Efficient Content Delivery through Fountain Coding in Opportunistic Information-Centric Networks

George Parisis, Vasilis Sourlas, Konstantinos V. Katsaros,
Wei Koong Chai, George Pavlou, Ian Wakeman

PII: S0140-3664(16)30649-1
DOI: 10.1016/j.comcom.2016.12.005
Reference: COMCOM 5420

To appear in: *Computer Communications*

Received date: 28 August 2016
Revised date: 29 November 2016
Accepted date: 3 December 2016

Please cite this article as: George Parisis, Vasilis Sourlas, Konstantinos V. Katsaros, Wei Koong Chai, George Pavlou, Ian Wakeman, Efficient Content Delivery through Fountain Coding in Opportunistic Information-Centric Networks, *Computer Communications* (2016), doi: 10.1016/j.comcom.2016.12.005

Efficient Content Delivery through Fountain Coding in Opportunistic Information-Centric Networks

George Parisis[a], Vasilis Sourlas[b], Konstantinos V. Katsaros[c], Wei Koong Chai[d], George Pavlou[b], Ian Wakeman[a]

[a] *School of Engineering and Informatics, University of Sussex, UK.*
[b] *Department of Electronic and Electrical Engineering, University College London, UK.*
[c] *Intracom Telecom S.A., Athens, Greece.*
[d] *Department of Computing and Informatics, Bournemouth University, UK.*

Abstract

Opportunistic networks can increase network capacity, support collaborative downloading of content and offload traffic from a cellular to a cellular-assisted, device-to-device network. They can also support communication and content exchange when the cellular infrastructure is under severe stress and when the network is down or inaccessible. Fountain coding has been considered as especially suitable for lossy networks, providing reliable multicast transport without requiring feedback from receivers. It is also ideal for multi-path and multi-source communication that fits exceptionally well with opportunistic networks. In this paper, we propose a content-centric approach for disseminating content in opportunistic networks efficiently and reliably. Our approach is based on Information-Centric Networking (ICN) and employs fountain coding. When tied together, ICN and fountain coding provide a comprehensive solution that overcomes significant limitations of existing approaches. Extensive network simulations indicate that our approach is viable. Cache hit ratio can be increased by up to five times, while the overall network traffic load is reduced by up to four times compared to content dissemination on top of the standard Named Data Networking architecture.

Keywords: information-centric networks, opportunistic networks, fountain coding, in-network caching, multi-source and multi-path content delivery.

1. Introduction

Smartphones are nowadays ubiquitous. They support multiple wireless technologies, such as LTE, Wi-Fi, Wi-Fi Direct and Bluetooth, which allow them

Email addresses: g.parisis@sussex.ac.uk (George Parisis), v.sourlas@ucl.ac.uk (Vasilis Sourlas), konkat@intracom-telecom.com (Konstantinos V. Katsaros), wchai@bournemouth.ac.uk (Wei Koong Chai), g.pavlou@ucl.ac.uk (George Pavlou), ianw@sussex.ac.uk (Ian Wakeman)

to flexibly access the Internet and communicate with devices in their vicinity. Modern smartphones are able to form wireless networks with other nearby devices opportunistically, while being connected to (and accessing the Internet through) Wi-Fi access points or cellular providers, *e.g.,* [1][2]. The formation of such networks can also be assisted by cellular providers, *e.g.,* [3][4]. The concept of opportunistic networking is about closing the gap between human and network behaviour through the exploitation of the natural human mobility as an ideal opportunity to facilitate content dissemination [5][6][7]. Mobility, which is usually perceived as a degrading factor for the network performance, is now considered as an opportunity rather than a challenge to cope with [8]. In an opportunistic network, devices spontaneously connect to each other (ideally without human intervention) and opportunistic network topologies change over time due to node mobility and energy preservation strategies (*e.g.,* when a mobile device that is for instance a Wi-Fi Direct Group Owner passes ownership to become a regular client).

Opportunistic networks operate following the *store-carry-forward* paradigm, where intermediate nodes locally store and carry content until a forwarding opportunity arises. Usually, the terms opportunistic and delay-tolerant networks (DTNs) are used interchangeably, but according to the DTN definition given in [6] opportunistic networks correspond to a more general concept that includes DTNs. According to [6], DTNs consist of a network of independent networks that are presented with occasional communication opportunities among them with known and isolated disconnection points, whereas in opportunistic networks both disconnection points and communication opportunities are usually random.

Opportunistic networks can increase network capacity [9][10], support collaborative content downloading [11] and offload traffic from a cellular to a cellular-assisted device-to-device network [3][12]. They can also support communication and content exchange when the cellular infrastructure is under severe stress [2][13][14], since it is un-economical to provide more capacity (*e.g.,* through portable cells or Wi-Fi access points) for events that take place rather infrequently. At the same time, opportunistic networks are the only means of communication when the network infrastructure is down or inaccessible due to natural disasters or government censorship (or just because it is not trusted[1]). In most of the above-mentioned scenarios communication is all about the exchange of content (*e.g.,* news, video-on-demand, emergency announcements, *etc.*), which in many cases can be of interest to multiple participants; *e.g.,* updates on concurrent sports matches, departure times for public transportation after a football match, assembly points for emergency scenarios or public demonstrations, *etc.*

In opportunistic networks, connectivity among devices is intermittent and communication can be very lossy. This might decrease the possibility for successful content forwarding. However, the fact that user movement and mobility

[1]See http://tinyurl.com/ogsz75o as an example of a real world scenario.

patterns have limited degree of freedom and variation, and rather exhibit structural patterns due to geographical and social constraints [8][15][16], minimizes this uncertainty. The dynamic and collaborative nature of opportunistic networks suggests that the usage of multiple content origins (along with caches) and multiple network paths to disseminate content would be extremely advantageous, especially when connectivity is not stable. The volatility of resource availability urges for efficient network utilization; *e.g.,* avoiding unnecessary (re-)transmissions. At the same time, all these need to be supported in the context of a lossy wireless environment, calling for mechanisms that ensure reliable content delivery. All in all, caching, multi-source, multi-path and multicast forwarding, along with reliable content delivery emerge as crucial properties that opportunistic networks should support.

However, opportunistic networks are mainly built on top of TCP/IP, which only supports unicast data transport among devices (even if the underlying wireless medium is a broadcast one). Sporadic connectivity results in network layer configuration and requires TCP connection re-establishment as the network topology changes. Moreover, wireless connectivity can be very lossy, bringing TCP, which reacts to stochastic losses as if congestion existed, to its knees. Current approaches require (usually application-level) coordination in order to fetch all data chunks from potentially multiple devices to complete a data transfer. A rarest-first approach is commonly followed (just like in BitTorrent) when deciding which data chunk to exchange so that the probability that a few (extinct or very rare) content chunks, which prevent the successful completion of content transmission, remains low.

In this paper, we deal with the challenges identified above by combining the Named Data Networking (NDN) [17] architecture and fountain coding [18][19], as follows:

- In NDN, communication is not based on end-to-end connectivity. Instead, the focus is on the content and routing is based on content names; therefore, topology changes do not result in broken connections that must be constantly rebuilt. In-network caching and multicast forwarding are inherently supported.
- In fountain coding, encoding symbols equally contribute to the decoding of the original content and, therefore, there is no need to retransmit lost data. Instead, the reception of new symbols will eventually lead to the successful decoding of content. Accordingly, there are no "rare" symbols that devices need to desperately keep alive in the network.
- With fountain coding, multiple sources (*e.g.,* devices that can fetch data from the Internet or have previously cached encoding symbols) can send symbols to a requesting device without any coordination among them. They only need to randomize the way symbols are created to avoid sending duplicate symbols.
- In fountain coding, there is no ordering of symbols; all received symbols decode the content (with some probability). Therefore, multiple paths

can be used in multi-hop opportunistic networks. Receivers can also receive symbols of the same content from different network interfaces (*e.g.,* Bluetooth and Wi-Fi).

In our approach, *Persistent Interests* (PIs) [20] (see Section 3) are forwarded (through scoped flooding [21]) to connected devices, requesting fountain-coded symbols for specific content. By employing scoped flooding of PIs at NDN's *strategy* layer [17] and fountain coding, we enable multi-source and multi-path forwarding. Additionally, we incorporate Bloom filters as part of the PI name [22] to minimize sending duplicate symbols from in-network caches (*i.e.,* other mobile devices in the network) (see Section 3.2). Network dynamics (*i.e.,* user mobility and frequent disconnections) are modelled by changing the set of devices that supply the network with newly requested content and by breaking already established connections among mobile devices.

The proposed approach successfully inherits the benefits of Information-Centric Networking, as realized by NDN [17], *i.e.,* in-network caching and multicast, while at the same time it avoids redundancy associated with multi-source and multi-path communication in the considered environment, without necessitating any complex error control or source coordination mechanisms. The proposed solution, not only overcomes the limitations of TCP/IP, but also utilizes available resources more efficiently compared to its standard NDN counterpart, achieving higher cache-hit ratio and lower traffic overhead in the network. In particular, we observe that the usage of fountain coding can increase the cache-hit ratio by up to five times compared to the standard NDN [17], maintaining at the same time the overall traffic load of the opportunistic network in less than 25% of the load of the standard NDN.

2. Background

In this section, we briefly discuss relevant research regarding information-centric networking and fountain codes that forms the foundations of our work.

2.1. Information-Centric Networking

Internet usage patterns have been constantly changing over the last decades, especially after the wide adoption of smartphones, reaching a situation that was not foreseen when it was originally designed. The engineering principles underpinning today's Internet architecture were created in the 1960s and 1970s with the assumption that Internet would be mainly used for host-to-host communications. Differently, nowadays the vast majority of Internet usage is related to content distribution and retrieval and this trend is forecast to continue in the foreseeable future [23]. This has altered the network communication paradigm from simply routing packets between hosts (using endpoint addresses) to binding information producers and consumers together.

