Computer Networks 57 (2013) 3128–3141

Contents lists available at ScienceDirect

Computer Networks

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

Caching in information centric networking: A survey

Guoqiang Zhang [a,c,*], Yang Li [b], Tao Lin [b]

[a] School of Computer Science and Technology, Nanjing Normal University, Nanjing, China
[b] High Performance Network Lab, Chinese Academy of Sciences, Beijing, China
[c] Provincial Key Laboratory for Computer Information Processing Technology, Soochow University, Suzhou, China

ARTICLE INFO

Article history:
Received 3 January 2013
Received in revised form 31 May 2013
Accepted 3 July 2013
Available online 8 August 2013

Keywords:
Information-centric networking
In-network cache
Cache coordination
Cache decision
Object availability

ABSTRACT

Internet usage has drastically shifted from host-centric end-to-end communication to receiver-driven content retrieval. In order to adapt to this change, a handful of innovative information/content centric networking (ICN) architectures have recently been proposed. One common and important feature of these architectures is to leverage built-in network caches to improve the transmission efficiency of content dissemination. Compared with traditional Web Caching and CDN Caching, ICN Cache takes on several new characteristics: cache is transparent to applications, cache is ubiquitous, and content to be cached is more ine-grained. These distinguished features pose new challenges to ICN caching technologies. This paper presents a comprehensive survey of state-of-art techniques aiming to address these issues, with particular focus on reducing cache redundancy and improving the availability of cached content. As a new research area, this paper also points out several interesting yet challenging research directions in this subject.

1. Introduction

Internet traffic has experienced rapid growth in the past several years. According to Cisco's VNI report [18], the global IP traffic has increased eight times in the past five years, and the CAGR (compound annual growth rate) is anticipated to be 29% during 2011–2016. Most of the traffic could be attributed to content retrieval applications. It is predicted that video traffic alone will account for 86% of all the IP traffic in 2016 [18]. With increasing demand for UGC (User-Generated Content), time-shift TV and high definition VoD, traffic generated by receiver-driven digital video content retrieval will continue its high growth rate [8].

To better cope with the Internet usage shift from sender-driven end-to-end communication paradigm to receiver-driven content retrieval paradigm, a handful of innovative information/content centric networking architectures have been proposed [15,32,28,65,5,1,3,30,31], e.g., DONA [32], CCN [28], NetInf [1], and PRISP [3]. A notable advantage of these architectures is to provide native support for scalable and highly efficient content retrieval, and meanwhile with enhanced capability for mobility and security. Interested readers can refer to Ref. [2,42,17] for a thorough survey of ICN architectures.

In ICN, users do not care *where* the content comes from, but are only interested in *what* the content is. The philosophy behind ICN is to promote content to a first-class citizen in the network. Instead of centering around IP addresses, ICN locates and routes content by unified content names, essentially decoupling content from its location. For example, the revolutionary proposal of CCN attempts to replace the current narrow waist of IP with named content chunks (see Fig. 1).

In order to alleviate the pressure that rapid traffic growth imposes on network bandwidth, a common approach of ICN is to provide transparent, ubiquitous in-network caching to speed up content distribution and improve network resource utilization. Though caching is already a useful tool employed by present Internet (e.g.,

* Corresponding author at: School of Computer Science and Technology, Nanjing Normal University, Nanjing, China. Tel.: +86 13851435685.
E-mail address: guoqiang@ict.ac.cn (G. Zhang).

1389-1286/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.comnet.2013.07.007

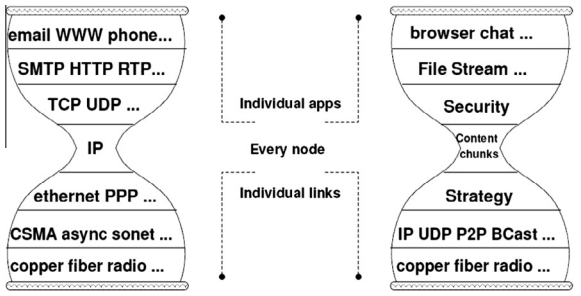


Fig. 1. In the CCN proposal, the current Internet narrow waist (left) is replaced by named content chunks (right).

Web, P2P [20,62,66]) to reduce bandwidth consumption, lack of unique identification of identical objects makes it hard to take advantage of caching. For example, the URL is used as both identifier and locator of Web objects. When two copies of the same object are placed in different servers owned by different content providers, different URLs will be used to identify and access the content. As a result, the Web caching system will treat them as different objects.

Consequently, although caching theory and techniques to optimize the caching system have already been extensively studied, new features of ICN caching–transparency, ubiquity and fine-granularity–have made traditional caching theories, models and optimization techniques developed for hierarchical Web caching and CDN caching systems unable to be directly and seamlessly ported to ICN caches.

This paper presents a thorough survey of caching techniques in ICN, in many cases tracing its origin to precursive Web caching or P2P caching. The following paper is structured as follows. Section 2 investigates new features of ICN caching and their impacts on ICN cache design. Techniques for ICN cache performance improvement are categorized and explored in detail in Section 3. Section 4 presents several challenging research directions in this subject, and finally Section 5 concludes this paper.

