Computer Communications 35 (2012) 1–32

Contents lists available at SciVerse ScienceDirect

Computer Communications

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

Review

A survey on content-centric technologies for the current Internet: CDN and P2P solutions ☆

Andrea Passarella *

IIT-CNR, Institute for Informatics and Telematics of the National Research Council, Via G. Moruzzi 1, 56124 Pisa, Italy

ARTICLE INFO

Article history:
Received 31 July 2009
Received in revised form 7 October 2011
Accepted 8 October 2011
Available online 14 October 2011

Keywords:
Content-centric Internet
Content Delivery Networks
P2P
Internet evolution

ABSTRACT

One of the most striking properties of the Internet is its flexibility to accommodate features it was not conceived for. Among the most significant examples, in this survey we consider the transition of the Internet from a reliable fault-tolerant network for host-to-host communication to a content-centric network, i.e. a network mostly devoted to support efficient generation, sharing and access to content. We survey this research area according to a top-down approach. We present a conceptual framework that encompasses the key building blocks required to support content-centric networking in the Internet. Then we describe in detail the two most important types of content-centric Internet technologies, i.e., Content-Delivery Networks (CDNs) and P2P systems. For each of them, we show how they cover the key building blocks. We then identify the functional components of CDN and P2P content management solutions, and discuss the main solutions proposed in the literature for each of them. We consider different types of content (both real time and non real time), and different networking environments (fixed, mobile, ...). Finally, we also discuss the main recent research trends focused on how to design the Future Internet as a native content-centric network.

1. Introduction

The impressive diffusion of the Internet worldwide has generated a significant shift in its usage, with respect to what it was originally conceived for. In the first place, the Internet was mainly designed as a robust, fault-tolerant network to connect hosts for military and scientific applications [1]. Nowadays, one of the main usages of the Internet is *content generation*, *sharing and access* [2,3]. Internet users are not that interested in connecting their computer with a particular host on the other end of the network, but are rather interested in accessing particular pieces of content they need at a given point in time. While accessing resources – rather than connecting to machines – has always been one of the Internet "use cases", there is today a massive increase in the number of users that generate content, and in the interest of the other users in accessing them. Enabling widespread access to content was already one of the goals of the conventional Web. More recently, it became the key focus of a range of applications having impressive popularity on the Internet. For example, User-Generated Content (UGC) services (such as Flickr and YouTube) place content at the very center of the picture, and enable users not only to consume content generated by a small set of well-identified providers, but also to produce and share their own content, thus becoming content *prosumers*. Online Social Networks are also very much oriented towards content generation, sharing and access. The possibility of using the Internet as a content-centric network is one of the many examples of the ability of the original Internet design to support unforeseen evolutions and new networking paradigms.

It is also worth noting that the role of a content-centric Internet usage is likely to become even more important as the share of resource rich mobile devices possibly connected to the Internet steadily increases. Last generation devices such as smartphones and tablets are equipped with a range of components for content generation (e.g., cameras, microphones) that can be exploited by users to generate content anytime and anywhere. The more those devices get connected to the Internet, the more the Internet will be used as a pervasive network to upload, share and consume content. This is already happening with Online Social Network applications for mobile devices, such as those provided by Facebook.

In this paper we provide a survey on the key solutions that have been proposed to address a very challenging problem: How to make the Internet behave as a content-centric network such that users can share and ask for content irrespective where and how the content is stored in the network. The motivation for this survey lies in the current debate on the Future Internet taking place worldwide, also stimulated by substantial research programmes [4,5]. Specifically, there is large consensus on the fact that

☆ This work was partially funded by the European Commission under the FET-AWARE RECOGNITION (257756), FIRE-SCAMPI (258414) and FIRE-EINS Projects.

* Tel.: +39 0503153269.
E-mail address: a.passarella@iit.cnr.it

0140-3664/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2011.10.005

content-centric networking will be one of the key features of the Future Internet [2]. Therefore, providing a clear view on how content-centric networking has been supported so far is both timely and needed.

Two broad classes of solutions have been proposed to this end, i.e. Content Delivery Networks (CDNs) and Peer-to-Peer Networks, which are therefore the main subject of this work. The main focus of this survey is on the CDN and P2P technologies proposed to store, retrieve and manage content, although – for completeness – we also briefly touch on other important aspects such as security, trust and cooperation. We take a rather broad view on the possible networking environments and content types. We cover technologies proposed for fixed and mobile networks, also including recent paradigms such as opportunistic networking [6]. With respect to the type of content, we clearly separate solutions designed for real-time and non-real-time content, as they provide quite different functionality. Note that – particularly in the part devoted to P2P – we present technologies also supporting recent content generation and consumption models, such as UGC. The key goal of the presented solutions is in general to (i) enable all interested users to access all pieces of content they want, and (ii) doing so by using Internet resources as efficiently as possible. Therefore, these are also the key references we use to discuss the different performance of the different solutions, and their improvements on each other.

In the rest of the paper, we first discuss a conceptual framework which encompasses the main building blocks proposed in the literature to provide content-centric networking services in the current Internet (Section 2). We also show how CDN and P2P solutions cover these blocks. Then we discuss in detail the different functional components of CDN (Section 3) and P2P (Section 4) technologies. Both for CDN and P2P technologies, we take a top-down approach, firstly identifying the most important functional elements, and then describing the various proposals available in the literature to implement each of them. We finally (Section 5) provide an outlook on recent proposals to design the Future Internet *from scratch* as a content-centric network.

2. Conceptual framework for content-centric networking in the current Internet

We take a top-down approach to discuss how content-centric networking is currently supported in the Internet. Therefore, we first highlight the main functional blocks that are conceptually needed to provide content centric networking services in the current Internet, and discuss how current technologies implement them. Fig. 1 provides a scheme to guide the following discussion.

The first point to note is that content-centric networking services are provided – in the current Internet – above the transport level, and thus the Internet core architecture is completely agnostic about them. This is coherent with the overall Internet architecture design. The Internet architecture was conceived to include in the core very simple best-effort services to connect particular endpoints identified by an IP address. A key Internet concept is that any further extensions of this basic functionality should not impact on the core, and should be implemented at the application level, on edge devices. As content-centric services are clearly something that was not included in the original Internet architectural design, they have been implemented above the TCP/IP level.

Abstracting from the specific solutions proposed in the literature for content-centric networking in the current Internet, it is possible to identify three main functional blocks: (i) replication and placement, (ii) indexing, search and advertisement, and (iii) transfer of content. They can be seen as functional building blocks that content-centric networking systems should provide to applications. *Replication and placement* solutions deal with issues concerning how largely any piece of content should be replicated, where it should be placed, and how replicas can be made consistent. *Indexing, search and advertisement* solutions mainly pertain to issues related to content search and access. Techniques to index the available pieces of content (and the available replicas), such that search for them can be optimised, also fall in this category. Finally, *transfer* solutions deal with methods to transfer the content from where it is stored to where it is requested. Techniques concerning content access and sharing from wireless and mobile devices, as well as aspects related to trust, security and cooperation are also important for supporting content-centric networking. They can be seen as cross issues that cut across the three main functional blocks identified in Fig. 1.

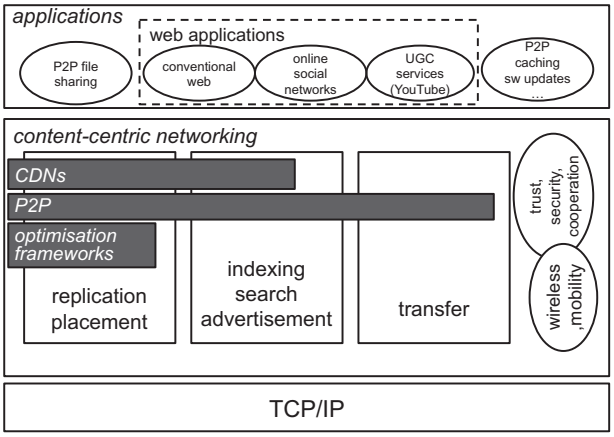

Fig. 1. Internet support to content-centric networking.

The first two horizontal grey bars in the figure map the solution space of Content Delivery Networks (CDN) and P2P solutions onto the three main building blocks. According to the CDN paradigm, a CDN operator places a set of machines at carefully identified locations in the Internet, and sells portions of the machine resources to content providers, which can replicate their content on the CDN. The CDN operator takes care of optimising several aspects of the CDN architecture, in order to guarantee a contractual level of service to the content provider. CDNs have been conceived for Web applications. Geographic replication of Web sites is fundamental for Web content providers to guarantee that their content is available across the whole Internet. By outsourcing replication to CDNs, they do not have to provide their own replication infrastructure. A CDN is typically owned, deployed and maintained by an individual operator, that charges content producers and/or Web site owners for its services. CDNs are totally transparent to Web users, as users are automatically (and transparently) re-directed to the replica site that is deemed most appropriate to optimise content access. To this end, CDNs also deploy advanced monitoring techniques to understand where to dynamically redirect each user request, taking into consideration the current load of both the network and the replica sites. With respect to the building blocks in Fig. 1, CDNs mainly provide solutions for replication, placement and indexing issues, while rely on vanilla TCP-based file transfer for non real-time content transfer, and on techniques inherited by the streaming communications literature in case of real-time transfer. Section 3 is devoted to describe these approaches, focusing mostly on how CDN solutions cover the first two building blocks.

Unlike CDNs, P2P systems "blossomed" at the edges of the Internet, by exploiting collaboration between Internet users. The bottomline concept of P2P technologies is that users contribute part of the resources of their devices, and receive content-centric services in return. Most of the time, P2P services are free of charge, distributed, and there is no concept of operator or central manager

(with some notable exceptions). For example, in file sharing services such as BitTorrent, each peer contributes parts of the files it locally stores, and is granted download bandwidth by other peers based on how much upload bandwidth it is contributing. While CDNs are mainly targeted at Web applications, the P2P concept has been used for a very large range of content-centric applications, from file sharing to multimedia streaming to video-on-demand. With respect to the building blocks in Fig. 1, P2P solutions include a significant variety of systems, and can address all the three blocks. We describe these solutions in Section 4.

Note that, as highlighted in [3], CDNs and P2P are complementary solutions. CDNs can provide robust services with guaranteed performance, exploiting the fact that there is a central entity that can enforce QoS levels. Clearly this comes at a non-negligible cost for CDN users. P2P solutions can seldom provide guaranteed services, but can provide forms of "statistical" guarantees thanks to their large diffusion: assuming all P2P users contribute sufficient resources to the P2P service they join, the more the service is used, the more resources are contributed, the better the level of the service that *most* of the users enjoy *most* of the time. P2P users receive services for free, by accepting to contribute resources and tolerating the possibility of receiving, once in a while, a lower-quality service.

The *optimisation frameworks* bar in Fig. 1 includes proposals concerning optimal replication and placement of content, as well as associated heuristics to approximate optimal solutions. In the literature, this is a body of work on its own. Optimisation techniques have such an important role on content-centric networking, that often CDN and P2P solutions simply incorporate one of the solutions proposed in the literature. Therefore, optimisation frameworks are presented when discussing CDN and P2P solutions directly in Sections 3 and 4.

For what concerns the trust, security and cooperation cross issues, it should be noted that P2P technologies usually take them into particular consideration (more than CDNs do), to the point that it is possible to identify specific research areas on trust, cooperation and security for P2P. Although this survey mostly focuses on content management techniques, we briefly cover these aspects in Section 4, due to their importance for P2P systems. On the other hand, CDNs, being typically owned and operated by a central entity, are not considered particularly exposed to trust and cooperation problems. Security issues are typically handled through standard Internet security solutions, such as using SSL for secure file transfer [7]. We do not deal with security, trust and cooperation issues for CDNs in the rest of the paper. With respect to wireless and mobility aspects, both CDN and P2P approaches have been originally designed having in mind a fixed Internet environment. However, solutions have been proposed to extend both CDN and P2P systems in wireless and mobile environments, and are thus included in Sections 3 and 4, respectively.

Finally, the top part of the Figure shows the most popular classes of applications that build on some form of content-centric networking. Several types of Web-based applications can be seen as content-centric applications. In conventional Web browsing, users issue requests for particular pieces of content, completely disregarding where this content is placed in the network, which is a typical "content-centric behaviour". Moreover, Web technologies have been used to support both Online Social Networks and User Generated Content applications such as YouTube and Flickr, which are examples of content-centric applications. This type of applications are typically supported by CDNs, and there is a clear separation between the application-level research topics, and the CDN research topics. Therefore, in the following of the paper we do not discuss applications based on CDNs further. On the other hand, a number of application types rely on P2P systems. P2P file sharing (such as Gnutella, BitTorrent, Kazaa, etc.) is a first class example. In the P2P family we also classify some "pilot" applications, less popular than file sharing applications, which are based on P2P technologies. For example, Microsoft research has developed P2P-based applications for caching in enterprise environments [8,9], or to implement a global software update service based on P2P and network coding transfer techniques [10–12]. With respect to CDNs, the separation between research on P2P applications and P2P content management systems is less evident, and it is difficult to draw a precise boundary between the two. Therefore, in Section 4 we also discuss research issues related to the most important classes of P2P applications.

As Fig. 1 hints, CDN and P2P systems typically integrate in a monolithic way the functions pertaining to the three building blocks, and there is no modular re-use of functionality across content-centric Internet services. Although this aspect is discussed in detail in Section 5, we can anticipate that this is a by-product of the Internet *ossification* [13]. Due to the original Internet design principles, components implementing any of the building blocks highlighted in Fig. 1 cannot be implemented as *core* Internet services. This pushes towards their implementation at the edge of the network, in closed and not-reusable solutions. Along similar lines, also applications are typically exclusively tailored either to CDN or P2P solutions. Web applications exploit CDNs, but – usually – do not use any P2P service, while P2P applications are typically built having in mind a well-defined implementation of a P2P service (e.g., the Squirrel cache service [9] is based on the Pastry DHT [14]). In general, also in this case we can notice a tendency towards a monolithic integration, while re-use of components, services, and modules is missing.

3. Content Delivery Networks

Content Delivery Networks [15–17] address a crucial problem of the conventional Web use, i.e., how to enable content publishers to make their content widely available. In order to make user access to Web content efficient, it is necessary to carefully provision the access link of the server(s), their capabilities in terms of CPU and disk bandwidth, etc. Even more importantly, dimensioning should be performed according to the expected popularity of the pieces of content, i.e. to the expected number of users that will access them, and to the expected temporal pattern of access. Unfortunately, content popularity is difficult to predict, as it usually happens that some piece of content becomes suddenly popular after particular events (e.g., the FIFA World Cup site during World Cup finals [18,19] or news sites after attacks or natural disasters [20]), or because an already popular Web site links that piece of content (the "Slashdot effect" [21]). Clearly, content publishers cannot provision their sites for sudden load peaks (usually called "flash crowds" [22]), as this is not economically viable.

CDNs aim at addressing this issue by replicating content on different sites and possibly on different independent ISP networks. The main idea is thus to replicate content, and re-direct Web requests to one of the replicas according to some selection rule (replicas are usually referred to as *surrogate servers*, while the original Web server is referred to as the *origin server*). As we describe in detail later on, replica selection takes into consideration network congestion and servers load, thus addressing the flash crowd problems [23]. In the CDN design, surrogates actually work as reverse proxies for the origin servers. With respect to server farms, CDN surrogates are not necessarily localised very close to the origin servers, and thus do not suffer from typical bottleneck problems in the link between the server farm and the Internet. With respect to conventional caching, CDN replication is not necessarily driven by client demands, and allow for richer and more flexible replication policies [24].

Usually, developing and maintaining a CDN is not affordable for an individual content publisher. Therefore, CDN services are most of the time implemented and commercialised by dedicated companies (e.g., Akamai [25], Mirror Image [26], LimeLight Networks [27]). Large companies operating in the ICT market (such as Cisco) also deploy Content Delivery Networks [28]. Note that however, a niche of solutions implementing open CDNs also exists (e.g., CoralCDN [29,30] CoDeeN [23,31,32] and Globule [33]). In general, since CDNs have started being investigated in the late nineties, the market share of CDN companies (and the associated benefits for CDN clients) has steadily increased now becoming a multibillion market [34,35].

3.1. CDN technologies for content-centric networking: an overall perspective

Fig. 2 presents a conceptual view of the key technical components of CDN solutions. Content Delivery Networks can be conceptually seen as made up of two key building blocks, namely a content distribution and replication service, and a request redirection service. The former block is responsible for identifying where to place surrogate servers, deciding how many copies of each piece of content to replicate, and where to store the replicas (on the surrogates). As such, it can be seen as the block interfacing the CDN with the content producers. We discuss the technical details of this block in Section 3.2 and 3.3. The latter block is responsible for receiving content requests, identify the place in the CDN where the requested content is stored such that it can be sent back to requesters. As such, it can be seen as the CDN module interfacing with the content consumers. We discuss the technical details of this block in Section 3.4. A couple of (conceptual) modules can be identified, which are related to management functions. In particular, a monitoring functionality is responsible for monitoring the status of the CDN surrogates, their load, the load of the network connecting them, and the user request load. We describe monitoring functionalities in Sections 3.2, 3.3 and 3.4. The accounting module typically deals with accounting mechanisms to monitor the use of CDN resources by clients, e.g., how much traffic a given Web site generates on the CDN. Although clearly important from a commercial standpoint, we do not cover accounting here, as it is of less technical interest for the scope of this paper. The interested reader is referred to [36,16,17]. The set of modules described so far provide the functionality required by traditional CDNs as they have been conceived in the beginning. A further set of modules can be identified, which provide extended functionality, corresponding to the evolution of CDN systems to cope with novel requirements and usage patterns. It is possible to identify extensions to deal with dynamic content (i.e. content generated on purpose upon a user request), described in Section 3.5. Extensions to make CDNs more dynamic have also been proposed. This corresponds, for example, to dynamically adapt the number of surrogates to the user request load. Dynamic CDN extensions are discussed in Section 3.6. Special features are typically required for streaming services, which require significant extensions of the original CDN design. These solutions are discussed in Section 3.7. Finally, supporting mobile users also requires to extend the conventional CDN concepts, as discussed in Section 3.8.

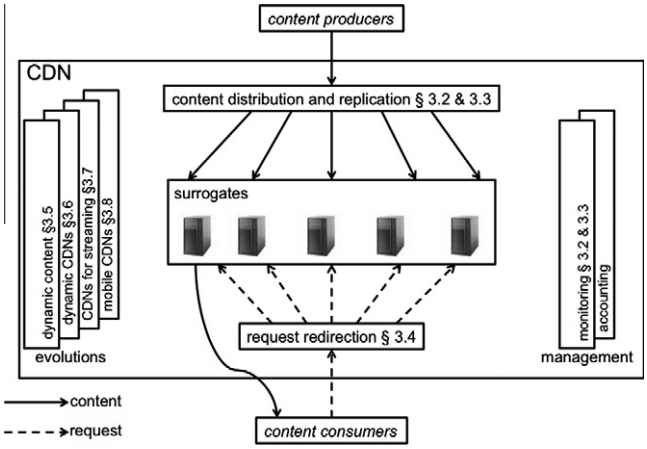

Fig. 2. Conceptual scheme of CDN systems.

In the following sections we do not go too much into the details of the possible alternative solutions proposed for each conceptual module, as dedicated surveys are already available in the literature [17,36,16]. The level of detail we provide is however sufficient to identify the role of CDNs with respect to the content-centric networking aspects of the current Internet.

3.2. CDN content distribution

The content distribution service of a CDN basically deals with three issues: identifying where to place surrogate servers, identifying what content (from a Web site) to replicate on the servers, and identifying on which surrogates to replicate each piece of content. We analyse them in detail.

Surrogate servers (surrogates) are the CDN nodes on which the Web content is replicated. Typically (at least in commercial CDNs such as Akamai) they are dedicated servers carefully placed at strategic locations in the Internet. Therefore, the decision about where to place the surrogates is critical for CDN success. Solutions are typically borrowed from the research on optimal placement of Web server replicas. From a theoretical standpoint, the problem of optimal placement of surrogates can be modelled as an instance of the facility location or minimum k-median problems [37], as well as the k-hierachical well-separated trees (k-HST) problem [38], which look at the optimal placement of a given number of M *facilities* in a set of N possible locations. Alternatively, [39] formulates it as a dynamic variation of the set-cover problem [40]. All these problems being NP-hard, heuristics have been proposed to approximate their optimal solutions [41–45] (see [39] for a general methodology to quantitatively compare them, based on the requirements of the particular CDN). Typically, all heuristics assume knowledge about the workload of the Web clients. Qui et al. propose in [43] the Greedy and HotSpot algorithms. At each step, the Greedy algorithm places one new facility, by identifying the location (out of the remaining available) that minimises the cost for clients. The HotSpot algorithm sorts locations based on the load generated by clients in their vicinity (i.e., by clients at a tunable maximum hop distance from them), and places the M facilities in the M top locations according to this ordering (HotZone, a similar algorithm using the expected latency between clients and possible locations is proposed in [46]). Greedy is shown to outperform HotSpot, and both outperform policies that do not take client workloads into account. Indeed, variations of the Greedy algorithm are used also in [42,44]. For example, [44] proposes the l-Greedy algorithm, in which at each step the algorithm places one facility, but it can relocate up to l previously placed facilities. Another approach (topology-informed placement strategy [47]) consists in placing the facilities in locations with decreasing order of outdegree, the rationale being that nodes with larger outdegree will be more easily reachable. Finally, [45] considers a dynamic placement of facilities driven by client requests, so as to guarantee a minimum pre-defined QoS level (e.g., in terms of delay), and also considers maximum capacities of facilities (thus providing an approximation of the constrained facility location problem).

The second aspect of CDN content distribution is the identification of what content in the origin server is to be replicated on the

CDN. Both full-site and partial-site replication are possible, although partial-site replication is typically preferred. Under partial-site replication, just a subset of the objects available on the Web site is replicated on the CDN, which is clearly a more scalable choice, as the number of objects of a Web site can be extremely large. However, under partial-site replication policies should be defined to decide what to replicate and what not to replicate (note that this problem is typically intertwined with prefetching solutions, that we will cover next in this section). Clustering of Web objects is usually adopted as a technique to define these policies [48,49]. Once clusters are defined, they are ranked according to some performance index (e.g., the expected delay reduction experienced by the users if the cluster is replicated on the CDN), and the set of clusters to be replicated is thus identified. The most flexible and best performing solutions are achieved when clusters are formed by individual objects [48]. This provides the finest granularity in the raking of objects, and thus allows for full customisation with respect to whatever performance index. However, clustering at the object level is typically not feasible from a computational standpoint, due to the huge number of objects to evaluate. Therefore, different clustering techniques are used. For example, [48] defines a distance metric between objects, which is a function of the objects spatial, temporal, or popularity locality (e.g., two objects are close to each other according to spatial locality if they are typically accessed from clients close to each other). Then, standard algorithms are used to identify the clusters. For example, clusters can be defined to minimise their maximum diameter subject to a maximum number of clusters, or minimise their number subject to a maximum allowed diameter. [50] takes an original approach to clustering, as it identifies "communities" of pages hosted at the same origin server by looking at the structure of the hyperlinks between them.

The last aspect of CDN content distribution is identifying how many copies of each piece of content to replicate, and on which surrogates. This task is typically termed "content outsourcing". Basically, content outsourcing does not tell *what* to replicate, but *how* to replicate it. The literature on this subject is extremely vast. Therefore, we prefer to present it in a dedicated section (Section 3.3). Hereafter, we provide a high level classification of the possible approaches.

Three approaches are typically considered, i.e., cooperative push based, non-cooperative pull based, and cooperative pull based outsourcing [16,17].

Cooperative push based outsourcing essentially corresponds to traditional pre-fetching. The key distinguishing feature of cooperative push based approaches is that content is *proactively* replicated on surrogates, before clients actually issue requests. Solutions in this class identify the optimal placement of a given set of objects across a given set of surrogates, such that some cost function is optimised. Typically, this problem turns out being NP-complete, and thus heuristics are defined. As discussed at the beginning of Section 2, with respect to the reference scheme in Fig. 1, this body of work represents the part of the "optimisation frameworks" box conceived for CDNs.

Note that cooperative push based outsourcing is different from the facility problem used for surrogate placement. In cooperative push based replication, clients have different access patterns to different objects (while in the facility problem each facility is equivalent for the clients). Furthermore, the same object can be replicated more than once on multiple locations or even not be replicated at all (while each facility must be placed exactly at one location). Cooperative push based algorithms are strictly intertwined with the object clustering solutions described before. Usually, clustering is applied to the part of the Web site that has to be replicated, and then one such optimisation algorithm is applied to the clusters, eventually identifying which clusters to replicate where. A joint solution of this type is presented in [48].

It should be noted that the most popular CDNs (either commercial or not) do not use such frameworks in practice. Most of the proposed heuristics assume knowledge about the location of Web clients and their request rates. This is a huge amount of information indeed, which is hardly available to, and manageable by, a CDN provider. Furthermore, as we discuss in Section 3.4, replicating objects according to such frameworks can result in inefficient performance of the request redirection solutions typically used by CDNs.

Non-cooperative pull based outsourcing corresponds to pure caching, as objects are fetched by surrogate servers only when a client request cannot be satisfied (i.e., upon a cache miss). The terms non-cooperative stands for the fact that there is no cooperation among different surrogates. Upon a miss, the surrogate to which the object has been requested retrieves it from the origin server and caches it. Therefore, object replication is driven by clients requests according to a very simple algorithm. Due to its extreme simplicity, and because they work well with their redirection algorithms, most commercial CDNs - e.g., Akamai [51] actually use non-cooperative pull based replication.

Also in *cooperative pull based* outsourcing solutions (e.g., CoralCDN [29,30]) content is not pre-fetched. However, upon a cache miss, surrogates cooperate in order to find other surrogates that possibly store the requested object. If such a surrogate is found, that content is replicated on the surrogate that experienced a miss (and returned to the client). Specifically, in CoralCDN surrogates form a Distributed Sloppy Hash Table (DHST) [52], that allows them to find *nearby* surrogates storing a given object. With respect to non-cooperative solutions, the replication process is still driven by clients requests, but fewer accesses to the origin servers are required. In addition, overhead is kept at a reasonable level, as, thanks to DSHTs, cooperation occurs between nearby surrogates, thus limiting the overall traffic on the CDN.

As a final note, replication oriented to fragment-based Web pages has also been proposed [53–56]. Fragment-based Web publishing is a way of composing Web pages out of individual fragments. Each fragment is a well-identified piece of content stored separately in the origin server, and several properties, such as validity and cacheability, are defined on a per-fragment, instead of a per-page, basis. Fragment-based publishing in general, and CDN replication organised around fragments in particular, provides several advantages. For example, the same fragment can be shared between different pages. This means that just one replica of the fragment has to be managed in the replication stage, thus reducing the bandwidth and storage cost of the CDN. Similarly, invalidation can be performed at the individual fragment level, which also reduces the traffic related to consistency management between the origin server and the surrogates. The main drawback of fragment-based solutions is that they still require significant manual configuration, as the fragment definition has to be hard-coded in the Web page design (although [55] investigates solutions to automatically detect fragments).

3.3. CDN object replication

The problem of optimally replicating a given set of objects on a given set of servers has been extensively studied starting from the early 90s [57], without necessarily special reference to CDN environments [58–67].

One of the reference papers focusing on a CDN environment is [68]. The authors show that the optimal object replication problem is NP-hard (this is also shown with respect to general Web replication in [69]). Therefore, they propose and compare four heuristics,

namely random, popularity, local greedy, and global greedy. Under *random*, at each step the object to be replicated and the surrogate on which to replicate it are chosen according to a uniform distribution. Under *popularity*, each surrogate independently ranks the objects according to the previous history of requests for the objects (their popularity). The ranking is used to select (at each surrogate) which objects to store. Under *local greedy* each surrogate ranks the objects according to the expected cost reduction if the object is locally replicated. Again, objects are stored at each surrogate according to this (local) ranking. Finally, under *global greedy* the ranking is performed across all surrogates, and, at each step, the pair (object, surrogate) is selected, which provides the highest cost reduction among all possible pairs. Results show that the *global greedy* policy significantly outperforms the others. This result is indirectly confirmed by the fact that several other papers adopt greedy heuristics. For example, similar policies are defined, with respect to similar environments, in [70,71].

More recently, other papers look at the problem from slightly different angles, often still reverting to greedy heuristics to approximate the optimal solutions. For example, [72] proposes the *lat-cdn* heuristic, in which objects are replicated irrespective of the expected request rates of the clients (while previous work assume that this information is available, and is one of the key parameters for defining the cost function). *Lat-cdn* starts from empty caches on surrogates, and goes through a series of iterations. At each iteration it computes the minimum distance between the origin server and the surrogates, and replicates the object corresponding to the maximum of the minimum distances. The algorithm iterates until all caches of the surrogates are full. This approach is extended in [73] by also considering the load of the surrogates.

While in the previous work the emphasis is on optimising some average cost metric (such as the delay or the hit rate), in [74] the focus is on providing performance guarantees. Also in this case, greedy heuristics are proposed. Specifically, in the *l-greedy insert heuristic* the algorithm starts from an empty system, and greedily replicates one object at a time, possibly replacing up to l previously allocated objects to insert $l+1$ objects. The *l-greedy delete* heuristic starts from a complete replication, and greedily removes the replicas, by considering all possibilities of re-inserting l replicas and removing $l+1$ replicas at each step.

