D. Bertsekas, J. Tsitsiklis, Parallel and Distributed Computation: Numerical Methods, Athena Scientific, 1997.
[5] Y. Cao, M. Xu, X. Fu, Delay-based congestion control for multipath TCP, in: Proc. of IEEE ICNP, Austin, Texas, USA, 2012.
[6] G. Carofiglio, M. Gallo, L. Muscariello, Bandwidth and storage sharing performance in information centric networking, in: Proc. of ACM SIGCOMM ICN, Torono, Canada, 2011.
[7] G. Carofiglio, M. Gallo, L. Muscariello, Bandwidth and storage sharing performance in information centric networking, Elsevier Sci. Comput. Netw. J. 57 (17) (2013).
[8] G. Carofiglio, M. Gallo, L. Muscariello, L. Papalini, Multipath congestion control in content-centric networks, in: Proc. of IEEE INFOCOM NOMEN, Turin, Italy, 2013.
[9] G. Carofiglio, M. Gallo, L. Muscariello, M. Papalini, S. Wang, Optimal multipath congestion control and request forwarding in information-centric networks, IEEE ICNP, Gottingen, Germany, 2013.
[10] G. Carofiglio, M. Gallo, L. Muscariello, D. Perino, Pending interest table sizing in named data networking, in: Proc. of ACM ICN, San Francisco, CA, USA, 2015.
[11] G. Carofiglio, M. Gallo, L. Muscariello, D. Perino, Scalable mobile backhauling via information-centric networking, in: Proc. of IEEE LANMLAN, Beijing, China, 2015.

[12] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, S. Moon, I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system, in: Proc. ACM SIGCOMM IMC, San Diego, CA, USA, 2007.
[13] R. Chiocchetti, D. Rossi, G. Rossini, G. Carofiglio, D. Perino, Exploit the known or explore the unknown: Hamlet-like doubts in information-centric networking, in: Proc. of ACM SIGCOMM ICN, Helsinki, Finland, 2012.
[14] R.B. D'Agostino, M.A. Stephens (Eds.), Goodness-of-fit Techniques, Marcel Dekker, Inc., New York, NY, USA, 1986.
[15] S.K. Fayazbakhsh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. Maggs, K. Ng, V. Sekar, S. Shenker, Less pain, most of the gain: Incrementally deployable icn, in: Proc. of ACM SIGCOMM, Hong Kong, China, 2013.
[16] C. Fricker, P. Robert, J. Roberts, A versatile and accurate approximation for cache performance, 24th International Teletraffic Congress, Krakow, Poland, 2012.
[17] W. Gong, Y. Liu, V. Misra, D. Towsley, On the tails of web file size distributions, in: Proc. of 39th Allerton Conference on Communication, Control, and Computing, Monticello, Illinois, USA, 2001.
[18] J.K. Hale, S.M. Lunel, Nonlinear Differential Equations and Dynamical Systems, Springer-Verlag New York, 1990.
[19] D. Han, A. Anand, F.e.a. Dogar, XIA: Efficient support for evolvable internetworking, in: Proc. USENIX NSDI, San Jose, CA, USA, 2012.
[20] H. Han, S. Shakkottai, e.a. Hollot, Multi-path TCP: a joint congestion control and routing scheme to exploit path diversity in the Internet, IEEE/ACM Trans. Netw. 14 (6) (2006) 1260–1271.
[21] C. Hollot, Y. Chait, Nonlinear stability analysis for a class of tcp/aqm networks, in: In Prof of IEEE Conference on Decision and Control, Orlando, FL, USA, 2001.
[22] V. Jacobson, D. Smetters, J. Thornton, al., Networking named content, in: Proc. of ACM CoNEXT, Rome, Italy, 2009.
[23] T. Janaszka, D. Bursztynowski, M. Dzida, On popularity-based load balancing in content networks, in: Proc. of ITC24, Krakow, Poland, 2012.
[24] F. Kelly, T. Voice, Stability of end-to-end algorithms for joint routing and rate control, ACM SIGCOMM CCR 35 (2) (2005) 5–12.
[25] F.P. Kelly, A.K. Maulloo, D.K. Tan, Rate control for communication networks: shadow prices, proportional fairness and stability, J. Oper. Res. soc. 49 (3) (1998) 237–252.
[26] T. Koponen, M. Chawla, B. Chun, A. Ermolinskiy, K. Kim, S. Shenker, I. Stoica, A data-oriented (and beyond) network architecture, in: Proc. of ACM SIGCOMM, Kyoto, Japan, 2007.
[27] T. Koponen, S. Shenker, H.e.a. Balakrishnan, Architecting for innovation, ACM SIGCOMM CCR 41 (3) (2011) 24–36.
[28] A. Kuzmanovic, E.W. Knightly, Receiver-centric congestion control with a misbehaving receiver: Vulnerabilities and end-point solutions, Elsevier Comput. Netw. 51 (10) (2007) 2717–2737.
[29] N. Laoutaris, H. Che, I. Stavrakakis, The LCD interconnection of LRU caches and its analysis, Perform. Eval. 63 (7) (2006) 609–634.
[30] E. Nordstrom, D. Shue, P.e.a. Gopalan, Serval: an End-Host Stack for Service–Centric Networking, in: Proc. USENIX NSDI, San Jose, CA, USA, 2012.
[31] D. Palomar, M. Chiang, A tutorial on decomposition methods for network utility maximization, Sel. Areas Commun. IEEE J. 24 (8) (2006) 1439–1451.
[32] I. Psaras, W.K. Chai, G. Pavlou, Probabilistic in-network caching for information-centric networks, in: Proc. of ACM SIGCOMM ICN, Helsinki, Finland, 2012.
[33] R. Srikant, The Mathematics of Internet Congestion Control (Systems and Control: Foundations and Applications), SpringerVerlag, 2004.
[34] M. Varvello, D. Perino, L. Linguaglossa, On the design and implementation of a wire-speed pending interest table, in: Proc. of IEEE INFOCOM NOMEN, Turin, Italy, 2013.
[35] F. Verhulst, Introduction to Functional Differential Equations, Springer-Verlag New York, 1993.
[36] W.-H. Wang, M. Palaniswami, S.H. Low, Optimal flow control and routing in multi-path networks, Perform. Eval. 52 (2–3) (2003) 119–132.
[37] D. Wischik, C. Raiciu, A. Greenhalgh, M. Handley, Design, implementation and evaluation of congestion control for multipath TCP, in: Proc. of USENIX NSDI, Boston, MA, USA, 2011.
[38] H. Xie, G. Shi, P. Wang, TECC: Towards collaborative in-network caching guided by traffic engineering, in: Proc. of IEEE INFOCOM Mini conference, Orlando, FL, USA, 2012.
[39] C. Yi, A. Afanasyev, L. Wang, B. Zhang, L. Zhang, Adaptive forwarding in named data networking, ACM SIGCOMM CCR 42 (3) (2012).
[40] H. Yuan, T. Song, P. Crowley, Scalable NDN Forwarding: Concepts, Issues and Principles, in: Proc. of IEEE ICCCN, Munich, Germany, 2012.
[41] L. Zhang, al., Named Data Networking (NDN) Project, 2010. http://named-data.net/ndn-proj.pdf.
[42] M. Zhang, J. Lai, A. Krishnamurthy, L. Peterson, R. Wang, A transport layer approach for improving end-to-end performance and robustness using redundant paths, in: Proc. of USENIX, San Diego, CA, USA, 2004.