The mismatch between the original design assumptions (host- or node-centric) and the current usage patterns (information-centric) has been partially addressed through application layer, problem-specific solutions overlaid on top

of IP, *e.g.,* Content Distribution Networks (CDNs). However, the lack of native network support for content distribution restricts the efficiency of such approaches, and also potentially hinders the evolution of the Internet as a whole. This has created a trend towards content-oriented networking, which has recently been realized through the ICN paradigm. ICN puts content itself in the forefront of attention when it comes to content delivery. That is, content can be delivered from any network location/device, provided that this device holds a valid copy of the requested content. Several ICN architectures [24] have been proposed to support information access and delivery based on location-independent names, instead of endpoint network addresses.

Given the information-oriented nature of ICN, each information item is uniquely identified and authenticated without being associated to a specific host. This, in turn, means that multiple devices can be used to relay and/or serve interests for requested data. Furthermore, the in-network, name-based caching and forwarding capability of ICN [17][25][26], enables ubiquitous content distribution. That is, every cache-enabled network node potentially caches items that traverses it, and, hence, may serve future matching interests faster than the content origin/server.

The important role that ICN has in mobile and opportunistic networks, has been extensively analysed in [27] by identifying several challenges and discussing appropriate solutions for them. The unique characteristics of ICN has led to an attempt to combine its capabilities with those of DTNs in order to assist content delivery in challenged scenarios. A protocol stack which integrates the DTN architecture in native NDN [17] to deal with network disruptions is presented in [28]. In particular, the authors in [28] extend NDN routing strategies to integrate the Bundle protocol (BP) of the DTN architecture. Integrating BP in NDN, enhances the connectivity options of NDN and allows it to deal with network disruptions. Also, in [29], the authors propose a disruption-tolerant information-centric ad-hoc network to provide low-cost, bandwidth-efficient operations for Vehicular ad-hoc networks (VANET). Their solution is inspired by the family of peer-to-peer data dissemination networks (*e.g.,* Haggle [30]) and the usage of specialized interest and cache summary messages to synchronize the nodes of the VANET.

Our approach here is orthogonal to the works presented so far and is based on the NDN architecture [17] mainly because of its inherent support for *mobility*, *multicast* and *caching*. NDN relies on *Interest* packets, as content chunk requests for its content access operations, whereby each Interest has a one-to-one correspondence to a *Data* packet. A NDN router undertakes the following steps when an Interest packet is received.

1) Content Store (CS) lookup – The router tries to satisfy the request *locally* by checking if the respective content is cached in its local CS. If found, the router replies by forwarding the requested Data packet to the face the Interest arrived from and discards the Interest packet.

2) Pending Interest Table (PIT) lookup – If the router cannot serve the Interest, it checks if there is a pending request for the same content; if so,

an Interest packet with the exact content name would exist in its PIT; the Interest's arrival face is added to the PIT entry's *Requesting Faces* list and the Interest is discarded.

3) Forwarding Information Base (FIB) lookup – The Interest is forwarded upstream towards the content provider based on a matching FIB entry. The arrival face is removed from the face list of the matching FIB entry, and if the resulting list is not empty, the packet is sent out to one or more of the remaining faces (in the case of multiple content origins/replicas and based on a *Forwarding Strategy* [17]). A new PIT entry that records the Interest and its arrival face is also created.

When the requested Data packet is found (or created), it is forwarded back to the requesting node following the reverse path of the Interest packet whereby each router forwards the packet based on the arrival face indicated in the corresponding PIT entry. At each hop, the Data packet may be cached based on specific caching policies (*e.g.,* [17][31][32]) that are applied locally at each network node.

It has been argued that the one-to-one correspondence of Interest to Data packets can waste bandwidth and burden routers with the maintenance of state [20]. Furthermore, if an Interest is lost, the corresponding Data packet will not be forwarded; a "transport protocol" (*e.g.,* ICP [33]) is required to handle retransmissions of Interest packets. Our approach is based on *Persistent Interests* (PIs) similarly to [20] (see Section 3). PIs are not consumed directly by the first Data packet. Instead, they persist in a router's PIT, as soft state, potentially "serving" the whole content. In our approach, a PI will match *any* encoding symbol for that specific content. A PI is eliminated either when a fixed timer is triggered (to avoid stale Interests) or when an adequate number of symbols has been matched by the Interest. This number is slightly larger than the number of content fragments carried in Data packets so that the receiver can decode the original data (see the description of Fountain Codes below). Another similar approach to PIs that allows push-based delivery using multicast for timely but efficient delivery is the mechanism described in [34] [35] and can be used for the realization of the proposed content delivery mechanism.

2.2. Fountain Codes

Fountain coding was originally designed for point-to-multipoint transmissions of large files over lossy channels [18][19][36], but its use in content distribution networks [37][38], collaborative downloading [39], content retrieval in sensor networks [40] and ICN [41][42] has demonstrated significant benefits. Fountain coding is ideal for supporting the following types of data transport, all of which are compatible with ICN and opportunistic network communication.

Multicast (One-to-Many) – Fountain coding has been a major component of reliable multicasting proposals [43], where clients acquire bulk data in the absence of feedback channels. Congestion control can be implemented

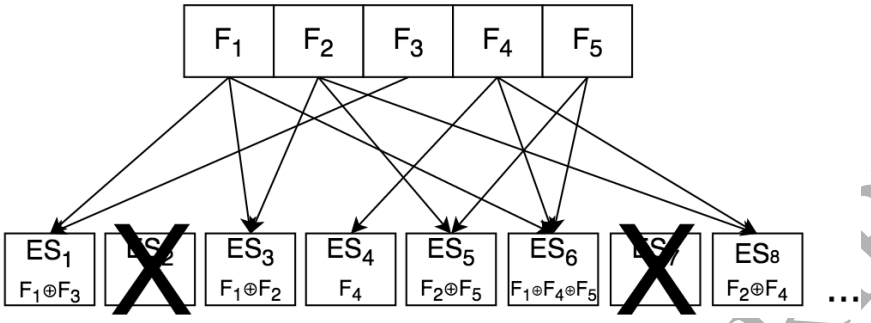

Figure 1: Fountain Coding Example

in a layered, receiver-driven fashion, where a client reacts to network congestion by registering to and un-registering from different layers (multicast groups).

Multi-Source (Many-to-One, Many-to-Many) – Regardless of the number of receivers, multiple senders can contribute to the production of encoding symbols. As long as all sources of the data produce different encoding symbols, then all symbols are of "equal value" to the decoding of the data at the receiver's side. This can be easily achieved by randomising the seeds that are used to produce the *degree* of each symbol (see below) and requires *no coordination* among senders and receivers.

Multi-Path and Multi-Home – Symbols may follow different paths in the network and be received through different network interfaces without affecting decoding efficiency. The receiver only requires a specific number of encoding symbols to decode the original data.

In the following, we overview the encoding and decoding operations of the Luby Transform codes [18] through the example illustrated in Fig. 1. A piece of content is first fragmented into a set of k equal size input data packets (*i.e.,* source symbols), $\mathcal{F} = \{F_1, F_2, \ldots, F_k\}$ where $k = |\mathcal{F}|$. From the set $\mathcal{F}$, a (very) large number of encoding symbols can be created ($\mathcal{ES} = \{ES_1, ES_2, \ldots, ES_N\}$) for transmission. Symbols have the same size as the original content fragments. They can be constructed on the fly or pre-constructed and cached prior to any request for the content. In total, the number of encoding symbols required to decode the requested content is slightly larger than the number of its fragments. Receivers can recover the requested item from *any such set of symbols*. Symbol losses and reordering are not important [18].

Encoding. To construct an encoding symbol, a sender first randomly generates a *seed*, which is associated to this symbol and seeds a pseudo-random generator, which underlies the degree and neighbour generators. After seeding the generator, the sender calculates the symbol's *degree*, d, which is sampled from a *degree distribution* $s(d)$ (*e.g.,* the *Robust Soliton* distribution in [18]). The degree specifies the number of content fragments that will be encoded (XORed) in

each symbol. An edge connects each encoding symbol with one of its neighbours in a bipartite graph as shown in Fig. 1. The sender then selects uniformly at random *degree* fragments as *neighbours* of the encoding symbol. The neighbours are XORed together to form an encoding symbol. For example, ES_1 has $d = 2$ and is the result of XORing fragment F_1 with F_3 (*i.e.,* $ES_1 = F_1 \oplus F_3$). The properties of $s(d)$ are crucial for efficient encoding and decoding as well as for minimizing the network overhead [18][19] from having to send a slightly larger amount of encoding symbols compared to the number of original fragments.

Decoding. In its simplest form, decoding is based on belief propagation [18]. A symbol with $d = 1$ represents a fragment of original data. A receiver utilizes such a symbol to start the decoding process (*e.g.,* ES_4 in Fig. 1) by XORing it with previously received symbols of higher degrees that have this fragment as a neighbour. This may trigger further decoding of other (or even all) previously un-decoded symbols.