An interesting research direction looks at the object replication problem not as an isolated issue, but focuses on joint optimisations of replication and other key components of a CDN system. The authors of [75] provides a formulation that jointly optimises the placement of the surrogates, the size of their memory devoted to replication, and the replication of pieces of content. As the problem is NP-hard, a simple greedy heuristic is proposed, according to which objects are placed iteratively on surrogates, picking at each iteration the pair (object, surrogate) that maximises the expected gain (assuming client request rates are known). The heuristic is extended to take into consideration the maximum load sustainable by the surrogates, as well as the possibility of peering between surrogates. A similar joint optimisation approach is also explored in [76,77].

Joint solutions for multiple CDN problems are also considered in [78,79]. The authors provide formulations for the joint surrogate placement, object replication and request routing problems, without, however, providing any heuristic to approximate the optimal solutions. These studies are very interesting, as they jointly consider the effect of replication strategies on request redirection policies. As we discuss in Section 3.4 the lack of practical algorithms dealing with this problem is one of the main reasons why commercial CDNs do not exploit optimal replication policies in their systems.

While the vast majority of proposals use greedy heuristics to approximate optimal solutions, a few papers propose heuristics that guarantee to find a solution whose value is within a guaranteed "distance" from the optimal. The interested reader is referred to [62,80] for examples of these approach. Finally, simulated annealing techniques have also been proposed [81]. Simulated annealing algorithms search for the optimal solution, by using mechanisms to "jump out" of possible local optimal solutions that may not correspond to the overall optimum.

Note that, whatever algorithm is used, several replicas of the same object may eventually be spread on different surrogates. If these objects can change over time, consistency issues arise. Conventional Web caching consistency mechanisms may be used to deal with this [82]. Other proposals modify the optimisation problems discussed above in order to take consistency issues into consideration. Such modifications are not trivial, as consistency management requires additional traffic between the origin servers and the surrogates. Bringing this cost into the analysis introduces a trade off between the advantage of replicating objects close to the clients, and the disadvantage of keeping consistent a large number of replicas located far away from the master copy at the origin server. This problem is studied, for example in [83,84] for TTL-based consistency, and in [85,74,76] for invalidation-based consistency.

3.4. CDN request redirection

Request redirection includes the set of mechanisms used by a CDN to map a client request for a Web object to its actual location in a CDN surrogate server (or in the origin server). Fig. 3 shows the typical sequence of events occurring when a client issues a request. Note that, at the Web *page* granularity (not necessarily at the Web *object* granularity) any redirection mechanism must be transparent to the end users, i.e. clients must get the requested Web pages as if they always got them from the origin server. Overall, the redirection process works as follows (we describe each step in detail in the rest of the section). A Web client issues a request for an object by providing its URL to the browser. This request is conveyed to a redirection block (step 1), which selects a surrogate where the requested object should be stored (step 2). The Web browser requests the object from the indicated surrogate (step 3). Surrogates may or may not be organised in cooperative clusters. If they are, an indexing mechanism is available within the cluster, mapping objects to their actual locations in the cluster surrogates. In this case, the index is looked up (step 4), and the appropriate surrogate storing the object is identified (step 5). The object is finally sent to the browser (step 6). If the object is not available in any surrogate of the cluster, a miss occurs and the browser is redirected to the origin server. If surrogates are not clustered, either the object is stored at the surrogate identified in step 2, or a miss occurs.

The first important architectural element is the *redirection* block, which is responsible for mapping the URL of the requested

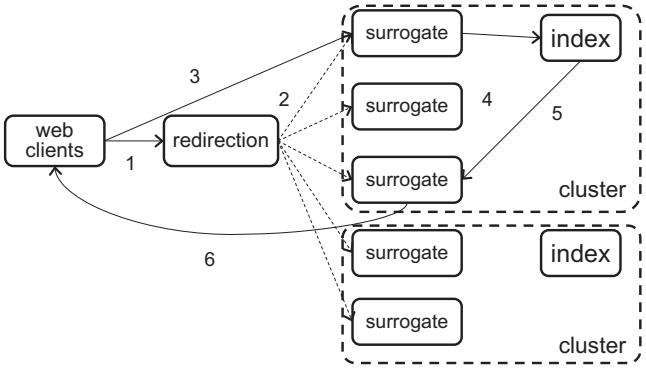

Fig. 3. Conceptual architecture of a CDN redirection system.

Web object to the IP address of a surrogate possibly storing it. This typically requires some levels of indirection, and thus the implementation of the redirection block must be very efficient [86]. The most popular redirection technique is DNS redirection [87–90], which is also used by commercial CDNs such as Akamai (see [91] for a compact yet precise description of the Akamai redirection mechanism). According to this scheme, the browser issues a standard request to a local DNS server to resolve the object's URL. As that URL is served through a CDN, the record in the local DNS points to a dedicated hierarchy of DNS servers managing the URL space of the CDN [36,17]. The DNS servers cooperatively select a surrogate (according to algorithms that we describe next) to redirect the client to. Another popular redirection mechanism is URL rewriting [68,92]. Although, conceptually, URL rewriting can still be represented as in Fig. 3, the detailed mechanisms are quite different. URL rewriting is typically used in case of partial-site replication, when the main HTML file is served through the origin server and surrogates are used for the embedded objects. When the origin server generates the HTML file for the client, it rewrites the URL or the embedded objects so that they point to the appropriate CDN surrogate. Scripts are usually provided by CDN operators to dynamically automate this process. Finally, other techniques proposed for request redirection include Global Server Load Balancing [93], HTTP redirection [36], anycasting (both at the IP [94] and at the application levels [95,96]) and CDN peering [97,98].

Besides the mechanisms used for redirecting requests, the *policies* used to select the surrogate upon a client request is a key part of any CDN solution. Not surprisingly, the literature on this topic is extremely rich. Broadly speaking, one can partition proposals as *non adaptive* and *adaptive* policies. The most trivial non adaptive policy is round robin [99,100]. Another approach consists in defining an expected load of each surrogate, and directing the requests according to these predictions [101]. Cisco Distributed Director [102] implements, between others, a policy assuming that more powerful surrogates should receive more requests. Finally, selections based on hash functions have also been proposed [103]. This is a mechanism similar to standard DHTs: The hash of the requested URL is computed, and the surrogate that is closest (in the hash ID space) to that value is selected. Several adaptive policies have also been proposed [33,104,105,102,51,106–108]. They usually define a virtual distance metric between the client and the surrogates. This metric may take into account a number of information, including the location of the client and the surrogates, the surrogates current load, the network congestion level between the client and the servers, etc. For example, Akamai uses a proprietary, non-disclosed solution of this type [51]. It is interesting to note that this policy turns out being so precise in estimating the quality of network paths, that [91] proposes to use it to also select regular multi-hop paths in the Internet. The drawback of adaptive solutions is that they often need to frequently probe the network status to gather statistics, which clearly generates some overhead. For example, Akamai periodically probes surrogates and network conditions by generating "virtual" clients that request content from the CDN.

All of the above surrogate selection policies can be seen as *server-side* policies, as Web clients have no role in the selection process. These are the typical solutions adopted in CDNs, as they allow CDN operators to have full control on the selection process. It is worth however mentioning the body of work related to *client-side* selection processes. A good survey of them is presented in [109]. In client-side selection it is assumed that a client, upon requesting a Web object, receives a complete list of the surrogates that can provide it. Then, the choice on how to download the object totally lies with the client. Two broad approaches are possible: single-server selection, and multiple-server selection. In the former case, one server is picked (usually the one performing best according to some performance index), and the object is downloaded from it. In multiple-server strategies, the object is divided in blocks, and several servers are used to download it. One block is requested from each server. In this case, several strategies are possible as far as the block definition, including having fixed-size blocks or variable-size blocks, varying according to measurements of the throughput performance. A detailed comparison of these strategies is presented in [109,110]. The main outcome is that single-server strategies are usually preferred, unless in cases where the object is very large.

It is worth finally spending a few words about how clustering of surrogates is organised (note that this clustering has nothing to share with the mechanisms for clustering objects on origin servers). For example, in the case of CoralCDN [30,52] indices are distributed, and Distributed Sloppy Hash Tables are used to maintain them and to identify the surrogates where the requested content is stored. Similar solutions have also been proposed for CoDeeN [32] and in [111–113]. Finally, several papers propose to use standard distributing caching techniques to manage objects among a cluster of surrogates [114–116].

3.5. Dynamic content in CDNs

Dynamically generated Web pages require a much richer conceptual architecture with respect to static content. For example, as described in [117], four layers can be identified in a Web system generating dynamic content. The *front-end* layer is the interface with the clients, i.e. the layer that delivers the final content to them. The *application* layer implements the application logic to generate the dynamic content. The *back-end* layer stores persistent data (e.g., in a database) that is needed by the application layer. Finally, the *user profile* layer stores information about the users, which allows the application to generate personalised content. In this framework, solutions have been proposed to exploit CDNs to replicate any of the four layers on surrogate servers.

Replication of the front-end layer typically corresponds to replicating static content only, according to one of the techniques described in Sections 3.2 and 3.3. Specifically, the dynamic parts of a Web page are generated by the origin server (where the other three layers are located), while the CDN is used for providing the static parts, such as embedded images or static fragments. Solutions for this type of replication have been already covered in the previous sections.

Replication of the application layer actually means replicating the application logic, instead of the content, on the CDN surrogates. This approach is typically referred to as *edge computing* [118,119]. A few solutions have been proposed to dynamically replicate copies of the application code onto the surrogate servers. For example, [120] proposes ACDN, which dynamically re-deploys application copies on surrogates based on client demands, and also routes requests to copies nearby and guarantees consistency among the copies. DotSlash [121,122] adopts a similar approach, even though application replication is triggered only when the origin server is getting overloaded. The advantage of replicating applications clearly lies in relieving the origin server from the computational burden of generating dynamic content upon each client request. However, this approach cannot be typically fully automated, as manual configuration is required, in general, to decide which components of the application should be replicated on the CDN. Solutions targeting automation of this process in small scale environments are presented in [123]. Finally, application replication does not help if the bottleneck of the Web system is the database layer [124].

Replicating the database layer yields a reduction of the load on the central database, and can be achieved through three alterna-

tives, i.e. content-blind caching, context-aware caching and full replication [117]. The latter solution is typically quite costly if the CDN is medium to large scale and sufficiently spread in the network. In content-blind caching, surrogates cache results of queries issued to satisfy previous user requests. Either the tuples of the result [125] or whole tables [126] can be replicated. In context-aware caching, each surrogate runs its own database server, which is typically proactively populated with a partial replication of the central database [127]. Replication on the surrogates is driven by access patterns of clients. While context-aware caching is theoretically more flexible, it typically requires some central controller in order to distribute queries on the databases and coordinate query execution. Therefore, content-blind caching is typically adopted by commercial solutions.

Finally, replication of the user-profile layer can be seen as a subset of the database layer replication problem, as user profiles are typically stored in a database [117]. However, it should be noted that full replication is never considered, as typically, during each session, the profile of a given user is required at a single surrogate only, and thus replicating all the user profiles does not make much sense. Therefore, either content-blind or context-aware replication models are adopted in this case. In addition, the profile of a given user might be migrated from a surrogate to another across two consecutive sessions. Standard solutions for state migration upon user mobility may be used to this end.

3.6. Dynamic CDNs

Research on dynamic CDNs essentially addresses a CDN dimensioning problem. CDNs have originally been designed to address scalability problems related e.g., to flash crowds and similar events. From a content provider standpoint, it is not easy to decide how many CDN resources are needed. It may be the case that CDN techniques are actually needed just in case of sudden spikes of popularity of particular content, while less powerful (and costly) solutions are needed most of the time. On the other hand, also from a CDN standpoint it is useful to include mechanisms in the CDN design to dynamically change the number, location and configuration of surrogates, based on the dynamic evolution of the clients demand.

A few proposals tackle this problem by investigating adaptable CDN solutions. In all the proposed approaches, some form of dynamic reconfiguration is required. In [128] authors organise surrogates in groups. During normal operation conditions, only one surrogate per group (named the primary surrogate) delivers content to the clients. When the primary surrogate starts becoming overloaded, it activates other surrogates of its group and instructs the DNS redirection mechanism to balance requests. Other solutions (typically referred to as "intermediate-layer" solutions [129]) exploit, in case of overload, intermediate network elements between the clients and the surrogates, which may be only "temporarily" enrolled in the CDN architecture. For example, in Flash Crowds Alleviation Networks (FCAN), standard Web proxies become surrogates on demand, when the conventional surrogates become overloaded. When the overload condition is over, they go back to behave as simple proxies. The drawback of these solutions is that they typically require updates of the DNS redirection mechanism, that might be quite a long process. Other proposals adopting a conceptually similar approach are solutions based on an intermediate P2P layer of cooperating nodes, such as CoralCDN [30,29] or BackSlash [130]. Another class of solutions rely only on clients to alleviate flash crowds. Typically, this requires a P2P network formed by the clients that serves as a P2P Web replication infrastructure. Moreover, "self-organising" CDNs have also been proposed to address this problem [131,132]. Self-organising CDNs have indeed a broader scope with respect to management of flash crowds only. The main idea is that surrogates can self-organise, and content placement decisions are taken autonomically based, e.g., on the current load on the network. Two such algorithms are proposed in [131]. In the "Survival of the Fittest" (SF) algorithm, each node associates a value to files passing by, which is basically the cost to retrieve it from the nearest surrogate. In the "Storage Renting" (SR) algorithm, the value is the cost to retrieve the file from the nearest surrogate, minus the cost of storing it. The latter cost increases over time, such that the value of storing a file diminishes with time. A greedy algorithm is used to place content, by which nodes store items with the highest value that pass by. These heuristics are enhanced with load-aware mechanisms. In particular, the cost to retrieve a file is not simply the hop count. A cost is associated to each link in the path to the nearest surrogate storing the file, which is a function of the load on the link. Thus, files stored on surrogates connected through congested paths becomes more valuable, and are more replicated. A conceptually similar system, COCOA, is proposed in [132]. COCOA also includes mechanisms to avoid that too many surrogates change the set of stored content item upon a flash crowd, as this can lead to excessive overhead and oscillations. Specifically, in COCOA some of the surrogates are frozen during flash crowds, to limit the impact of these problems.

It is interesting to note that CDN peering can actually be seen as an alternative method to alleviate flash crowds. In typical CDN peering architectures, different CDNs establish peering Service Level Agreements to avail of each other's infrastructure. When some content is not available on one CDN, it is looked for and retrieved from one of the other CDNs that are part of the agreement. The same mechanism can be also used to temporally alleviate flash crowds conditions on one CDN, if the other CDNs are not overloaded too [133]. An architecture for CDN peering that can support such mechanisms is presented in [97,134].

Finally, an orthogonal problem with respect to dynamic CDNs is how to detect true overload conditions due to flash crowds, and distinguish them from DoS attacks. [135] summarises the characteristics of real flash crowds, and compares them with those of DoS attacks. Both events share similar features, such as the dramatic increase of the request rates and the saturation of the network bandwidth. However, in real flash crowds the number of clients (and not only the total request rate) dramatically increases, while during DoS attacks a few clients generate a disproportionate amount of requests. Moreover, analysis of popularity statistics about the content accessed during a flash crowd shows a Zipf-like distribution, in which a small percentage of content (typically, less than 1%) is responsible for a very large fraction of requests (typically, up to 90%) [136]. In addition, a significant fraction of content requested during the startup of a flash crowd must be provided by the origin server, as it is typically not cached already (this percentage was reported to be about 60% in the case investigated in [135]). All of these statistical features are hardly observed in DoS attacks.

3.7. Streaming in CDNs

Delivery of multimedia content with real time requirements over the Internet is one of the key content-centric services. Both stored content or live streams are delivered over the Internet. It is thus natural that CDNs have been enhanced to support this kind of content delivery services, also resulting in important commercial results (e.g., Microsoft Silverlight http://www.microsoft.com/silverlight/). This is not trivial though. The peculiarities of streaming content is such that several design choices of CDNs for non real-time content must be rethought.

First of all, streaming files are typically very large. In addition, they are usually divided in smaller units, either in the time domain, or in the quality domain [137]. In the first case, a large multimedia file is composed of a sequence of units (called *segments*) that

should be reproduced one after the other [138–140]. In the second case, the file is composed of several *layers*, each improving the quality of the stream obtained by combining all the previous layers [141,142]. This organisation of streaming files has a significant impact on the CDN architecture. Replicating the whole files on surrogates means generating a significant traffic over the network, which should be accounted for. Users might just access portions (either in time or in quality) of the file, thus making full replication unnecessarily costly. Typically, a two level architecture is adopted. The first level consists of the surrogates on which the pieces of the files are replicated from the origin server. The second level is the set of clients that download the pieces of content from the surrogates. Transfer from the origin server to the surrogates occur via conventional non real-time transfer mechanisms (either unicast or multicast), while transfer from the surrogates to the clients occur over multimedia protocols such as RTP and RTSP. For example, the SProxy system in [137] adopts such approach, and works as follows. An SProxy (essentially, a surrogate in the CDN terminology) is composed of three blocks, i.e. a streaming engine, a local content manager and scheduler (LCMS) and a segmentation-enabled cache engine. The streaming engine interacts with the clients, and issues to the LCMS requests for segments that are required by the clients to continue reproducing the stream. The LCMS converts the requests into standard HTTP requests. It also takes care of monitoring the sequence of segments requested by the clients, and decides which segments should be prefetched in order to meet the clients QoS requirements and avoid jitter. HTTP requests are handed over to the segmentation-enabled cache engine, that forwards them to the origin server in case of local miss. Essentially, SProxy implements a non-cooperative pull-based content replication mechanisms. It uses prefetching of segments to avoid blocking of the playback and/or jitters. A similar hierarchical architecture is also used by the iTVP platform described in [143].

The fact that multimedia files are typically large also has some impact on a more theoretical side, in the case where content is transferred through multicast from the origin server to the surrogates (and not only from the surrogates to the clients). Usually the cost of the multicast from the surrogates to the clients decreases if surrogates are placed close to the clients, and with the number of surrogates. However, having many surrogates far away from the origin usually results in less efficient multicast from the origin server to the surrogates. Therefore, a trade-off must be met. From a theoretical standpoint, this means a re-formulation of the optimal replication problems presented in Section 3.2 and 3.3. For example, in [144] a minimum cost solution is found taking into consideration replica placement and routing paths. Simpler versions of this problem, i.e., considering the availability of a fixed number of replicas, have been considered in [145–147,138].

Cooperation between users joining a unique streaming session, is another issue related to CDN support to streaming applications, in the case where the streaming service supports "VCR-like" actions, such as rewinding, fast-forwarding, skipping portions, etc. The streaming service should support these actions, coordinate among possibly conflicting actions of different users, ensuring that all users are synchronised. To tackle this issue, the COMODIN system has been proposed in [148,149]. Essentially, COMODIN exploits a conventional CDN architecture, but enhances the surrogates functionality to provide coordination for VCR-like actions. This is achieved through the HCOCOP protocol and a set of additional modules. Specifically, a Collaborative Playback Session Manager (CPSM) is responsible for managing membership of the streaming session. A Collaborative Playback Control Server (CPCS) is co-located with each CDN surrogate, and provides the interface for the coordinated VCR-like operations. As different members of the session might access the streaming content from different surrogates, a coordination module (called CPCS Coordination Channel, CCC) is responsible to coordinate the various CPCS, possibly arbitrating among conflicting users requests, and synchronising the streaming playback.

3.8. Mobility management in CDNs

To the best of our knowledge, mobility issues have not widely been addressed in the literature so far. Some papers address the problem of surrogate selection in presence of mobile clients [150,151]. Bin et al. [151] assumes CDN surrogates are organised hierarchically, and focuses on streaming applications. Surrogates at lower tiers of the hierarchy are closer to the clients, and thus can provide greater performance in terms of QoS. However, higher-tier surrogates serve larger physical areas, and can thus reduce the number of handoffs. The idea of the request redirection mechanism is to balance between the client requirements in terms of QoS, and the minimisation of the number of handoffs during the same session. To this end, it exploits estimates about the mobile client mobility patterns, and firstly assigns requests to the highest possible tier that meets the QoS requirements. Then, based on the actual client mobility pattern, it evaluates if the client should be served by the higher or lower tier.

In [150] authors focus on multimedia streaming with mobile clients too, and propose the Mobile Streaming Media CDN (MSM-CDN). It adopts file segmentation in the time domain as well as segment pre-fetching techniques as described in Section 3.7. One of the key features of MSM-CDN is how to adapt a standard CDN request redirection scheme based on URL rewriting to mobile users. The authors of [150] propose to exploit the SMIL language [152] to describe how the segments of the content are related to each other, as well as the best surrogate from which the clients should request them. When clients move, they request an update of the SMIL file to a portal server. The portal queries a Content Location Manager, that identifies the best surrogates for the current position of the client, based on its location, the network conditions, and the load on the surrogates. An updated SMIL file is finally sent to the client, in which the URLs to be used to request segments are rewritten according to the CLM indications.

Other papers focus on dynamic content replication in presence of mobile clients (e.g. [153,154]). The main focus of those papers is actually on the dynamic re-evaluation of objects replication. Specifically, authors note that clients' demands usually change over time, and therefore statically computed optimal replication layouts might not be suitable to dynamic environments. They tackle this issue through a standard optimisation problem similar to those discussed in Section 3.3. However, they do not assume that client requests for surrogates are known in advance, but they estimate the request rates by monitoring their evolution over time.

4. P2P technologies for content-centric networking

We now focus on the other key technology used in the current Internet to provide content-centric networking services, i.e., peer-to-peer. According to the P2P paradigm, the network is formed by peers that equally share the burden of providing services to each other in a cooperative fashion. Each peer contributes parts of its resources (network bandwidth, disk storage, etc.), and avails of the distributed service(s) provided by the P2P network.

Several key features make P2P networks particularly suitable for content management and delivery services [3]. P2P networks do not usually require central controllers, but implement distributed algorithms for network management, which provide resilience to peer churns and failures. Furthermore, as peers voluntarily join the network and contribute resources, there is virtually no cost in running and joining a P2P network besides the

cost of Internet access. From a technical standpoint, P2P networks just require peers to run a piece of software, and do not require any support from the core of the Internet besides a conventional TCP/IP stack. Finally, P2P networks are *in principle* self scalable (at least, in wireline environments). Each new peer not only consumes a share of the overall network capacity, but it also contributes additional capacity provided it contributes own resources to the network. Therefore, the total capacity *scales up* with the size of the network, which is something completely unfeasible in conventional operator networks.

All these features enable services that scale to huge numbers of users and huge amounts of content, by means of completely decentralised algorithms that do not require any specific operator support, at virtually no cost. Just to give an example, a recent measurement study of PPLive [155], one of the most popular free P2P IPTV services, has been reported to have served 200000 users watching the Spring Festival Gala on Chinese New Year 2006, accounting for an aggregate bit rate close to 100 Gbit/s [156]. It is thus not surprising that P2P technologies have successfully been established for content management and delivery, and provide today key technical platforms for delivering heterogeneous content, ranging from ordinary files to multimedia streaming, from video-on-demand (VoD) to phone calls.

4.1. P2P technologies for content-centric networking: an overall perspective

To assist the presentation of content-centric P2P technologies, we consider the conceptual architecture in Fig. 4. It is not easy to identify an overall, clean, architecture in which the various P2P services related to content centric networking can be accommodated. However, we can broadly group P2P services for content delivery applications in three main levels. The bottom level provides the basic networking abstractions, i.e., the P2P overlay networks. We present them in Section 4.2, following the conventional classification into structured, unstructured and hybrid overlays.

The middle level provides additional P2P services for content management and delivery. Above them, we can identify P2P applications. These can be either built directly on top of P2P overlay networks, such as Voice over P2P (VoP2P) services – at least in its most successful incarnation, Skype. Or, they may require additional network abstractions (grouped in the Figure in the grey shaded box) helping the content management and delivery process. These abstractions may provide tree structures for multicast communication, meshes, gossip services, or even integrate P2P and CDN delivery mechanisms. We present components belonging to this middle level in Sections 4.3–4.5.

P2P file sharing, streaming and Video-on-Demand (VoD) are typical examples of applications built on top of these abstractions. We discuss them in Section 4.6. Finally, mobility, security, trust and cooperation are seen as cross-layer issues, and will be addressed in dedicated sections. Also, we will discuss problems related to mismatches between the traffic generated by P2P applications and peering agreements among ISPs, which are also a cross-layer issue. Sections 4.7, 4.8, 4.9 are devoted to these aspects.

This classification in levels can hardly be seen as a neat layered architecture as in the classical Internet view. As will be clear in the following, most of the time services implemented at the upper level (and even applications) are totally customised with respect to the particular overlay network they are designed for, or even for its particular implementation. For example, search techniques designed for a structured overlay are completely useless in unstructured networks and vice versa. The most popular example of mesh-oriented solution for content delivery, i.e., BitTorrent, defines its own implementation of what can be roughly seen as an unstructured overlay network. Popular streaming applications such as PPLive [155] implement their own delivery structures and overlays. Such a vertical integration is a typical characteristic of the bottom-up process according to which P2P services are created and establish themselves. They typically are created to answer a very concrete need of a community of users, and are designed to provide a "quick& dirty" solution to these need. Skype [157] and BitTorrent [158] are clear examples of this approach. Another example of "layer-less" P2P components is gossiping. As we will explain in Section 4.5.2, gossiping techniques can be used in a range of P2P-related components, from construction of the overlay structure to data dissemination.

4.2. P2P overlay networks

In this section we describe the main solutions to provide structured, unstructured and hierarchical overlay networks. We intentionally keep the presentation of this part brief and concentrated. Overlay networks have been the first area being investigated, and a number of surveys already cover most of the research carried out in this field. Thus, we prefer to devote more space to the other blocks highlighted in Fig. 4, that have recently provided very interesting results. The interested reader is referred to [159,160] for extended surveys on P2P overlay networks.

4.2.1. Structured overlays

The general idea of structured overlays is defining a virtual address space (usually, in the form of a string of hexadecimal digits) and assign overlay addresses to peers in this space. The space is organised according to a given geometry (e.g., a ring, a multidi-

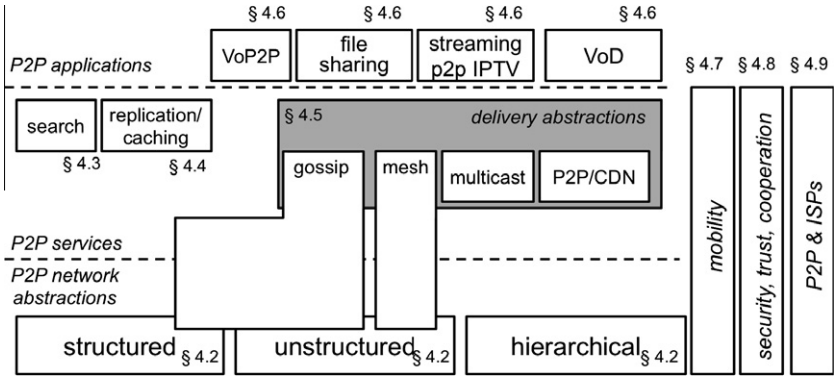

Fig. 4. Content-centric P2P technologies.

mensional cartesian space, a hybercube) which defines neighbourhood relationships between peers. Each peer is responsible for a certain portion of the address space, specifically for all identifiers to which it is the closest available peer in the overlay (the definition of distance is coupled with that of the geometry defining the overlay structure). The most common use of such an overlay is for storing and locating objects. To this end, objects are also assigned an address in the same logical space of the addresses of peers. An object to be stored in the overlay is stored on the peer responsible for the address of the object.

The core functions of a structured overlay network are address assignment, join and leave operations, structure maintenance, routing and forwarding. To give an example of how they can be implemented, we focus on Pastry [14]. As several other proposals, Pastry organises peers in a logical ring, addresses being positions in the ring. Addresses are computed by hashing some identifier of peers (such as the IP address) or objects (such as a key value associated with the object) through a collision resistant cryptographic hash function. Forwarding in Pastry is the process through which a message addressed to a given identifier in the ring reaches the node responsible for that id, possibly following a multi-hop path on the overlay. The forwarding algorithm exploits the prefix-match concept, exemplified in Fig. 5. At each step[1], the node holding the message computes the prefix match between its id and the id of the destination, and forwards it to a peer whose match is longer than its own. Nodes along the path thus share an increasing number of digits with the destination address, until the peer responsible for the destination address if found. The Pastry routing procedures fill up several structures, specifically a routing table, a leaf set and a neighbour set. These structures, filled according to the Pastry routing procedures, guarantee that nodes can always find a next hop whose match with the destination is longer than that with the current node. We omit the details of such procedures as they are quite long and do not add much to the presentation. It is just worth pointing out that these procedure generate background traffic to keep the routing structures consistent in case of any modification of the overlay (i.e., peers joining or leaving). To join a Pastry network, a new node must get in touch with a peer already in the overlay. It then collects a number of entries from the routing table of that peer and other peers, which allow it to fill its routing structures in a consistent way (the routing structures of the contacted peers might change as well). No specific procedure for a leaving node is specified, i.e. nodes leaving events are handled as node failures by the routing procedures.

The forwarding mechanism used by Pastry is actually shared by several other structured overlay networks. The first proposal exploiting the same concept is PRR [161]. PRR uses suffix routing, instead of prefix routing, i.e., hops along the path share an increasingly longer suffix match with the destination id. Starting from the PRR work, other proposals have improved this basic scheme with structure creation and management functionality. Tapestry [162,163], P-Grid [164,165], Bamboo [166,166] and Z-Ring [167] all use this approach.