2. New features of ICN caching

2.1. Cache transparency

Traditional cache systems are closed, application-dependent systems designed for one particular traffic class, e.g., Web, CDN or P2P. Though Web caching is based on open HTTP protocol, Web contents follow domain-based naming convention. Two copies of the same object in different domains have different names. So to the caching system, objects are logically segregated by domain boundaries. P2P applications typically use proprietary protocols, which makes each P2P application a closed system. In order to overcome this, researchers are also trying to make the cache more transparent to applications. For example, the IETF DECADE proposal [23,56] intends to provide a shared cache infrastructure so that each application can manage the cache space independently. However, lack of unified naming convention still makes it hard to achieve cross-application cache sharing. The problem of *protocol closeness* and *naming inconsistency* can be gracefully addressed within the ICN infrastructure. Firstly, ICN names content in a unified and consistent way. Often, these names are also self-certifiable [32,3,1,5,26,2], simplifying security checking of content.[1] Secondly, ICN makes its routing and caching decisions on unified content names, essentially making these names network-aware. This feature makes caching become a general, open, and transparent service, independent of applications.

Cache transparency, however, also poses several challenges, including:

- *Inconsistency between caching objectives*
  Objectives of traditional caching systems are often simple, but may vary with each other. For example, Web cache intends to reduce network traffic and user-perceived latency, whereas P2P cache primarily concerns the reduction of network traffic. As a new network infrastructure, ICN is expected to serve a variety of network traffic including Web, VoD, file sharing, etc. These traffic types have varied caching objectives, so ICN should make reasonable choice of its caching objective to balance between diverse traffic types.
- *Cross-application competitive sharing of cache space*
  Different types of traffic differ significantly in their population scale, object size and object popularity [25]. For example, the population of Web objects is enormous, in the order of billions, but object size is typically small. In contrast, the population size of VoD traffic is much smaller, in the order of 10^5, but the object size is much larger. The high heterogeneity of these traffic places new challenges on traditional dedicated, closed caching systems, requiring ICN to be able to efficiently share cache resources between different traffic types.
- *Line-rate operation of caches*Transparency of in-network caches also poses new requirement on the operation speed of ICN caches. It is suggested that ICN caches should operate in line rate, which makes the cache management quite different from traditional disk-based management [46,43].

2.2. Cache ubiquity

In traditional caching systems, cache locations are often predetermined and typically the cache topology is purposefully formed in a linear cascading structure (Fig. 2(a)) or hierarchical tree structure (Fig. 2(b)). Content placement and coordination between caches can be determined by solving the analytical model established by prior traffic demand and cache structure; in ICN, caches are ubiquitous, and caching points are no longer fixed. Topology of the cache network evolves from hierarchical trees to arbitrary graphs (Fig. 2(c)), so the fixed parent-child

[1] Self-certification is a key enabler for the shift from end-to-end paradigm to content-centric paradigm. Without this, security checking of transmitted content should rely on end-to-end authentication of endpoint hosts or users. While with self-certification, content can be retrieved from anywhere without worrying about its validity.

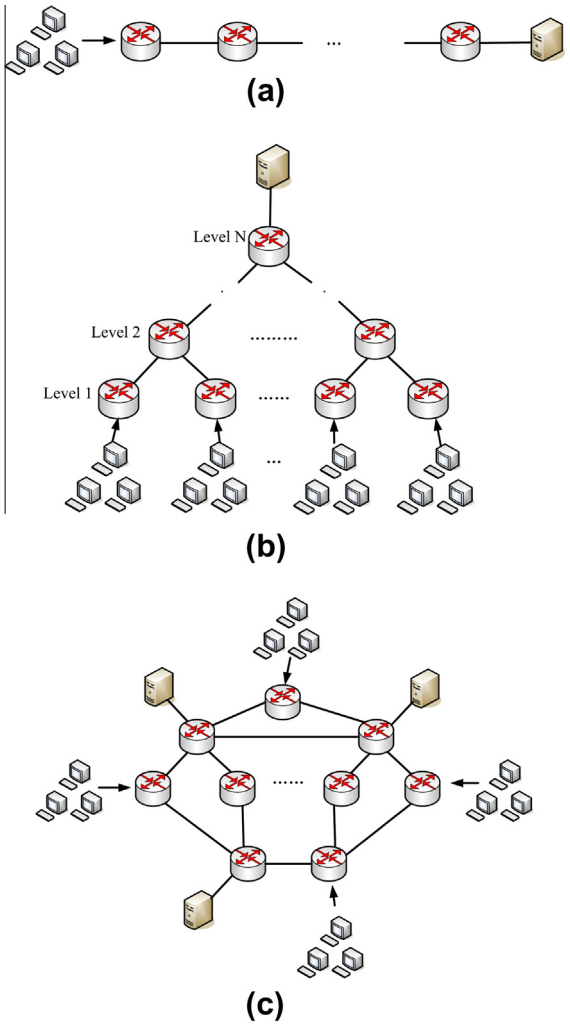


Fig. 2. Cache topologies: (a) linear cascading cache structure; (b) hierarchical tree cache structure; (c) general graph cache network.