Let us assume that a receiver receives the encoding symbols in the following order:

$$ES_1,\ ES_6,\ ES_3,\ ES_4,\ ES_5,\ ES_8.$$

We further assume lossy transmission whereby ES_2 and ES_7 are erased (*e.g.,* corrupted and dropped). We note that, in our example, the encoding symbols arrived "out-of-order" (quoted since there is no concept of ordering in fountain coding) but as mentioned, this has no effect on the decoding process. With each received symbol, its associated seed is also sent. The receiver can then recalculate its degree and neighbours by seeding its own pseudo-random generator with the symbol's seed (all devices run identical pseudo-random generators; *e.g.,* Mersenne Twister). The first symbol received is ES_1 which encodes (XORed) two original fragments, F_1 and F_3. The receiver, at this stage, is unable to decode it and thus stores it locally. It also stores symbols ES_6 and ES_3 for the same reason. The next symbol received, ES_4, is the original fragment F_4. It is XORed with all previously stored symbols that contain fragment F_4 (in this case, only with symbol ES_6). This action effectively removes the edge between nodes F_4 and ES_6 in the bipartite graph of Fig. 1. ES_6 now contains fragments F_1 and F_5 whereby fragment F_4 has been *XORed out* of symbol ES_6. The next received symbol, ES_5, with $d = 2$, contains fragments F_2 and F_5, none of which is yet decoded. Therefore it is stored until it can be decoded. The last symbol, ES_8, encodes fragments F_2 and F_4. F_4 has been previously decoded and it can be used to decode F_2 by XORing it out of ES_8 (*i.e.,* removing the edge between nodes F_4 and ES_8 in Fig. 1). In turn, this enables a chain decoding. The newly decoded fragment F_2 can be used to decode F_1 and F_5 from the previously received symbols ES_3 and ES_5 respectively. Finally, F_3 can be decoded from ES_1 using F_1. At this point, the original content is fully decoded.

Degree Distributions, Computational and Network Overhead. In the description above, we assumed the usage of LT Codes for encoding and decoding content. In LT codes, the degree of each encoding symbol is calculated based on the Robust Soliton Distribution (RSD) [18]. RSD is characterised by the parameters c and δ. δ is the probability that decoding will fail after $N =$

$(1+\epsilon)k$ encoding symbols have been received, where k is the number of source symbols and ϵ is the overhead[2]. c is a positive constant that heavily affects the performance of LT codes, in terms of network and computational overhead. Moreover, both δ and c affect the average degree of the produced encoding symbols, which in turn affects the computational overhead for encoding and decoding[3]. A thorough analysis on how these parameters affect the performance of LT codes can be found in [44] (in the Evaluation section we use the values $c = 0.2$ and $\delta = 0.3$ that are inline with the analysis of [44]). Note that the erasure probability of the channel is not related to the decoding failure probability. In other words, the receiver will successfully decode the received symbols with probability δ after receiving $N = (1+\epsilon)k$, regardless of the number of encoding symbols that were discarded during transmission; the receiver just needs to receive N encoding symbols. Fountain codes, such as R10 [45] and RaptorQ [36], support failure probabilities of around 10^{-6} with an overhead of only a few symbols, regardless of the number of source symbols. This means that for large pieces of content (*e.g.,* 56,403 source symbols for RaptorQ codes), the overhead is essentially negligible for practical applications. Equally importantly, encoding and decoding performance is linear to the number of source symbols, k. Note that we chose LT codes only for presentation simplicity and there is nothing that would prevent us from using RaptorQ codes in our ICN-based approach.

3. Information-centric Content Dissemination with Fountain Codes

In the following, we describe the targeted communication environment and scenarios and present our approach as a set of core components, focussing on the integration of fountain coding within the NDN architecture.

3.1. Communication Environment

Our work focuses on scenarios where a large concentration of mobile users is observed *e.g.,* during a sports event, concert, or demonstration. Although users are mobile, the nature of the considered events results in limited mobility. Mobile devices are assumed to be equipped with both cellular and Wi-Fi network interfaces, enabling Internet access through a cellular base station (BS) or Wi-Fi access point (AP), respectively. They are also enabled for opportunistic, infrastructure-less communication with other devices through Wi-Fi Direct and/or Bluetooth. Internet access is assumed to be a scarce resource either because the large concentration of users stresses the infrastructure or due to the infrastructure being damaged/malfunctioning (*e.g.,* after a natural disaster) or because it is costly (*e.g.,* group of tourists with expensive roaming plans). We

[2]Note that LT codes are asymptotically optimal; i.e. $\epsilon \to 0$ as $k \to \infty$. However, in practical applications ϵ can be as low as 5% [44].

[3]A larger average symbol degree means that more source symbols are encoded in each encoding symbol and therefore more XOR operations are required for each encoding symbol.

also assume that only a limited amount of devices can have access to the Internet at the same time; this set of devices changes over time and therefore there is no single point in the opportunistic network where data can be downloaded from. Instead, many (or all) devices undertake that role through time. Depending on the application scenario the number of devices that have access to the Internet at any time and the duration that they can stay connected can vary. For example, in a very crowded network environment the bottleneck is the network provider that could potentially be unable to serve all devices. In such a scenario all devices try to connect to the Internet (*e.g.,* through their cellular network interface) but only a few get access at specific times (based on the ISP's capacity). In a natural disaster scenario, access to the Internet can be very difficult and only in specific regions (*e.g.,* where a network cell is still operating). Finally, there can be cases where Internet access may be constantly available but, because of privacy concerns (*e.g.,* in public demonstrations) or access costs (*e.g.,* when roaming charges apply), users actively disable it and exchange information in a peer-to-peer fashion (*e.g.,* by chatting to other users through the opportunistically created network in public demonstration scenario).

Users are considered to be interested in information related to the attended event (*e.g.,* other football match scores, song lyrics, news updates) or the general context (*e.g.,* after a natural disaster), although this is not a strict requirement. The shared interest in specific content motivates the sharing of network resources and the collaboration in the form of opportunistic networking, in an effort to address the scarcity of bandwidth. However, energy constraints of mobile devices, along with intermittent connectivity between them, even under the assumption of limited mobility, and the error-prone nature of the wireless medium present significant challenges in sharing content.

3.2. Core Architectural Components

In this work, opportunistic collaboration between users takes place in the following forms:

Seeding – Mobile devices (called *edge* devices as they reside at the edge of the network being associated with an AP/BS) that have access to the Internet download content and provide it to other devices, which have expressed interest in it. Multiple edge devices may be downloading the same content from one or more requesting devices to speed up delivery time and ensure that the content will be successfully delivered.

Forwarding – Devices forward content to other devices. In dense environments, this may result in the formation of multicast trees where a single device forwards content to multiple receivers or other devices that merely forward content. In multi-hop opportunistic networks, multiple paths can be used in order to fully utilize resources among connected devices.

Caching – Devices cache content as they forward it; they can later serve it to other interested devices.

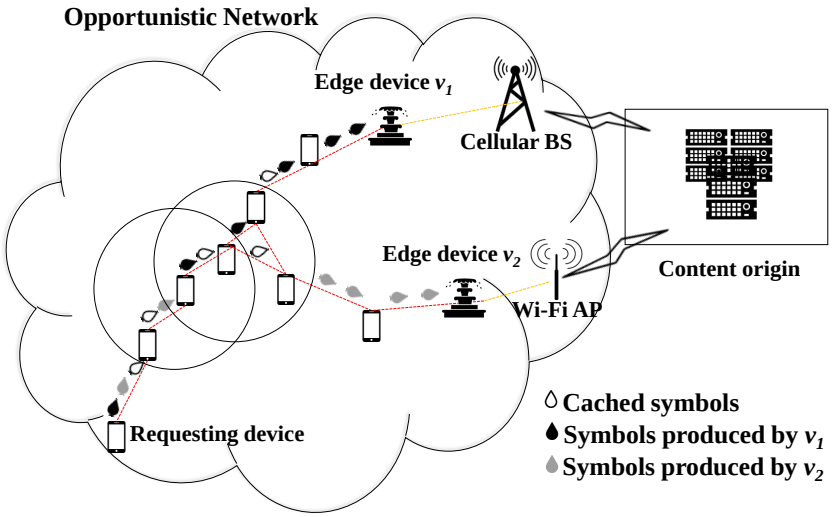

Figure 2: Sending symbols from multiple devices through multiple paths

The integration of fountain coding in the way content is disseminated in the NDN architecture ensures that the required collaboration is efficiently implemented. The core architectural components of our approach are as follows:

Naming – Content is named following NDN's hierarchical scheme [17] *e.g.,* `/a/b/c`. In the context of our work, naming granularity for PIs is at the content level; *i.e.,* no specific chunks are identified in each PI, which requests fountain coded symbols for some content. As explained below, we further extend PI with a *Retrieved Information Base (RIB)* name component, which is used to minimize the dissemination of duplicate encoding symbols. On the returning path, data packets carry the same name augmented with the value of the seed used to calculate the degree and neighbours' set for the encapsulated encoding symbol (*e.g.,* `/a/b/c/seed-x`). Receivers however cannot know (and do not care about) the origin of each symbol *i.e.,* the appended seed is not used to identify a host. `/a/b/c/seed-x` uniquely identifies an encoding symbol for content with name `/a/b/c`.

Forwarding Persistent Interests – When a user requests access to some content, a PI is created and forwarded to the devices it is currently connected with. Since connectivity and content availability are subject to constant change, in our design we do not consider the population of FIB tables in the NDN-enabled devices (as in standard NDN routers); requests must be disseminated as fast as possible to all available devices to ensure that the content is successfully transferred. PIs are propagated throughout the network through limited flooding, *e.g.,* as in [21]. The scope of flooding presents a trade-off with respect to the incurred signalling overhead; its effectiveness is also related to the availability of the content *i.e.,*

the more the seeding edge users, the smaller the required flooding scope to discover them. As we show in Section 4, in most cases, a small scope value is enough to discover the content; a finding in agreement to [21] [46]. When content cannot be found with the default scope (*i.e.,* after an associated timer has expired), the scope is gradually increased. Interests are soft state in the mobile devices. They expire after a timeout is triggered. A device also erases a PI when the number of forwarded encoding symbols matching the PI reaches a specific threshold. Both the timeout and threshold can be dynamically set based on the size of the requested content[4].