The use of prefix routing guarantees that forwarding takes $O(\log N)$ steps, where N is the size of the overlay. Using other types of structured overlay results in the same complexity. Chord [168,169] organises the virtual address space in a logical ring. Each node maintains a finger table that partitions the space (as seen by that peer) in increasingly larger intervals following a logarithmic rule. Each interval is assigned to a peer that is used as next hop. Specifically, upon forwarding, a node looks in its finger table for the peer in the space between the node and the destination, which is responsible for the largest interval, and forwards the message to the node responsible for that interval. A similar approach is also taken by DKS [170] and Chord# [171,172].

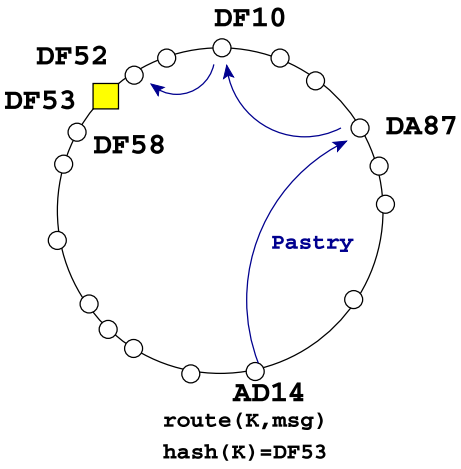

Fig. 5. Routing in a Pastry overlay.

The structured overlays mentioned so far typically require $O(\log N)$ memory space to maintain the overlay structures. Other proposals use fixed-size routing structures, irrespective of the network size. This has the advantage of a lower memory and maintenance overhead, although the drawback is an increase in the complexity of the routing and maintenance algorithms. Examples of these networks are Koorde [173], Ulysses [174] and Cycloid [175,176].

Other approaches use different metrics for defining peers proximity. For example, CAN [177,178] organises the overlay in a d-dimensional Euclidean space, achieving path lengths in the order of $(d/4)(N^{1/d})$. Kademilia [179] uses a non-Euclidean distance, i.e. the exclusive OR (XOR) function.

One of the design decisions for structured overlays is the trade off between the routing state and the path length. As we have seen, prefix routing schemes achieve $O(\log N)$ path length with $O(\log N)$ routing state. On the other hand, in proposals using fixed routing state, the path length increases linearly with N. At the opposite side of the spectrum, other proposals target constant $O(1)$ path length, irrespective of the network size. This is of course paid in terms of a steep increase (relative to prefix routing approaches) of the routing state size. An interesting study of this trade off is presented in [180]. Overlay networks targeting $O(1)$ path length are Kelips [181], OneHop [182], EpiChord [183,184] and D1HT [185,186].

4.2.2. Unstructured overlays

Unlike structured overlays, unstructured systems do not enforce any particular structure in the network. From a topological standpoint, unstructured networks typically resemble random graphs (we come back on the properties of unstructured overlays from a graph theoretic point of view at the end of the section). In some unstructured P2P networks peers do have not all the same role. With respect to this point, they are typically divided in three categories, i.e., (i) hybrid decentralised, (ii) purely decentralised, and (iii) partially centralised (or hierarchical) [160].

Hybrid decentralised systems have been one of the first types of unstructured networks implemented at large scale, typically for the purpose of sharing contents. Each peer contributes some content, but a central server is used to store an index of what content is available where. From a topological standpoint, it is even difficult to identify an overlay network of any kind, as all peers query the server, and then establish a regular TCP connection to the peer that stores the content. Napster [187] and Publius [188] are examples of such systems.

Purely decentralised systems are the most pure version of unstructured P2P systems, as all peer have the same role and share

[1] For the sake of clarity, we present a slightly simplified version of the actual Pastry forwarding algorithm.

the same responsibilities. Gnutella version 0.4 [189] is the most notable example in this class. In Gnutella a new peer joining the network establishes a first link (in the overlay) by connecting to a bootstrap node, i.e., another peer already in the overlay that accepts the joining peer as a new neighbour. The joining peer then establishes additional links by flooding a limited-TTL Ping message through the bootstrap node. Each node receiving such a message (i) decrements the TTL (ii) forwards the Ping message – if the TTL permits – to its neighbours, and (iii) sends back a Pong message to the joining peer if it is willing to have it as a new neighbour. Each peer in a Gnutella overlay tries to keep the number of neighbours between a lower (LB) and an upper bound (UB). While the current number of neighbours is below LB, new Ping messages are generated. When it is between LB and UB, the peer does not look for new neighbours, but accepts as neighbours peers that send Ping requests (i.e., it sends back a Pong reply). When the number reaches UB, the peer stops accepting new neighbours. FreeHaven [190] is another example of solutions in this class. Note that the BitTorrent overlay network can be seen as a purely decentralised one, although trackers represent a "spurious" element from this standpoint. We do not describe BitTorrent here, as we prefer to discuss it while presenting mesh overlay abstractions.

Partially centralised systems divide peers in two class. More powerful peers (either in terms of bandwidth, or resource in general), usually called superpeers, and ordinary peers. Superpeers form a conventional unstructured network. Ordinary peers can only connect to superpeers, and do not have links between them. This is a very pragmatic solution to exploit the advantages of the purely decentralised approach, while taking into consideration the common heterogeneity of peer resources. Several popular overlay networks adopt this approach, e.g., FastTrack [191], Gnutella version 0.6 [192], Skype [157].

The basic mechanism used to build unstructured networks permits several degrees of freedom in the choice of the neighbours. This can be exploited to obtain certain properties in the overlay structure. This is one of the motivations for the work carried out in the IETF Application-Layer Traffic Optimization (ALTO) Working Group [193], which aims at exploiting network- and link-level information to guide the peer selection process. Moreover, a few proposals look at choosing neighbours based on their users' interests. The rationale behind this choice is that people sharing common interests are likely to share similar content, and thus searches for a particular type of content is more efficient if peers likely to store that content type are neighbours. Examples of this approach are presented in [194,195], and systems using the same concept, although more implicitly, are proposed in [196,197].

As mentioned at the beginning of the section, topologies in unstructured networks can be modelled with random graphs, at least when the overlay construction and management algorithms do not dictate any special rule to select neighbours. The properties of these random graphs have been studied for Gnutella and similar networks in [198–200], and, more recently, in [201]. Unlike previous studies, [201] shows that the degree distribution of superpeers does not follow a power law, but most of the superpeers keep around 30 neighbours. This is intuitive, as 30 is the target number of peers for the Gnutella implementations in the studied network. Despite not showing a power law degree distribution, the network still shows a remarkably small average path, of about 4 hops in a network in the order of 100 thousands nodes. This is an indication of a "small-world" behaviour. Interestingly enough, the same work also analyses the stability of links between peers, highlighting an "onion-like" structure. Specifically, peers tend to have stable connections with other peers that stay in the network for a similar amount of time or for longer times. Thus, very stable nodes tend to form an extremely connected core component.

4.2.3. Hierarchical overlays

Conceptually, unstructured architectures with superpeers are hierarchical networks, the superpeers forming the top level, and ordinary peers the bottom level. However, this form of overlays is usually still classified in the unstructured family [160,202]. With hierarchical overlays we thus denote a more complex architecture. Nodes are organised in different groups, and each group runs its own overlay. A top-level overlay is then formed by one representative per group. Such an architecture is more flexible than the conventional unstructured architecture with superpeers, as each group can possibly implement a different overlay network (either structured or unstructured), which can be different also from the top-level overlay. This solution permits greater flexibility, as each group can use the most appropriate type of overlay. Also, the different overlays can be isolated from an administrative point of view. Examples of such hierarchical overlays are presented in [203–205].

4.3. Search in P2P networks

Searching for content is one of the key services for which overlay networks are used, and thus it has attracted – and still attracts – a lot of attention. In this section, we firstly present issues related to indexing, then we survey search solutions for unstructured and structured overlays, as well as solutions that jointly exploit both types of overlays (hybrid search).

4.3.1. Indexing issues

Indexing is the service that maps available content to places in the network where the content can be found. In the area of P2P networks, indices can be of three types: centralised, localised, or distributed. Centralised indices typically rely on a unique entity in the network that stores the index. Peers have to query the central index in order to know where the content of interest is stored. Systems adopting centralised indices are Napster and iMesh [206]. BitTorrent can also be seen as adopting this approach for the identification of peers in a swarm, as we will explain in Section 4.6.1.

Using localised indices is what most unstructured networks do (e.g., Gnutella). In this case, each peer indexes only content available locally. A query looking for a particular content must thus be propagated among peers in order to check each local index. In unstructured networks with superpeers, typically superpeers keep a (sort of) index for the ordinary peers attached to them [192].

Finally, in systems implementing a distributed index, the index is distributed among the peers. DHTs can be used to this end, basically according to two schemes. In the first scheme, each piece of content is stored on the peer responsible for the content id (the hashed value of a key associated with the content, see Section 4.2.1). In this case, the DHT itself – actually, the hash function – defines an implicit index. In the second scheme, the peer responsible for the content id stores a reference to the node where the content is stored (e.g., its IP address). The hash function is thus a way to access the index, and the index is distributed on the peers. The KAD system [207] adopts this scheme.

4.3.2. Search in unstructured networks

Search techniques in unstructured networks are very widely investigated in the literature. One of the reasons is that unstructured topologies are very robust with respect to flash crowd [208], making them suitable for distributed content retrieval systems. However, as no content-centric topological structure is enforced (as in the case of structured networks), search typically results in the exploration of the overlay. Making such exploration efficient in terms of overhead, without sacrificing performance in terms of success ratio, is an exciting and non-trivial problem.

More specifically, as mentioned in the previous section, unstructured networks typically adopt localised indices. Thus, queries must be propagated in the network until they hit a content matching the object of the query. The most trivial approach to achieve this is by flooding the overlay network. This is actually the algorithm used in the first version of Gnutella. As flooding generates too much overhead, more intelligent mechanisms have been investigated. The first proposed improvements have been random walks and expanding ring searches [209]. The expanding ring technique floods the network with increasing TTL values, until the content is found. In case of random walks, the query is forwarded at each step by selecting a random peer among the set of neighbours of the forwarding node. A random walk ends when the content is found, or when a stopping condition is reached. [209] proposes, as stopping condition, either a TTL value, or that each node reached by the walk checks back with the issuing peer whether to proceed or not. The latter condition is preferred. The most effective version of the random walk technique issues a given number (k) of parallel random walks from the querying peer. It has been shown [210,209] that k parallel walks visit, in a given amount of time t, approximately the same number of distinct nodes visited by a single random walk in a time equal to kt. The parallel random walk technique has shown to significantly outperform both flooding and expanding ring [209], and is thus one of the most investigated approaches. The analytical properties of random walks are investigated in [210,211].

Random walks essentially correspond to blind searches, as the next step of the walk is chosen completely at random. Intuitively, it is possible to optimise the search performance if the choice of the walk steps is performed not completely blindly, but introducing some bias in the choices (this class of random walks are usually named *biased* random walks). Without special reference to P2P systems, [212] shows that improvements over unbiased random walks can be achieved if random walks are biased towards high degree nodes, and if nodes are aware of the content available at their neighbours. It is clear that, if nodes are aware of the content of the neighbours, higher degree nodes have more information about where content is located with respect to lower degree nodes. This has been proved analytically and experimentally in [213]. The idea proposed in [212] has been exploited in the framework of P2P networks by Gia [214]. However, the drawback of the approach proposed in [212] is that high degree nodes easily become overloaded. To avoid this, Gia modifies the unstructured topology formation process with respect to vanilla Gnutella, as peers attach preferentially to high *capacity* nodes. The combined effect is that random walks are biased towards high degree nodes, that however have enough capacity to handle the load. Proposals to bias the random walk according to previous results obtained by peers have also been proposed [215].

Adamic et al. [212] show a clear performance improvement when indices are not totally localised, but are replicated one-hop away (the analytical model in [213] actually considers the case of few-hop replication also). This idea is also exploited in [216], where each node advertises its content further away than just one hop. But at each hop, the precision of the information about where content is stored decreases. Specifically, Exponentially Decay Bloom Filters are used to this end, in which fewer bits are used to represent content stored by peers further away. After some point, the information about far away nodes is indistinguishable from noise in the Bloom Filter. Search is performed through standard random walks. The effect of indices advertisement is that the random walk proceeds blindly until arriving in a few-hop neighbourhood of the queried content. Afterwards, the walk is attracted very quickly towards the content. Finally, replicating indices is also the idea of [217]. The authors assume that the overlay network has power-law degree distribution. Each peer generates a random walk whose length is sub-linear with the network size, and the peer's index is replicated at each peer visited by the walk. An analogous walk is performed by queries, that thus also gets "implanted" in the overlay. Then, queries are propagated according to a probabilistic broadcast, i.e., each peer propagates the query to each of its neighbours with a given probability, which is a function of the network percolation threshold. [217] shows that, following this approach, contents are found with probability approaching 1 in $O(\log N)$ steps.

The search techniques described so far assume that pieces of content are replicated as peers request, find and fetch them on the local nodes. Another important direction focuses on search-driven *proactive* replication, i.e. looks at how to proactively replicate content in order to optimise the search functionality. This body of work is actually shared between search and replication techniques. However, we present it here as the replication policy is usually strictly tied with the specific search algorithm, while in Section 4.4 we present replication techniques that are agnostic to the search solutions. A fundamental reference is the square-root replication strategy, proposed and investigated in [218,209]. The authors assume that the search process samples a random set of peers (all peers having the same probability of being sampled). This is almost equivalent to what is achieved by a non-biased random walk, as proven in [210]. Under this assumption, authors of [218,209] prove that the policy producing the lowest overhead in the search process replicates contents according to the square root of their popularity (measured as the request rate). The "square-root principle" has drawn significant attention also beyond proactive replication techniques. Interestingly enough, [219,220] have shown that the same effect of square-root replication can be achieved by biasing the overlay topology formation. Specifically, each node should have a number of peers proportional to the square root of the popularity of the content at the peer. Finally, [221] shows that the very same effect can also be achieved without imposing any particular topology, but by biasing the random walk at each node according to the popularity of the content stored at the peers. The key result of [221] is that such a biased random walk achieves, at each node storing a particular content, the same visiting probability (by one random walk) of square-root replication with unbiased random walks.

While the work related to square-root replication targets the optimisation of the search cost overhead, another research stream looks at replicating and searching for content so that the search failure probability can be analytically bounded [222,223]. These proposals build on the following observation. Assume a given content is replicated on d random places, and a search process visits q random places, all places being chosen independently. Then, the probability that there is no intersection between the places storing a replica and the visited places is e^{-qd}. Exploiting this result, [222] proposes Bubblestorm. As the search failure probability depends on the product of q and d only, in Bubblestorm q and d are selected such that the resulting traffic is minimised. [223] provides a simple way to practically select the peers onto which to replicate, and the peers to visit. Specifically, it exploits the result in [210] about random walks and sampling nodes, which states that, after an initial number of steps in the order $\Omega(\log n)$, the next steps represent a uniform random sample of the nodes in the network. Therefore, in [223] both replications and searches are implemented through random walks of length $c \log n + \sqrt{n \log n}$, and replica and search functions are performed only in the last $\sqrt{n \log n}$ steps. It is worth noting the similarity between this body of work looking at replicating *content* and the work looking at replicating *indices* [217] building on a similar analytical background.

The last approach we highlight related to search in unstructured networks is Local Minima Search (LMS) [224]. On the one hand, LMS shares an approach similar to Bubblestorm. Both replica and

search are replicated on independently sampled nodes, so as to provide a bounded failure probability in the search process. In addition, LMS provides a DHT-like search semantic on unstructured networks. As in DHTs, LMS assigns ids both to nodes and to contents. The LMS search and replication processes are conceptually identical, and include a random and a deterministic walk. A walk is performed for any replica to be stored and for any search for a given content. The walk is a standard random walk. During the deterministic walk each node evaluates the distance between the content id and the ids of its neighbourhood, typically at in a range of 1 or 2 hops. The deterministic walk is directed towards the neighbour with closest id, until it reaches a *local* minimum for that id. At this point either a new replica is stored, or the result of the search operation is returned. Among other reasons, LMS is an important reference point as it shows that a DHT-like search semantic can be implemented in unstructured networks too. A similar approach (although less elaborated) is also used in FreeNet [225].

4.3.3. Search in structured networks

Structured overlays provide the perfect support for exact match (or "pin") queries, i.e., queries issued on the values of keys used for storing the contents in the overlay. In these cases users query for "content whose key is k", and, due to the overlay mechanism, the only place where the content can be is at the peer responsible for the address $hash(k)$. Therefore, a search operation results in a forwarding operation, and is as efficient as forwarding on the overlay.

The challenge in structured networks is providing efficient "free-text" (or keyword) search, which is a much more general case in which users asks for any piece of content containing a given set of keywords, throughout referred to as K. Note that in unstructured networks this type of queries is easily supported, essentially because indices are mostly local. For example, when random walks are used, each peer visited by the random walk can easily lookup the index for content having the set of keywords K.

A possible solution, proposed in [226], can be applied when the number of keywords (Q) in the query is known in advance and fixed, and is equal to the number of keywords that each content can contain. In this case, an extended DHT mechanism can be used. A Q-dimensional virtual coordinate space can be defined, and each peer is responsible for a portion of the virtual space through a mapping function (conceptually equivalent to hash functions for conventional DHTs). A reference to a piece of content can only be stored at the node responsible for the set of keywords the content contains, and can thus be retrieved according to a forwarding mechanism in the Q-dimensional coordinate space.

More generally, the typical way to support keyword search in structured networks is by building inverted indices. For each keyword k, the inverted index stores a pointer to all the peers that store a piece of content containing that keyword. Therefore, a conceptually possible solution is to store the (inverted) index for keyword k at the node responsible for address $hash(k)$. A keyword search for a set of keywords $K \triangleq \{k_1,\ldots,k_Q\}$ can thus be answered by (i) retrieving the inverted indices relative to each keyword k_i, $i = 1,\ldots,Q$, and (ii) intersecting those indices to obtain the set of contents containing all keywords. This is the basic mechanism used by the Kademilia DHT system [227,179,207,228,229].

Although conceptually simple, and guaranteed to provide optimal recall rates, this approach typically imposes a huge overhead on the network, as inverted indices have to be sent somewhere in the network, when the intersection has to be performed. As inverted indices can be very large (e.g., if the keyword looked for is extremely common in the content stored in the network), this can generate a huge overhead. Usually, this approach is therefore seen as too costly [230]. A more efficient solution has been proposed in [231]. Search is still based on inverted indices. However, a Bloom filter is generated from each index, and the filters are transmitted in order to perform the intersection. Bloom filters represent a compressed version of the index, that can be dimensioned to control the inaccuracy of the stored information. The idea is thus to trade – in a controlled way – precision in the recall process for efficiency in traffic overhead.

A further improvement over the work in [231] is proposed in [232]. Inverted indices do not only store a pointer to the content containing the keyword. In addition, they also store a "pagerank" value, a Bloom filter related to all the keywords that content contains (referred to as BF_c), and the precision value of that Bloom filter. The pagerank value measures how "important" that content is, and is used to order indices at any peer. A query for a set of keywords K is answered as follows. First of all, the querying node asks the other peers responsible for keywords k_i, $i = 1,\ldots,Q$ for the length of their indices, and organises a query execution plan which orders those peers from the one having the *shorter* index on. Then, a Bloom filter for the query is generated (BF_Q), and passed to the first peer in the plan. This peer navigates the index starting from the highest pagerank value. Recall that each item in the list represents a content containing the keyword that node is responsible for. However, to be a possible result for the query, the content should contain *all* the keywords in K. Therefore, the Bloom filter of the content BF_c is compared with the Bloom filter of the query BF_Q. If there is a match, the content may be in the result set. As the Bloom filter is an imprecise representation, the Bloom filter precision is used as a measure of the probability that the content actually contains all keywords in K, i.e. that the content actually belongs to the result set. Contents at the peer are navigated until the total sum of the Bloom filters' precision of contents possibly in the result set exceeds a given threshold. At this stage, the search process has identified a set of candidate results, i.e., a set of contents that, with some probability, actually contain all keywords. This set is then sent to the second peer in the execution plan, which simply checks if all candidate results actually contain the keyword it is responsible for. If a candidate result received from the previous peer in the plan is not available in the current peer's index, it means that the candidate result does not actually contains one keyword, and is thus removed from the result set. Although more complex from an operational standpoint, this algorithm allows the search process to significantly reduce the amount of data to be transferred over the network. Other proposals investigating similar optimisations can be found in [233,234].

The last approach to search in structured network that we mention is "Structella", which has been proposed in [235]. The authors show that it is possible to efficiently perform random walks also in structured overlay networks. Based on this result, it is clearly possible to inherit most of the body of work presented in Section 4.3.2 also in the case of structured overlays. In this sense, [235] is dual to LMS [224], which provides a structured-search semantic on top of an unstructured network.

As a concluding remark on the search problem in P2P networks, it should be clear that no clear winner can be identified between structured and unstructured approaches. The performance depends significantly on the particular overlay solution that is adopted, and on the type of query that is required. For example, [236] shows that, unlike it is commonly believed, a Pastry network can be optimised to achieve low maintenance overhead, can exploit heterogeneity of peers, and support complex queries at least as efficiently as unstructured networks. Finally, [237] compares structured, flat-unstructured and hierarchical-unstructured (i.e., unstructured with superpeers) networks, and conclude that: (i) structured networks are the best choice in terms of response time, but have a high cost for publishing indices, (ii) hierarchical unstructured networks are those with the lowest overhead in

general, and (iii) random walk techniques are the most efficient techniques in terms of publishing overhead, while are the worst techniques in terms of delay response, unless multiple random walks are used.

Other research areas related to search in P2P networks include semantic search and range queries. The interested reader is referred to [238] for a complete survey of these approaches.

4.3.4. Hybrid search and rare objects

We conclude the description about search schemes by considering approaches that exploit both structured and unstructured networks, i.e. hybrid solutions. The key remark behind these solutions is the fact that objects and keywords being searched greatly differ in terms of popularity, and thus, typically, in terms of replication. It has been shown that search for rare items in unstructured networks can be quite inefficient [239–241]. Specifically, according to [239,240] up to 18% of available objects may not be found, just because they are not replicated widely enough, being not so popular. In addition, the query delay is also related to popularity, and queries for less popular items are satisfied with higher delays [239]. For unpopular items structured networks are a better option, as – in some cases – they can guarantee 100% recall rate. However, unstructured networks are clearly more efficient for locating highly popular – and thus largely replicated – items. Hybrid search schemes therefore try to identify popular and unpopular items, and use an unstructured network to search for the former, and a structured network to search for the latter. Note that the structured network required by hybrid search schemes typically indexes just a few items, and therefore the associated overhead is affordable.

To the best of our knowledge, the first proposals of this kind are presented in [239,240]. At each node, local information related to the local content and to the content available on direct neighbours is used to estimate which item is popular and which one is unpopular. Unpopular objects are indexed on PIERsearch, which provides (partial) inverted indices on top of a DHT (see Secion 4.3.3). When a query is issued, the unstructured network is searched first. If no results are returned after a given timeout, it is assumed that the item is unpopular, and PIERsearch is queried. According to this straightforward approach, an initial search on the unstructured network is always issued, which results in overhead. [242] uses a gossip mechanism to collect global statistics about which items are popular and which are unpopular. Based on this information it is straightforward to issue a query on the unstructured or structured overlay, respectively. A more refined strategy is proposed in [243]. First of all, the unstructured overlay is built such that peers sharing similar content are close to each other in the network. This clusters similar items together, thus facilitating recall. Second, a structured network is used to store "hints" about where – in the *unstructured* network – content matching some search criteria are located with high probability. Third, the structured network is also used to locate references to unpopular items. Queries are first fired on the unstructured network. In case of failure, the structured network is looked up for "hints". In case also this stage fails, the items are searched in the structured network. Finally, [244] still uses a hybrid approach, but the role of the structured network is quite different. Items are replicated in the unstructured network according to a rule similar to the one used by BubbleStorm [222]. Such rules require to replicate both items and queries on a number of independent nodes which depend on the size of the network. In [244] the structured network is used to collect data that are used to estimate the size of the network and to generate independent sets of nodes. Other solutions using hybrid search schemes are proposed in [245–247].

Search optimisations in case of rare objects are not necessarily addressed through hybrid schemes. For example, [248] uses a hierarchical unstructured overlay only. The topology is formed in a way similar to Gia [214]. As also shown in other unstructured search solutions (see Section 4.3.2) replicating indices on neighbours, and biasing random walks towards high-degree nodes provides efficient search performance. In [248] this idea is brought one step forward by replicating indices over two hops. Different replication policies are defined for replicating indices from first-hop neighbours to second-hop neighbours. The one providing the best tradeoff between recall rate and overhead replicates on a random set of second-hop neighbour, whose size is proportional to the square root of the first hop neighbourhood.

More recently, the problem of optimising search in case of rare objects has been further addressed in [249]. Specifically, this work considers the problem of search for items that are *not* available in unstructured networks. In standard unstructured search, a possible long process involving multiple long random walks is needed to conclude that an item is not available. This clearly represents a significant overhead. In [249] queries are forwarded over a limited-size random walk. A query terminates when either of the following three conditions are met: (i) the item is found, in which case the query is satisfied, and the issuing node becomes *positive* about the existence of the item; (ii) the walk reaches a node that is *negative* about the existence of the item, case in which the issuing node also becomes negative, and the query is deemed failed; (iii) the query TTL expires, and the issuing node becomes negative. By using this approach a lower overhead is required in order to understand that an item is not available in the overlay.

4.4. Replication and caching in P2P environments

Replication solutions are typically tied with the underlying P2P overlays they are designed for. In the case of unstructured overlays, replication and search issues are usually handled jointly, as described in Section 4.3.2.

In the case of structured P2P networks, a typical approach consists in exploiting closeness in the ID space to proactively replicate objects. One of the most popular solutions in this class is PAST [8]. PAST is built on Pastry, and each object is replicated k times, k being a system parameter. Specifically, the k nodes with closest ID to the object ID are chosen as replication points. In addition, PAST nodes may cache data objects if they have unused memory space to contribute. Each time an object request passes by a node (meaning, the node is on the path between the requester and one of the replica nodes), the node evaluates if caching that object or not. The well-known Greedy-Dual Size policy is used for replacement. Approaches conceptually similar to PAST are proposed by CFS [250] (where objects are split into blocks, and replication is handled on a block-by-block basis), and in LessLog [251] (where the underlying structured overlay has a tree shape). Other proposals slightly modify this approach, by considering also physical information to decide the replication nodes. For example, in [252] load measurements are used to decide the replication points, and the standard DHT routing protocol is modified to point requests towards replica nodes. More recently, [253] proposes Plover, which organises nodes in a hierarchical DHT in which ordinary nodes connect to supernodes which are in physical proximity, and the DHT of supernodes is also organised according to physical proximity (neighbour supernodes are also physically close to each other). Physical proximity is also used to decide where to replicate objects when nodes supposed to store them (according the the DHT semantic) becomes overloaded.

Other proposals approach the replication problem according to standard optimisation frameworks. The main difference with respect to the body of work presented in Section 3.3 is that in this case the target environment are P2P systems rather than CDNs. Beehive [254] still assumes a structured DHT, but replication is performed according to the solution of an optimisation problem.

By assuming that objects popularity (and thus the request process) follows a Zipf-like distribution, the optimal solution guarantees that the average lookup delay is bounded by a given constant, instead of being order $O(\log N)$ as in conventional DHTs. Distributed monitoring of the required problem parameters is used to derive the optimal solution in practice.

Optimisation frameworks are also proposed in [255–257]. With respect to Beehive, these solutions are less strictly coupled with a specific type of P2P network. They basically differ in the way they define their optimisation problem. In [255] authors assume a general P2P environment (one in which nodes cooperate to store each other's files), and aim at optimising the download cost, measured as the average number of links traversed on the download path from the place where a copy of the object is stored to the requesting nodes. The authors find that the optimal number of copies for each object is proportional to the object request rate. [256] focuses on a similar P2P environment, although authors consider that the cooperating nodes form a P2P community with access to the Internet, and other nodes in the Internet are not part of the P2P community, although they can store files that members of the community may download. The bottomline assumption is that downloading is cheap from within the community, and costly from outside the community. Therefore, authors find the optimal replication strategy of objects within the P2P community, i.e., the one that optimises the expected hit rate of requests issued by community members. With respect to [255], they consider the possibility of nodes being temporarily disconnected. They assume the user request rates are known and do not change over time. They find that the optimal replication policy is logarithmic with the request rates. Finally, they provide a heuristic to approximate the optimal solution. Basically, nodes store objects in a decreasing order of locally perceived request rate. The presence of a DHT such as Chord or Pastry is assumed, by way of which clients identify the top-K nodes closest to the ID (in the DHT) of the object they require. Clients query these nodes sequentially until the object is found.

A more general definition of cost is used in [257]. Specifically, authors consider the cost related to downloading the object from replica nodes and the cost of a miss, and evaluate different functions to compute the cost. They propose an optimisation framework in two steps. In the first step, the minimum number of copies is found that achieves a given performance target (e.g., reducing the miss rate below a given threshold). In the second step, given the number of copies for each object, they optimise the object location to minimise the cost metric. The resulting optimisation framework is fairly complex, and only heuristics can be practically adopted. In [257], random and greedy assignments are compared, showing that in general the greedy alternative is preferable.