Giovanna Carofiglio received the Dr Ing degree in telecommunication engineering and in electronic and telecommunication engineering, both from Politecnico di Torino, Italy, in 2004, and the PhD in telecommunication engineering jointly from Politecnico di Torino and Telecom Paris- Tech, Paris, France, in 2008. Her graduate reseach focused on stochastic analysis of wired and wireless networks and has been performed at Politecnico di Torino and at Ecole Normale Superieure (ENS Ulm) in the INRIA-TREC group. She spent more than six years at Bell Labs as hea d of the research department on content networking. She works currently at Cisco Systems as Distinguished Engineer. She was general co-chair of ACM ICN 2014. She is a member of the IEEE.

Massimo Gallo received the Bachelor and the Master of Science degrees in computer and communication networks from Politecnico di Torino in 2006 and 2008, the Ph.D. in computer science at Telecom ParisTech (Paris, France) in 2012. He is member of technical staff at Bell Labs, now subsidiary of Nokia (previously Alcatel-Lucent). His main research interests are on performance evaluation, simulation and experimentation of caching and transport mechanisms for Information-Centric Networking and Peer to Peer network analysis. He is a member of the IEEE.

Luca Muscariello received the Dr Ing degree in telecommunication engineering in 2002 and the Ph.D. in electronics and communications engineering in 2006 both from Politecnico di Torino, Italy. He works at Cisco Systems as Principal Engineer and is a research associate at the IRT SystemX. He spent ten years working at Orange Labs (former France Telecom R&D) on doing research and innovation in networking. He was program co-chair of Valuetools 2013, TPC chair of ACM ICN 2014 and general co-chair of ACM ICN 2014. He is a member of the ACM and a senior member of the IEEE and SEE.