Forwarding Data (Encoding Symbols) – In our approach, edge devices create and forward encoding symbols throughout the lifetime of a PI (*i.e.,* until it expires). Caches also forward (but cannot create new) symbols. As shown in Fig. 2, each edge device streams encoding symbols, which follow the reverse PI path (edge devices are shown as digital fountains and encoding symbols are shown as drops of different colours based on their origin). Each device generates a different stream, which contains statistically unique symbols (see Section 2.2). Edge devices may decide to download the content through their Internet connection and start sending symbols in response to a PI or just ignore the request and let the Interest packet expire (*e.g.,* when their battery levels are low)[5]. The level of mobility may drastically vary for different opportunistic network scenarios. In very dynamic scenarios, reverse paths only refer to specific wireless interfaces from which a device can broadcast symbols. Controlling the flow at which devices (edge and in-network caches) send encoding symbols can be done through approaches similar to layered multicasting [47] where PIs for the same content can be used to dynamically vary the rate at which symbols are sent[6]. Designing and evaluating such a layered approach for controlling symbols' flows in the network is part of our future work. Finally, it is worth highlighting that our approach does not require nor assume global adoption of NDN. Instead, edge devices can download content from the Internet through their co-existing TCP/IP stack, similar to [48].

Caching – As a PI propagates through the network, it may encounter devices that currently have one or more encoding symbols for the requested content cached. Given that PIs represent requests for *any* encoding symbol of the requested content item, a cache hit results in the delivery of all relevant

[4]Note that the size of the content must be sent along with encoding symbols in order for receivers to decode the content.

[5]Providing appropriate incentives for users to participate in opportunistic networks is a separate research area; *e.g.,* in [2] trust arises from the sense of camaraderie among fans of the same team.

[6]Note that this can be easily supported using fountain coding where all symbols contribute to the decoding of the original content.

cached symbols. The nature of the communication is such that an encoding symbol is cached at more than one devices along a path. However, the cache-everything-everywhere approach of NDN [17] has been shown to be suboptimal, *e.g.,* [31][32]. For this reason, we adopt opportunistic caching where each user caches a passing by symbol with some probability p, regardless of whether the symbol is cached elsewhere along the path or not. In our approach, multiple devices, including caches, may be sending symbols to one or more receivers at the same time. Given the probabilistic nature of caching, this means that multiple caches may store the same symbol and try to replay it at a later time. Sending identical symbols is a waste of network resources, therefore their existence should be eliminated or, at least, minimized.

We ensure that a receiver will never receive the same symbol from multiple caches by employing a name-based mechanism we call *Retrieved Information Base (RIB)* [22]. RIB is a name component that is appended at the end of the name included in each PI. This new name component represents the seeds of all encoding symbols that are cached by routers (at the time the Interest was processed by each device) on the path that has been followed by the PI so far. Essentially, with the RIB component, each router can know which of its currently cached symbols have not been already replayed to the receiver(s) by other routers in the path. RIB is updated at each hop along the path followed by the Interest. A client initially sends an Interest with an empty RIB. Each router along a path towards a content origin includes in the RIB the seeds of the matching cached encoding symbols that will be sent back to the client in response to the received Interest. Finally, the content origin, upon receiving an Interest, sends to the client symbols that are not included in the RIB until either the PI expires or the user notifies the server that the content item has been retrieved (and fully decoded).

A realization of the RIB could be using *Bloom filters* [49]. A simple Bloom filter-based approach would require each device in the network to support a fixed set of hash functions for producing Bloom filters. Every time a device receives an Interest packet, for each locally cached encoding symbol, it uses the hash functions to calculate the bits of the filter that will be set for the respective seed values. Note that the Bloom filter is of fixed size and does not grow with the number of hops nor the number of cached encoding symbols. In response to an Interest packet, a device will only forward encoding symbols that have not been previously forwarded to the requesting device by testing whether each one of its cached symbols has been previously added to the Bloom filter (by other devices downstream to the requesting client device). The bits that are used to test the membership in the filter are calculated based on the fixed set of hash functions. Given that Bloom filters are of fixed size, their usage may yield *false positives*, which in our case it means that a router might assume that cached packets have already been transmitted to the requesting user. Note that false positives only result in some extra (but not redundant) network traffic, since more encoding symbols should be transmitted from another device further along the path or

from the content origin itself (because the device erroneously assumed that a specific cached encoding symbol had been already sent to the requesting device from some other device that previously processed the PI). The probability of a false positive depends on the number of bits used and the number of hash functions [50]. For example, using four bits per seed we can create filters for ten thousand encoded packets using less than a five KB packet and yielding an accuracy in the area of 85% (less than 15% of false positives). Higher accuracy can be achieved using more bits and larger packets or compressed Bloom filters.

4. Evaluation

In this section, we evaluate the performance of the proposed data dissemination approach and compare it to NDN architecture.

4.1. Evaluation Setup and Metrics

We implemented the proposed approach in a discrete event simulator. The simulator relies on a set of parameters which can be tuned to control the behaviour of the system. We assume that $V = |\mathcal{V}|$ users holding a mobile device each are randomly deployed in a square area of $D = 150 \times 150$ distance units. Each user $v \in \mathcal{V}$ can communicate with a subset of other users that are within a range R. A range $R = \Theta\left(\sqrt{\frac{\log V}{V}}\right) \cdot D \approx 22$ distance units, is adopted to make the whole network fully connected (*i.e.,* for the default number of users parameter), which has already been verified in [51]. Also the formed default network topology has a diameter D' equal to 20 hops.

Each device is equipped with a cache of same capacity, C (*i.e.,* each user contributes the same amount of network storage), defined as a percentage of the entire content catalogue size, M. Request generation at each device follows a Poisson process and the corresponding λ parameter of the exponential distribution is equal to $\lambda = 0.01$. The request rate, r_v^m, of user v for item m is given by $r_v^m = \lambda \cdot \frac{1/j^z}{\sum_{i=1}^{M} 1/i^z}$ (assuming that the Zipf exponent of the item's popularity is z and the item is ranked j-th out of the M information items within the Zipf distribution). Generally, the popularity of each item may vary between users. In our model, we assume that the request popularity exponent has the same value z at each user, but the ranking/order of the items within the Zipf distribution is different. It is well known that the user demand is characterized by a) the total volume of requests for each item in the network, which defines the global content popularity (GCP), and b) the number of locations where each content is requested, defining the geographic distribution of the interests (GDI) [52] and [53]. Here, despite the fact that we assume a very specific and small area where users are deployed we assume the same model (different ranking/order of the items within the Zipf distribution at each user) to examine the performance of the proposed approach in a more generic scenario, since homogeneous popularity of items among the users would definitely benefit the overall system performance.

Table 1: Default evaluation parameter values

Parameter	Value
Number of users/devices (V)	100
Number of content items (M)	10^4
Cache size at each device (C)	$0.5\%M = 50$
Number of edge devices (S)	5
Caching probability at each device (p)	0.5
Popularity Zipf exponent (z)	0.7
Fragments of each item (k)	1000
Network diameter (D' in hops)	20

In order to satisfy the scalability constraint of our simulator, while preserving a good approximation of a realistic content catalogue, we consider a list of $M = |\mathcal{M}| = 10^4$ content items. Additionally, without loss of generality, we assume that each item is fragmented into $k = 10^3$ equally sized fragments, *i.e.,* source symbols. Assuming that each fragment is 1500 bytes long (*i.e.,* typical MTU size), the size of each item $m \in \mathcal{M}$ is equal to 1.5 MB. Although 10^4 items may not seem representative of the current Internet content space, here we focus on overcrowded areas (*e.g.,* football stadiums, music festivals) [2] or disaster scenarios [54] where the network infrastructure is heavily stressed and the available Internet access bandwidth through Wi-Fi APs or cellular BSs gets swallowed up very quickly. In such cases, it appears that the likely desired data services (*e.g.,* updates from other music scenes, travel information, evacuation plans) are limited [2]. It is also important to highlight that even if the size of some content is very large, fountain coding approaches require fragmenting it into smaller encoding blocks (*e.g.,* blocks of 56,403 source symbols each [36])[7]. This is also a realistic assumption as the fragmentation of items into equally sized chunks is a requirement of many replication mechanisms, *e.g.,* [55], [56], and is also present in various content distribution systems such as BitTorrent and modern streaming technologies (*e.g.,* Apple HLS, MPEG DASH) that usually operate on fixed duration segments. Finally, we assume that at each time instance, each content item can be served by S different edge devices. In order to model the dynamics of an opportunistic network, which can be the result of mobility and battery preservation strategies, we assume that the set of users serving a particular item changes randomly every 25 seconds. This further validates our design choice for the usage of PIs since in such a volatile environment

[7]Note that for LT codes, the overhead depends on the total size of the content and therefore fragmenting the content to smaller encoding blocks does not affect the performance of encoding and decoding. For RaptorQ codes the overhead is extremely small compared to the maximum number of source symbols (a few encoding symbols for a decoding failure probability of 10^{-6}) therefore fragmenting the content does not result in any significant extra overhead.

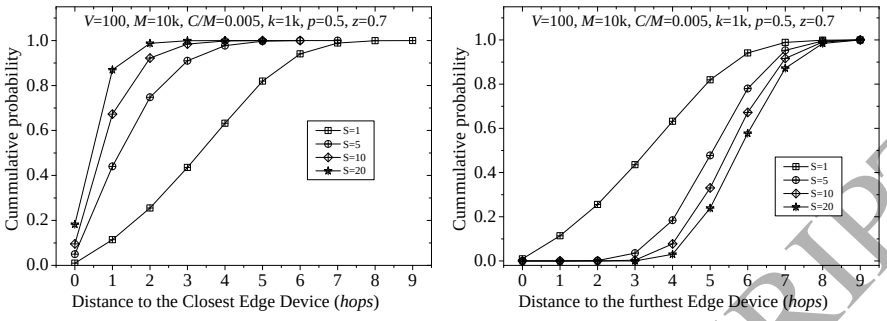

Figure 3: The distance to edge devices (in hops).