Finally, [258] considers the original angle of user mistreatment. Specifically, it starts from the observation that, in an environment where nodes are not under the control of a single authority, optimal policies aiming at the global social optimal (i.e., the optimal performance of the system as a whole) may result in some nodes receiving lower performance than they would if they were greedy. Therefore, authors propose a game theoretical formulation in which no such conditions occur, while the system as a whole significantly outperforms the one in which all nodes are greedy. This formulation is amenable to an efficient distributed implementation, in which nodes just require knowledge about their own local request rates and the content stored by the other nodes of the network.

Caching is another key topic in distributed systems and therefore in P2P systems. However, no particularly original solutions have been proposed with specific reference to P2P environments. Typically, in P2P systems caching techniques like the one proposed in PAST [8] are adopted, in which objects are cached on the path between their location and the requester. A similar concept is also used in FreeNet [225], which however considers an unstructured overlay. Note that the typical caching approach in unstructured overlays is to cache objects at nodes that explicitly request them (also named passive replication). An interesting piece of work on caching in P2P system is presented in [259,260], where authors consider caching of *portions* of objects. This is relevant for P2P systems, as objects might well be very large files, and chopping of objects in chunks is a typical feature of P2P file sharing systems (e.g., BitTorrent). In [259,260], authors show that replacement policies can be more effective if they consider also information related to the *actual range* of the objects that has been served from the node.

As a complement to replication policies, researchers have also investigated consistency solutions for P2P environments. Clearly, replication and consistency management are intertwined topics in distributed systems. Consistency management is an extremely well investigated topic in the caching area. Some of these results have been applied to (structured and unstructured) P2P systems, as well. For example, in [261] several classical schemes are presented: (i) push schemes, consisting in pushing invalidations from objects owners to replica nodes upon changes; (ii) pull schemes, consisting in replica nodes checking objects' validity when needed; (iii) hybrid schemes using both pull and push approaches. The simple application of these schemes to P2P environments sounds – in some cases - somewhat naive. For example, in the case of push schemes for unstructured networks it is assumed that the object owners do not know the set of replica nodes, thus invalidations are broadcast on the whole network. Other works take the specificity of P2P environments in greater account. [262] focuses on systems where replicas of any object are spread on a large number of nodes. Replica nodes of a given object are organised in a hybrid overlay network where ordinary nodes connected to the same supernode are physically close to each other. When the object is updated, a tree is built (on top of the supernodes overlay) connecting together all interested supernodes, and the update is propagated. The idea of using overlay multicast trees is also used in [263].

Finally, the systems described in [264,265] are designed for unstructured networks. [264] defines, between replica nodes, a "virtual chain": each replica nodes knows about a given number of neighbours in the chain, which are contacted in case of an update. On the other hand, [265] uses standard gossiping techniques to propagate updates when needed.

4.5. Delivery abstractions

In this section we discuss a set of research areas that aim at providing further networking abstractions starting from basic P2P overlay networks (see Fig. 4). Such abstractions are typically used for several content delivery P2P applications, such as streaming, Video-on-Demand, file sharing, etc. In this section we discuss multicast abstractions and gossiping systems. A discussion of mesh structures is postponed to Section 4.6, as they are typically designed for a specific class of applications. Solutions to integrate CDN and P2P techniques are also postponed to Section 4.10, as they are included in a more general discussion about complementarity and co-existence of these approaches.

4.5.1. P2P multicast

P2P multicast solutions belong to the more general class of Application-Level Multicast protocols (ALM) [266]. ALM is an alternative to conventional IP-level (native) multicast, which exploits only end hosts to deliver multicast services. Historically, ALM developed as the deployment of IP-level multicast requires non-trivial modifications to the Internet core that did not keep the pace with the demand for multicast services of the user community.

ALM implements multicast at the application level, and requires plain unicast IP support from the core. End hosts organise themselves in overlay structures that are then used to implement multicast services. Although requiring more bandwidth with respect to IP-level multicast, ALM is widespread as it is extremely quick and easy to deploy.

Historically, two main multicast structures have been used for ALM: trees and meshes. For example, the End-system multicast [267] builds a mesh structure among participating nodes, and then uses a routing protocol (on top of the mesh) to deliver content to multicast members. The problem with ALM mesh structures is that they are prone to content duplication, and require additional routing logic besides the logic to build the mesh. Therefore, to the best of our knowledge, P2P multicast solutions have preferentially used tree structures. In addition, they typically target structured overlay networks. Forms of multicast in unstructured networks are achieved through mesh structures, which are however customised for the particular application they support. Therefore we present them in Section 4.6. Hereafter, we present three reference systems for multicast over structured networks, i.e., Scribe [268,269], Bayeux [270] and Borg [271].

Scribe and Bayeux build shared multicast trees (one tree for each multicast group) on top of Pastry and Tapestry, respectively. In both systems a group is identified by an ID. The node responsible for that ID is termed *root* of the group. To subscribe to a group, nodes send a `join` message to the ID of the group. In Scribe, this message also establishes the forwarding state for delivery of messages. In detail, a `join` message is propagated towards the root until a branching point in the multicast tree is encountered (or until the message reaches the root). At each hop (on the overlay) the receiving node enrolls the sending node as one of its children in the tree and forwards the message towards the root (thus implicitly enrolling in the group as well). This way, Scribe builds a tree rooted at the root node, which consists of the union of the paths from the leaf nodes to the root. Messages are always sent to the root first, and are then delivered according to the tree structure built by join operations. The Scribe strategy is known as reverse path multicast (as state is setup on the reverse path between the root and the leaves). On the other hand, Bayeux uses forward path multicast, i.e., state is setup on the forward path from the root to the leaves. To this end, `join` messages are forwarded up to the root without generating any multicast state at intermediate nodes. The root replies to the joining nodes, and this message establishes the multicast state. Therefore, in Bayeux a multicast tree is the union of the paths between the root and the leaf nodes.

In typical DHTs, nodes with close IDs are quite far away from each other on the Internet. It is easy to note that, thus, in reverse path multicast messages traverse shorter and shorter (in terms of path stretch) edges from the root to the leaves, while in forward path multicast messages traverse longer and longer edges from the root to the leaves. As in reverse path multicast nodes close to the root are more likely to be shared by several leaves, reverse path forwarding is in general more efficient.

Borg exploits the different properties of reverse and forward path multicast in a different way. Trees in Borg are actually two-level trees, the leaves of the top tree being the roots of the bottom trees. In the bottom trees Borg uses reverse path multicast, while in the top tree it can use either reverse or forward path multicast, depending which one is less costly to connect any two nodes of the tree.

A common critique to tree-based P2P multicast is that leaf nodes are less loaded than nodes in the core of the tree, which is somewhat at odds with the P2P principles. To this end, several systems try to meet a better balance among the load of the nodes, while still retaining the bottom-line idea of tree-based multicast. For example, SplitStream [272] "splits" the content in k separate streams, and multicasts each stream on a different tree involving the same set of group members. This way, nodes are leaves on some of the k trees and interior nodes in others, thus achieving a better load balance. Other similar approaches are CoopNet [273] and MutualCast [274].

We finally mention another type of P2P multicast approach, i.e., CAN-multicast [275]. This system, built on the CAN overlay network, exploits a very simple idea. A dedicated CAN is built for each multicast group. Then, multicasting corresponds to broadcasting on the CAN corresponding to the multicast group. Although relevant in principle, it has been shown that tree-based multicast typically outperforms this solution [269].

4.5.2. Gossiping

Gossiping is a simple yet effective mechanism with numerous applications in various fields of distributed computing [276]. It builds upon a very simple mechanism to distribute a generic piece of information in a distributed system [277]. Each peer participating in a gossiping process contacts a random number of other peers it is aware of (the number of peers, also known as *fanout*, being a system parameter). In push gossip, the contacted peers receive a piece of information from the contacting peer. In pull gossip, the contacted peers send a piece of information to the contacting peer. The set of peers to contact can be chosen completely at random, or according to some preferential rule [278].

Several components of P2P systems can be built by exploiting this simple mechanism. One of them is data dissemination, the piece of gossiped information being the data to be disseminated [279,280]. It has been shown that, by using a fanout f, the proportion of nodes eventually receiving the data (ρ) satisfies the equation $\rho = 1 - e^{\rho f}$. Gossip-based data dissemination has also been used for implementing multicast services [281].

Gossip mechanisms have also been used as underlying services to build both structured and unstructured networks. Indeed, gossiping can be used to provide nodes with lists of peers that can be used for various reasons. Basically, this corresponds to providing random sampling of peers [282,283]. It has been shown that if nodes know about a limited number of peers, and exchange and merge their lists through gossiping (possibly dropping peers if space is required) they obtain a set of peer which is remarkably close to a random sample from the entire network [284].

Based on such service, arbitrary overlay topologies can be built, both structured and unstructured. The key point is clearly how to select peers as overlay neighbours, starting from the set of peers provided through gossiping. If they are selected with uniform probability, the obtained structure resembles traditional random graphs [276]. Preferential rules can be defined to obtain ring structures [285] or even conventional structured overlays [286].

4.6. P2P applications

In this section we discuss relevant content-centric applications built on top of P2P technologies. Actually those applications typically define their own variant of overlay network and delivery mechanism. Specifically, we focus on file sharing, IPTV streaming, Video-on-Demand, and Voice-over-P2P.

4.6.1. File sharing

File sharing is one of the main applications of P2P technologies. Techniques to build overlay networks, or to implement search and replication described in the previous sections are typically part of file sharing applications such as EMule, KAD, ecc. In this section, we focus on BitTorrent, as (i) it is probably the most popular P2P file sharing system, and (ii) it is based on P2P mesh structures, that we have not covered yet.

BitTorrent implements an unstructured overlay network customized for file sharing. Nodes of the overlay are called peers. Peers both download and upload parts of the shared files (called *chunks*). Therefore, each file is downloaded in parallel from several peers, instead of from a single node as in client/server models, resulting in a higher overall capacity [287]. To download a file a peer needs to firstly download a *torrent* – a very small file – from one of a set of well-known servers. A torrent contains the name of the file's *tracker*, a node that constantly tracks which peers have parts of the file. The tagged peer receives from the tracker a random list of peers, from which it starts to download the file. Peers participating in the download of the same file are collectively called a *swarm*. The set of peers with which a tagged peer is in contact at any point in time is the neighbour set of the tagged peer. This set dynamically changes according to the "Tit-for-Tat" policy. Each peer preferentially selects, for uploading parts of the shared files, those neighbours from which it can download at the highest rate. In addition to that, once every 30 s neighbours are selected completely at random. This allows peers to discover new neighbours, and allow new peers in a swarm to start-up the download. Finally, each peer downloads from neighbours chunks according the Rarest First policy (i.e., chunks which are less spread are downloaded first). It is thus clear that BitTorrent builds an unstructured overlay network in the form of a mesh, which is very customised to the particular application (file sharing) it is designed for. This is a typical example of vertically integrated P2P design patterns.

Due to its extreme popularity, BitTorrent has been extensively analysed, by means of measurements [288–292], simulation [293,294], and analysis [295–297].

In mesh structures like the one realised by BitTorrent there are several degrees of freedom that can be exploited to further optimise its performance. For example, in standard BitTorrent (i) the number of neighbours, (ii) the algorithm to choose the neighbours, and (iii) the algorithm to decide how much bandwidth to contribute can be modified. Researchers have looked at how to exploit such degrees of freedom. One such proposal is BitThief [298]. In BitThief, peers continuously contact the tracker to get more neighbours to connect with. The idea is that out of a large number of neighbours some of them will pick the BitThief peer irrespective of its upload rate. Therefore, a BitThief client may hope to download files without contributing anything (i.e., being a free rider). A greedy (but not free riding) BitTorrent client called BitTyrant is proposed in [299] (a similar solution is also proposed in [300]). The idea of BitTyrant is to fine tune the upload bandwidth of peers so as to achieve the maximum possible download rate. [301] analyses BitTyrant in detail, and shows that (i) the main performance gains of BitTyrant comes for the increased number of neighbours it maintains; (ii) the BitTyrant algorithm actually results in an altruistic behaviour, especially in the initial phase of the content distribution; (iii) the advantages of using BitTyrant disappear as the fraction of peers using it increases. In case of a widespread adoption, actually vanilla BitTorrent is a better option.

The problem of selfishness in BitTorrent is widely analysed. We specifically mention the results in [302]. It is shown, by modelling BitTorrent neighbour selection as a game, that selfish nodes can actually "reap" significant resources in a network of mostly well-behaving peers. Furthermore, when the vast majority of peers are selfish, also non-selfish nodes can gain, as – on average - the peer selection strategy is already extremely optimised. Other works looking at the selfishness problem are [303–305].

BitMax [306] uses a totally different approach with respect to BitTyrant. BitMax is designed for P2P cooperative environments, in which, e.g., peers are set-top-boxes under the control of a unique operator [307]. In this case, peers can be assumed not to have selfish behaviour, and strategies can be defined to maximise the overall social benefit, i.e., the overall download capacity. To this end, a BitMax peer tries to saturate its uplink capacity by maximising the upload rate towards a few neighbours at any time. The idea is that if chunks are pushed quickly to fast downloading neighbours, they will then become quickly available for others to download too. Other strategies for modified neighbour selection with respect to plain BitTorrent have been proposed in [308,309].

4.6.2. P2P IPTV and P2P Video-on-Demand

P2P technologies for streaming TV content over the Internet has recently gained a lot of attention. The main approaches can be broadly classified based on the multicast abstractions they use, in tree-based push or mesh-based pull systems. Examples of the former class are, for example, [310–312]. The majority of proposals actually fall in the mesh category, and builds networks conceptually similar to BitTorrent. Examples of mesh-based solutions are presented in [313–323]. Given their popularity, mesh-based approaches have been extensively studied either through measurements [156,324], simulation [320] or experiments [318,319]. Furthermore, significant effort has been put in modelling their behaviour and performance (e.g. [325–333]). Moreover, several commercial solutions have also been deployed (PPLive [155], Coolstreaming [313,314], PPStream [334], UUSee [335], SopCast [336], TVAnts [337], VVSky [338]). The most popular cases are probably PPLive [155] and CoolStreaming [313,314].

P2P IPTV solutions come as an alternative to native IP streaming, by leveraging the P2P paradigm. The idea is that clients not only download the stream they are interested into, but also cooperatively contribute it to other clients. To this end, streams are divided in chunks and each peer downloads missing chunks from other peers. More specifically, a typical P2P IPTV system includes the concept of a tracker, that provides to peers the set of available streaming channels, as well as initial sets of peers for any channel. Starting from this information, peers get in touch with each other and exchange chunks.

Although conceptually similar to BitTorrent, P2P IPTV systems are technically different due to the particular features of the target application [156]. First of all, in BitTorrent file sharing, chunks can be received in any order, as files are used by the user once they are complete. In IPTV systems, chunks must be played in a precise order, and therefore chunk scheduling is necessary. To this end, peers keep a sliding window covering chunks that have been recently played and chunks that should be played in the near feature. A bitmap is associated to the window, showing which chunks are available at the peer. Neighbours exchange these bitmaps, and decide what to fetch from each other, usually according to a rarest chunk first policy. The second difference with respect to BitTorrent networks is the absence of a tit-for-tat policy. The fact that chunks should be played in a precise order gives less freedom from this standpoint, as sometimes chunks must be transferred in order to guarantee smooth playback irrespective of the cooperation level of the peers. Finally, IPTV systems are designed for much larger scales with respect to BitTorrent. Therefore, it is not feasible to store at trackers a full list of peers of a given channel. Trackers store only partial lists, while gossiping protocols (see Section 4.5.2) are implemented between peers to continuously exchange lists of peers.

Although IPTV P2P systems are already very popular, several open issues still exist [156]. It has been shown that when a peer joins a channel, it typically needs some time for the stream to start. This is clearly at odds with the typical behaviour of TV users, who may require fast switching between channels. It has also been shown that the uplink bandwidth required to peers is not well balanced, and some peers must contribute a lot more bandwidth for the system to work properly. It is questionable to what extent this is sustainable, and it is suggested that some dedicated proxy peers with plenty of bandwidth be installed in the network. Finally, IPTV

systems typically yield large time lags in the playback across different users. This is also something to improve.

P2P Video-on-Demand (VoD) systems (such as GridCast [339]) provide to users stored real-time content on demand (e.g., movies). VoD might seem very similar to IPTV, and actually several architectural components are shared. For example, both types of services use mesh structures for delivery, trackers, gossiping for peer discovery, and rarest-first chunk selection policies [340]. The major difference between VoD and IPTV is that in VoD applications users might want to use "VCR-like" features, such as jumping from one point of the stream to another, rewind and fast-forward, and are not synchronised on a (theoretically) simultaneous playback schedule. This difference results in several technical differences. In P2P-VoD systems, peers are required to contribute some significant shared portion of their memory (typically, in the order of 1 GB), and replication strategies for the available content is required. Peers might pre-fetch parts of the content they are watching, but also store content they are not interested into for the sake of cooperation. Replacement policies must be put in place. Standard caching techniques are used (such as LRU and LFU), as well as utility based policies. Note that, typically, evictions are decided at the whole content granularity, not at a chunk granularity. A complete overview of the architectural blocks of typical P2P-VoD systems is presented in [340].

The performance of a VoD system, with specific reference to the scheduling policy, is analysed in [341]. This work proposes a two-tiered transfer policy for chunks. Videos are assumed to be chopped in segments, each segment consisting of several blocks. The performance of the VoD system is analysed with respect to a range of possible transfer scheduling policies. The main outcome of this work is the fact that network coding can help a lot. Specifically, the best performing system is one in which blocks are transferred by means of network coding techniques, while segments are scheduled essentially according to a rarest-first policy. It is also observed that network coding cannot be applied to segments, as the mixing it creates hinders smooth playback.

Moreover, we mention the results presented in [342], which is a reference work for the European NanoDataCenters (NADA) project [343]. The key idea of NADA is to design a P2P-VoD system in which peers are users set-top-boxes operated by a unique operator. In this case peers can be assumed to be completely cooperative, rather predictable, and essentially out of the user's control (the user can just turn it on and off, and watch programs). Therefore, several policies can be investigated [342] to proactively push content to set-top-boxes, so that users are very likely to find the content they need on set-top-boxes nearby.

The IETF Peer to Peer Streaming Protocol (PPSP) [344] also focuses on P2P streaming systems. In the PPSP model, nodes can be divided in "peers" and "trackers". Peers are the nodes that send and receive the streamed content, while trackers are well-known nodes, typically with stable connectivity, which maintain meta information about the status of the streams and the peers that take part to it (note the similarity with the BitTorrent architecture). The PPSP WG aims at designing signaling and control protocols both between trackers and peers, and between peers.

4.6.3. P2P Voice-over-IP

The last class of applications we consider is Voice-over-IP based on P2P technologies (P2P-VoIP). This represents an extremely viable alternative to conventional VoIP. In conventional VoIP a telco operator sets up a VoIP service, and charges users for the service. Standard infrastructure deployment is needed. In P2P-VoIP clients just need Internet connectivity, and collectively implement the typical services provided by operators in conventional VoIP. Peers can place VoIP call for free, as long as both endpoints are connected to the Internet. Clearly, the most popular example of P2P-VoIP service is Skype [345].

Although Skype technical details are not fully disclosed, several researchers performed reverse-engineering exercises to understand its architecture and study its performance [157,346–348]. Skype implements a hierarchical overlay network. Each peer having sufficient resources can be promoted as a supernode. A peer joining the Skype network has to connect to a supernode. To this end, each peer keeps a Host Cache, i.e. a list of possible supernodes to try to connect upon joining the network. After the peer has established one such connection, it authenticates with a central server (the only centralised entity in Skype). Information about Skype clients is kept in the overlay network, and overlay search is used to look for clients starting from their skype id. Calls are supported through a TCP signalling connections and UDP transfers for voice packets. In case both endpoints have public IP addresses, a direct flow is established between the caller and the callee. Otherwise, in case either the caller or the callee (or both) are behind a NAT firewall, a third Skype peer (having a public IP address) acts as an application-level proxy for the voice call.

An important topic related to P2P-VoIP is how to design SIP protocols in this decentralised environment where central control is absent. This is the focus of the IETF Peer-to-Peer Session Initiation Protocol (P2PSIP) Working Group [349]. The main goal is providing a service to discover available resources. To this end, "P2PSIP peers" build an overlay network, where information about resource availability and location can be stored. Only a subset of nodes have to be peers, while "P2PSIP clients" access the overlay network to locate needed resources.

4.7. Mobility in P2P

All P2P systems described so far require to build and maintain some type of overlay structure. Even unstructured networks and meshes require that peers establish and maintain overlay links with their neighbours. Mobility of peers is therefore a great challenge for the stability and performance of P2P systems. In a conventional TCP/IP stack, whenever a peer moves, the IP configuration might change. This usually means that all TCP connections have to be torn down and re-opened again. Often, this also means that the ID of the peer in the overlay structures changes (as usually the ID is obtained as a function of the IP address), as if it were a completely different and new peer.

One of the fields, in the mobile networking area, where P2P technologies seem directly applicable is Mobile Ad hoc Networks (MANETs) [350]. MANETs are completely self-organised networks built by users devices only without any infrastructure support. Nodes run full TCP/IP stacks, and cooperatively provide to each other the routing services usually provided by core Internet routers. Conceptually, such a decentralised and "anarchical" environment is perfect for P2P solutions. However, although this is certainly true in principle, things are quite different in practice. Wireless links between nodes are typically not as stable as wired links. This means that nodes may appear and disappear quite often in such networks, just because the conditions of the wireless channel fluctuates, and the ad hoc routing protocols need some time to reconfigure. At the P2P level, these events appear as *churns* that are handled through standard overlay structure mechanisms. This results in a "wind-up" effect that may lead the ad hoc network to an excessive congestion and even to collapse: The more the wireless links fluctuate, the more the peers appear to churn, the more maintenance traffic is generated. This increases the load on the network, that contributes to make links even less stable, thus creating even more churns and so on. Churn problems originating from mobility have been investigated in [351–353]. Furthermore, these aspects have been widely investigated in the European

MobileMAN project [354]. Specifically, it has been confirmed [355–357] that current P2P technologies may be adopted in ad hoc networks of limited scale only (a few tens of devices), and in quasi-static configurations.

Research on P2P for mobile networks is thus a very challenging and open area. In the following, we will specifically discuss proposals for optimising P2P networks, P2P meshes (specifically, BitTorrent-like meshes), P2P multicast and gossiping systems in MANET environments. To the best of our knowledge, these are the topics that have received most attention in the mobile networking area.

4.7.1. Mobile P2P overlays

The bottomline approach of P2P overlay networks for mobile environments is providing lighter solutions, with respect to the traditional P2P network design, which require less bandwidth for maintenance and operation, and that avoid the "wind-up" effect discussed above in case of instability.

A first cluster of solutions use a sort of hierarchical design that separates mobile nodes from conventional static peers connected to the Internet. [358] uses stealth and service nodes. Service nodes run a conventional P2P overlay, while stealth nodes neither participate to the routing nor to the maintenance features of the P2P network. They simply register with the service node responsible for their ID, which keeps track of their location, and acts as a sort of proxy on their behalf. In Bristle [359] static nodes run a P2P network that is used to route messages between mobile nodes. When a mobile node needs to communicate with a peer, it sends the message on the overlay through the static peer responsible for its ID (similar to the stealth mechanism). That peer forwards the message on the overlay to the peer responsible for the destination ID. Finally, Warp [360] adopts a similar approach and runs a location service for mobile nodes on the static P2P overlay, but allows groups of nodes moving together (mobile crowds) to update their location at once.

The above solutions all assume that a conventional P2P network can be run by exploiting static nodes. The following set of proposals, instead, design P2P overlay systems for fully mobile networks. The common treat of them is exploiting the cross-layer paradigm [361]. Cross-layer designs break, in a controlled way, the rigid separation between layers of the conventional OSI (and Internet) stack. In case of wireless networks this can result in significant performance gains. Several cross-layer architectural solutions exist. The proposal in [361] is to augment conventional stacks with a Network Status vertical layer (NeSt). Any protocol, through a well-defined interface, can publish and subscribe to information in the NeSt. Thanks to this, information collected at a particular layer can be exploited by protocols running at other layers to have a better understanding of the current network conditions, and therefore optimise their operation. Architectures like the one proposed in [361] allow mobile network protocols to gain from cross-layer interactions, without incurring in the maintenance problems of spaghetti-like architectures, which have been highlighted in [362].

With specific reference to overlay construction and maintenance, this paradigm has been exploited in CrossROADS, Ekta, VRR, XL-Gnutella and a few others. CrossROADS [363,364] implements a standard Pastry-like interface (see Section 4.2), through a design optimised for mobile networks. CrossROADS requires a proactive routing protocol running at the network level below. Periodic link state advertisements are used to convey also information about the fact that the node issuing the advertisement is actually a Pastry node with a given ID. Essentially, CrossROADS relies on the L3-routing process to build, at the same time, the overlay. Furthermore, the convergence of the routing protocol guarantees that all peers become aware of each other. Therefore, in CrossROADS paths at the overlay level are always one hop away. This of course is paid in terms of additional status at each node. However, as flat MANETs are not a suitable technology for large-scale wireless networks, this is not a significant drawback. CrossROADS has shown to significantly outperform Pastry in terms of network overhead [356,355].

Similar to CrossROADS, also Ekta [365] exploits the concept of building the overlay network through L3-routing traffic. Unlike CrossROADS, Ekta relies on reactive routing (DSR in particular). It maintains standard Pastry structures, by exploiting overheard information about DSR route requests. When a message needs to be sent along an overlay logical hop, a DSR source route is generated looking for the best destination (longest prefix match).

The original trait of Virtual Ring Routing [366] is to completely forget about the distinction between L3 and overlay routing. VRR provides a Pastry-like routing structure. In VRR nodes do not have an IP address, but are identified directly (i.e., above L2) by their virtual ID. Each node in VRR maintains information about a physical set *pset* and a virtual set *vset* of neighbours. With respect to a tagged node, peers in the pset are physically one-hop away, and are kept in the pset as long as the physical link is stable enough. Peers in the vset are virtual neighbours, and can be connected to the node through a physical multi-hop path. Paths towards vset neighbours are kept proactively, while each node also stores information about vset paths that happen to pass through that particular node. By appropriately selecting nodes in the vset, routing coherence is guaranteed (as it is in Pastry). Other approaches conceptually similar to VRR have been proposed, e.g. [367–369].

Finally, we describe an example of cross-layer unstructured overlay network, i.e., XL-Gnutella [370]. XL-Gnutella exploits the same bottomline idea of CrossROADS. It is also based on a proactive routing protocol. By simply piggybacking their availability to be peers in a Gnutella network in L3 link state advertisements, nodes in XL-Gnutella become aware of possible neighbours, and can establish and maintain a Gnutella network without any additional overhead. [370] shows that in MANET environments this approach results in much more stable and efficient unstructured overlays. Other proposals exploiting similar concepts are presented in [371–373].

4.7.2. Delivery abstractions for wireless environments

In this section we consider modifications to the delivery abstractions presented in Section 4.5 that consider the specificity of mobile and wireless environments. Specifically, we focus on BitTorrent-like meshes, multicasting and gossiping.

A few works have looked at the issues related to establish mesh-like structures (similar to that of BitTorrent) in wireless networks. Overall, the common remark is that the neighbour selection process in BitTorrent should be modified, in a wireless network, taking into consideration the properties of the physical links. Using a completely random selection policy results in peer possibly selecting far away nodes (in terms of physical hops), which leads to low throughput and download performance.

Nandan [374] focuses on a vehicular networking environment, and designs a content sharing system. Content is divided into chunks as in standard BitTorrent. The authors assume the presence of static gateways to which nodes (cars) can connect on passing by. A new peer joining the network fetches, from the first gateway it encounters, the pieces of the content it is interested into stored at the gateway, as well as a list of other peers interested in the same content. As centralised trackers are appropriate in this environment, additional peers are discovered through gossip mechanisms. Peers to connect to are then chosen based on the distance, in number of hops, while pieces to download are chosen based on both the rarity in the neighbourhood and the distance at which they are located.

In [375] authors propose a set of modifications to the BitTorrent protocol in order to adapt it to static MANETs (mesh networks). Also in this case, they observe that neighbours should be selected according to proximity metrics. More specifically, each node establishes connections with a maximum number of neighbours. To find them, an expanding ring technique is used, with a limited maximum TTL. Peers exchange a bitmap showing which chunks of the downloaded content they possess locally. Chunk selection is performed according to the standard rarest-first policy. Finally, a tit-for-tat strategy is used to decide whether to serve or not requests coming from neighbours.

Sbai [376] proposes a more advanced neighbour selection policy. Also in this case, authors start by analysing the standard BitTorrent neighbour selection algorithm in static MANETs. They confirm that the fact that neighbours are selected completely at random results in long multi-hop paths, which in turn causes severe degradation of the download performance. However, they also notice that selecting neighbours only in the physical proximity may reduce the mixing of chunks in the whole network. Therefore, they propose a selection strategy that picks most of the neighbours within a 2-hop neighbourhood, but picks also some far away peers to increase chunks mixing.

Multicasting for MANETs has been mainly addressed at L3 or at the application layer, but without relying on P2P paradigms. Examples of multicast at L3 are MAODV [377] and ODMRP [378], while application-level multicast has been proposed e.g., in ALMA [379] and PAST-DM [380]. It should be noted that the difference between L3 and application-level multicast is not evident in ad hoc networks as it is in the static Internet, as in MANETs user nodes play also the role of core routers, and therefore modification to the "core" is not costly as in the case of the conventional Internet.