relationship vanishes. These factors add difficulties to the mathematical modeling and analysis of the caching system, and also make explicit coordination harder to achieve. Cache ubiquity also makes the availability of cached content more subtle. In traditional Web and CDN caches, whether a cached content is available to a request is clear. In CDN, content is proactively pushed and copied to edge servers based on prior knowledge of access demand and network structure. The system is based on mechanisms such as redirection and DNS resolution to ensure global availability of these content copies. In hierarchical Web caches, only the content located along the path from the requesting point to the root is available to a given request, while content cached outside the route is not available to that request. But in ICN, the situation is changed. The caching system in ICN exhibits high dynamics because its general cache network topology, ubiquity of in-network caches and volatility of cached content. If the presence information of all the volatile objects cached in in-network caches is to be announced either to the global registration or routing system, the system scalability will be severely degraded due to high volume of update messages. In addition, high dynamics also makes it hard to maintain system consistency.

2.3. Fine-granularity of cached content

Most ICN proposals use the technique of slicing large files into small self identifiable chunks, and perform cache operations on the unit of chunks [28,10,7,30,31]. This change of cache unit raises the following issues:

- *Change of popularity*. Object popularity in the file-level has received extensive studies. For instance, it is well established that the access frequencies of Web objects and P2P objects follow Zipf distribution [9] and Mandelbrot-Zipf distribution [54,27] respectively. However, file-level object popularity cannot be simply extended to chunk-level object popularity because different chunks of a single file can have different access frequencies. For example, users will often make their decision of whether to watch the whole video or not depending on the forepart of the video, which results in differentiated access frequencies for different chunks. To date, there is neither analytical modeling nor experimental study of the chunk-level object popularity.
- *Failure of independent reference assumption*. Traditional file-based caches are typically based on the assumption that requests follow the so-called *independent reference model*, i.e., the probability that a given object will be requested is only determined by its popularity, independent of previous requests. This assumption, however, is invalidated in chunk-level caching. Requests for different chunks of the same file are often correlated, e.g., in sequential order.
- *Opportunity for more efficient use of the cache space*. Caching and replacing at chunk-level instead of file-level objects with unified names makes it possible to retrieve different parts of the same file from different nodes, which speeds up the retrieval rate and improves the space utilization. Though P2P also uses a similar tactic, the lack of unified naming across different P2P applications makes cross-application cache reuse impossible.

3. Techniques for ICN performance optimization

Since caching moves from a middle-ware optimization mechanism in the present Internet to a fundamental architectural building block in ICN to realize efficient content retrieval, it is thus of vital importance to optimize its performance. Transparency, ubiquity and fine granularity of the ICN caching, not only place new challenges, but also offer new opportunities to the development of caching techniques. Performance optimization for ICN caching can be carried out in a wide range of dimensions. In the following, we will discuss these mechanisms in detail.

3.1. Cache dimensioning

When other settings are identical, larger cache storage size means more content can be cached, hence higher

cache hit ratio. However, larger cache size also means additional lookup overhead for a single cache. Since an ICN cache should operate at line rate, the cache size that can be installed at each caching node is thus limited. There are two issues that remain to be addressed:

1. The first question is how large the cache space should be to have noticeable performance improvement? In traditional Web caching, no limitation is imposed on storage size, in contrast, requirement for line-rate operation limits the cache size in ICN. But meanwhile, it is expected that ICN will be used to carry all sorts of content transmission, even larger than the aggregate of Web content. Hence, if the cache size is too small, it is possible that the expectation will not be met. So, it is preferred to configure the cache size based on the router's performance disparity, to meet the line-rate operation requirement. Thus, it is necessary to have a thorough understanding of the details of each hardware memory technologies, such as access speed, maximum allowed size, cost per storage unit and power consumption. This knowledge can serve as useful input to the decision choice of different memory technologies for different functional components in ICN [43]. For example, in CCN, if a node operates at linerate of 1–100 Gbps, each Interest packet is 40 bytes, each PIT entry takes 56 bits, and each request is stored in the PIT for an average lifetime of 80ms, then the PIT should be sized between 14M bits to 1.4G bits.[2] So, we should consider choosing RLDRAM rather than SRAM, since the latter is not large enough to satisfy this purpose.
2. The second question is how to allocate the storage resource across different cache nodes so that the cache performance of the whole system can be optimized, given that the total storage resource is fixed? This is indeed a network dimensioning problem under a given budget. Cache space allocation needs to take account of network topology and traffic demands. It has been shown that allocating cache size based on centrality[3] can only have marginal effect on performance improvement, and this marginal improvement can even be achieved with a much simpler allocation scheme—degree based allocation [53], i.e., the cache capacity allocated to a node is proportional to its node degree. Assuming the node degree of content router i is $d(i)$, and the total cache space is fixed to be C, then the cache capacity allocated to node i is:

$$C(i) = \frac{d(i)}{\Sigma_j d(j)} \quad (1)$$

In another study, it was shown that allocating more storage resource to edge routers rather than core routers is beneficial for performance improvement [46]. These studies together show that when network topology is considered, a resource allocation scheme should not simply be based on the node's static topological centrality, but should be based on the distance from the caching node and users as well as its serviced user population. Meanwhile, since simple cache space allocation alone cannot achieve considerable performance improvement, it is necessary to integrate the cache space allocation with object placement and object query policies.