FIB entries quickly become obsolete.

We have simulated four different dissemination schemes. Specifically, we evaluated schemes where the proposed dissemination approach (*FC*) is used with and without the RIB name component (denoted as *FC-RIB* and *FC-noRIB* accordingly), as well as the standard NDN dissemination approach with and without the RIB component (denoted as *NDN-RIB* and *NDN-noRIB* accordingly). Our evaluation is based on the following metrics:

- *Cache Hit Ratio*: The ratio of the retrieved Data packets (encoding symbols or plain packets depending on the dissemination scheme) found cached in the network (and not fetched by edge devices) and used for the decoding of an item in the FC schemes and for the retrieval of an item in the standard NDN schemes.

- *Transmission Overhead* (in number of *items*): The average number of extra items that a requesting user receives from the network in order to retrieve/decode the requested content. This is the amount of content that the RIB component (wherever used) and the duplicate dropping mechanisms of the NDN architecture did not manage to save due to the multipath and multi-source forwarding of the Interests.

- *Link Stress* (in *Mbps*): We present both the average link stress as well as the maximum link stress. The *Average Link Stress* is the mean load of traffic that traverses each link of the network. The *Maximum Link Stress* is the maximum load of traffic that traverses the most constrained/congested link of the network. The link stress metrics are indicative of the load balancing capabilities of each examined scheme.

4.2. Performance Evaluation

In the following, for each dissemination scheme, we evaluate *i*) the influence of the number of users/devices V participating in the opportunistic network,

ii) the number of edge devices S serving each information item, *iii*) the storage/cache capacity C available at each participating device, *iv*) the probability p at which each en-route device caches passing by items, *v*) the value of the Zipf distribution parameter z of the content popularity, *vi*) the scope value of the scoped flooding disseminations of the PIs, and finally *vii*) the transmission link failure probability. For the first six experiments we assume that no link failures occur nor any packets are lost in order to examine the impact of the examined parameters. With the last experiment we investigate the behaviour of the proposed approach for various transmission failure probabilities (per wireless link).

For the rest of this section, unless otherwise stated, we consider the reference values presented on Table 1 for the different system parameters. For these baseline values, Fig. 3 shows the cumulative distribution of the distance for finding the closest and all (*i.e.,* distance to the furthest) edge devices for four different values of the edge devices per item accordingly. From the figures, we observe that in our baseline setup almost 96% of the issued Interests can reach at least one edge device within 3-4 hops away from the requesting user, a finding in agreement to recent empirical observations [46]. In the same setup, almost 80% of the issued Interests can reach all the edge devices ($S = 5$ in the corresponding setup) in less than 6 hops away from the requesting user. Based on the above observation, we consider a default scope value of 5 hops, for the Interest packet propagation (the baseline opportunistic network has a diameter equal to 20 hops).

4.2.1. Impact of the number of users/devices

We investigate the impact of the number of users/devices V that participate in the formation of the opportunistic network in the performance of the examined schemes. The results are shown in Fig. 4. The number of users/devices in the network varies from 10 to 250.

It is obvious that the different number of devices, as well as the way the network is formed result in different network topologies. Fig. 4 shows a representative result, where we observe that a linear increase in the number of devices results in almost an exponential increase in the cache hit ratio performance of the fountain coding schemes, whereas such an increase has limited impact on the performance of the NDN schemes. An increase in the number of devices increases the number of available paths towards the edge devices. This, along with the versatility of the different encoding symbols available at each cache allows the fountain coding-based schemes to retrieve more unique symbols from the network and decode more items without contacting the content origin. For instance, a number of devices larger than 150, which is very moderate for a festival or a football match or even a train platform, allows the retrieval of more than 50% of the requested items locally. On the other hand, the NDN schemes keep replicating the same packets along the different paths, minimizing the potential benefits of multi-path and multi-source Interest forwarding, something that is evident from the overhead plot, where the NDN schemes produce up to five times more overhead compared to the FC ones. Finally, the usage of the

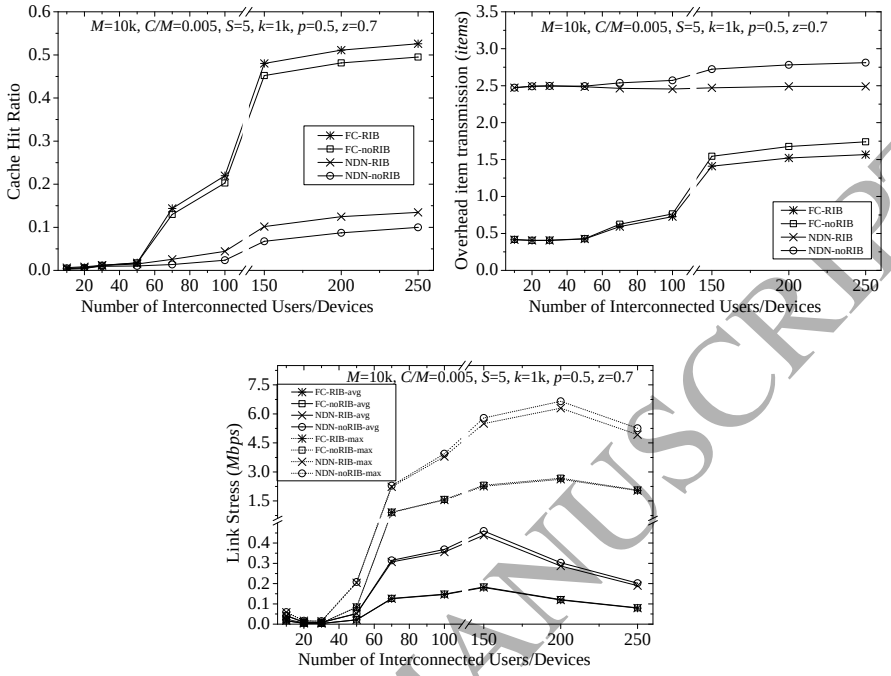

Figure 4: The impact of the number of devices participating in the opportunistic network in the performance of the examined schemes.

RIB might not have significant impact for small networks, but can save up to 10% of overhead transmissions for relatively large opportunistic networks.

Regarding the Link stress metric, we initially observe an increase for both the maximum and the average load of the links with the increase of the network size, since a larger and denser network implies more and longer paths to exploit for cached content. In other words, the 5 hops limitation in the forwarding of the interests cannot be spent for small networks leading to less responses as also shown from the other plots. However, for larger topologies interests can reach more edge devices and intermediate users something that increase the load of the links as well. When the topology is relatively large the interests spend their "quotas" faster and they don't reach the whole network, something that decreases slightly the average link stress (*i.e.,* more available links and smaller the radius of each interest).

4.2.2. Impact of the number of edge devices per item

We investigate the impact of the number of different edge devices S per item in the performance of the examined schemes. The results are shown in Fig. 5. The number of different edge devices varies from 1 to 30. Note that as with all other experiments, edge devices change over the course of each experiment (the set of edge devices for each item change randomly every 25 seconds).

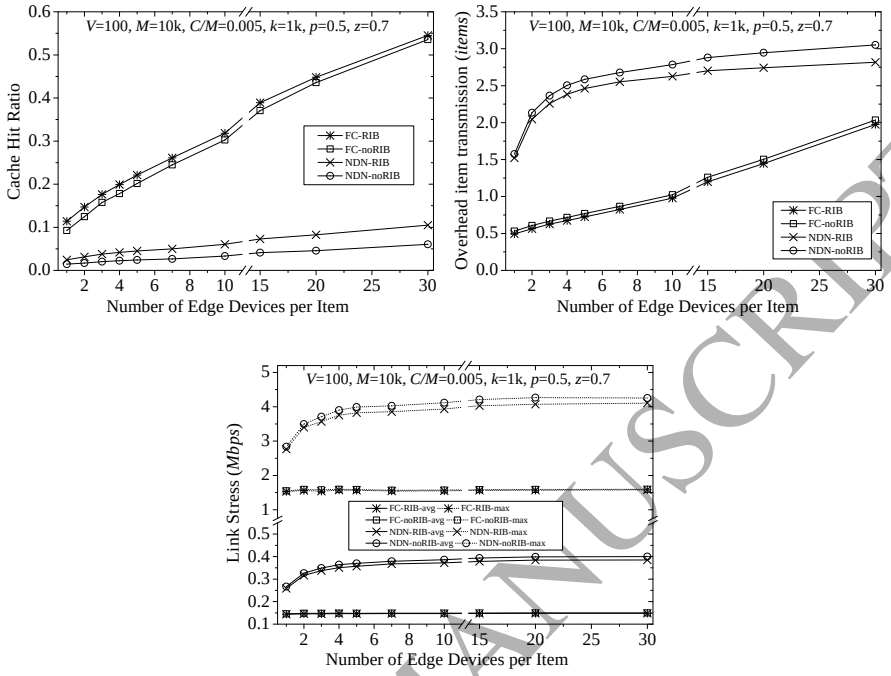

Figure 5: The impact of the number of edge devices for each information item in the performance of the examined schemes.

Generally, we observe that the usage of fountain coding significantly outperforms the standard NDN schemes even when there is only one edge device available each time, despite the overhead (in terms of required encoding symbols) that is associated with the decoding of the original data. As explained above, this out-performance is mainly due to the multipath forwarding of the Interest packets and the caching of the same packets along a path of the NDN mechanism. Note that in a given network topology (the basic topology used in the Evaluation section) there exist three to twelve different paths (overlapped or not) between any two nodes of the network. The fountain coding schemes are not affected by the number of edge devices regarding the link stress metrics and they perform up to 4.5 times better regarding the cache hit ratio compared to the NDN schemes. Also, the usage of the RIB mechanism can save up to 8% of the overhead traffic comparing similar schemes (even without being combined with fountain coding) and is useful for the minimization of the overhead even in the NDN architecture.