One of the few examples of P2P multicast solutions for ad hoc networks is XScribe [381,382]. XScribe is a cross-layer extension of Scribe (see Section 4.5.1). It is built on top of CrossROADS, and exploits L3 link state advertisements to disseminate information about multicast group membership. Thanks to this, all members of a multicast group are aware of each other, and there is no need to maintain a tree structure like in the original Scribe design (similar to CrossROADS, XScribe exploits the fact that MANETs are not likely to be of very large size). Therefore, when a member wants to send a message to the multicast group, it simply sends it to all the members. In [381,382] it is shown through an experimental testbed that XScribe significantly reduces message loss rate and delays with respect to standard Scribe in MANETs.

We finally briefly mention some relevant proposals for using gossiping schemes in MANET environments. As described in Section 4.5.2, gossiping just requires that each node is aware of a set of peers to communicate with. This idea is particularly suitable for ad hoc networks, as partial views of the network are commonplace. Indeed, gossiping mechanisms have been used for implementing routing [383], multicast [384] and data dissemination [385] systems for ad hoc networks. It is interesting to note that gossiping schemes can be seen actually at the basis of several communication mechanisms in an emerging mobile networking paradigms called opportunistic networking [6]. Opportunistic networks are a form of MANETs in which no simultaneous end-to-end path is assumed between nodes. Network partitions and nodes disconnections are seen as a rule rather than an exception, and no attempt is done to keep an updated topological view of the network as in MANET routing protocols. Messages are opportunistically transferred from one node to the next one, if the latter is deemed more suitable to bring it to the eventual destination. The action of moving messages opportunistically from one node to the other resembles quite closely the spread of an epidemic or a gossiping protocol. Indeed, several protocols for routing (e.g. [386–388]) and disseminating content (e.g. [389–391]) in opportunistic networks can be seen as instances of the gossiping scheme.

4.8. Security issues in P2P

For completeness, in this section we briefly discuss issues related to security in a broad sense (i.e., security, trust, privacy, cooperation). The interested reader is referred to [160,392,393,202] for more detailed surveys.

The authors of [202] (see Chapter 14) highlight several key security issues and attacks in P2P networks. *File leakage* consists of masking access-restricted files in popular formats (such as MP3) and disseminating them through P2P networks. Wrapster [394] is a tool that can be used for this purpose. Another well known problem is bandwidth clogging, consisting in the fact that sharing of large multimedia files might clog bandwidth resources of public and private institutions. Clearly, Denial of Service attacks targeted at saturating resources in terms of bandwidth, memory, storage are possible in P2P environments. In addition to classic DoS attacks, *distributed DoS* are possible, in which a set of peers cooperate to produce a DoS attack [395–397]. In addition, other types of attacks can be produced, oriented towards the specific features of overlay structures. In *NodeID attacks* [398] malicious nodes obtain a specific ID in the DHT, through which they control a large share of important objects. In *sybil attacks* [399] malicious nodes try to obtain a large number of distinct IDs and then act as multiple nodes. In *routing table attacks* overlay routing tables are manipulated for malicious intent. *Forwarding attacks* are also possible, in which malicious nodes compromise the correct behaviour of the forwarding process [398,400]. Mechanisms to counteract these attacks partly rely on standard security techniques, such as standard DoS avoidance measures or cryptographic systems. In other cases, specific mechanisms are designed, e.g., to guarantee secure nodeID assignment, secure routing table maintenance, secure message forwarding [398].

Another important facet of security for P2P networks is clearly related to trust, and particularly to establishment of reputation. Typically, establishing reputation of peers requires collecting information about previous interactions. [401] distinguishes between probabilistic estimation and social-based estimation. In the former case [402,403], reputation is established based on a restricted sample of the complete information about previous interactions. In the latter case [401], aggregates of the whole available information are used to establish trustworthiness.

Finally, cooperation enforcement are other key topics related to security in P2P environments. The tit-for-tat policy of BitTorrent is one popular example of cooperation enforcement. Other examples are micropayment [404] and incentive systems [405].

4.9. P2P and ISP traffic imbalance

To conclude our presentation of P2P technologies for a content-centric Internet, we highlight an important aspect related to the impact of P2P technologies on ISP policies. P2P solutions are most of the time network agnostic, in the sense that they do not take in great consideration the physical characteristics of the network paths they generate. Having agnostic P2P solutions can cause problems at different levels to ISPs. It has been shown [406,407] that "blind" selections of neighbours in P2P networks may result in unnecessary traversal of multiple links inside an ISP. Furthermore, it can significantly impact on the shape and amount of traffic among different ISPs, which may result being quite different from what foreseen in ISPs peering agreements (with non negligible economic impact).

Conventional approaches to face these problems work at either the network or at the P2P levels. Network-level solutions include detailed forecasting of P2P traffic and to determine routing polices based on this. However, this can be completely disrupted by the fact that P2P structures may adapt themselves in a different way from what foreseen by ISPs. This can even cause oscillations and lead to instability [406]. Otherwise, it has been proposed to place caches in ISP networks to reduce cross-ISP traffic between P2P peers [408,407]. However, caching poses additional issues, such as ISP liability for possibly caching illegally shared content, additional ISP costs of deploying and maintaining caches, and the fact that caches are usually optimised for specific applications, rather than for P2P services in general. Finally, more drastic solutions include rate-limiting P2P peers (e.g., [409]), which is however very badly perceived by ISP subscribers. On the other hand, solutions working exclusively at the P2P level try to optimise the overlay structures according to information about peers physical proximity [407] (this is a classic extension that DHT algorithms incorporate [14]). However, this information has to be inferred e.g., through monitoring, and can thus be costly and less precise than what available already at the network level.

Recent solutions to address this problem advocate a sort of cross-layer interaction between ISPs and P2P applications. Aggarwal [410] assumes that P2P peers can exploit an oracle on selecting neighbours. The oracle is implemented by the ISP, and ranks possible neighbours of any given peer according, e.g., to the cost associated to connecting to them. P2P applications can thus exploit the oracle to jointly optimise the application performance and the cost on the ISP network. A similar approach is also used in the P4P architecture [406], in which however the concept of oracle is extended with an architectural perspective. P2P applications and ISPs are supposed to communicate through a `p4p-portal` interface, which provides information about the costs associated to different possible neighbours. This can be used to build more efficient P2P structures. Finally, [411] proposes to exploit information that can be gathered by CDNmeasurements. The bottomline idea is that if two possible neighbours are redirected by a CDN service to the same replica, then it is very likely that they are physically close to each other. By exploiting this idea, authors show that 33% of the times their CDN-biased selection leads peers to pick neighbours within the same autonomous system, which result in less traffic overhead on the ISP network.

4.10. Complementarity and integration of CDN and P2P solutions

Being the most successful solutions for a content-centric Internet, it is natural to ask whether CDNs and P2P are complementary or alternative, and which one is preferable. With respect to the CDN approach, P2P technologies do not require any service operator, but run efficiently as soon as a critical mass of users join the P2P service. Being virtually free, they represent an alternative for the vast majority of users that cannot afford contracts with a CDN operator. Legal aspects related to the distribution of copyrighted content is clearly an issue, and P2P technologies are still largely used to illegally exchange content. However, technical solutions are being investigated to implement digital rights management (DRM) systems suitable for P2P environments (see, e.g., Chapter 1 of [202]). For example, content can be distributed according to a conventional P2P service, but users need to obtain a licence from the copyright owner to be able to use the content. Besides these advantages, as highlighted in [3], P2P technologies still present some issues that may make them less suitable than CDNs in particular cases. For example, it is much more difficult in P2P systems to provide service guarantees. Enforcing cooperation among peers is an issue unless for some particular configuration [343,342], as there is usually no central controller. The mismatch between P2P traffic patterns and ISP peering agreements might be an issue, as discussed in Section 4.9, and pricing schemes are more difficult to define. Based on these remarks, it is possible to conclude, as argued in [3], that at this stage CDNs and P2P technologies are complementary: CDNs can provide robust delivery of content with service guarantees, while P2P technologies can provide cheap, flexible delivery of content with *statistical* guarantees, and are very suitable to support participatory content generation and sharing.

Moreover, it is interesting to note that integration of CDN and P2P techniques has also been proposed, aiming to breed the best of the two worlds.

A first example is proposed in [412]. The scenario is that of a CDN, with clients issuing requests for Web content. A P2P network is established out of Web caches, that is used to improve the performance perceived by clients. Specifically, upon a client request, the Web cache close to the client (e.g., a proxy on the client LAN) looks up in the P2P network for the requested content. Edge replica nodes of the CDN also participate to the P2P network, and fetch content from the CDN if it is not available in the P2P network. This solution clearly reduces the load of the CDN, and is one of the few proposals exploring a very interesting direction, i.e., moving P2P technologies towards Internet nodes other than edge client machines.

Xu et al. [413,414] design a content distribution system that exploits a P2P network or a CDN depending on the popularity of the content. As soon as a new object is generated it is not diffused enough, and therefore it is published on the CDN. Clients requesting content are supposed to run a P2P network on which objects are replicated. The more they become replicated, the less the advantage of serving them from within the CDN. As a given object becomes popular enough (i.e., enough spread in the P2P network), it is removed from the CDN.

The main goal of [415] is to exploit a hybrid P2P/CDN architecture to reduce the number of points of presence a CDN operator should deploy to achieve a given performance target. The idea is that a P2P network is run on nodes residing under the same ISP. These P2P networks complement the CDN, as clients can download objects also from other clients in the same ISP. Note that confining P2P traffic within the same ISP also tries to address the well-known problem of the imbalance between P2P traffic and ISPs peering agreement, that we will discuss in Section 4.9. A similar solution is also proposed in [416].

Finally, a hybrid P2P/CDN solution for a very specific application has been proposed in [417]. The reference application is Video-on-Demand, and the hybrid architecture is used to assist new clients to catch up with a broadcast stream as soon as they join it. Specifically, they assume that streams are broadcast using CDN services. A P2P architecture is used to enable clients joining at a random point in time to quickly fetch the content previously streamed on the broadcast channel. To this end, clients form a P2P network, and a new joining client fetches the content broadcast up to that point in time from clients through a BitTorrent-like protocol.

5. An outlook on content-centric evolution of the Internet

We have highlighted in Section 2 that CDNs and P2P solutions are the most important technologies available today to make the Internet work as a content-centric network. Very much aligned with the Internet spirit, the evolution of the Internet towards being a content-centric network mostly happened in a "creative disorganised" way. Content-centric services have been built on top of the standard Internet as the need arose, without any well-identified plan of architectural evolution towards content-centric services.

It is interesting to note that all the mechanisms required by CDNs and P2P technologies have been implemented at the application level, while the Internet core has not been modified at all. Recently, researchers started discussing whether this is still affordable, or if content-centric support should be at least in part become a standard feature of the Internet core [2,418]. This is actually one of the key aspects of a broader debate on the subject of the Internet *ossification* [13]. The Internet ossification is widely recognised as one of the key reasons behind the lack of evolution of the core Internet architecture in general, and towards content-centric networking in particular. [13] highlights that no significant modifications to the core Internet architecture (at layers 3 and 4) has happened since early 90s. Even important research areas such as IP multicast [419], QoS [420,421] and IP mobility [422] did not result in significant modifications of the Internet core architecture. This can be seen as a by-product of the Internet "narrow waist" design principle. According to it, the Internet permits large differentiation above and below the TCP/IP level (which represents the narrow waist), but does not permit any modification in the waist itself. This design approach permitted to integrate in the Internet numerous physical- and MAC-layer technologies, as well as to support significant innovations at the middleware and application levels. Modifications below and above the waist occurred in an incremental way, without requiring global coordination to incorporate them. This is a great advantage of such a narrow-waist design. Ossification is the drawback. Modifying features implemented in the waist is very difficult, as it often requires coordination at large, and results in non-negligible costs for ISPs. ISPs are typically reluctant to switch to new features before their effectivness is convincingly proved. However, their effectiveness usually cannot be proved before a reasonably large scale deployment is achieved and experimentation performed, which would require large adoption by ISPs. It is clear that this status favours the development of innovative solutions either as closed, proprietary solutions (like CDNs), or as technologies implemented at the edge of the Internet (like P2P overlays), but seldom permits innovation of the core Internet architecture.

Despite the clear ossification of the Internet architecture, researchers are wondering if this might be the right time to "enlarge" the narrow waist of the Internet. Work in this direction is undertaken by several projects both in Europe (e.g., ANA [423]), in the US (e.g., [424]) and in Asia (e.g., [425]). Content-centric networking could play a key role from this standpoint, as it may have the critical mass required to impact on the core architecture of the Future Internet [426,3,418,427,2,428–430]. Modifications of the core Internet architecture have often been "imposed" by compelling needs that resulted from scalability limitations of the previously adopted solutions. Examples of such modifications are the transition from the `hosts.txt` file [431] to the DNS system for name resolution, the adoption of BGP [432], and congestion control [433]. All of them have been "last-minute" modifications in response to the risk that the original architecture collapsed due to scalability issues. Content-centric networking might be a similar driving force for even drastic modifications of the Future Internet architecture. For example, the explosion of User-Generated Content [434], Online Social Networks [435,436], as well as the penetration of smart phones and other resource-rich mobile devices are likely to result in an unprecedented amount of content generated and shared by users, which might pose severe scalability challenges to the content-centric services currently adopted in the Internet. The answer to this might be incorporating some of the features provided today by CDNs and P2P systems in the core of the Internet. Conceptually, modifying the narrow-waist Internet design paradigm breaks one of the key features of the original Internet, i.e. having fast "dumb" pipes in the core and pushing all the intelligence on end devices. It should however be noted that this principle is often already bypassed in the current Internet. The wide use of CDNs is a typical example. CDNs provide a de facto content centric infrastructure on top of the conventional Internet. From the standpoint of end devices, CDNs functions could as well be implemented as core Internet functions. The importance of CDNs and P2P systems both from an academic and industrial standpoint, show that these additional features, with respect to the standard Internet services, are needed by the vast majority of applications. This is a strong motivation to identify a subset of such functions as candidates to be included in the core of the Internet. Moreover, the wide diffusion of mobile devices with content generation capabilities also calls for an extended set of Internet core services, with respect to the original "dumb" pipes. Mobile devices cannot be assumed to be always connected with high-bandwidth links as fixed PCs. Therefore, in a network where a very large fraction of the nodes is mobile, systems like conventional P2P overlays cannot be used anymore. Content generated by mobile devices need to be available in the network despite mobile nodes disconnections or poor connectivity. This also calls for functions such as caching and replication to be part of the "standard" network support in the Future Internet.

Actually, there are already significant results pointing in this direction such as the DONA architecture developed at Berkeley [429], the clean-slate research program at Stanford [437], the Content Centric Networking (CCN) project at PARC [438]. The main focus of the DONA architecture is re-defining the naming architecture of the current Internet, such that the Internet *core* can map data names to the location(s) where they are stored. To this end, DONA defines a pub/sub-like architecture, whose API mainly consists of two primitives: `REGISTER` and `FIND`. The `REGISTER` primitive is used by content producers to advertise the availability of a piece of content, and installs in the core the state needed for resolving names at a later time. The `FIND` primitive exploits the state installed by a `REGISTER` to locate the required content, which is transferred according to a standard transport connection. To implement these primitives, DONA defines Internet core devices (Resolution Handlers), which exploit the routing state to forward `FIND` packets. Note that Resolution Handlers play the role, in the DONA content-centric network, of IP routers in the conventional Internet. The DONA architecture is able to provide three fundamental properties for content-centric networking, i.e. persistence (pieces of content should remain available irrespective of possible physical relocations), availability (pieces of content should remain available irrespective of possible failures of the physical locations where they are stored), and authenticity (users should be sure that the pieces of content they receive actually come from the legitimate source). CCN [438] shares similar goals with respect to DONA, as it also looks at how to modify the Internet core to directly support content-centric access. Unlike DONA, CCN fully exploits conventional Internet routing machineries to advertise content. Content availability is broadcast in routing advertisement to all CCN "routers", which can then build shortest-hop routes to the content (as conventional IP routers do for hosts). CCN content request packets can then be forwarded to the correct place by using a mechanism very similar to standard IP forwarding. While forwarded, such packets also establish the reverse path that content will take to reach the requesting user. Furthermore, pieces of content are temporarily stored at CCN routers, which thus naturally implement in-network caching.

DONA and CCN are the two most important examples of new architectures proposing a redesign of the Internet core towards native content-centric networking. While this would have clear advantages, there are also several risks. For example, it is still unclear whether the core devices could be fast and scalable enough to efficiently support the required content-centric features. Also, there are several concerns related to network neutrality issues

and possible control of ISPs over the content available in the Internet. However, the advantages would also be significant. Custom solutions provided today as proprietary technologies or implemented in monolithic and not re-usable designs might become standard, open and reusable features of the Internet, pretty much like TCP/IP services are today. Furthermore, content centric services might become straightforward to implement and provide, without forcing intricacies such as the ones implemented – though efficiently – by CDNs. The debate on these issues is still totally open.

References

[1] P. Baran, On distributed communications networks, IEEE transactions on Communications Systems 12 (1) (1964) 1–9, doi:10.1109/TCOM.1964.1088883.

[2] The Second COST-NSF Workshop on Future Internet (NeXtworking '07) (19–20 April 2007, <http://www.net.t-labs.tu-berlin.de/NeXtworking/>).

[3] P. Rodriguez, S.-M. Tan, C. Gkantsidis, On the feasibility of commercial, legal p2p content distribution, SIGCOMM Computer Communications Review 36 (1) (2006) 75–78. http://dx.doi.org/10.1145/1111322.1111339.

[4] The European Commission FIRE (Future Internet Research and Experimentation) Initiative, <http://cordis.europa.eu/fp7/ict/fire/>.

[5] NSF Networking Technology and Systems (NeTS), <http://www.nsf.gov/funding/pgm_summ.jsp?pims_id=503307&org=CNS>.

[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] A. Technologies, Secure content delivery, 2003.

[8] A. Rowstron, P. Druschel, Storage management and caching in past, a large-scale, persistent peer-to-peer storage utility, SIGOPS Operating Systems Review 35 (5) (2001) 188–201. <http://doi.acm.org/10.1145/502059.502053>.

[9] S. Iyer, A. Rowstron, P. Druschel, Squirrel: a decentralized peer-to-peer web cache, in: PODC '02: Proceedings of the twenty-first annual symposium on Principles of distributed computing, ACM, New York, NY, USA, 2002, pp. 213–222. http://doi.acm.org/10.1145/571825.571861 .

[10] C. Gkantsidis, P. Rodriguez, Network coding for large scale content distribution, in: Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2005, IEEE, vol. 4, 2005, pp. 2235–2245, doi:10.1109/INFCOM.2005.1498511.

[11] J. Chesterfield, P. Rodriguez, Deltacast: efficient file reconciliation in wireless broadcast systems, in: MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services, ACM, New York, NY, USA, 2005, pp. 93–106. http://doi.acm.org/10.1145/1067170.1067181.

[12] C. Gkantsidis, J. Miller, P. Rodriguez, Anatomy of a p2p content distribution system with network coding, in: Proceedings of the Fifth International Workshop on Peer-to-Peer Systems (IPTPS 2006), 2006.

[13] M. Handley, Why the internet only just works, BT Technology Journal 24 (3) (2006) 119–129. http://dx.doi.org/10.1007/s10550-006-0084-z.

[14] A. Rowstron, P. Druschel, Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems, in: IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), 2001, pp. 329–350.

[15] N. Bartolini, E. Casalicchio, S. Tucci, A walk through content delivery networks, in: LNCS 2965: Performance Tools and Applications to Networked Systems, Springer-Verlag, 2004.

[16] G. Pallis, A. Vakali, Insight and perspectives for content delivery networks, Communications of the ACM 49 (1) (2006) 101–106. http://doi.acm.org/10.1145/1107458.1107462.

[17] M. Pathan, R. Buyya, A Taxonomy of CDNs, First ed., Springer-Verlag, R. Buyya, M. Pathan, A. Vakali (Eds.), 2008, (Chapter 2), <http://dx.doi.org/10.1007/978-3-540-77887-5>.

[18] M. Arlitt, T. Jin, A workload characterization study of the 1998 world cup web site, Network, IEEE 14 (3) (2000) 30–37, doi:10.1109/65.844498.

[19] A. Lipsman, Official FIFA World Cup Web Site Attracts Millions of Viewers And Billions of Page Views from Around the World in June, ComScore Press Release, <http://www.comscore.com/press/release.asp?press=934> (July 2006).

[20] V.N. Padmanabhan, K. Sripanidkulchai, The case for cooperative networking, in: IPTPS '02: Revised Papers from the First International Workshop on Peer-to-Peer Systems, Springer-Verlag, London, UK, 2002, pp. 178–190.

[21] S. Adler, The Slashdot Effect, An Analysis of Three Internet Publications, Linux Gazette 38.

[22] J. Jung, B. Krishnamurthy, M. Rabinovich, Flash crowds and denial of service attacks: characterization and implications for cdns and web sites, in: WWW '02: Proceedings of the 11th International Conference on World Wide Web, ACM, New York, NY, USA, 2002, pp. 293–304. http://doi.acm.org/10.1145/511446.511485.

[23] L. Wang, V. Pai, L. Peterson, The effectiveness of request redirection on cdn robustness, SIGOPS Operating Systems Review 36 (SI) (2002) 345–360. http://doi.acm.org/10.1145/844128.844160.

[24] M. Pathan, R. Buyya, A. Vakali, content delivery networks: state of the art, insights and imperatives, in: Content Delivery Networks, Springer-Verlag, 2008.

[25] Akamai technologies (2009, <http://www.akamai.com>).

[26] Mirror image internet (2009, <http://www.mirror-image.com/>).

[27] Limelight networks (2009, <http://www.limelightnetworks.com/>).

[28] Cisco Content Delivery Systems (2009, <http://www.cisco.com/en/US/products/ps7191/Products_Sub_Category_Home.html>).

[29] M.J. Freedman, E. Freudenthal, D. Mazières, Democratizing content publication with coral, in: NSDI'04: Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation, USENIX Association, Berkeley, CA, USA, 2004, pp. 18–18.

[30] M.J. Freedman, Democratizing content distribution, Ph.D. Thesis, New York, NY, USA, adviser-Mazieres, David, 2007.

[31] V.S. Pai, L. Wang, K. Park, R. Pang, L. Peterson, The dark side of the web: an open proxy's view, SIGCOMM Computer Communications Review 34 (1) (2004) 57–62. http://doi.acm.org/10.1145/972374.972385.

[32] L. Wang, K.S. Park, R. Pang, V. Pai, L. Peterson, Reliability and security in the codeen content distribution network, in: ATEC '04: Proceedings of the Annual Conference on USENIX Annual Technical Conference, USENIX Association, Berkeley, CA, USA, 2004, pp. 14–14.

[33] G. Pierre, M. van Steen, Globule: a collaborative content delivery network, IEEE Communications Magazine 44 (8) (2006) 127–133. http://www.globule.org/publi/GCCDN_commag2006.html.

[34] WinterGreen Research, Inc., Streaming Media: Market Opportunities, Strategies, and Forecasts, 2004 to 2009, Market Research Report, <www.wintergreenresearch.com/reports/Streaming_Media_Brochure.pdf>.

[35] AccuStream iMedia Research, CDN Growth and Market Share Shifts: 2002-2006, Market Research Report, <www.researchandmarkets.com>.

[36] A. Vakali, G. Pallis, Content delivery networks: status and trends, Internet, Computing IEEE 7 (6) (2003) 68–74, doi:10.1109/MIC.2003.1250586.

[37] P.B. Mirchandani, R.L. Francis, Discrete Location Theory, Wiley, 1990.

[38] Y. Bartal, Probabilistic approximation of metric spaces and its algorithmic applications, in: FOCS '96: Proceedings of the 37th Annual Symposium on Foundations of Computer Science, IEEE Computer Society, Washington, DC, USA, 1996, p. 184.

[39] M. Karlsson, C. Karamanolis, Choosing replica placement heuristics for wide-area systems, in: ICDCS '04: Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04), IEEE Computer Society, Washington, DC, USA, 2004, pp. 350–359.

[40] V. Vazirani, Approximation Algorithms, Springer-Verlag, 2001.

[41] S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, L. Zhang, On the placement of internet instrumentation, vol. 1, 2000, pp. 295–304, doi:10.1109/INFCOM.2000.832199.

[42] P. Krishnan, D. Raz, Y. Shavitt, The cache location problem, IEEE/ACM Transactions on Networking 8 (5) (2000) 568–582, doi:10.1109/90.879344.

[43] L. Qiu, V. Padmanabhan, G. Voelker, On the placement of web server replicas 3 (2001) 1587–1596, doi:10.1109/INFCOM.2001.916655.

[44] E. Cronin, S. Jamin, C. Jin, A. Kurc, D. Raz, Y. Shavitt, Constrained mirror placement on the internet, selected areas in communications, IEEE Journal on 20 (7) (2002) 1369–1382, doi:10.1109/JSAC.2002.802066.

[45] Y. Chen, R.H. Katz, J. Kubiatowicz, Dynamic replica placement for scalable content delivery, in: IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems, Springer-Verlag, London, UK, 2002, pp. 306–318.

[46] M. Szymaniak, G. Pierre, M. van Steen, Latency-driven replica placement, in: 2005. Proceedings of the Symposium on Applications and the Internet, 2005, pp. 399–405. doi:10.1109/SAINT.2005.37.

[47] P. Radoslavov, R. Govindan, D. Estrin, Topology-informed internet replica placement, Computer Communications 25 (4) (2002) 384–392.

[48] Y. Chen, L. Qiu, W. Chen, L. Nguyen, R. Katz, Efficient and adaptive web replication using content clustering, IEEE Journal on Selected Areas in Communications 21 (6) (2003) 979–994, doi:10.1109/JSAC.2003.814608.

[49] N. Fujita, Y. Ishikawa, A. Iwata, R. Izmailov, Coarse-grain replica management strategies for dynamic replication of web contents, Computer Networks 45 (1) (2004) 19–34. http://dx.doi.org/10.1016/j.comnet.2004.02.006.

[50] A. Sidiropoulos, G. Pallis, D. Katsaros, K. Stamos, A. Vakali, Y. Manolopoulos, Prefetching in content distribution networks via web communities identification and outsourcing, World Wide Web 11 (1) (2008) 39–70. http://dx.doi.org/10.1007/s11280-007-0027-8.

[51] J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, B. Weihl, Globally distributed content delivery, Internet Computing, IEEE 6 (5) (2002) 50–58, doi:10.1109/MIC.2002.1036038.

[52] M. Freedman, D. Mazières, Sloppy hashing and self-organizing clusters, in: IPTPS '03: Revised Papers from the Second International Workshop on Peer-to-Peer Systems, 2003.

[53] Edge side includes (2009, <http://www.esi.org>).

[54] J. Challenger, P. Dantzig, A. Iyengar, K. Witting, A fragment-based approach for efficiently creating dynamic web content, ACM Transactions Interet Technol. 5 (2) (2005) 359–389. http://doi.acm.org/10.1145/1064340.1064343.

[55] L. Ramaswamy, A. Iyengar, L. Liu, F. Douglis, Automatic fragment detection in dynamic web pages and its impact on caching, IEEE Transactions on Knowledge and Data Engineering 17 (6) (2005) 859–874, doi:10.1109/TKDE.2005.89.

[56] C. Yuan, Y. Chen, Z. Zhang, Evaluation of edge caching/off loading for dynamic content delivery, IEEE Transactions on Knowledge and Data Engineering 16 (11) (2004) 1411–1423, doi:10.1109/TKDE.2004.73.

[57] A. Leff, J. Wolf, P. Yu, Replication algorithms in a remote caching architecture, IEEE Transactions on Parallel and Distributed Systems 4 (11) (1993) 1185–1204, doi:10.1109/71.250099.

[58] I. Cidon, S. Kutten, R. Soffer, Optimal allocation of electronic content, INFOCOM 2001, Proceedings of Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 3, IEEE, 2001, pp. 1773–1780, doi:10.1109/INFCOM.2001.916675.

[59] K. Kalpakis, K. Dasgupta, O. Wolfson, Optimal placement of replicas in trees with read, write, and storage costs, IEEE Transactions on Parallel and Distributed Computing. 12 (6) (2001) 628–637. http://dx.doi.org/10.1109/71.932716.

[60] S.S.H. Tse, Approximate algorithms for document placement in distributed web servers, IEEE Transactions on Parallel and Distributed Computing. 16 (6) (2005) 489–496. http://dx.doi.org/10.1109/TPDS.2005.63.

[61] M.R. Korupolu, C.G. Plaxton, R. Rajaraman, Placement algorithms for hierarchical cooperative caching, in: SODA '99: Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1999, pp. 586–595.

[62] I.D. Baev, R. Rajaraman, Approximation algorithms for data placement in arbitrary networks, in: SODA '01: Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001, pp. 661–670.

[63] T. Loukopoulos, I. Ahmad, Static and adaptive distributed data replication using genetic algorithms, Journal of Parallel and Distributed Computing 64 (11) (2004) 1270–1285. http://dx.doi.org/10.1016/j.jpdc.2004.04.005.

[64] M. Yang, Z. Fei, A model for replica placement in content distribution networks for multimedia applications, in: IEEE International Conference on Communications, 2003. ICC '03, vol. 1, 2003, pp. 557–561, doi:10.1109/ICC.2003.1204238.