3.2. Cache sharing mechanism

A direct consequence of cache transparency is that different types of traffic/applications have to contend with and share the storage space in a single cache node. These different types of traffic/applications vary in traffic characteristics and cache optimization objectives. How to effectively share limited resources between different types of traffic, while at the same time having the ability to provide differentiated services, becomes an urgent task that ICN caching needs to address.

3.2.1. Differentiated caching service

The significance of providing differentiated caching services is emphasized in the study by Lu [39], which can be summarized from three points of view.

- From *an end user's point of view*, cache is more important and useful for users with fast access speed. If the bottleneck is located in the core network, then cache can substantially reduce the transmission delay, otherwise, if the network bottleneck is in the edge network, e.g., the access link, then the utility of cache will be greatly reduced. Hence, cache should serve users with faster access speed with higher priority. Realization of this, depends on the network's ability to *distinguish between different access technologies*, e.g., Wireless Markup Language (WML) developed for low speed wireless access networks.
- From *the content point of view*, it may be possible to improve client-perceived performance by caching the most "noticeable" content more often. For example, it has been observed that different classes of web contents contribute differently to users' perception of network performance. User-perceived performance depends more on the downloading latency of HTML pages than on the downloading latency of embedded objects.[4] Hence, the caching system should treat HTML pages as a premium class to improve the quality of experience of users. This requires that the system should be able to *distinguish between different content classes*.
- From *a content provider's point of view*, ISPs may have different contracts with different content providers to provide some sites with a better service for a negotiated price. This requires the caching system to *distinguish between different content providers*, which, for example, can be realized by the URL or the principal identifer of the content.

[2] Take 1Gps as an example, the PIT size is calculated as $10^9/(40 \times 8) \times 80 \times 10^{-3} \times 56 = 14 \times 10^6$ bits.

[3] Centrality is a general graph property that characterizes how central a node is located within the graph. There are several centrality measures, such as degree centrality, betweenness centrality and closeness centrality.

[4] This assumption may no longer be true today, since HTML pages are embedded with more and more objects today. But the general argument still holds.

3.2.2. Techniques for cache space sharing

In order to realize differentiated caching services, it is necessary to have supporting cache sharing techniques. The main choice in designing a sharing mechanism is whether the sharing is fixed or dynamic, as discussed in the following:

- *Cache sharing based on fixed partition*
  This refers to partitioning the cache space into fixed shares so that each class of applications gets its own share which will not be used by other traffic classes even if it is underutilized. In this way, the performance of each traffic class will not be interfered with by other traffic classes. There are two subtle questions remaining to be addressed. The first question is how to partition the space across different traffic classes? The second is whether each node should follow the same partition ratio, or different nodes could have different partition ratios for different traffic classes? To the former question, Ref. [39] proposed an approach to adaptively adjust the partition ratio based on the dynamic feedback control theory. To the latter issue, it is observed that to achieve the same hit ratio, a VoD application requires much smaller cache space than UGC, file sharing and Web applications [25]. Hence it is desirable to dedicate the lowest-level cache closest to users to VoD applications alone. This approach, in some sense, suggests different partition ratios for different nodes according to node property.
  Although fixed partition-based sharing is easy to implement, it is however, hard to achieve efficient and dynamic sharing between different application classes. It is often the case that caching systems set the lifetime for cached objects.[5] Once a TTL is associated with a specific application, the volume of effective data for that application is about the product of its arrival rate and the TTL. As a result, if fixed partition is used, some applications will not exhaust their allocated cache space at all times, while other applications may at some time interval run out of their cache space.
- *Dynamic cache sharing*
  Dynamic sharing allows one traffic class to use the cache space if no other traffic class at that moment needs it. Two dynamic sharing policies are typically used: priority-based sharing and weighted fair sharing [12]. Priority-based sharing gives some applications higher service priority. When an object arrives at and needs to be cached at a node, but there is no space to accommodate this object, this cache node repeatedly removes from its cache space those objects belonging to applications with lower or equal priorities[6] until sufficient free space is reserved for this object. Pure priority-based schemes, however, may severely degrade the performance of low priority applications if high priority traffic arrives at a constant high rate. Similar to fixed partition, weighted fair sharing also intends to share cache resources between different application classes according to predefined weights, but different from fixed partition, weighted fair sharing allows one traffic class to take the share of other traffic classes if the space is unused at that moment, hence increasing utilization.