4.2.3. Impact of the cache capacity of each device

We investigate the impact of the cache capacity C of each device, expressed as the fraction of the items' population in the performance of the examined schemes. The results are shown in Fig. 6. The cache capacity varies from 0.1% to 15% of the content catalogue M.

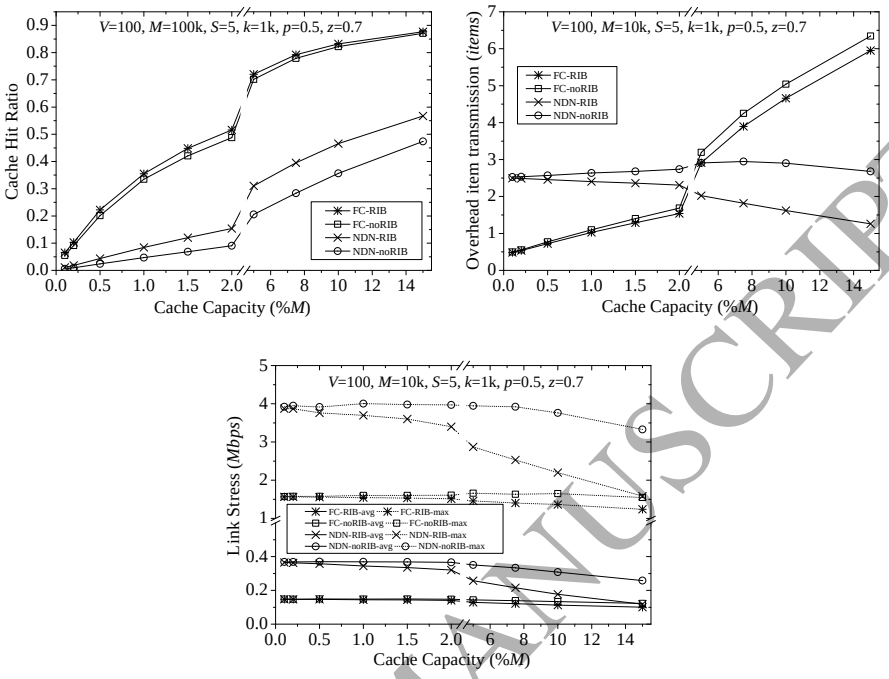

Figure 6: The impact of the cache capacity of each device participating in the opportunistic network in the performance of the examined schemes.

Again, the FC schemes perform significantly better compared to the rest, even for very small cache capacities. For larger cache capacity sizes ($C/M > 0.05$), we observe that the cache hit ratio is twice as high as in the NDN scheme. For smaller capacities, the performance of the FC schemes is even more impressive, performing for instance 4.2 times better when the cache of each device can accommodate 0.5% of the content catalogue. This means that in real-world scenarios (*i.e.*, devices with relatively small cache capacity), more than 25% of the requested items are found in the network, which, in turn, means that combining fountain coding with our proposed RIB approach could be very useful for future ICN applications in opportunistic device-to-device networks. Also, as in the previous experiment, the usage of the RIB component can save up to 10% of the duplicate transmissions (16% in the non-encoding schemes) even for very small cache capacities.

4.2.4. Impact of the caching probability

We investigate the impact of the caching probability p in the performance of the examined schemes. The results are shown in Fig. 7. The caching probability varies from 0.01 to 1.

From Fig. 7 we observe that the caching probability has almost no impact on the overhead and link stress metrics; and as in every other conducted experiment, the FC schemes perform significantly better than the NDN schemes.

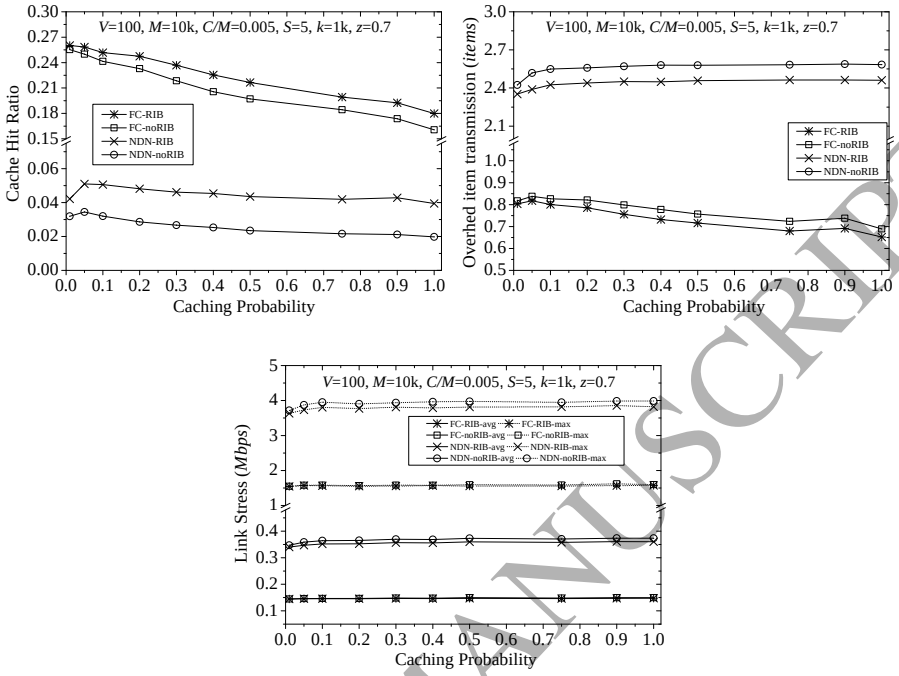

Figure 7: The impact of the caching probability of each packet at each passing by device in the performance of the examined schemes.

Also, the caching probability has minimum impact to the cache hit ratio of the NDN schemes. This is mainly due to the fact that devices along the path tend to replicate the same packets even for very small values of p. On the other hand, when fountain coding is used the almost infinite number of different encoding symbols allows devices to cache different content and the actual value of the caching probability severely impacts the performance of the system. For example, we observe a 30% decrease in the cache hit ratio when the caching probability changes from 0.01 to 1. Finally, as already shown in [31][32], from Fig. 7, we also observe that the cache-everything-everywhere approach (*i.e.,* $p = 1$) performs worse, regarding cache hit ratio, than any other caching mechanism for every examined scheme.

4.2.5. Impact of the content popularity Zipf exponent

We investigate the impact of the content popularity distribution, which is driven by the Zipf parameter z in the performance of the examined schemes. The results are shown in Fig. 8. Parameter z varies from 0 to 2.

The distribution of the content popularity becomes more uniform as z decreases. In the cases where z is in the order of $0 - 0.5$, FC schemes perform almost 3.5 times better regarding cache hit ratio. However, various studies, *e.g.,* [57] [58], suggest that the Zipf parameter z for web traffic lies in the range of $0.64 - 0.84$, and is larger than 1.2 for online video applications. In those real

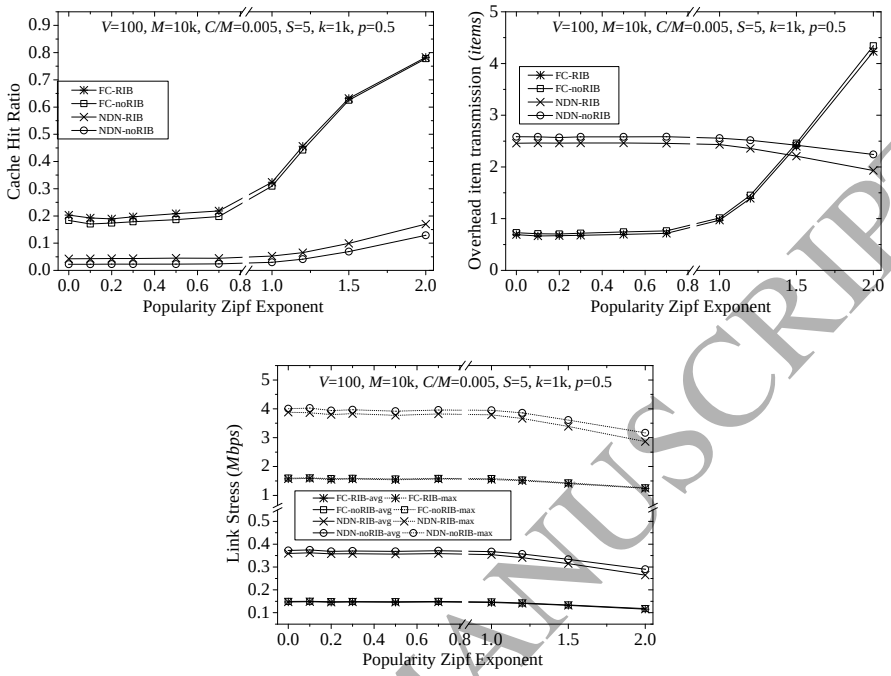

Figure 8: The impact of the content popularity distribution in the performance of the examined schemes.

world environments, where the distribution of content popularity is less uniform (especially in video delivery systems), the proposed FC schemes perform up to 6 times better that the NDN schemes regarding cache hit ratio. The link stress metrics, on the other hand, are not affected by the content popularity dynamics.

4.2.6. Impact of the Interest Scope value

We investigate the impact of the interest scope value in the performance of the examined schemes. The results are shown in Fig. 9 for the scope value varying from 1 to 20.