[65] H. Yu, A. Vahdat, Minimal replication cost for availability, in: PODC '02: Proceedings of the twenty-first annual symposium on Principles of distributed computing, ACM, New York, NY, USA, 2002, pp. 98–107. http://doi.acm.org/10.1145/571825.571839.

[66] L. Zhuo, C.-L. Wang, F.C.M. Lau, Load balancing in distributed web server systems with partial document replication, in: ICPP '02: Proceedings of the 2002 International Conference on Parallel Processing, IEEE Computer Society, Washington, DC, USA, 2002, p. 305.

[67] B.-J. Ko, D. Rubenstein, Distributed self-stabilizing placement of replicated resources in emerging networks, IEEE/ACM Transactions on Networking 13 (3) (2005) 476–487. http://dx.doi.org/10.1109/TNET.2005.850196.

[68] J. Kangasharju, K.W. Ross, J.W. Roberts, Performance evaluation of redirection schemes in content distribution networks, Computer Communications 24 (2) (2001) 207–214.

[69] L.-C. Chen, H.-A. Choi, Approximation algorithms for data distribution with load balancing of web servers, in: CLUSTER '01: Proceedings of the 3rd IEEE International Conference on Cluster Computing, IEEE Computer Society, Washington, DC, USA, 2001, p. 274.

[70] B. Wu, A. Kshemkalyani, Objective-optimal algorithms for long-term web prefetching, IEEE Transactions on Computers 55 (1) (2006) 2–17, doi:10.1109/TC.2006.12.

[71] Y. Guo, Z. Ge, B. Urgaonkar, P. Shenoy, D. Towsley, Dynamic cache reconfiguration strategies for a cluster-based streaming proxy, Web content caching and distribution: Proceedings of the 8th International Workshop, 2004, 139–157.

[72] G. Pallis, A. Vakali, K. Stamos, A. Sidiropoulos, D. Katsaros, Y. Manolopoulos, A latency-based object placement approach in content distribution networks, 2005, p. 8, doi:10.1109/LAWEB.2005.3.

[73] G. Pallis, K. Stamos, A. Vakali, D. Katsaros, A. Sidiropoulos, Replication based on objects load under a content distribution network, in: ICDEW '06: Proceedings of the 22nd International Conference on Data Engineering Workshops, IEEE Computer Society, Washington, DC, USA, 2006, p. 53. http://dx.doi.org/10.1109/ICDEW.2006.127.

[74] X. Tang, J. Xu, Qos-aware replica placement for content distribution, IEEE Transactions on Parallel and Distributed Systems 16 (10) (2005) 921–932, doi:10.1109/TPDS.2005.126.

[75] N. Laoutaris, V. Zissimopoulos, I. Stavrakakis, On the optimization of storage capacity allocation for content distribution, Computer Networks 47 (2005) 409–428.

[76] J. Xu, B. Li, D.L. Lee, Placement problems for transparent data replication proxy services, IEEE Journal on Selected Areas in Communications 20 (7) (2002) 1383–1398, doi:10.1109/JSAC.2002.802068.

[77] X. Zhang, W. Wang, X. Tan, Y. Zhu, Data replication at web proxies in content distribution network, in: LNCS 2642: Web Technologies and Applications, Springer-Verlag, 2003.

[78] T. Bektas, O. Oguz, I. Ouveysi, Designing cost-effective content distribution networks, Computers and Operations Research 34 (8) (2007) 2436–2449. http://dx.doi.org/10.1016/j.cor.2005.09.013.

[79] T. Bektaş, J.-F. Cordeau, E. Erkut, G. Laporte, Exact algorithms for the joint object placement and request routing problem in content distribution networks, Computers and Operations Research 35 (12) (2008) 3860–3884. http://dx.doi.org/10.1016/j.cor.2007.02.005.

[80] N. Laoutaris, V. Zissimopoulos, I. Stavrakakis, Joint object placement and node dimensioning for internet content distribution, Information Processing Letters 89 (6) (2004) 273–279. http://dx.doi.org/10.1016/j.ipl.2003.12.002.

[81] T. Bektas, J.-F. Cordeau, E. Erkut, G. Laporte, A two-level simulated annealing algorithm for efficient dissemination of electronic content, Journal of the Operational Research Society 59 (2008) 1557–1567.

[82] P. Venkatesh, S.N. Sivanandam, R. Venkatesan, A review of consistency mechanisms for qos aware content distribution networks, Academic Open Internet Journal 14 (2005).

[83] X. Tang, H. Chi, S.T. Chanson, Optimal replica placement under ttl-based consistency, IEEE Transactions on Parallel and Distributed Computing 18 (3) (2007) 351–363. http://dx.doi.org/10.1109/TPDS.2007.47.

[84] X. Tang, S.T. Chanson, Analysis of replica placement under expiration-based consistency management, IEEE Transactions on Parallel and Distributed Computing. 17 (11) (2006) 1253–1263. http://dx.doi.org/10.1109/TPDS.2006.147.

[85] X. Jia, D. Li, H. Du, J. Cao, On optimal replication of data object at hierarchical and transparent web proxies, IEEE Transactions on Parallel and Distributed Computing. 16 (8) (2005) 673–685. http://dx.doi.org/10.1109/TPDS.2005.94.

[86] C.-M. Chen, Y. Ling, M. Pang, W. Chen, S. Cai, Y. Suwa, O. Altintas, Scalable request routing with next-neighbor load sharing in multi-server environments, in: AINA '05: Proceedings of the 19th International Conference on Advanced Information Networking and Applications, IEEE Computer Society, Washington, DC, USA, 2005, pp. 441–446. http://dx.doi.org/10.1109/AINA.2005.303.

[87] A. Shaikh, R. Tewari, M. Agrawal, On the effectiveness of dns-based server selection, INFOCOM 2001: Proceedings of Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 3, IEEE, 2001, pp. 1801–1810, doi:10.1109/INFCOM.2001.916678.

[88] Z.M. Mao, C.D. Cranor, F. Douglis, O. Spatscheck, J. Wang, A precise and efficient evaluation of the proximity between web clients and their local dns servers, in: In Proceedings of USENIX Annual Technical Conference, USENIX Association, 2002, pp. 229–242.

[89] RFC 3568: Known Content Network Request-Routing Mechanisms, 2003.

[90] J. Pan, Y.T. Hou, B. Li, An overview of dns-based server selections in content distribution networks, Computer Networks 43 (6) (2003) 695–711. http://dx.doi.org/10.1016/S1389-1286(03)00293-7.

[91] A.-J. Su, D.R. Choffnes, A. Kuzmanovic, F.E. Bustamante, Drafting behind akamai (travelocity-based detouring), SIGCOMM Computer Communications Review 36 (4) (2006) 435–446. http://doi.acm.org/10.1145/1151659.1159962.

[92] B. Krishnamurthy, C. Wills, Y. Zhang, On the use and performance of content distribution networks, in: IMW '01: Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement, ACM, New York, NY, USA, 2001, pp. 169–182. http://doi.acm.org/10.1145/505202.505224.

[93] M. Hofmann, L.R. Beaumont, Content Networking: Architecture, Protocols, and Practice (The Morgan Kaufmann Series in Networking), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[94] C. Partridge, T. Mendez, W. Milliken, RFC 1546: Host Anycasting Service, 1993.

[95] Z.-M. Fei, S. Bhattacharjee, E. Zegura, M. Ammar, A novel server selection technique for improving the response time of a replicated service, vol. 2, pp. 783–791, 1998, doi:10.1109/INFCOM.1998.665101.

[96] M.J. Freedman, K. Lakshminarayanan, D. Mazières, Oasis: anycast for any service, in: NSDI'06: Proceedings of the 3rd conference on Networked Systems Design & Implementation, USENIX Association, Berkeley, CA, USA, 2006, pp. 10–10.

[97] R. Buyya, A.-M.K. Pathan, J. Broberg, Z. Tari, A case for peering of content delivery networks, Distributed Systems Online 7 (10) (2006), doi:10.1109/MDSO.2006.57. pp. 3–3.

[98] A.-M. Khan Pathan, R. Buyya, Economy-based content replication for peering content delivery networks, in: CCGRID '07: Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid, IEEE Computer Society, Washington, DC, USA, 2007, pp. 887–892. http://dx.doi.org/10.1109/CCGRID.2007.48.

[99] M. Szymaniak, G. Pierre, M. van Steen, Netairt: A DNS-based redirection system for Apache, in: Proceedings of the IADIS International Conference on WWW/Internet, Algarve, Portugal, 2003, <http://www.globule.org/publi/NDBRSA_icwi2003.html>.

[100] V.S. Pai, M. Aron, G. Banga, M. Svendsen, P. Druschel, W. Zwaenepoel, E. Nahum, Locality-aware request distribution in cluster-based network servers, SIGOPS Operating Systemc Review 32 (5) (1998) 205–216. http://doi.acm.org/10.1145/384265.291048.

[101] A. Aggarwal, M. Rabinovich, Performance of dynamic replication schemes for an internet hosting service, Tech. rep., ATT TR HA6177000-981030-01-TM (1998).

[102] K. Delgadillo, Cisco distributeddirector, cisco inc. white paper, 1997.

[103] D. Karger, A. Sherman, A. Berkheimer, B. Bogstad, R. Dhanidina, K. Iwamoto, B. Kim, L. Matkins, Y. Yerushalmi, Web caching with consistent hashing, Computer Networks 31 (11-16) (1999) 1203–1213. http://dx.doi.org/10.1016/S1389-1286(99)00055-9.

[104] M. Andrews, B. Shepherd, A. Srinivasan, P. Winkler, F. Zane, Clustering and server selection using passive monitoring, vol. 3, 2002, pp. 1717–1725, doi:10.1109/INFCOM.2002.1019425.

[105] O. Ardaiz, F. Freitag, L. Navarro, Improving the service time of web clients using server redirection, SIGMETRICS Performance Evaluation RevIEW 29 (2) (2001) 39–44. http://doi.acm.org/10.1145/572317.572324.
[106] N. Ball, P. Pietzuch, D istributed content delivery using load-aware network coordinates, in: Proceedings of the 3rd International Workshop on Real Overlays and Distributed System (ROADS'08), Madrid, Spain, 2008. <http://www.doc.ic.ac.uk/prp/doc/research/roads08-cachenc-final.pdf>.
[107] B. Wong, A. Slivkins, E.G. Sirer, Meridian: a lightweight network location service without virtual coordinates, in: SIGCOMM '05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications, ACM, New York, NY, USA, 2005, pp. 85–96. http://doi.acm.org/10.1145/1080091.1080103.
[108] T. Wu, D. Starobinski, A comparative analysis of server selection in content replication networks, IEEE/ACM Transactions on Networking 16 (6) (2008) 1461–1474. http://dx.doi.org/10.1109/TNET.2007.909752.
[109] M. Conti, E. Gregori, W. Lapenna, Content delivery policies in replicated web services: client-side vs. server-side, Cluster Computing 8 (1) (2005) 47–60.
[110] M. Conti, E. Gregori, W. Lapenna, Client-side content delivery policies in replicated web services: parallel access versus single server approach, Performance Evaluation 59 (2005) 137–157.
[111] J. Ni, D. Tsang, Large-scale cooperative caching and application-level multicast in multimedia content delivery networks, Communications Magazine, IEEE 43 (5) (2005) 98–105, doi:10.1109/MCOM.2005.1453429.
[112] J. Ni, D. Tsang, I. Yeung, X. Hei, Hierarchical content routing in large-scale multimedia content delivery network, in: IEEE International Conference on Communications, 2003. ICC '03, vol. 2, 2003, pp. 854–859.
[113] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, D. Lewin, Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the world wide web, in: STOC '97: Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing, ACM, New York, NY, USA, 1997, pp. 654–663. http://doi.acm.org/10.1145/258533.258660.
[114] K. Stamos, G. Pallis, A. Vakali, Integrating caching techniques on a content distribution network, in: LNCS 4152: Advances in Databases and Information Systems, Springer-Verlag, 2005.
[115] S. Bakiras, T. Loukopoulos, Combining replica placement and caching techniques in content distribution networks, Computer Communications 28 (9) (2005) 1062–1073.
[116] K. Stamos, G. Pallis, C. Thomos, A. Vakali, A similarity based approach for integrated web caching and content replication in cdns, in: IDEAS '06: Proceedings of the 10th International Database Engineering and Applications Symposium, IEEE Computer Society, Washington, DC, USA, 2006. http://dx.doi.org/10.1109/IDEAS.2006.5.
[117] C. Canali, V. Cardellini, M. Colajanni, R. Lancellotti, Content delivery and management, in: R. Buyya, M. Pathan, A. Vakali (Eds.), Content Delivery Networks, Springer-verlag, 2008.
[118] A. Davis, J. Parikh, W.E. Weihl, Edgecomputing: extending enterprise applications to the edge of the internet, in: WWW Alt. '04: Proceedings of the 13th International World Wide Web conference on Alternate track papers & posters, ACM, New York, NY, USA, 2004, pp. 180–187. http://doi.acm.org/10.1145/1013367.1013397.
[119] S. Sivasubramanian, M. Szymaniak, G. Pierre, M.v. Steen, Replication for web hosting systems, ACM Computing Surveys 36 (3) (2004) 291–334. http://doi.acm.org/10.1145/1035570.1035573.
[120] M. Rabinovich, Z. Xiao, A. Aggarwal, Computing on the edge: a platform for replicating internet applications, Web content caching and distribution: proceedings of the 8th international workshop, 2004, pp. 57–77.
[121] W. Zhao, H. Schulzrinne, Dotslash: handling web hotspots at dynamic content web sites, vol. 4, 2005, pp. 2836–2840, doi:10.1109/INFCOM.2005.1498572.
[122] W. Zhao, H. Schulzrinne, Enabling on-demand query result caching in dotslash for handling web hotspots effectively, in: Hot Topics in Web Systems and Technologies, 2006. HOTWEB '06. 1st IEEE Workshop on, 2006, pp. 1–12, doi:10.1109/HOTWEB.2006.355257.
[123] G. Pacifici, M. Spreitzer, A. Tantawi, A. Youssef, Performance management for cluster-based web services, IEEE Journal on Selected Areas in Communications 23 (12) (2005) 2333–2343, doi:10.1109/JSAC.2005.857208.
[124] E. Cecchet, A. Chanda, S. Elnikety, J. Marguerite, W. Zwaenepoel, Performance comparison of middleware architectures for generating dynamic web content, in: In Proceedings of the ACM/IFIP/USENIX International Middleware Conference (Middleware 2003), Rio de, 2003.
[125] L. Rilling, S. Sivasubramanian, G. Pierre, High availability and scalability support for web applications, in: International Symposium on Applications and the Internet, 2007. SAINT 2007, 2007, pp. 5–5, doi:10.1109/SAINT.2007.14.
[126] C. Bornhövd, M. Altinel, S. Krishnamurthy, C. Mohan, H. Pirahesh, B. Reinwald, Dbcache: middle-tier database caching for highly scalable e-business architectures, in: SIGMOD '03: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, ACM, New York, NY, USA, 2003. http://doi.acm.org/10.1145/872757.872849. pp. 662–662.
[127] S. Sivasubramanian, G. Alonso, G. Pierre, M. van Steen, Globedb: Autonomic data replication for web applications, in: In WWW '05: Proceedings of the 14th International World-Wide Web conference, 2005, pp. 33–42.
[128] J. Jung, B. Krishnamurthy, M. Rabinovich, Flash crowds and denial of service attacks: characterization and implications for cdns and web sites, in: WWW '02: Proceedings of the 11th International Conference on World Wide Web, ACM, New York, NY, USA, 2002, pp. 293–304. http://doi.acm.org/10.1145/511446.511485.
[129] N. Yoshida, Dyamic CDN Against Flash Crowds, in: Content Delivery Networks, Springer-Verlag, 2008.
[130] T. Stading, P. Maniatis, M. Baker, Peer-to-peer caching schemes to address flash crowds, in: IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems, Springer-Verlag, London, UK, 2002, pp. 203–213.
[131] T. Wauters, J. Coppens, F. De Turck, B. Dhoedt, P. Demeester, Replica placement in ring based content delivery networks, Computer Communications 29 (2006) 3313–3326, doi:10.1016/j.comcom.2006.05.008. http://dl.acm.org/citation.cfm?id=1646661.1647004.
[132] J. Coppens, T. Wauters, F. De Turck, B. Dhoedt, P. Demeester, Design and performance of a self-organizing adaptive content distribution network, in: Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP, 2006, pp. 534 –545, doi:10.1109/NOMS.2006.1687582.
[133] M. Pathan, R. Buyya, J. Broberg, internetworking of cdns, in: Content Delivery Networks, Springer-Verlag, 2008.
[134] A.-M.K. Pathan, J.A. Broberg, K. Bubendorfer, K.H. Kim, R. Buyya, An architecture for virtual organization (vo)-based effective peering of content delivery networks, in: UPGRADE '07: Proceedings of the Second Workshop on Use of P2P, GRID and Agents for the Development of Content Networks, ACM, New York, NY, USA, 2007, pp. 29–38. http://doi.acm.org/10.1145/1272980.1272989.
[135] J. Jung, B. Krishnamurthy, M. Rabinovich, Flash crowds and denial of service attacks: characterization and implications for cdns and web sites, in: WWW '02: Proceedings of the 11th international conference on World Wide Web, ACM, New York, NY, USA, 2002, pp. 293–304. http://doi.acm.org/10.1145/511446.511485.
[136] V.N. Padmanabhan, K. Sripanidkulchai, The case for cooperative networking, in: IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems, Springer-Verlag, London, UK, 2002, pp. 178–190.
[137] S. Chen, B. Shen, S. Wee, X. Zhang, Sproxy: a caching infrastructure to support internet streaming, IEEE Transactions on Multimedia 9 (5) (2007) 1062–1072, doi:10.1109/TMM.2007.898943.
[138] B. Wang, S. Sen, M. Adler, D. Towsley, Optimal proxy cache allocation for efficient streaming media distribution, INFOCOM 2002: Proceedings of Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 3, IEEE, 2002, pp. 1726–1735, doi:10.1109/INFCOM.2002.1019426.
[139] R. Rejaie, M. Handley, M. H, H. Yu, D. Estrin, Proxy caching mechanism for multimedia playback streams in the internet, in: Proceedings of the 4th International Web Caching Workshop, 1999.
[140] K.-L. Wu, P.S. Yu, J.L. Wolf, Segment-based proxy caching of multimedia streams, in: WWW '01: Proceedings of the 10th international conference on World Wide Web, ACM, New York, NY, USA, 2001, pp. 36–44. http://doi.acm.org/10.1145/371920.371933.
[141] R. Rejaie, H. Yu, M. Handley, D. Estrin, Multimedia proxy caching mechanism for quality adaptive streaming applications in the internet, INFOCOM 2000, Proceedings of Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 2, IEEE, 2000, pp. 980–989, doi:10.1109/INFCOM.2000.832273.
[142] R. Rejaie, M. Handley, D. Estrin, Quality adaptation for congestion controlled video playback over the internet, SIGCOMM Computer Communications Review 29 (4) (1999) 189–200. http://doi.acm.org/10.1145/316194.316222.
[143] M. Czyrnek, C. Mazurek, E. Kusmierek, M. Stroinsk, itvp: platform for digital audio and video delivery over broadband ip networks, in: NOSSDAV '08: Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video, ACM, New York, NY, USA, 2008, pp. 119–120. http://doi.acm.org/10.1145/1496046.1496077.
[144] J. Almeida, D. Eager, M. Vernon, S. Wright, Minimizing delivery cost in scalable streaming content distribution systems, IEEE Transactions on Multimedia 6 (2) (2004) 356–365, doi:10.1109/TMM.2003.822796.
[145] J. Almeida, D. Eager, M. Ferris, M. Vernon, Provisioning content distribution networks for streaming media, INFOCOM 2002: Proceedings of Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 3, IEEE, 2002, pp. 1746–1755, doi:10.1109/INFCOM.2002.1019428.
[146] D.L. Eager, M.C. Ferris, M.K. Vernon, Optimized caching in systems with heterogeneous client populations, Performance Evaluation 42 (2-3) (2000) 163–185. http://dx.doi.org/10.1016/S0166-5316(00)00029-8.
[147] O. Verscheure, C. Venkatramani, P. Frossard, L. Amini, Joint server scheduling and proxy caching for video delivery, Computer Communications 25 (4) (2002) 413–423.
[148] G. Fortino, W. Russo, C. Mastroianni, C. Palau, M. Esteve, Cdn-supported collaborative media streaming control, Multimedia, IEEE 14 (2) (2007) 60–71, doi:10.1109/MMUL.2007.29.
[149] G. Fortino, C. Mastroianni, W. Russo, Collaborative media streaming services based on CDNs, in: Content Delivery Networks, Springer-Verlag, 2008.
[150] T. Yoshimura, Y. Yonemoto, T. Ohya, M. Etoh, S. Wee, Mobile streaming media cdn enabled by dynamic smil, in: WWW '02: Proceedings of the 11th international conference on World Wide Web, ACM, New York, NY, USA, 2002, pp. 651–661. http://doi.acm.org/10.1145/511446.511530.
[151] M.M. Bin Tariq, R. Jain, T. Kawahara, Mobility aware server selection for mobile streaming multimedia content distribution networks, Web content

caching and distribution: proceedings of the 8th international workshop, 2004, pp. 1–18.
[152] W3C, Synchronized multimedia integration language 3.0 (2008, <http://www.w3.org/TR/2008/REC-SMIL3-20081201/>).
[153] W. Aioffi, G. Mateus, J. de Almeida, A. Loureiro, Dynamic content distribution for mobile enterprise networks, IEEE Journal on Selected Areas in Communications 23 (10) (2005) 2022–2031, doi:10.1109/JSAC.2005.854126.
[154] W.M. Aioffi, G.R. Mateus, J.M. Almeida, D.S. Mendes, Mobile dynamic content distribution networks, in: MSWiM '04: Proceedings of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems, ACM, New York, NY, USA, 2004, pp. 87–94. http://doi.acm.org/10.1145/1023663.1023682.
[155] PPLive, <http://www.pplive.com/en/index.html>.
[156] X. Hei, C. Liang, J. Liang, Y. Liu, K. Ross, A measurement study of a large-scale p2p iptv system, IEEE Transactions on Multimedia 9 (8) (2007) 1672–1687, doi:10.1109/TMM.2007.907451.
[157] S.A. Baset, H.G. Schulzrinne, An analysis of the skype peer-to-peer internet telephony protocol, in: INFOCOM 2006: Proceedings of 25th IEEE International Conference on Computer Communications, 2006, pp. 1–11, <http://dx.doi.org/10.1109/INFOCOM.2006.312>.
[158] Bram Cohen's BitTorrent, <http://bramcohen.com/BitTorrent/>.
[159] K. Lua, J. Crowcroft, M. Pias, R. Sharma, S. Lim, A survey and comparison of peer-to-peer overlay network schemes, Communications Surveys & Tutorials, IEEE (2005) 72–93. http://ieeexplore.ieee.org/xpls/abs_all.jsp? arnumber=1528337 .
[160] S. Androutsellis-Theotokis, D. Spinellis, A survey of peer-to-peer content distribution technologies, ACM Computing Surveys 36 (4) (2004) 335–371, doi:10.1145/1041680.1041681. http://dx.doi.org/10.1145/1041680.1041681.
[161] C.G. Plaxton, R. Rajaraman, A.W. Richa, Accessing nearby copies of replicated objects in a distributed environment, in: SPAA '97: Proceedings of the Ninth Annual ACM Symposium on Parallel Algorithms and Architectures, ACM, New York, NY, USA, 1997, pp. 311–320. http://doi.acm.org/10.1145/258492.258523.
[162] K. Hildrum, J.D. Kubiatowicz, S. Rao, B.Y. Zhao, Distributed object location in a dynamic network, in: SPAA '02: Proceedings of the Fourteenth Annual ACM Symposium on Parallel Algorithms and Architectures, ACM, New York, NY, USA, 2002, pp. 41–52. http://doi.acm.org/10.1145/564870.564877.
[163] K. Hildrum, J.D. Kubiatowicz, S. Rao, B.Y. Zhao, Distributed object location in a dynamic network, Theory of Computing Systems 37 (3) (2004) 405–440.
[164] K. Aberer, P. Cudré-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Punceva, R. Schmidt, P-grid: a self-organizing structured p2p system, SIGMOD Record 32 (3) (2003) 29–33. http://doi.acm.org/10.1145/945721.945729.
[165] K. Aberer, A. Datta, M. Hauswirth, Efficient, self-contained handling of identity in peer-to-peer systems, IEEE Transactions on Knowledge and Data Engineering 16 (7) (2004) 858–869, doi:10.1109/TKDE.2004.1318567.
[166] S. Rhea, D. Geels, T. Roscoe, J. Kubiatowicz, Handling churn in a dht, in: ATEC '04: Proceedings of the annual conference on USENIX Annual Technical Conference, USENIX Association, Berkeley, CA, USA, 2004, pp. 10–10.
[167] Q. Lian, W. Chen, Z. Zhang, S. Wu, B. Zhao, Z-ring: fast prefix routing via a low maintenance membership protocol, in: IEEE International Conference on Network Protocols, 2005. ICNP 2005. 13th 2005, doi:10.1109/ICNP.2005.45.
[168] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, H. Balakrishnan, Chord: A scalable peer-to-peer lookup service for internet applications, in: SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, ACM, New York, NY, USA, 2001, pp. 149–160. http://doi.acm.org/10.1145/383059.383071.
[169] I. Stoica, R. Morris, D. Liben-Nowell, D. Karger, M. Kaashoek, F. Dabek, H. Balakrishnan, Chord: a scalable peer-to-peer lookup protocol for internet applications, IEEE/ACM Transactions on Networking 11 (1) (2003) 17–32, doi:10.1109/TNET.2002.808407.
[170] L. Alimal, S. El-Ansary, P. Brand, S. Haridi, Dks(n, k, f): a family of low communication, scalable and fault-tolerant infrastructures for p2p applications, in: Proceedings of 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid, 2003, CCGrid 2003, 2003, pp. 344–350. doi:10.1109/CCGRID.2003.1199386.
[171] T. Schutt, F. Schintke, A. Reinefeld, Structured overlay without consistent hashing: Empirical results, in: Sixth IEEE International Symposium on Cluster Computing and the Grid Workshops, 2006, vol. 2, 2006, pp. 8–8. doi:10.1109/CCGRID.2006.1630903.
[172] T. Schütt, F. Schintke, A. Reinefeld, Range queries on structured overlay networks, Computer Communications 31 (2) (2008) 280–291. http://dx.doi.org/10.1016/j.comcom.2007.08.027.
[173] F.M. Kaashoek, D.R. Karger, Koorde: A simple degree-optimal distributed hash table, in: Lecture Notes in Computer Science: Peer-to-Peer Systems II, Springer-Verlag, 2003, pp. 98–107. http://www.springerlink.com/content/unmqcqy0yxpu32xp.
[174] A. Kumar, S. Merugu, J. Xu, X. Yu, Ulysses: a robust, low-diameter, low-latency peer-to-peer network, in: Proceedings of 11th IEEE International Conference on Network Protocols, 2003, pp. 258–267. doi:10.1109/ICNP.2003.1249776.
[175] H. Shen, C.-Z. Xu, G. Chen, Cycloid: a constant-degree and lookup-efficient p2p overlay network, in: Proceedings of 18th International Symposium on Parallel and Distributed Processing, 2004, p. 195-216. doi:10.1109/IPDPS.2004.1302935.
[176] H. Shen, C.-Z. Xu, G. Chen, Cycloid: a constant-degree and lookup-efficient p2p overlay network, Performance Evaluation 63 (3) (2006) 195–216. http://dx.doi.org/10.1016/j.peva.2005.01.004.
[177] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Schenker, A scalable content-addressable network, in: SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, ACM, New York, NY, USA, 2001, pp. 161–172. http://doi.acm.org/10.1145/383059.383072.
[178] S. Ratnasamy, M. Handley, R. Karp, S. Schenker, Topologically aware overlay construction and server selection, in: SIGCOMM '02: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications, 2002.
[179] P. Maymounkov, D. Maziãres, Kademlia: A peer-to-peer information system based on the xor metric, Lecture Notes in Computer Science: Peer-to-Peer Systems: First InternationalWorkshop, IPTPS 2002 Cambridge, MA, USA, March 7–8, 2002, Springer-Verlag, 2002, pp. 53–65. Revised Papers http://www.springerlink.com/content/2ekx2a76ptwd24qt .
[180] R. Rodrigues, C. Blake, When multi-hop peer-to-peer routing matters, in: 3rd International Workshop on Peer-to-Peer Systems (IPTPS'04), 2004.
[181] I. Gupta, K. Birman, P. Linga, A. Demers, R. van Renesse, Kelip s: Building an efficient and stable p2p DHT through increased memory and background overhead, in: Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03), 2003. <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.13.3464>.
[182] A. Gupta, B. Liskov, R. Rodrigues, Efficient routing for peer-to-peer overlays, in: NSDI'04: Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation, USENIX Association, Berkeley, CA, USA, 2004, pp. 9–9.
[183] B. Leong, B. Liskov, E. Demaine, Epichord: parallelizing the chord lookup algorithm with reactive routing state management, in: Proceedings. 12th IEEE International Conference on Networks. (ICON 2004), vol. 1, 2004, pp. 270–276, doi:10.1109/ICON.2004.1409145.
[184] B. Leong, B. Liskov, E.D. Demaine, Epichord: Parallelizing the chord lookup algorithm with reactive routing state management, Computer Communications 29 (9) (2006) 1243–1259.
[185] J. Buford, A. Brown, M. Kolberg, Analysis of an active maintenance algorithm for an o(1)-hop overlay, in: Global Telecommunications Conference, 2007. GLOBECOM '07, IEEE, 2007, pp. 81–86, doi:10.1109/GLOCOM.2007.23.
[186] L. Monnerat, C. Amorim, D1ht: a distributed one hop hash table, in: Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International, 2006, doi:10.1109/IPDPS.2006.1639278.
[187] B. Carlsson, R. Gustavsson, The rise and fall of napster – an evolutionary approach, in: AMT '01: Proceedings of the 6th International Computer Science Conference on Active Media Technology, Springer-Verlag, London, UK, 2001, pp. 347–354.
[188] A.D.R. Marc Waldman, L.F. Cranor, Publius: A robust, tamper-evident, censorship-resistant, web publishing system, in: Proceedings 9th USENIX Security Symposium, 2000, pp. 59–72.
[189] The Annotated Gnutella Protocol Specification v0.4, <http://rfc-gnutella.sourceforge.net/developer/stable/index.html>.
[190] R. Dingledine, M.J. Freedman, D. Molnar, The free haven project: distributed anonymous storage service, in: International Workshop on Designing Privacy Enhancing Technologies, Springer-Verlag, New York, Inc., New York, NY, USA, 2001, pp. 67–95.
[191] J. Liang, R. Kumar, K.W. Ross, The fasttrack overlay: a measurement study, Computer Networking 50 (6) (2006) 842–858. http://dx.doi.org/10.1016/j.comnet.2005.07.014.
[192] Gnutella 0.6, <http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html>.
[193] IETF ALTO WG, <https://datatracker.ietf.org/wg/alto/charter/>.
[194] R. Zhang, Y.C. Hu, Assisted peer-to-peer search with partial indexing, IEEE Transactions on Parallel and Distributed Systems 18 (8) (2007) 1146–1158. http://doi.ieeecomputersociety.org/10.1109/TPDS.2007.1035.
[195] C.-J. Lin, Y.-T. Chang, S.-C. Tsai, C.-F. Chou, Distributed social-based overlay adaptation for unstructured p2p networks, in: IEEE Global Internet Symposium, 2007, pp. 1–6, doi:10.1109/GI.2007.4301422.
[196] K. Sripanidkulchai, B. Maggs, H. Zhang, Efficient content location using interest-based locality in peer-to-peer systems, vol. 3, 2003, pp. 2166–2176.
[197] E. Cohen, A. Fiat, H. Kaplan, Associative search in peer to peer networks: harnessing latent semantics, vol. 2, 2003, pp. 1261–1271.
[198] J. Liang, R. Kumar, K.W. Ross, The kazaa overlay: A measurement study, Computer Networks (Special Issue on Overlaps) (2005).
[199] S. Saroiu, K.P. Gummadi, S.D. Gribble, Measuring and analyzing the characteristics of napster and gnutella hosts, Multimedia Systems 9 (2) (2003) 170–184. <http://dx.doi.org/10.1007/s00530-003-0088-1>.
[200] M. Ripeanu, I. Foster, A. Iamnitchi, Mapping the gnutella network: properties of large-scale peer-to-peer systems and implications for system, IEEE Internet Computing Journal 6 (2002) 2002.
[201] D. Stutzbach, R. Rejaie, S. Sen, Characterizing unstructured overlay topologies in modern p2p file-sharing systems, IEEE/ACM Transactions on Networking 16 (2) (2008) 267–280, doi:10.1109/TNET.2007.900406.
[202] J. Buford, H. Yu, E.K. Lua, P2P Networking and Applications, Morgan Kaufmann Publishers Inc., 2009.