3.3. Cache decision policy

Cache decision policy determines which objects are to be placed at which cache nodes. It is an active and possibly the hottest research area in ICN caching. In the traditional Web cache or CDN cache, it is sometimes possible to figure out the optimal object placement based on apriori knowledge of network topology and traffic demand. This approach is typically called explicitly coordinated cache decision. But in ICN, cache nodes are no longer fixed, traffic classes are diverse, and cache operation needs to be line-rate. These factors have made explicit cache coordination algorithms unsuitable for ICN due to their high complexity and communication overhead. ICN needs more simplified yet effective cache decision policies.

One simplest policy called leave copy everywhere (LCE) is to copy the object at each node along the downloading path, which is the default cache decision policy in most ICN architectures (e.g., CCN). This approach, however, introduces high *cache redundancy*, i.e., the same object is unnecessarily copied at multiple nodes, which reduces the diversity of the whole system's cached contents. In order to improve the whole cache system's utility, intuitively it is necessary to: (1) push high popularity contents quickly to the edge of the network, so that user's downloading latency can be reduced and network resource utilization can be improved, and (2) improve the whole network's *cache diversity*, especially within the same ISP, so that a user's request can be served by the same ISP, resulting in greatly reduced inter-domain traffic [60,37].

Reducing cache redundancy and improving cache diversity need simple and effective coordination between cache nodes. These coordination schemes can be classified in several ways. According to the degree of autonomy in the cache decision making process, present cache coordination schemes can be classified as either *explicit* or *implicit*. According to the decision point, these coordination schemes can be classified as either *centralized*or *decentralized*.

3.3.1. Explicit cache coordination decision

In a typical *explicit cache coordination* scheme, object access pattern, cache network topology and each cache's state are used as inputs for the calculation of the placement position of each object. This prerequisite information is either obtained offline or by online communication. Nodes participating in the explicit coordination can vary in scope. Common approaches can be classified into three categories: *global coordination*, *path coordination* and *neighborhood coordination*.

Global coordination means cache coordination involves all cache nodes, which is often used in CDN. In a CDN, the set of cache nodes, network distances between them,

[5] Two motivations drive the setting of TTLs for cached objects: content will get outdated, e.g., live streaming data becomes useless in seconds; TTL-based opportunistic caching can improve cache scalability without the support of explicit cache coordination.

[6] When priorities are the same, cache replacement policy determines which object is to be evicted.

and the object access frequencies at each cache node are assumed to be known apriori, then the provider needs to work out the optimal object placement in order to minimize users' access cost, either in a centralized manner [33] or a decentralized manner [8].

Path coordination means the coordination only involves the cache nodes along the path from the request hit place to the requesting client, e.g., coordinated en-route web caching (CERWC) [57,29]. Objects can only be (but not necessarily) copied along this path. This approach typically piggybacks the information needed for coordination in the content requesting packet. The information typically includes the state of each cache node and the object access frequency at each cache node. When the request arrives at a hit node, the hit node extracts all the needed state information of the path, and calculates the optimal object placement positions as well as the objects that need to be replaced at those positions.

Neighborhood coordination means coordination takes place among a node's neighborhood. How the neighborhood is defined varies with different schemes. For example, it can include a node's direct neighbors, or two-hop neighbors. Cooperative In-Network Caching (CINC) [37] belongs to this category which is based on a hash function. Fig. 3 shows how it operates. When a chunk arrives at a cache node, it uses a hash function to determine in which of its neighbors (including itself) to cache this chunk. In this way, it prevents the same object from being duplicated in a node's neighborhood. When a request comes, the hash function is also used to consult the responsible node to determine whether the chunk is cached or not. In [49], two other approaches are mentioned. In the first approach, a node caches an object only when this object has no copy in all of its neighbors, and when replacement is needed, those objects which have copies in this node's neighborhood are first replaced. In the second approach, a node caches an object only when its parent node does not contain a copy of the object, and when replacement is needed, those objects which have copies in this node's parent node are first replaced.

Global coordination, path coordination and neighborhood coordination reduce cache redundancy respectively in the global scope, path scope and neighborhood scope. However, all these explicit cache coordination schemes require pre-known network cache state information and user's access frequency information, and also incur considerable information exchange and computation overhead to make the final cache decision, which in most cases are too complex for ICN.

3.3.2. Implicit cache coordination

Different from explicit cache coordination, in *implicit cache coordination*, each cache node does not need to know the state information of other cache nodes or only needs to exchange very little information with other cache nodes, before it can autonomously determine whether to cache the object or not.

Hierarchical caching is one example of implicit coordination. The special cache network topology makes the edge network capable of identifying popular requests, and pulling high-popularity content objects down to the network edge. Due to the filtering effect, high-level cache nodes will only cache objects with moderate or low popularity, which to some extent realizes coordination. However, in ICN, the cache network topology evolves to an arbitrary graph, so it is hard to define hierarchy relationship between different cache nodes. In addition, exogenous requests can occur at any cache node. So, the above scheme is no longer effective for ICN.