In Fig. 9 we also depict the satisfied interest ratio, which is the ratio of the issued interests that where finally satisfied by either cached content or by content fetched through the edge nodes. This metric is inline with Fig. 3 for $S = 5$, where we observe that with only 4-5 hops radius all interests are satisfied, which means that at least one edge node was found. This is also obvious in the rest plots, where we observe an exponential increase in the cache hit ratio for the fountain coding schemes with the increase of the interest scope, whereas this increase follows a linear pattern for the NDN schemes. On the other hand, we have an inverse pattern in the overhead metric, where the NDN schemes result in an exponential increase in the overhead responses due to the multiple replication of the same items at the intermediate nodes. Regarding link stress metric, we observe that max link stress is not influenced by the scope value,

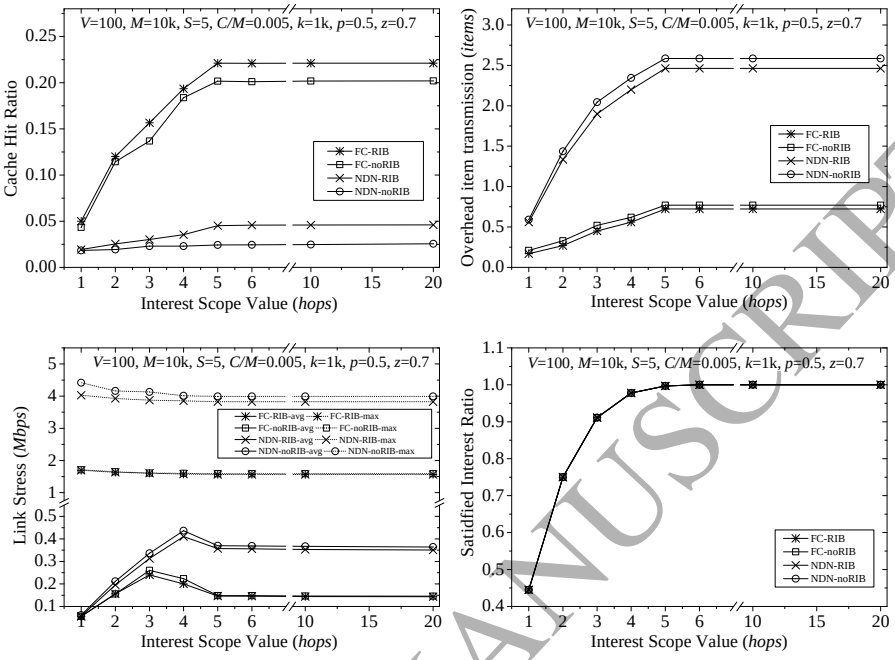

Figure 9: The impact of the Interest scope in the performance of the examined schemes.

since this usually refers to the a link that connects an edge node with the rest of the network and as long as an interest reaches an edge node is typically stable. This is not the case for the average link stress metric though, where we initially observe an increase with the increase of the scope value, since this implies more and longer paths to exploit for cached content, that result in more responses. However, after 5 hops all the source nodes are found and the interests are not propagated further in the network. This is also observed in the rest of the metrics, where any scope value larger than 5 is redundant (*i.e.,* the interests are not forwarded further; all available source nodes have been found/reached).

4.2.7. Impact of the transmission link failure probability

Communications in opportunistic networks are highly susceptible to losses. We investigate next the impact of the transmission link fail probability in the performance of the examined schemes. The results are shown in Fig. 10 for the transmission link fail probability varying from 0 to 0.1.

Despite the advances in the wireless communication area for the avoidance of collisions and dropped packets (*i.e.,* CSMA/CD and RTS/CTS in 802.11 WLANS), wireless networks still suffer from dropped packets due to various reasons (*e.g.,* hidden terminal problem and/or exposed node problem). In the above experiments, we assumed that there are no failures during the transmission of the packets. Here, we examine the performance of the proposed schemes

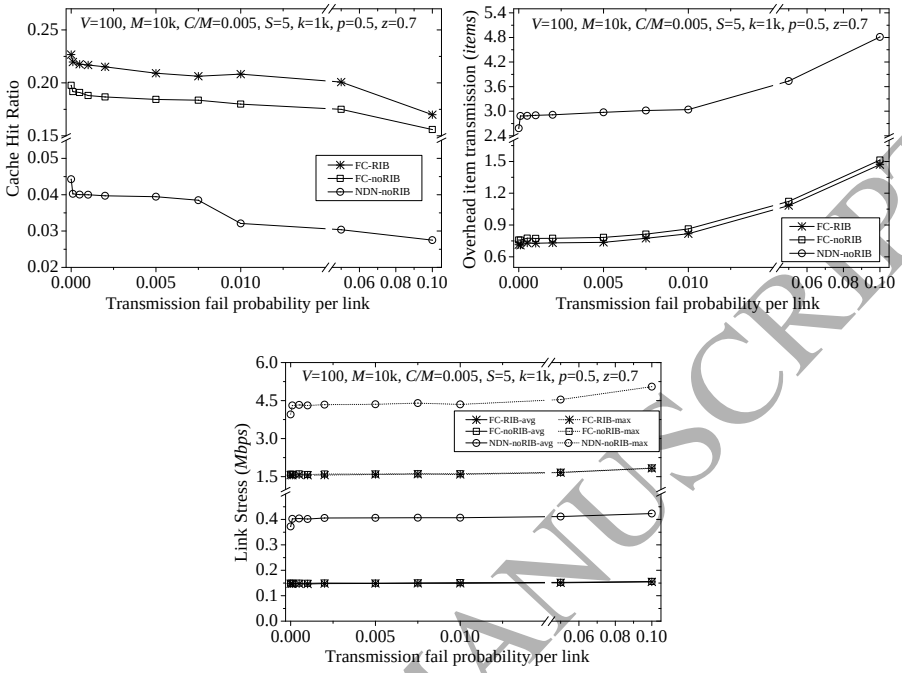

Figure 10: The impact of the transmission link failure probability in the performance of the examined schemes.

when each transmission over a wireless link fails with a given probability. Note that when failures exist, we cannot enable the RIB mechanism for the NDN schemes, since a failed transmission might prevent a content origin to send a packet of an item that is set to be delivered according to a RIB entry (*i.e.,* failed transmission from an intermediate device). However, the usage of fountain coding allows the usage of RIB even in the case of failures, since an edge device dos not necessarily has to transmit specific symbols. For that reason, we don't examine here the NDN-RIB scheme.

From Fig. 10 we observe that the transmission fail probability has a significant impact on the cache hit ratio of every examined scheme. In particular, we observe up to 25% decrease in the cache hit ratio when the fail probability is 0.1 compared to the failure free scenario. Also, the increased transmission failure probability has a similar impact on the overhead transmissions, since after a failure a packet/symbol has to be retransmitted. As in every other conducted experiment, the FC schemes perform significantly better than NDN and the RIB mechanism can still save a non-trivial portion of extra retransmissions, even when the fail probability is considerably high (more than 8% even for fail probabilities larger that 0.01).

5. Conclusions

In this paper we proposed a content-centric approach for disseminating content in opportunistic, device-to-device networks. Opportunistic networks should support crucial functionality, such as in-network caching capabilities, multi-source, multi-path and multicast forwarding along with the requirement for reliable content delivery. To support these, we depart from traditional TCP/IP network approaches and, instead, we combine inherent characteristics of information-centric networking with key properties of fountain codes. We also proposed an enhancement to the NDN Interest forwarding mechanism to minimize duplicate transmissions. The proposed approach follows the content-centric principles of the network and is crucial in the fountain coding-based content dissemination. In addition, we employed scoped flooding and the use of Persistent Interests to fully exploit the benefits of multi-source and multi-path forwarding, which are inherent in opportunistic networks. Our extensive evaluations show that the proposed solution benefits from the ICN, while at the same time the use of fountain codes minimizes redundancy (and the resulting waste of network resources) without requiring any centralized coordination mechanisms. We believe this is a promising approach to maximize the benefits of future opportunistic networks.

References

[1] S. Trifunovic, B. Distl, D. Schatzmann, F. Legendre, WiFi-Opp: Ad-hoc-less Opportunistic Networking, in: ACM CHANTS, 2011, pp. 37–42.

[2] I. Wakeman, S. Naicken, J. Rimmer, D. Chalmers, C. Fisher, The Fans United will Always be Connected: Building a Practical DTN in a Football Stadium, in: International Conference on Ad Hoc Networks, 2013, pp. 162–177.

[3] B. Han, P. Hui, V. A. Kumar, M. V. Marathe, G. Pei, A. Srinivasan, Cellular Traffic Offloading Through Opportunistic Communications: A Case Study, in: ACM CHANTS, 2010, pp. 31–38.

[4] Y. Wang, H. Zhang, C. C. Tan, X. Du, B. Sheng, WiGroup: A Lightweight Cellular-Assisted Device-to-Device Network Formation Framework, in: IEEE Conference on Ubiquitous Intelligence and Computing, 2015, pp. 292–299.

[5] M. Conti, M. Kumar, Opportunities in Opportunistic Computing, IEEE Computer 43 (1) (2010) 42–50.

[6] L. Pelusi, A. Passarella, M. Conti, Opportunistic networking: data forwarding in disconnected mobile ad hoc networks, IEEE Communications Magazine 44 (11) (2006) 134–141.

[7] S. H. Masood, S. A. Raza, M. Coates, Content Distribution Strategies in Opportunistic Networks, CoRR abs/1308.0786.
URL http://arxiv.org/abs/1308.0786

[8] T. E. Amah, M. Kamat, W. Moreira, K. A. Bakar, S. Mandala, M. A. Batista, Towards next-generation routing protocols for pocket switched networks, Journal of Network and Computer Applications 70 (2016) 51 – 88.

[9] M. Grossglauser, D. N. C. Tse, Mobility Increases the Capacity of Ad Hoc Wireless Networks, IEEE/ACM ToN 10 (4) (2002) 477–486.

[10] V. Vukadinović, G. Karlsson, Spectral Efficiency of Mobility-assisted Podcasting in Cellular Networks, in: International Workshop on Mobile Opportunistic Networking, 2010, pp. 51–57.