[203] L. Garces-Erice, E. Biersack, P.A. Felber, K.W. Ross, G. Urvoy-Keller, Hierarchical peer-to-peer systems, in: Proceedings of ACM/IFIP International Conference on Parallel and Distributed Computing (Euro-Par, 2003, pp. 643–657.
[204] M. Kleis, E.K. Lua, X. Zhou, Hierarchical peer-to-peer networks using lightweight superpeer topologies, in: Computers and Communications, 2005. ISCC 2005. Proceedings. 10th IEEE Symposium on, 2005, pp. 143–148. doi:10.1109/ISCC.2005.78.
[205] E.K. Lua, X. Zhou, Network-aware superpeers-peers geometric overlay network, in: Computer Communications and Networks, 2007. ICCCN 2007. Proceedings of 16th International Conference on, 2007, pp. 141–148. doi:10.1109/ICCCN.2007.4317810.
[206] iMesh, <http://www.imesh.com/>.
[207] D. Carra, E.W. Biersack, Building a reliable p2p system out of unreliable p2p clients: the case of kad, in: CoNEXT '07: Proceedings of the 2007 ACM CoNEXT Conference, ACM, New York, NY, USA, 2007, pp. 1–12. http://doi.acm.org/10.1145/1364654.1364690.
[208] D. Rubenstein, S. Sahu, Can unstructured p2p protocols survive flash crowds? IEEE/ACM Transactions on Networking 13 (3) (2005) 501–512, doi:10.1109/TNET.2005.845530.
[209] Q. Lv, P. Cao, E. Cohen, K. Li, S. Shenker, Search and replication in unstructured peer-to-peer networks, in: ICS '02: Proceedings of the 16th International Conference on Supercomputing, ACM, New York, NY, USA, 2002, pp. 84–95. http://doi.acm.org/10.1145/514191.514206.
[210] C. Gkantsidis, M. Mihail, A. Saberi, Random walks in peer-to-peer networks, in: INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies, vol. 1, 2004, p. 130. doi:10.1109/INFCOM.2004.1354487.
[211] G. Hasslinger, S. Kempken, Efficiency of random walks for search in different network structures, in: ValueTools '07: Proceedings of the 2nd international conference on Performance evaluation methodologies and tools, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, 2007, pp. 1–8.
[212] L. Adamic, R.M. Lukose, A.R. Puniyani, B.A. Huberman, Search in power-law networks, Physical Review E 64.
[213] C. Gkantsidis, M. Mihail, A. Saberi, Hybrid search schemes for unstructured peer-to-peer networks, vol. 3, 2005, pp. 1526–1537, doi:10.1109/INFCOM.2005.1498436.
[214] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, S. Shenker, Making gnutella-like p2p systems scalable, in: SIGCOMM '03: Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, ACM, New York, NY, USA, 2003, pp. 407–418. <http://doi.acm.org/10.1145/863955.864000>.
[215] B. Yang, H. Garcia-Molina, Improving search in peer-to-peer networks, in: Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on, 2002, pp. 5–14, doi:10.1109/ICDCS.2002.1022237.
[216] A. Kumar, J. Xu, E. Zegura, Efficient and scalable query routing for unstructured peer-to-peer networks, in: INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE, vol. 2, 2005, pp. 1162–1173, doi:10.1109/INFCOM.2005.1498343.
[217] N. Sarshar, P. Boykin, V. Roychowdhury, Percolation search in power law networks: making unstructured peer-to-peer networks scalable, in: Peer-to-Peer Computing, 2004. Proceedings. Fourth International Conference on, 2004, pp. 2–9, doi:10.1109/PTP.2004.1334925.
[218] E. Cohen, S. Shenker, Replication strategies in unstructured peer-to-peer networks, SIGCOMM Computer Communications Review 32 (4) (2002) 177–190. <http://doi.acm.org/10.1145/964725.633043>.
[219] B.F. Cooper, Quickly routing searches without having to move content, in: Peer-to-Peer Systems IV, Springer-Verlag, 2005, pp. 163–172.
[220] B.F. Cooper, An optimal overlay topology for routing peer-to-peer searches, in: Middleware '05: Proceedings of the ACM/IFIP/USENIX 2005 International Conference on Middleware, Springer-Verlag, New York, Inc., New York, NY, USA, 2005, pp. 82–101.
[221] M. Zhong, K. Shen, Popularity-biased random walks for peer-to-peer search under the square-root principle, in: In Proceedings of the 5th International Workshop on Peer-to-Peer Systems (IPTPS, 2006.
[222] W.W. Terpstra, J. Kangasharju, C. Leng, A.P. Buchmann, Bubblestorm: resilient, probabilistic, and exhaustive peer-to-peer search, SIGCOMM Computer Communications Review 37 (4) (2007) 49–60. <http://doi.acm.org/10.1145/1282427.1282387>.
[223] R.A. Ferreira, M.K. Ramanathan, A. Awan, A. Grama, S. Jagannathan, Search with probabilistic guarantees in unstructured peer-to-peer networks, in: P2P '05: Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing, IEEE Computer Society, Washington, DC, USA, 2005, pp. 165–172. <http://dx.doi.org/10.1109/P2P.2005.33>.
[224] R. Morselli, B. Bhattacharjee, M.A. Marsh, A. Srinivasan, Efficient lookup on unstructured topologies, IEEE Journal on Selected Areas in Communications 25 (1) (2007) 62–72, doi:10.1109/JSAC.2007.07007.
[225] I. Clarke, O. Sandberg, B. Wiley, T.W. Hong, Freenet: A distributed anonymous information storage and retrieval system, Lecture Notes in Computer Science 2009 (2001) 46. <http://citeseer.ist.psu.edu/clarke00freenet.html>.
[226] Y.-J. Joung, L.-W. Yang, C.-T. Fang, Keyword search in dht-based peer-to-peer networks, IEEE Journal on Selected Areas in Communications 25 (1) (2007) 46–61, doi:10.1109/JSAC.2007.070106.
[227] M. Steiner, E.W. Biersack, T. En-Najjary, Actively monitoring peers in KAD, in: Proceedings of IPTS, 2007.
[228] M. Steiner, T. En-Najjary, E.W. Biersack, A global view of kad, in: IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement, ACM, New York, NY, USA, 2007, pp. 117–122. <http://doi.acm.org/10.1145/1298306.1298323>.
[229] D. Stutzbach, R. Rejaie, Improving lookup performance over a widely-deployed dht, in: INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings, 2006, pp. 1–12. doi:10.1109/INFOCOM.2006.329.
[230] J. Li, B.T. Loo, L. Joseph, J.M. Hellerstein, D.R. Karger, R. Morris, M.F. Kaashoek, On the feasibility of peer-to-peer web indexing and search, in: In IPTPS'03, 2003, pp. 207–215.
[231] P. Reynolds, A. Vahdat, Efficient peer-to-peer keyword searching, in: Middleware '03: Proceedings of the ACM/IFIP/USENIX 2003 International Conference on Middleware, Springer-Verlag, New York, Inc., New York, NY, USA, 2003, pp. 21–40.
[232] K.-H. Yang, J.-M. Ho, Proof: A dht-based peer-to-peer search engine, in: Web Intelligence, 2006. WI 2006. IEEE/WIC/ACM International Conference on, 2006, pp. 702–708, doi:10.1109/WI.2006.137.
[233] K. Sankaralingam, S. Sethumadhavan, J. Browne, Distributed pagerank for p2p systems, in: Proceedings of 12th IEEE International Symposium on High Performance Distributed Computing, 2003, pp. 58–68.
[234] C. Tang, S. Dwarkadas, Hybrid global-local indexing for effcient peer-to-peer information retrieval, in: NSDI'04: Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation, USENIX Association, Berkeley, CA, USA, 2004, pp. 16–16.
[235] M. Castro, M. Costa, A. Rowstron, Peer-to-peer overlays: structured, unstructured, or both? Tech. Rep. MSR-TR-2004-73, Microsoft Research, 2004.
[236] M. Castro, M. Costa, A. Rowstron, Debunking some myths about structured and unstructured overlays, in: NSDI'05: Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation, USENIX Association, Berkeley, CA, USA, 2005, pp. 85–98.
[237] Y. Yang, R. Dunlap, M. Rexroad, B.F. Cooper, Performance of full text search in structured and unstructured peer-to-peer systems, in: INFOCOM 2006, Proceedings of 25th IEEE International Conference on Computer Communications, 2006, pp. 1–12. doi:10.1109/INFOCOM.2006.309.
[238] J. Risson, T. Moors, Survey of research towards robust peer-to-peer networks: search methods, Computer Networks 50 (17) (2006) 3485–3521. <http://dx.doi.org/10.1016/j.comnet.2006.02.001>.
[239] B.T. Loo, J.M. Hellerstein, R. Huebsch, S. Shenker, I. Stoica, Enhancing p2p file-sharing with an internet-scale query processor, in: VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases, VLDB Endowment, 2004, pp. 432–443.
[240] B.T. Loo, R. Huebsch, I. Stoica, J.M. Hellerstein, The Case for a Hybrid P2P Search Infrastructure, in: Proceedings of IPTPS, 2004.
[241] Y. Qiao, F.E. Bustamante, Structured and unstructured overlays under the microscope: a measurement-based view of two p2p systems that people use, in: ATEC '06: Proceedings of the annual conference on USENIX '06 Annual Technical Conference, USENIX Association, Berkeley, CA, USA, 2006, pp. 31–31.
[242] M. Zaharia, S. Keshav, Gossip-based search selection in hybrid peer-to-peer networks: Research articles, Concurrency Computation: Practice and Experience 20 (2) (2008) 139–153. <http://dx.doi.org/10.1002/cpe.v20:2>.
[243] R. Zhang, Y. Hu, Assisted peer-to-peer search with partial indexing, IEEE Transactions on Parallel and Distributed Systems 18 (8) (2007) 1146–1158, doi:10.1109/TPDS.2007.1035.
[244] X. Luo, Z. Qin, J. Han, H. Chen, Dht-assisted probabilistic exhaustive search in unstructured p2p networks, in: Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on, 2008, pp. 1–9, doi:10.1109/IPDPS.2008.4536254.
[245] H. Chen, H. Jin, Y. Liu, L. Ni, Difficulty-aware hybrid search in peer-to-peer networks, in: Parallel Processing, ICPP 2007. International Conference on, 2007, pp. 6–6, doi:10.1109/ICPP.2007.35.
[246] W. Acosta, S. Chandra, Understanding the practical limits of the gnutella p2p system: an analysis of query terms and object name distributions, in: Proceedings of the ACM/SPIE Multimedia Computing and Networking MMCN '08, 2008.
[247] S. Ioannidis, P. Marbach, On the design of hybrid peer-to-peer systems, in: SIGMETRICS '08: Proceedings of the 2008 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, ACM, New York, NY, USA, 2008, pp. 157–168. <http://doi.acm.org/10.1145/1375457.1375476>.
[248] K. Puttaswamy, A. Sala, B. Zhao, Searching for rare objects using index replication, in: INFOCOM 2008, The 27th Conference on Computer Communications. IEEE, 2008, pp. 1723–1731, doi:10.1109/INFOCOM.2008.234.
[249] S. Ioannidis, P. Marbach, Absence of Evidence as Evidence of Absence: A Simple Mechanism for Scalable P2P Search, in: INFOCOM 2009, 28th IEEE International Conference on Computer Communications. IEEE, 2009.
[250] F. Dabek, M.F. Kaashoek, D. Karger, R. Morris, I. Stoica, Wide-area cooperative storage with cfs, in: SOSP '01: Proceedings of the eighteenth ACM symposium on Operating systems principles, ACM, New York, NY, USA, 2001, pp. 202–215. <http://doi.acm.org/10.1145/502034.502054>.
[251] K.-L. Huang, T.-Y. Huang, J. Chou, Lesslog: a logless file replication algorithm for peer-to-peer distributed systems, in: Proceedings of 18th International

Symposium on Parallel and Distributed Processing, 2004, doi:10.1109/IPDPS.2004.1303021.
[252] V. Gopalakrishnan, B. Silaghi, B. Bhattacharjee, P. Keleher, Adaptive replication in peer-to-peer systems, in: Proceedings of 24th International Conference on Distributed Computing Systems, 2004, pp. 360–369, doi:10.1109/ICDCS.2004.1281601.
[253] H. Shen, Y. Zhu, A proactive low-overhead file replication scheme for structured p2p content delivery networks, Journal of Parallel and Distributed Computing 69 (5) (2009) 429–440. <http://dx.doi.org/10.1016/j.jpdc.2009.02.008>.
[254] V. Ramasubramanian, E.G. Sirer, Beehive: O(1)lookup performance for power-law query distributions in peer-to-peer overlays, in: NSDI'04: Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation, USENIX Association, Berkeley, CA, USA, 2004, pp. 8–8.
[255] S. Tewari, L. Kleinrock, Proportional replication in peer-to-peer networks, in: INFOCOM 2006, Proceedings of 25th IEEE International Conference on Computer Communications, 2006, pp. 1–12, doi:10.1109/INFOCOM.2006.132.
[256] J. Kangasharju, K. Ross, D. Turner, Optimizing file availability in peer-to-peer content distribution, in: INFOCOM 2007, 26th IEEE International Conference on Computer Communications. IEEE, 2007, pp. 1973–1981, doi:10.1109/INFCOM.2007.229.
[257] J. Ni, J. Lin, S. Harrington, N. Sharma, Designing file replication schemes for peer-to-peer file sharing systems, in: Communications, 2008. ICC '08. IEEE International Conference on, 2008, pp. 5609–5613, doi:10.1109/ICC.2008.1051.
[258] N. Laoutaris, O. Telelis, V. Zissimopoulos, I. Stavrakakis, Distributed selfish replication, IEEE Transactions on Parallel and Distributed Systems 17 (12) (2006) 1401–1413, doi:10.1109/TPDS.2006.171.
[259] A. Wierzbicki, N. Leibowitz, M. Ripeanu, R. Wozniak, Cache replacement policies for p2p file sharing protocols, European Transactions on Telecommunications 15 (6) (2004) 559–569.
[260] A. Wierzbicki, N. Leibowitz, M. Ripeanu, R. Wozniak, Cache replacement policies revisited: the case of p2p traffic, in: Cluster Computing and the Grid, 2004. CCGrid 2004. IEEE International Symposium on, 2004, pp. 182–189, doi:10.1109/CCGrid.2004.1336565.
[261] X. Liu, J. Lan, P. Shenoy, K. Ramaritham, Consistency maintenance in dynamic peer-to-peer overlay networks, Computer Networks 50 (6) (2006) 859–876, doi:10.1016/j.comnet.2005.07.010. <http://www.sciencedirect.com/science/article/B6VRG-4H0J7J4-1/2/c915569e03061038e3dc41afe78c71ec>. overlay Distribution Structures and their Applications..
[262] Z. Li, G. Xie, Z. Li, Efficient and scalable consistency maintenance for heterogeneous peer-to-peer systems, IEEE Transactions on Parallel and Distributed Systems 19 (12) (2008) 1695–1708, doi:10.1109/TPDS.2008.46.
[263] X. Chen, S. Ren, H. Wang, X. zhang, Scope: scalable consistency maintenance in structured p2p systems, INFOCOM 2005, Proceedings of 24th Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 3, IEEE, 2005, pp. 1502–1513, doi:10.1109/INFCOM.2005.1498434.
[264] Z. Wang, S.K. Das, M. Kumar, H. Shen, An efficient update propagation algorithm for p2p systems, Computer Communications 30 (5) (2007) 1106–1115. <ttp://dx.doi.org/10.1016/j.comcom.2006.11.005>.
[265] A. Datta, M. Hauswirth, K. Aberer, Updates in highly unreliable, replicated peer-to-peer systems, in: ICDCS '03: Proceedings of the 23rd International Conference on Distributed Computing Systems, IEEE Computer Society, Washington, DC, USA, 2003, p. 76.
[266] C.K. Yeo, B.S. Lee, M.H. Er, A survey of application level multicast techniques, Computer Communications 27 (15) (2004) 1547–1568. <http://www.sciencedirect.com/science/article/B6TYP-4C9P7J3-2/2/6275461a2da90490db074e2b5549204d>.
[267] Y.-h. Chu, S.G. Rao, H. Zhang, A case for end system multicast (keynote address), in: SIGMETRICS '00: Proceedings of the 2000 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, ACM, New York, NY, USA, 2000, pp. 1–12. <http://doi.acm.org/10.1145/339331.339337>.
[268] M. Castro, P. Druschel, A.M. Kermarrec, A.I.T. Rowstron, Scribe: a large-scale and decentralized application-level multicast infrastructure, IEEE Journal on Selected Areas in Communications 20 (8) (2002) 1489–1499, doi:10.1109/JSAC.2002.803069. <http://dx.doi.org/10.1109/JSAC.2002.803069>.
[269] M. Castro, M.B. Jones, A.M. Kermarrec, A. Rowstron, M. Theimer, H. Wang, A. Wolman, An evaluation of scalable application-level multicast built using peer-to-peer overlays, in: INFOCOM 2003, Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies, IEEE, vol. 2, 2003, pp. 1510–1520 <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1208986>.
[270] S.Q. Zhuang, B.Y. Zhao, A.D. Joseph, R.H. Katz, J.D. Kubiatowicz, Bayeux: an architecture for scalable and fault-tolerant wide-area data dissemination, in: NOSSDAV '01: Proceedings of the 11th international workshop on Network and operating systems support for digital audio and video, ACM, New York, NY, USA, 2001, pp. 11–20. <http://doi.acm.org/10.1145/378344.378347>.
[271] R. Zhang, C.Y. Hu, Borg: A hybrid protocol for scalable application-level multicast in peer-to-peer networks, in: NOSSDAV '03: Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video, ACM, ACM Press, New York, NY, USA, 2003, pp. 172–179. <http://dx.doi.org/10.1145/776322.776349>.
[272] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, A. Singh, Splitstream: high-bandwidth multicast in cooperative environments, in: SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles, ACM, New York, NY, USA, 2003, pp. 298–313. <http://doi.acm.org/10.1145/945445.945474>.
[273] V.N. Padmanabhan, H.J. Wang, P.A. Chou, K. Sripanidkulchai, Distributing streaming media content using cooperative networking, in: NOSSDAV '02: Proceedings of the 12th international workshop on Network and operating systems support for digital audio and video, ACM, New York, NY, USA, 2002, pp. 177–186. <http://doi.acm.org/10.1145/507670.507695>.
[274] J. Li, Mutualcast: A serverless peer-to-peer multiparty real-time audio conferencing system, in: IEEE International Conference on Multimedia and Expo, 2005, ICME 2005, 2005, pp. 602–605, doi:10.1109/ICME.2005.1521495.
[275] S. Ratnasamy, M. Handley, R.M. Karp, S. Shenker, Application-level multicast using content-addressable networks, in: NGC '01: Proceedings of the Third International COST264 Workshop on Networked Group Communication, Springer-Verlag, London, UK, 2001, pp. 14–29.
[276] A.-M. Kermarrec, M. van Steen, Gossiping in distributed systems, SIGOPS Operating Systemc Review 41 (5) (2007) 2–7. <http://doi.acm.org/10.1145/1317379.1317381>.
[277] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, D. Terry, Epidemic algorithms for replicated database maintenance, in: PODC '87: Proceedings of the sixth annual ACM Symposium on Principles of distributed computing, ACM, New York, NY, USA, 1987, pp. 1–12. <http://doi.acm.org/10.1145/41840.41841>.
[278] M.-J. Lin, K. Marzullo, Directional gossip: Gossip in a wide area network, in: EDCC-3: Proceedings of the Third European Dependable Computing Conference on Dependable Computing, Springer-Verlag, London, UK, 1999, pp. 364–379.
[279] P. Eugster, R. Guerraoui, A.-M. Kermarrec, L. Massoulie, Epidemic information dissemination in distributed systems, Computer 37 (5) (2004) 60–67, doi:10.1109/MC.2004.1297243.
[280] M. Deshpande, B. Xing, I. Lazardis, B. Hore, N. Venkatasubramanian, S. Mehrotra, Crew: A gossip-based flash-dissemination system, in: 26th IEEE International Conference on Distributed Computing Systems, 2006, ICDCS 2006, 2006, pp. 45–45, doi:10.1109/ICDCS.2006.24.
[281] K.P. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, Y. Minsky, Bimodal multicast, ACM Transactions on Computer Systems 17 (2) (1999) 41–88. <http://doi.acm.org/10.1145/312203.312207>.
[282] P.T. Eugster, R. Guerraoui, S.B. Handurukande, P. Kouznetsov, A.-M. Kermarrec, Lightweight probabilistic broadcast, ACM Transactions on Computer Systems 21 (4) (2003) 341–374. <http://doi.acm.org/10.1145/945506.945507>.
[283] S. Voulgaris, D. Gavidia, M. Steen, Cyclon: Inexpensive membership management for unstructured p2p overlays, Journal of Network and Systems Management 13 (2) (2005) 197–217, doi:10.1007/s10922-005-4441-x. <http://dx.doi.org/10.1007/s10922-005-4441-x>.
[284] M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M. Kermarrec, M. van Steen, Gossip-based peer sampling, ACM Transactions on Computer Systems 25 (3) (2007) 8. <http://doi.acm.org/10.1145/1275517.1275520>.
[285] M. Jelasity, O. Babaoglu, T-man: Gossip-based overlay topology management, in: In 3rd International Workshop on Engineering Self-Organising Applications (ESOA'05, 2005, pp. 1–15. <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.62.9444>.
[286] J. Sacha, J. Dowling, R. Cunningham, R. Meier, Discovery of stable peers in a self-organising peer-to-peer gradient topology, in: Proceedings of Distributed Applications and Interoperable Systems, 2006.
[287] D. Towsley, The internet is flat: a brief history of networking in the next ten years, in: PODC '08: Proceedings of the twenty-seventh ACM symposium on Principles of distributed computing, ACM, New York, NY, USA, 2008, pp. 11–12. <http://doi.acm.org/10.1145/1400751.1400753>.
[288] A. Legout, G. Urvoy-Keller, P. Michiardi, Rarest first and choke algorithms are enough, in: IMC '06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement, ACM, New York, NY, USA, 2006, pp. 203–216. <http://doi.acm.org/10.1145/1177080.1177106>.
[289] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, X. Zhang, Measurements, analysis, and modeling of bittorrent-like systems, in: IMC '05: Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement, USENIX Association, Berkeley, CA, USA, 2005, pp. 4–4.
[290] M. Izal, G. Urvoy-Keller, E.W. Biersack, P.A. Felber, Al, L. Garcés-Erice, Dissecting bittorrent: Five months in a torrent's lifetime, in: Passive and Active Network Measurement, 2004, pp. 1–11. <http://www.springerlink.com/content/fg8hqw4136t0vtx9>.
[291] A. Legout, N. Liogkas, E. Kohler, L. Zhang, Clustering and sharing incentives in bittorrent systems, in: SIGMETRICS '07: Proceedings of the 2007 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, ACM, New York, NY, USA, 2007, pp. 301–312. <http://doi.acm.org/10.1145/1254882.1254919>.
[292] J.A. Pouwelse, P. Garbacki, D.H.J. Epema, H.J. Sips, The bittorrent p2p file-sharing system: Measurements and analysis, in: 4th International Workshop on Peer-to-Peer Systems (IPTPS), 2005. <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.59.3191>.
[293] X. Yang, G. de Veciana, Performance of peer-to-peer networks: service capacity and role of resource sharing policies, Performance Evaluation 63 (3) (2006) 175–194. <http://dx.doi.org/10.1016/j.peva.2005.01.005>.