Quite some implicit cache coordination schemes are proposed to alleviate the high cache redundancy introduced by LCE [34,35,16,24,38,46,41]. Fig. 4 illustrates some typical schemes, which are described in detail in the following:

- *Leave Copy Down (LCD)* [34,35]: When a cache hit occurs, this scheme only caches the object at the direct downstream node, which avoids a large number of copies of the same object. LCD also implies that several number of requests are needed to pull the object down to the edge network, which implicitly considers the objects' access frequencies.
- *Move Copy Down (MCD)* [34,35]: When a cache hit occurs, this scheme moves the object from the hit node to its direct downstream node, and deletes the object from the hit node. Comparing with LCD, MCD reduces object redundancy more aggressively.

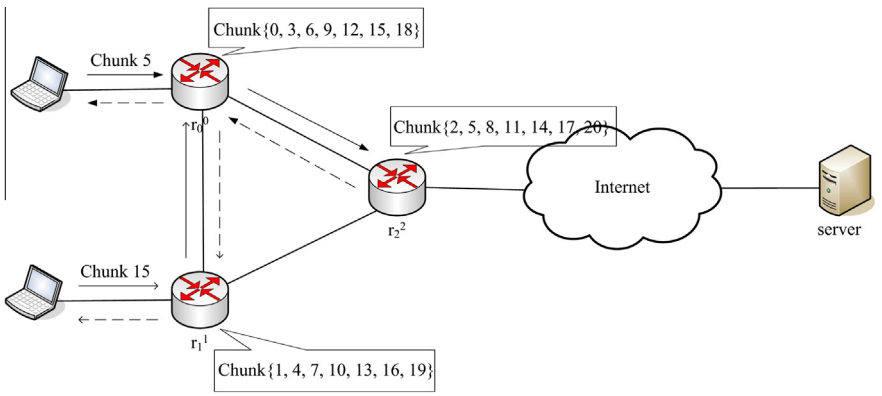


Fig. 3. The operation of coordinated in-network caching (CINC). Here a simple modular 3 hash function is used for chunk-level object placement and search.

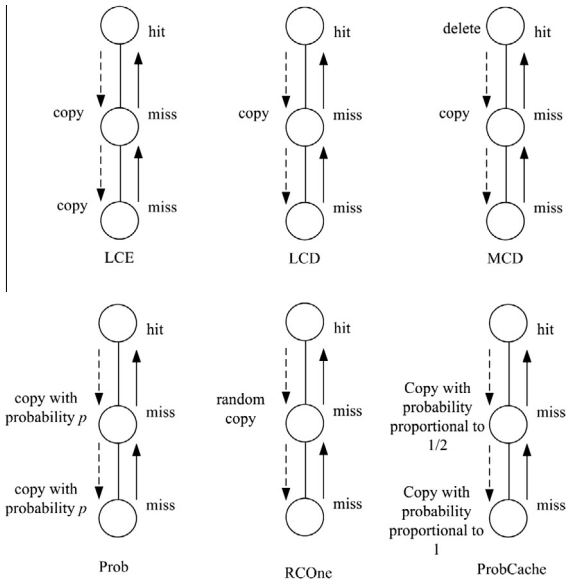


Fig. 4. Illustrative operation of different implicit cache decision schemes.

- *Copy with Probability (Prob)* [34]: The requested object is copied with a given probability p at each node along the returning path. This approach can be thought of as a generalization of LCE. When $p = 1$, it degenerates to LCE.
- *Randomly Copy One (RCOne)* [24]: This scheme copies the requested object at one random node along the returning path. For a hierarchical cache with l-levels, this approach equates the Prob approach with $p = 1/l$.
- *Probabilistic Cache (ProbCache)* [46]: In this scheme, the requested object is copied at each node with a probability. But, for each node, the probability varies. The probability is inversely proportional to the distance from the requester and this node. Hence, if the node is close to the requester, the object will be copied with high probability. Conversely, if the node is far from the requester, the object has little chance of being copied. This approach has the advantage that it can quickly push the copy to the network edge and meanwhile reduce the number of copies.

3.3.3. Timing of cache decision

In the aforementioned schemes, cache decision is only made when a new object arrives. But, cache decision can also be made at the moment of cache replacement. When an object is to be replaced, simply removing it from the cache system may not be the best choice, instead, the object can be pushed back to one-level upstream the cache hierarchy for caching, possibly incurring a cascading of object replacements, e.g., CLS [38] and Demote [63]. The rational behind this idea is that objects cached closer to the network edge often have higher access frequencies than objects cached upstream in the cache hierarchy.

3.3.4. Correlation between cache decisions

Most of the researches mentioned so far make cache decision independently of each other, i.e., whether an object is to be cached is independent of other objects. However, in ICN, requests for chunks can be correlated, e.g., in a sequential order. This reality thus calls for correlated cache decision.