[11] P. Jassal, K. Yadav, A. Kumar, V. Naik, V. Narwal, A. Singh, Unity: Collaborative downloading content using co-located socially connected peers, in: IEEE Pervasive Computing and Communications Workshops, 2013, pp. 66–71.

[12] H. Luo, R. Ramjee, P. Sinha, L. E. Li, S. Lu, UCAN: A Unified Cellular and Ad-hoc Network Architecture, in: ACM MobiCom, 2003, pp. 353–367.

[13] J. Erman, K. Ramakrishnan, Understanding the Super-sized Traffic of the Super Bowl, in: ACM Internet Measurement Conference, 2013, pp. 353–360.

[14] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, S. Venkataraman, J. Wang, A First Look at Cellular Network Performance During Crowded Events, in: ACM SIGMETRICS, 2013, pp. 17–28.

[15] E. Cho, S. A. Myers, J. Leskovec, Friendship and Mobility: User Movement in Location-based Social Networks, in: ACM Conference on Knowledge Discovery and Data Mining, 2011, pp. 1082–1090.

[16] Y. Tanahashi, J. R. Rowland, S. North, K.-L. Ma, Inferring Human Mobility Patterns from Anonymized Mobile Communication Usage, in: Conference on Advances in Mobile Computing and Multimedia, 2012, pp. 151–160.

[17] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, R. L. Braynard, Networking Named Content, in: ACM CoNEXT, 2009, pp. 1–12.

[18] M. Luby, LT codes, in: IEEE Symposium on Foundations of Computer Science, 2002, pp. 271–280.

[19] A. Shokrollahi, Raptor Codes, IEEE/ACM ToN 14 (SI) (2006) 2551–2567.

[20] C. Tsilopoulos, G. Xylomenos, Supporting Diverse Traffic Types in Information Centric Networks, in: ACM Workshop on ICN, 2011, pp. 13–18.

[21] L. Wang, S. Bayhan, J. Ott, J. Kangasharju, A. Sathiaseelan, J. Crowcroft, Pro-Diluvian: Understanding scoped-flooding for content discovery in information-centric networking, in: International Conference on ICN, 2015, pp. 9–18.

[22] G. Parisis, V. Sourlas, K. V. Katsaros, W. K. Chai, G. Pavlou, Enhancing Multi-source Content Delivery in Content-Centric Networks with Fountain Coding, in: CoNEXT CCDWN Workshop, 2015, pp. 1–6.

[23] Cisco, Cisco Global Cloud Index: Forecast and Methodology, 2015-2020, Tech. rep. (2015).

[24] G. Xylomenos, C. Ververidis, V. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. Katsaros, G. Polyzos, A Survey of Information-Centric Networking Research, IEEE Communications Surveys Tutorials 16 (2) (2014) 1024–1049.

[25] D. Trossen, G. Parisis, Designing and realizing an information-centric internet, IEEE Communications Magazine 50 (7) (2012) 60–67.

[26] G. Parisis, D. Trossen, D. Syrivelis, Implementation and evaluation of an information-centric network, in: IFIP Networking, 2013, pp. 1–9.

[27] C. Anastasiades, T. Braun, V. A. Siris, Information-Centric Networking in Mobile and Opportunistic Networks, 2014, pp. 14–30.

[28] H. M. A. Islam, A. Lukyanenko, S. Tarkoma, A. Ylä-Jääski, Towards Disruption Tolerant ICN, CoRR abs/1510.04436.
URL `http://arxiv.org/abs/1510.04436`

[29] Y.-T. Yu, J. Joy, R. Fan, Y. Lu, M. Gerla, M. Sanadidi, DT-ICAN: A Disruption-Tolerant Information-centric Ad-Hoc Network, in: IEEE MILCOM, 2014, pp. 1021–1026.

[30] J. Su, J. Scott, P. Hui, J. Crowcroft, E. De Lara, C. Diot, A. Goel, M. H. Lim, E. Upton, Haggle: Seamless Networking for Mobile Applications, in: Conference on Ubiquitous Computing, 2007, pp. 391–408.

[31] W. K. Chai, D. He, I. Psaras, G. Pavlou, Cache less for more in information-centric networks (extended version), Computer Communications 36 (7) (2013) 758 – 770.

[32] I. Psaras, W. K. Chai, G. Pavlou, In-Network Cache Management and Resource Allocation for Information-Centric Networks, IEEE Transactions on Parallel and Distributed Systems 25 (11) (2014) 2920–2931.

[33] G. Carofiglio, M. Gallo, L. Muscariello, ICP: Design and evaluation of an Interest control protocol for content-centric networking, in: IEEE INFOCOM Workshops, 2012, pp. 304–309.

[34] J. Chen, M. Arumaithurai, L. Jiao, X. Fu, K. K. Ramakrishnan, Copss: An efficient content oriented publish/subscribe system, in: ACM/IEEE Symposium on Architectures for Networking and Communications Systems, 2011, pp. 99–110.

[35] J. Chen, M. Arumaithurai, X. Fu, K. K. Ramakrishnan, G-copss: A content centric communication infrastructure for gaming applications, in: IEEE LANMAN, 2011, pp. 1–6.

[36] M. Luby, A. Shokrollahi, M. Watson, T. Stockhammer, L. Minder, RFC 6330 - RaptorQ Forward Error Correction Scheme for Object Delivery, Tech. rep., IETF (2011).
URL `http://tools.ietf.org/html/rfc6330`

[37] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, K. Ramchandran, Network coding for distributed storage systems, IEEE Transactions on Information Theory 56 (9) (2010) 4539–4551.

[38] C. Gkantsidis, P. R. Rodriguez, Network coding for large scale content distribution, in: IEEE Conference of the IEEE Computer and Communications Societies., Vol. 4, 2005, pp. 2235–2245.

[39] P. Maymounkov, D. Mazières, Rateless Codes and Big Downloads, in: IPTPS, 2003, pp. 247–255.

[40] A. G. Dimakis, V. Prabhakaran, K. Ramchandran, Ubiquitous access to distributed data in large-scale sensor networks through decentralized erasure codes, in: International Symposium on Information Processing in Sensor Networks, 2005, pp. 111–117.

[41] G. Parisis, D. Trossen, Filling the gaps of unused capacity through a fountain coded dissemination of information, Mobile Computing and Communications Review 18 (1) (2014) 46–54.

[42] M.-J. Montpetit, C. Westphal, D. Trossen, Network Coding Meets Information-centric Networking: An Architectural Case for Information Dispersion Through Native Network Coding, in: ACM NoM Workshop, 2012, pp. 31–36.

[43] J. W. Byers, M. Luby, M. Mitzenmacher, A. Rege, A Digital Fountain Approach to Reliable Distribution of Bulk Data, in: ACM SIGCOMM, 1998, pp. 56–67.

[44] P. Cataldi, M. P. Shatarski, M. Grangetto, E. Magli, Implementation and Performance Evaluation of LT and Raptor Codes for Multimedia Applications, in: Conference on Intelligent Information Hiding and Multimedia, 2006, pp. 263–266.

[45] M. Luby, A. Shokrollahi, M. Watson, T. Stockhammer, RFC 5053 - Raptor Forward Error Correction Scheme for Object Delivery, Tech. rep., IETF (2007).
URL `http://tools.ietf.org/html/rfc5053`

[46] S. Bayhan, E. Hyytiä, J. Kangasharju, J. Ott, Two Hops or More: On Hop-Limited Search in Opportunistic Networks, in: ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2015, pp. 115–124.

[47] S. McCanne, V. Jacobson, M. Vetterli, Receiver-driven Layered Multicast, in: ACM SIGCOMM, 1996, pp. 117–130.

[48] I. Psaras, K. V. Katsaros, L. Saino, G. Pavlou, LIRA: A Location Independent Routing Layer based on Source-Provided Ephemeral Names, CoRR abs/1509.05589.

[49] B. H. Bloom, Space/Time Trade-offs in Hash Coding with Allowable Errors, ACM Communications.

[50] J. W. Byers, J. Considine, M. Mitzenmacher, S. Rost, Informed content delivery across adaptive overlay networks, IEEE/ACM ToN 12 (5) (2004) 767–780.

[51] P. Gupta, P. R. Kumar, The capacity of wireless networks, IEEE Transactions on Information Theory 46 (2) (2000) 388–404.

[52] D. Tuncer, M. Charalambides, R. Landa, G. Pavlou, More Control Over Network Resources: An ISP Caching Perspective, in: IEEE/IFIP Conference on Network and Service Management, 2013, pp. 26–33.

[53] D. Tuncer, V. Sourlas, M. Charalambides, J. Famaey, G. Pavlou, F. De Turck, Scalable cache management for isp-operated content delivery services, IEEE Journal on Selected Areas in Communications.

[54] V. Sourlas, L. Tassiulas, I. Psaras, G. Pavlou, Information resilience through user-assisted caching in disruptive Content-Centric Networks, in: IFIP Networking, 2015, pp. 1–9.

[55] S. Borst, V. Gupta, A. Walid, Distributed caching algorithms for content distribution networks, in: IEEE INFOCOM, 2010, pp. 1–9.

[56] N. Golrezaei, K. Shanmugam, A. Dimakis, A. Molisch, G. Caire, Femtocaching: Wireless video content delivery through distributed caching helpers, in: IEEE INFOCOM, 2012, pp. 1107–1115.

[57] G. Dán, N. Carlsson, Power-law Revisited: Large Scale Measurement Study of P2P Content Popularity, in: IPTPS, 2010.

[58] Y. Sun, S. K. Fayaz, Y. Guo, V. Sekar, Y. Jin, M. A. Kaafar, S. Uhlig, Trace-Driven Analysis of ICN Caching Algorithms on Video-on-Demand Workloads, in: ACM CoNEXT, 2014, pp. 363–376.