[294] A.R. Bharambe, C. Herley, V.N. Padmanabhan, Analyzing and improving a bittorrent networks performance mechanisms, in: 25th IEEE International Conference on Computer Communications (INFOCOM 2006), 2006, pp. 1–12, <http://dx.doi.org/10.1109/INFOCOM.2006.328>.
[295] D. Qiu, R. Srikant, Modeling and performance analysis of bittorrent-like peer-to-peer networks, in: SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications, ACM, New York, NY, USA, 2004, pp. 367–378. <http://doi.acm.org/10.1145/1015467.1015508>.
[296] B. Fan, D.-m. Chiu, J. Lui, The delicate tradeoffs in bittorrent-like file sharing protocol design, in: ICNP '06: Proceedings of the Proceedings of the 2006 IEEE International Conference on Network Protocols, IEEE Computer Society, Washington, DC, USA, 2006, pp. 239–248. <http://dx.doi.org/10.1109/ICNP.2006.320217>.
[297] B. Fan, D.-M. Chiu, J. Lui, Stochastic differential equation approach to model bittorrent-like p2p systems, in: IEEE International Conference on Communications, 2006, ICC '06, vol. 2, 2006, pp. 915–920, doi:10.1109/ICC.2006.254824.
[298] T. Locher, P. Moor, S. Schmid, R. Wattenhofer, Free Riding in BitTorrent is Cheap, in: Proceedings of HotNets-V, 2006.
[299] M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, A. Venkataramani, Do incentives build robustness in bittorrent? in: Proceedings of 4th USENIX Symposium on Networked Systems Design & Implementation (NSDI 2007), USENIX, Cambridge, MA, 2007.
[300] N. Liogkas, R. Nelson, E. Kohler, L. Zhang, Exploiting BitTorrent for Fun (But Not Profit), in: Proceedings of the Fifth International Workshop on Peer-to-Peer Systems (IPTPS 2006), 2006.
[301] D. Carra, G. Neglia, P. Michiardi, On the impact of greedy strategies in bittorrent networks: The case of bittyrant, in: Peer-to-Peer Computing, 2008. P2P '08. Eighth International Conference on, 2008, pp. 311–320, doi:10.1109/P2P.2008.45.
[302] N. Laoutaris, G. Smaragdakis, A. Bestavros, J. Byers, Implications of selfish neighbor selection in overlay networks, in: IEEE INFOCOM 2007. 26th IEEE International Conference on Computer Communications, 2007, pp. 490–498.
[303] A. Fabrikant, A. Luthra, E. Maneva, C.H. Papadimitriou, S. Shenker, On a network creation game, in: PODC '03: Proceedings of the twenty-second annual symposium on Principles of distributed computing, ACM, New York, NY, USA, 2003, pp. 347–351. <http://doi.acm.org/10.1145/872035.872088>.
[304] B.-G. Chun, R. Fonseca, I. Stoica, J. Kubiatowicz, Characterizing selfishly constructed overlay routing networks, in: INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies, vol. 2, 2004, pp. 1329–1339.
[305] J. Feigenbaum, S. Shenker, Distributed algorithmic mechanism design: recent results and future directions, in: DIALM '02: Proceedings of the 6th international workshop on Discrete algorithms and methods for mobile computing and communications, ACM, New York, NY, USA, 2002, pp. 1–13. <http://doi.acm.org/10.1145/570810.570812>.
[306] N. Laoutaris, D. Carra, P. Michiardi, Uplink allocation beyond choke/unchoke: or how to divide and conquer best, in: CONEXT '08: Proceedings of the 2008 ACM CoNEXT Conference, ACM, New York, NY, USA, 2008, pp. 1–12. <http://doi.acm.org/10.1145/1544012.1544030>.
[307] N. Laoutaris, P. Rodriguez, L. Massoulie, Echos: edge capacity hosting overlays of nano data centers, SIGCOMM Computer Communications Review 38 (1) (2008) 51–54. <http://doi.acm.org/10.1145/1341431.1341442>.
[308] R. Bindal, P. Cao, W. Chan, J. Medved, G. Suwala, T. Bates, A. Zhang, Improving traffic locality in bittorrent via biased neighbor selection, in: 26th IEEE International Conference on Distributed Computing Systems, 2006. ICDCS 2006. 2006, pp. 66–66, doi:10.1109/ICDCS.2006.48.
[309] P.B. Godfrey, S. Shenker, I. Stoica, Minimizing churn in distributed systems, SIGCOMM Computer Communications Review 36 (4) (2006) 147–158, <http://doi.acm.org/10.1145/1151659.1159931>.
[310] D. Pendarakis, S. Shi, D. Verma, M. Waldvogel, Almi: an application level multicast infrastructure, in: Proceedings of the 3rd Conference on USENIX Symposium on Internet Technologies and Systems, vol. 3, USITS'01, USENIX Association, Berkeley, CA, USA, 2001, pp. 5–5, <http://portal.acm.org/citation.cfm?id=1251440.1251445>.
[311] S. Banerjee, B. Bhattacharjee, C. Kommareddy, Scalable application layer multicast, in: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications, SIGCOMM '02, ACM, New York, NY, USA, 2002, pp. 205–217. <http://doi.acm.org/10.1145/633025.633045>.
[312] V. Venkataraman, K. Yoshida, P. Francis, Chunkyspread: Heterogeneous unstructured tree-based peer-to-peer multicast, in: Proceedings of the 2006 IEEE International Conference on Network Protocols, IEEE Computer Society, Washington, DC, USA, 2006, pp. 2–11, doi:10.1109/ICNP.2006.320193. <http://portal.acm.org/citation.cfm?id=1317535.1318351>.
[313] S. Xie, B. Li, G.Y. Keung, X. Zhang, Coolstreaming: Design, Theory, and Practice, IEEE Transactions on Multimedia 9 (8) (2007) 1661–1671, doi:10.1109/TMM.2007.907469. <http://dx.doi.org/10.1109/TMM.2007.907469>.
[314] X. Zhang, J. Liu, B. Li, Y.-S. Yum, Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming, in: INFOCOM 2005, Proceedings IEEE of 24th Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 3, 2005, pp. 2102–2111, doi:10.1109/INFCOM.2005.1498486.
[315] V. Pai, K. Kumar, K. Tamilmani, V. Sambamurthy, A.E. Mohr, E.E. Mohr, Chainsaw: Eliminating trees from overlay multicast, in: IPTPS, 2005, pp. 127–140.
[316] M. Zhang, L. Zhao, Y. Tang, J.-G. Luo, S.-Q. Yang, Large-scale live media streaming over peer-to-peer networks through global internet, in: P2PMMS'05: Proceedings of the ACM workshop on Advances in peer-to-peer multimedia streaming, ACM, New York, NY, USA, 2005, pp. 21–28. <http://doi.acm.org/10.1145/1099384.1099388>.
[317] C. Dana, D. Li, D. Harrison, C.-N. Chuah, Bass: Bittorrent assisted streaming system for video-on-demand, in: IEEE 7th Workshop on Multimedia Signal Processing, 2005, pp. 1–4, doi:10.1109/MMSP.2005.248586.
[318] X. Liao, H. Jin, Y. Liu, L.M. Ni, D. Deng, Anysee: Peer-to-peer live streaming, in: Proceedings of INFOCOM 2006, 25th IEEE International Conference on Computer Communications, 2006, pp. 1–10. doi:10.1109/INFOCOM.2006.288.
[319] F. Pianese, J. Keller, E. Biersack, Pulse, a flexible p2p live streaming system, in: Proceedings of INFOCOM 2006, 25th IEEE International Conference on Computer Communications, 2006, pp. 1–6, doi:10.1109/INFOCOM.2006.42.
[320] A. Vlavianos, M. Iliofotou, M. Faloutsos, Bitos: Enhancing bittorrent for supporting streaming applications, in: INFOCOM 2006. Proceedings of 25th IEEE International Conference on Computer Communications, 2006, pp. 1–6. doi:10.1109/INFOCOM.2006.43.
[321] N. Magharei, R. Rejaie, Understanding mesh-based peer-to-peer streaming, in: NOSSDAV '06: Proceedings of the 2006 international workshop on Network and operating systems support for digital audio and video, ACM, New York, NY, USA, 2006, pp. 1–6. <http://doi.acm.org/10.1145/1378191.1378204>.
[322] T. Piotrowski, S. Banerjee, S. Bhatnagar, S. Ganguly, R. Izmailov, Peer-to-peer streaming of stored media: the indirect approach, SIGMETRICS Performance Evaluation Review 34 (1) (2006) 371–372. <http://doi.acm.org/10.1145/1140103.1140325>.
[323] X. Hei, Y. Liu, K. Ross, IPTV over P2P streaming networks: the mesh-pull approach, IEEE Communications Magazine 46 (2) (2008) 86–92, doi:10.1109/MCOM.2008.4473088. <http://dx.doi.org/10.1109/MCOM.2008.4473088>.
[324] S. Ali, A. Mathur, H. Zhang, Measurement of Commercial Peer-To-Peer Live Video Streaming, in: In Proceedings of ICST Workshop on Recent Advances in Peer-to-Peer Streaming, Weaterloo, Canadda, 2006.
[325] R. Kumar, Y. Liu, K. Ross, Stochastic fluid theory for p2p streaming systems, in: INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE, 2007, pp. 919–927, doi:10.1109/INFCOM.2007.112.
[326] S. Tewari, L. Kleinrock, Analytical model for bittorrent- based live video streaming, in: 3rd IEEE International Workshop on Networking Issues (NIME'07), Las Vegas, NV, 2007.
[327] N. Magharei, R. Rejaie, Mesh or multiple-tree: A comparative study of live p2p streaming approaches, in: in INFOCOM 2007, Anchorage, Alaska,6-12, 2007, pp. 1424–1432.
[328] S. Liu, R. Zhang-Shen, W. Jiang, J. Rexford, M. Chiang, Performance bounds for peer-assisted live streaming, in: Proceedings of the 2008 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, SIGMETRICS '08, ACM, New York, NY, USA, 2008, pp. 313–324. <http://doi.acm.org/10.1145/1375457.1375493>.
[329] D. Wu, Y. Liu, K. Ross, Queuing network models for multi-channel p2p live streaming systems, in: INFOCOM 2009, IEEE, 2009, pp. 73 –81. doi:10.1109/INFCOM.2009.5061908.
[330] D. Carra, R. Lo Cigno, E.W. Biersack, Graph based analysis of mesh overlay streaming systems, EEE Journal on Selected Area in Communications (JSAC) 25 (9) (2007) 1667–1677.
[331] M. Brinkmeier, G. Schäfer, T. Strufe, Optimally dos resistant p2p topologies for live multimedia streaming, IEEE Transactions on Parallel and Distributed Computing. 20 (2009) 831–844, doi:10.1109/TPDS.2008.265. <http://portal.acm.org/citation.cfm?id=1550403.1550575>.
[332] M. Zhang, S. Member, Q. Zhang, S. Member, L. Sun, S. Yang, Understanding the power of pull-based streaming protocol: Can we do better? IEEE JSAC 25 (8).
[333] C. Feng, B. Li, Understanding the Performance Gap Between Pull-Based Mesh Streaming Protocols and Fundamental Limits, 2009, pp. 891–899. doi:10.1109/INFCOM.2009.5061999. <http://dx.doi.org/10.1109/INFCOM.2009.5061999>.
[334] PPStream, <http://www.ppstream.com>.
[335] UUSee, <http://www.uusee.com>.
[336] SopCast, <http://www.sopcast.org>.
[337] TVAnts, <http://www.tvants.com>.
[338] VVKsy, <http://www.vvsky.com.cn>.
[339] GridCast, <http://www.gridcast.cn>.
[340] Y. Huang, T.Z.J. Fu, D.M. Chiu, J.C.S. Lui, C. Huang, Challeng es, design and analysis of a large-scale p2p-vod system, in: SIGCOMM '08: Proceedings of the ACM SIGCOMM 2008 conference on Data communication, ACM, New York, NY, USA, 2008, pp. 375–388. <http://dx.doi.org/http://doi.acm.org/10.1145/1402958.1403001>.
[341] S. Annapureddy, S. Guha, C. Gkantsidis, D. Gunawardena, P. Rodriguez, Exploring vod in p2p swarming systems, in: INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE, 2007, pp. 2571–2575. doi:10.1109/INFCOM.2007.323.
[342] K. Suh, C. Diot, J. Kurose, L. Massoulie, C. Neumann, D. Towsley, M. Varvello, Push-to-peer video-on-demand system: Design and evaluation, IEEE Journal on Selected Areas in Communications 25 (9) (2007) 1706–1716, doi:10.1109/JSAC.2007.071209.

[343] NADA, <http://www.nanodatacenters.eu/>.
[344] IETF PPSP WG, <https://datatracker.ietf.org/wg/ppsp/charter/>.
[345] Skype, <http://www.skype.com/>.
[346] K. Suh, D. Figueiredo, J. Kurose, D. Towsley, Characterizing and detecting relayed traffic: A case study using Skype, Tech. Rep. UMASS-TR-2005-50, UMass Computer Science (2005).
[347] S. Guha, N. Daswani, R. Jain, An experimental study of the skype peer-to-peer voip system, in: IPTPS'06: The 5th International Workshop on Peer-to-Peer Systems, Microsoft Research. <http://saikat.guha.cc/pub/iptps06-skype.pdf>.
[348] K.-T. Chen, C.-Y. Huang, P. Huang, C.-L. Lei, Quantifying skype user satisfaction, in: SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications, ACM, New York, NY, USA, 2006, pp. 399–410. <http://doi.acm.org/10.1145/1159913.1159959>.
[349] IETF P2PSIP WG, <https://datatracker.ietf.org/wg/p2psip/charter/>.
[350] I. Chlamtac, M. Conti, J.J. Liu, Mobile ad hoc networking: imperatives and challenges, Ad Hoc Networks 1 (1) (2003) 13–64, doi:10.1016/S1570-8705(03)00013-1. <http://dx.doi.org/10.1016/S1570-8705(03)00013-1>.
[351] H.-C. Hsiao, C.-T. King, Mobility churn in dhts, in: 25th IEEE International Conference on Distributed Computing Systems Workshops, 2005, pp. 799–805, doi:10.1109/ICDCSW.2005.98.
[352] J. Li, J. Stribling, R. Morris, M. Kaashoek, T. Gil, A performance vs. cost framework for evaluating dht design tradeoffs under churn, in: INFOCOM 2005, Proceedings IEEE of 24th Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 1, 2005, pp. 225–236 vol. 1. doi:10.1109/INFCOM.2005.1497894.
[353] H. Pucha, S. Das, Y. Hu, How to implement DHTs in mobile ad hoc networks, in: Proceedings of the 10th ACM International Conference on Mobile Computing and Network (MobiCom 2004), 2004.
[354] The MobileMAN Project, <http://cnd.iit.cnr.it/mobileMAN/>.
[355] E. Borgia, M. Conti, F. Delmastro, E. Gregori, Experimental comparison of routing and middleware solutions for mobile ad hoc networks: legacy vs cross-layer approach, in: Proceedings of the 2005 ACM SIGCOMM workshop on Experimental approaches to wireless network design and analysis, ACM, New York, NY, USA, 2005, pp. 82–87.
[356] E. Borgia, M. Conti, F. Delmastro, L. Pelusi, Lessons from an ad hoc network test-bed: Middleware and routing issues, Ad Hoc & Sensor Wireless Networks 1 (1-2) (2005) 125–157.
[357] E. Borgia, M. Conti, F. Delmastro, Mobileman: design, integration, and experimentation of cross-layer mobile multihop ad hoc networks, IEEE Communications Magazine 44 (7) (2006) 80–85.
[358] A. MacQuire, A. Brampton, I. Rai, L. Mathy, Performance analysis of Stealth DHT with mobile nodes, in: Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, 2006. PerCom Workshops 2006, 2006, p. 5.
[359] H. Hsiao, C. King, Bristle: A mobile structured peer-to-peer architecture, in: Proceedings of the 17th International Symposium on Parallel and Distributed Processing, IEEE Computer Society, Washington, DC, USA, 2003, pp. 33–37.
[360] B. Zhao, L. Huang, A. Joseph, J. Kubiatowicz, Rapid mobility via type indirection, in: Proceedings of IPTPS, Springer, 2004.
[361] M. Conti, G. Maselli, G. Turi, S. Giordano, Cross-layering in mobile ad hoc network design, Computer 37 (2) (2004) 48–51.
[362] V. Kawadia, P. Kumar, B. Technol, M. Cambridge, A cautionary perspective on cross-layer design, IEEE [see also IEEE Personal Communications] Wireless Communications 12 (1) (2005) 3–11.
[363] F. Delmastro, From Pastry to CrossROAD: Cross-layer ring overlay for ad hoc networks, in: Third IEEE International Conference on Pervasive Computing and Communications Workshops, 2005. PerCom 2005 Workshops, 2005, pp. 60–64.
[364] F. Delmastro, M. Conti, E. Gregori, P2P common API for structured overlay networks: A cross-layer extension, in: World of Wireless, Mobile and Multimedia Networks, 2006. WoWMoM 2006. International Symposium on a, 2006, p. 5.
[365] H. Pucha, S.M. Das, Y.C. Hu, Ekta: An efficient dht substrate for distributed applications in mobile ad hoc networks, in: WMCSA '04: Proceedings of the Sixth IEEE Workshop on Mobile Computing Systems and Applications, IEEE Computer Society, Washington, DC, USA, 2004, pp. 163–173. <http://dx.doi.org/10.1109/MCSA.2004.11>.
[366] M. Caesar, M. Castro, E. Nightingale, G. O'Shea, A. Rowstron, Virtual ring routing: network routing inspired by DHTs, Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications, Vol. 36, ACM, New York, NY, USA, 2006, pp. 351–362.
[367] A. Awad, R. German, F. Dressler, Exploiting Virtual Coordinates for Improved Routing Performance in Sensor Networks, IEEE Transactions on Mobile ComputingAvailable online: 10.1109/TMC.2010.218. <http://dx.doi.org/10.1109/TMC.2010.218>.
[368] C.-H. Lin, B.-H. Liu, H.-Y. Yang, C.-Y. Kao, M.-J. Tsai, Virtual-coordinate-based delivery-guaranteed routing protocol in wireless sensor networks with unidirectional links, in: INFOCOM 2008. The 27th Conference on Computer Communications. IEEE, 2008, pp. 351–355. doi:10.1109/INFOCOM.2008.79.
[369] M.-J. Tsai, F.-R. Wang, H.-Y. Yang, Y.-P. Cheng, Virtualface: An algorithm to guarantee packet delivery of virtual-coordinate-based routing protocols in wireless sensor networks, in: INFOCOM 2009, IEEE, 2009, pp. 1728 –1736. doi:10.1109/INFCOM.2009.5062092.
[370] M. Conti, E. Gregori, G. Turi, A cross-layer optimization of gnutella for mobile ad hoc networks, in: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing, ACM, New York, NY, USA, 2005, pp. 343–354.
[371] S. Rajagopalan, C. Shen, A cross-layer decentralized BitTorrent for mobile ad hoc networks, in: Mobile and Ubiquitous Systems-Workshops, 2006. 3rd Annual International Conference on, 2006, pp. 1–10.
[372] A. Klemm, C. Lindemann, O. Waldhorst, A special-purpose peer-to-peer file sharing system for mobile ad hoc networks, in: IEEE VEHICULAR TECHNOLOGY CONFERENCE, vol. 4, Citeseer, 2003, pp. 2758–2763.
[373] G. Ding, B. Bhargava, Peer-to-peer File-sharing over Mobile Ad hoc Networks, in: Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on, 2004, pp. 104–108.
[374] A. Nandan, S. Das, G. Pau, M. Gerla, M. Sanadidi, Co-operative downloading in vehicular ad-hoc wireless networks, in: Proceedings of IEEE/IFIP International Conference on Wireless On demand Network Systems and Services, St. Moritz, Switzerland, 2005, pp. 32–41.
[375] P. Michiardi, G. Urvoy-Keller, Performance analysis of cooperative content distribution in wireless ad hoc networks, in: Wireless on Demand Network Systems and Services, 2007. WONS'07. Fourth Annual Conference on, 2007, pp. 22–29.
[376] M. Sbai, C. Barakat, J. Choi, A. Hamra, T. Turletti, Adapting BitTorrent to wireless ad hoc networks, Lecture Notes in Computer Science 5198 (2008) 189–203.
[377] E. Royer, C. Perkins, Multicast ad hoc on-demand distance vector (MAODV) routing, IETF MANET Working Group Internet Draft.
[378] S. Lee, M. Gerla, C. Chiang, On-demand multicast routing protocol, in: 1999 IEEE Wireless Communications and Networking Conference, 1999. WCNC, 1999, pp. 1298–1302.
[379] M. Ge, S. Krishnamurthy, M. Faloutsos, Application versus network layer multicasting in ad hoc networks: the ALMA routing protocol, Ad Hoc Networks 4 (2) (2006) 283–300.
[380] C. Gui, P. Mohapatra, Overlay multicast for MANETs using dynamic virtual mesh, Wireless Networks 13 (1) (2007) 77–91.
[381] F. Delmastro, A. Passarella, M. Conti, P2p multicast for pervasive ad hoc networks, Pervasive and Mobile Computing 4 (1) (2008) 62–91. <http://dx.doi.org/10.1016/j.pmcj.2007.03.001>.
[382] A. Passarella, F. Delmastro, M. Conti, Xscribe: a stateless, cross-layer approach to p2p multicast in multi-hop ad hoc networks, in: MobiShare '06: Proceedings of the 1st international workshop on Decentralized resource sharing in mobile computing and networking, ACM, New York, NY, USA, 2006, pp. 6–11. <http://doi.acm.org/10.1145/1161252.1161255>.
[383] Z. Haas, J. Halpern, L. Li, Gossip-based ad hoc routing, IEEE/ACM Transactions on Networking (TON) 14 (3) (2006) 479–491.
[384] J. Luo, P. Eugster, J. Hubaux, Route driven gossip: Probabilistic reliable multicast in ad hoc networks, in: IEEE INFOCOM, vol. 3, INSTITUTE OF ELECTRICAL ENGINEERS INC (IEEE), 2003, pp. 2229–2239.
[385] V. Drabkin, R. Friedman, G. Kliot, M. Segal, RAPID: Reliable probabilistic dissemination in wireless ad-hoc networks, in: 26th IEEE International Symposium on Reliable Distributed Systems, 2007. SRDS 2007, 2007, pp. 13–22.
[386] C. Boldrini, M. Conti, A. Passarella, Exploiting users' social relations to forward data in opportunistic networks: The hibop solution, Pervasive and Mobile Computing 4 (5) (2008) 633–657. <http://dx.doi.org/10.1016/j.pmcj.2008.04.003>.
[387] P. Hui, J. Crowcroft, E. Yoneki, BUBBLE Rap: Social-based Forwarding in Delay Tolerant Networks, IEEE Transactions on Mobile Computing.
[388] T. Spyropoulos, K. Psounis, C.S. Raghavendra, Efficient routing in intermittently connected mobile networks: the multiple-copy case, IEEE/ACM Transactions Netwing 16 (2008) 77–90. <http://dx.doi.org/10.1109/TNET.2007.897964>.
[389] C. Boldrini, M. Conti, A. Passarella, Design and performance evaluation of contentplace, a social-aware data dissemination system for opportunistic networks, Computer Networks 54 (2010) 589–604. <http://dx.doi.org/10.1016/j.comnet.2009.09.001>.
[390] J. Reich, A. Chaintreau, The age of impatience: optimal replication schemes for opportunistic networks, in: Proceedings of the 5th international conference on Emerging networking experiments and technologies, CoNEXT '09, ACM, New York, NY, USA, 2009, pp. 85–96. <http://doi.acm.org/10.1145/1658939.1658950>.
[391] V. Lenders, M. May, G. Karlsson, C. Wacha, Wireless ad hoc podcasting, SIGMOBILE Mobium Computing Communications Review 12 (2008) 65–67. <http://doi.acm.org/10.1145/1374512.1374535>.
[392] R. Brooks, Disruptive security technologies with mobile code and peer-to-peer networks, CRC Press, 2004.
[393] R. Steinmetz, K. Wehrle, Peer-to-Peer Systems and Applications (Lecture Notes in Computer Science), Springer-Verlag, New York, Inc. Secaucus, NJ, USA, 2005.
[394] Wrapster, <http://www.unwrapper.com>.
[395] L. Garber, Denial-of-service attacks rip the Internet, Computer 33 (4) (2000) 12–17.
[396] J. Mirkovic, P. Reiher, A taxonomy of ddos attack and ddos defense mechanisms, SIGCOMM Computer Communications Review 34 (2) (2004) 39–53. <http://doi.acm.org/10.1145/997150.997156>.

[397] N. Naoumov, K. Ross, Exploiting p2p systems for ddos attacks, in: Proceedings of the 1st international conference on Scalable information systems, ACM, New York, NY, USA, 2006.
[398] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, D.S. Wallach, Secure routing for structured peer-to-peer overlay networks, SIGOPS Operating Systemc Review 36 (SI) (2002) 299–314. <http://doi.acm.org/10.1145/844128.844156>.
[399] J.R. Douceur, The sybil attack, in: IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems, Springer-Verlag, London, UK, 2002, pp. 251–260.
[400] D. Wallach, A survey of peer-to-peer security issues, Lecture Notes in Computer Science (2003) 42–57.
[401] Z. Despotovic, K. Aberer, P2P reputation management: Probabilistic estimation vs. social networks, Computer Networks 50 (4) (2006) 485–500.
[402] L. Xiong, L. Liu, A reputation-based trust model for peer-to-peer e-commerce communities, in: IEEE International Conference on E-Commerce, 2003. CEC 2003, 2003, pp. 275–284.
[403] K. Wongrujira, T. Hsin-Ting, A. Seneviratne, Avoidance routing to misbebaving nodes in P2P by using reputation and variance, in: Advanced Communication Technology, 2004. The 6th International Conference on, Vol. 2, 2004.
[404] N. Salem, L. Buttyan, J. Hubaux, M. Jakobsson, Node cooperation in hybrid ad hoc networks, IEEE Transactions on Mobile Computing 5 (4) (2006) 365.
[405] T. Ngan, D. Wallach, P. Druschel, Enforcing fair sharing of peer-to-peer resources, Proceedings IPTPS 2003.
[406] H. Xie, Y. Yang, A. Krishnamurthy, Y. Liu, A. Silberschatz, P4P: Provider portal for (P2P) applications, in: Proceedings of ACM SIGCOMM, 2008.
[407] T. Karagiannis, P. Rodriguez, K. Papagiannaki, Should internet service providers fear peer-assisted content distribution? in: Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement, USENIX Association Berkeley, CA, USA, 2005, pp. 6–6.
[408] Shen, HPTP: Relieving the Tension between ISPs and P2P, in: Proceedings of IPTPS, 2007.
[409] F5 White Paper, Bandwidth management for peer-to-peer applications <http://www.f5.com/solutions/technology/rateshaping_wp.html>, January 2006.
[410] V. Aggarwal, A. Feldmann, C. Scheideler, Can ISPs and p2p users cooperate for improved performance? SIGCOMM Computer Communications Review 37 (3) (2007) 29–40.
[411] D. Choffnes, F. Bustamante, Taming the torrent: a practical approach to reducing cross-isp traffic in peer-to-peer systems, Proceedings of SIGCOMM.
[412] K.-A. Skevik, V. Goebel, T. Plagemann, Design of a hybrid cdn, in: Proceedings of Interactive Multimedia and Next Generation Networks, 2004.
[413] D. Xu, S.S. Kulkarni, C. Rosenberg, H. keung Chai, A cdn-p2p hybrid architecture for cost-effective streaming media distribution, Computer Networks 44 (2004) 353–382.
[414] D. Xu, S.S. Kulkarni, C. Rosenberg, H.K. Chai, Analysis of a CDN-P2P hybrid architecture for cost-effective streaming media distribution, Springer Multimedia Systems 11 (2006) 383–399.
[415] C. Huang, A. Wang, J. Li, K.W. Ross, Understanding hybrid cdn-p2p: why limelight needs its own red swoosh, in: NOSSDAV '08: Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video, ACM, New York, NY, USA, 2008, pp. 75–80. <http://doi.acm.org/10.1145/1496046.1496064>.
[416] D. Pakkala, J. Latvakoski, Towards peer to peer extended content delivery network, in: Proceedings of the 14th IST Mobile and Wireless Communications Summit, 2005.
[417] A. MacQuire, A. Brampton, N.J.P. Race, L. Mathy, M. Fry, A case for hybrid content distribution for interactive video-on-demand, in: NGMAST '08: Proceedings of the 2008 The Second International Conference on Next Generation Mobile Applications, Services, and Technologies, IEEE Computer Society, Washington, DC, USA, 2008, pp. 556–561. <http://dx.doi.org/10.1109/NGMAST.2008.68>.
[418] V. Jacobson, If a clean slate is the solution, what was the problem? Stanford University Seminars, 2006.
[419] S.E. Deering, RFC 1112: Host extensions for IP multicasting, 1989.
[420] R. Braden, D. Clark, S. Shenker, RFC1633: Integrated Services in the Internet Architecture: an Overview, 1994.
[421] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, RFC 2475: An Architecture for Differentiated Service, 1998.
[422] RFC 3344: IP Mobility Support for IPv4, 2002.
[423] The FET-IST FP6 ANA Project, <http://www.ana-project.org/>.
[424] The Stanford Clean Slate Research Programme, <http://cleanslate.stanford.edu/index.php>.
[425] The Japanise AKARI Project, <http://akari-project.nict.go.jp/eng/index2.htm>.
[426] B.T. Loo, J.M. Hellerstein, I. Stoica, R. Ramakrishnan, Declarative routing: extensible routing with declarative queries, in: SIGCOMM '05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications, ACM, New York, NY, USA, 2005, pp. 289–300. <http://doi.acm.org/10.1145/1080091.1080126>.
[427] A. Venkataramani, D. Towsley, NSF-FIND Project: A swarming architecture for Internet data transfer (2007, <http://www.nets-find.net/Funded/Swarming.php>).
[428] H. Schulzrinne, S. Seetharaman, V. Hilt, NSF FIND Proejct: NetSerV V Architecture of a Service-Virtualized Internet 2007, <http://www.nets-find.net/Funded/Netserv.php>.
[429] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K.H. Kim, S. Shenker, I. Stoica, A data-oriented (and beyond) network architecture, in: SIGCOMM '07: Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications, ACM, New York, NY, USA, 2007, pp. 181–192. <http://dx.doi.org/10.1145/1282380.1282402>.
[430] NetArch 2009 Symposium, March 2009, <http://www.netarch2009.net/>.
[431] M.D. Kudlick, RFC 608: Host names on-line, 1974.
[432] K. Lougheed, Y. Rekhter, RFC 1163: Border Gateway Protocol (BGP) (1990).
[433] V. Jacobson, Congestion avoidance and control, in: SIGCOMM '88: Symposium proceedings on Communications architectures and protocols, ACM, New York, NY, USA, 1988, pp. 314–329. <http://doi.acm.org/10.1145/52324.52356>.
[434] 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: IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement, ACM, New York, NY, USA, 2007, pp. 1–14. <http://doi.acm.org/ 10.1145/1298306.1298309>.
[435] Myspace, <http://www.myspace.com>.
[436] Facebook, <http://www.facebook.com>.
[437] The Clean-Slate Research Program at Stanford, <http://cleanslate.stanford.edu/>.
[438] V. Jacobson, D.K. Smetters, J.D. Thornton, M.F. Plass, N.H. Briggs, R.L. Braynard, Networking named content, in: Proceedings of the 5th international conference on Emerging networking experiments and technologies, CoNEXT '09, ACM, New York, NY, USA, 2009, pp. 1–12. <http://doi.acm.org/10.1145/1658939.1658941>.