WAVE [16] is an attempt towards this trend. Fig. 5 illustrates the operation of this strategy. For each file, WAVE adjusts the number of chunks cached at each node based on the file's popularity. When the number of requests for a file increases, WAVE reacts with exponential increase in the number of chucks cached for this file. A content router in WAVE explicitly sets the cache indication mark (e.g., in CCN's data packet), which makes its direct downstream content router cache this chunk. Once the chunk is cached, the cache indication mark is cleared. This mechanism is similar to LCD, i.e., the chunk is pushed one hop towards the requester each time. However, different from LCD, WAVE counts the access frequency of a given file. When a file is first accessed, the first chunk will be marked to be cached. When the file is accessed for the second time, the next two chunks are marked to be cached. In general, when the file is accessed for the nth time, the next 2^{n-1} chunks are marked. As a result, if a file is accessed frequently, dissemination of the file will be sped up. In contrast, files with low access frequencies will consume very few cache resources.

3.3.5. Comparison

Cache decision policy is a hot research topic in ICN caching. Table 1 compares the caching coordination schemes mentioned so far from several aspects: whether the coordination is explicit or implicit, the timing of cache decision, the basis for cache decision, whether the cache decision takes request correlation into account, the degree of resulting cache redundancy, and the dissemination speed of objects down to network edge.

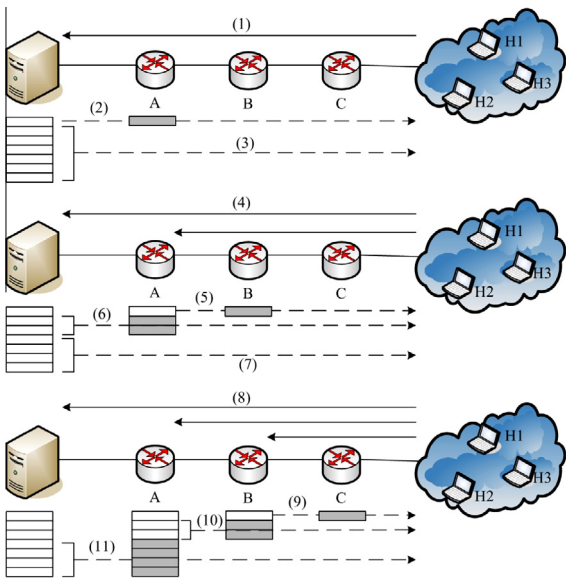


Fig. 5. Operation of WAVE.

Table 1
Comparison between different cache decision policies. The acronyms are mentioned in previous subsections.

	Coordination manner	Decision timing	Decision basis	Correlation	Cache redundancy	Dissemination speed
LCE	No coordination	Object arrival	No	No	High	Fast
CERWC	Path explicit	Object arrival	Cache gaining	No	Low	Fast
CINC	Neighborhood explicit	Object arrival	Chunk identifier	No	Low	Fast
LCD	Implicit	Object arrival	Object popularity	No	Medium	Slow
MCD	Implicit	Object arrival	Object popularity	No	Low	Slow
Prob	Implicit	Object arrival	Random decision	No	Depends on p	Depends on p
RCOne	Implicit	Object arrival	Random decision	No	Low	Slow
ProbCache	Implicit	Object arrival	Distance between cached node and requesting node	No	Medium	Fast
CLS	Implicit	Object arrival and object replacement	Object popularity	No	Low	Slow
Demote	Implicit	Object arrival and object replacement	Object popularity	No	Low	Slow
WAVE	Implicit	Object arrival	File-level object popularity	Yes	Low	Slow

3.4. Cache replacement algorithm

Cache replacement algorithms have received extensive research in the Web Cache literature. See Ref. [44] for a thorough survey on this topic. However, in ICN, cache replacement should be performed as fast as possible, so complex replacement algorithms are not suitable for this purpose. The requirement for simplicity overruns the need for efficiency. Recent studies have also shown that in ICN, a simple random replacement algorithm suffices to achieve similar performance attained by LRU [52,24]. As a result, cache replacement algorithm is not a hot research topic in ICN cache.

3.5. Object availability

Whether a cached object is available to a content request largely affects the cache performance. In a typical setting widely adopted in web caching, only the objects cached along the path from the requester to the root cache are available to the request. There is no coordination between sibling caches.

In general, object availability is determined by two factors: *object visibility* and *object search scheme*. Object visibility means to what extent the presence information of a cached object will be propagated so that other cache nodes know its existence. While an object search scheme refers to how to search the object when an request comes before forwarding the control of the request to the next hop towards the source.

There are two extremes of object visibility: one extreme is that a cached object is only visible to the resident node, whereas the other extreme is that a cached object is visible to all nodes. Between these two extremes, there is wide design space. A cached object can be visible to a subset of cache nodes. The subset can vary for different schemes. Visibility can also vary by its degree. For example, one could maintain a direct mapping between a cached object's identifier and its resident node, which gives this node explicit information of how to contact the resident node. Or, it is also possible to only maintain routing hints or index abstraction information for the cached object.

According to different object visibility schemes, different object search schemes can be used to make the cached object vary in availability. For instance, when an object is only locally visible, then to make the object globally available, it is necessary to use a more aggressive search scheme, such as request flooding. If a conservative search scheme is used, e.g., the requested object is only searched locally at each node enroute to the source node, then the cached object is only available along the request forwarding path.

Generally speaking, wider object visibility means higher chance for the request to be routed to appropriate cache nodes. However, improving object visibility also incurs additional overhead due to the following two reasons. On one hand, presence information needs to be advertised within a much larger scope. Since cached objects present much higher dynamism than objects in original content repositories, the presence information should also be updated with much higher frequency. Conjunction of these two issues will lead to uncontrollable volume of advertisement information. On the other hand, high dynamism also makes it hard to maintain consistency state for the system as a whole. So, a hot research topic in ICN is how to improve the object availability at an acceptable cost level. According to the scope of object availability, present researches can be classified into the following three categories.

3.5.1. Path availability

In traditional Web cache and most of ICN cache studies, a content request message is routed either by unicast or anycast to the source and searched locally at each cache node enroute, until a cache hit occurs during the process [14,47,48,57,29,55,28,32]. In this paradigm, only cached objects along the path are available to the content request. Even if the requested object is cached in some node next to a node along the path, and the cost of fetching the content from this node is far cheaper than fetching it from the upstream node along the path, the request will still be routed

to the source, which obviously, is inferior. Hence, a common recognition is that in ICN, either the routing system or registration system should be utilized to enlarge the visibility of cached objects, so that these objects are available to those requests even if these requests would not pass through those nodes with the desired objects cached [2].

3.5.2. Global availability

In CDN, content copies are placed on demand. Presence of these copies is registered in a global resolution system so that every copy is visible and thus available to content requests. It is the responsibility of the resolution mechanism that determines which copy is to be used. Instead of using a global resolution system, an alternative is to advertise the presence of an object through the global routing system to ensure that every copy be globally visible and hence available. In ICN, the global routing system or registration system is used in the similar way, but only to publish the presence information of objects located in source repository nodes, e.g., CCN and DONA. Objects cached in intermediate cache nodes, however, are not published globally. Hence, if searching is performed locally, objects in in-network caches still only have path availability to content requests.

Because of the large volume and volatile nature of cached objects in ICN in-network caches, it is inappropriate to register these objects in a global resolution system or publish the presence of these objects into the global routing system. Doing so will cause unendurable overhead and raise scalability concern. However, path availability alone prevents us from using the full power of networked cache system. Consequently, how to balance between the scope of object availability to requests and the incurred overhead, is an urgent issue that needs to be addressed by ICN researches.

3.5.3. Partial availability

One possible solution to the aforementioned issue is to make the volatile objects cached in in-network caches be partially visible to the network, in hope of achieving good tradeoff between object availability and system scalability. Specifically, there are several ways to do this.

A straightforward way is to perform cache query. When a cache miss occurs, do not forward the request upstream immediately. Instead, use a cache query protocol, such as ICP [61], to query its neighboring nodes to determine whether the requested object is cached at these neighbors. If none of its neighbors has the requested object, the node forwards the control of the request to the next hop towards the source of the requested object. However, blindly querying neighboring cache nodes without any knowledge of the contents stored at them will increase the bandwidth overhead as well as perceived latency.

One improvement that can be made is to index the objects cached within a node's neighborhood. When a cache miss occurs, the index table is first looked up to find out whether an appropriate neighboring node can be contacted. This approach has received extensive research in the distributed cache and P2P cache systems [48,58,13,40,64], but has only recently received attention in the ICN field [67,37]. In CONIC [67], contents are placed at end nodes, while indices of contents are placed at content routers. So, CONIC can be viewed as something between a pure P2P network where routers are agnostic to contents and a pure content-centric networking architecture where contents are stored at and perceivable to routers. In CINC [37], contents and indices are all stored at content routers. A hash function is used to determine the chunk numbers each neighborhood node is responsible for.

An alternative to store direct content indices is to maintain indirect routing hints so that potential cached objects can be found by following these hints. This approach is exemplified by Breadcrumb [49] and CATT [59]. The idea of this approach, however, is not new. It can be traced back to object replication and searching in P2P networks, e.g., Routing Indices (RIs) [21] and Freenet [19].

In the Routing Indices proposal, each participating peer node maintains a table of routing hints, indicating the number of objects (and optionally a measure of routing cost, e.g., number of hops) that can be retrieved for any subject through each neighboring peer. When a peer receives a request, it makes an optimal choice to forward the request based on the subject of the request and its own routing hints table. Freenet identifies each object based on its content hash. Each node in Freenet maintains a local datastore and a dynamic routing table. The dynamic routing table contains the addresses of other nodes and the object identifiers these nodes are thought to hold. When a request arrives at a node, this node first searches its local store. If the requested object is not not found, then this node looks up the nearest identifier in its routing table to the identifier requested and forwards the request to the corresponding node. If that request is ultimately successful and returns with the object, the node will pass the object back to the upstream requestor, cache the object in its own datastore, and create a new entry in its routing table associating the actual object source with the requested object identifier.

Breadcrumb borrows similar ideas from these P2P inventions. In the proposed Breadcrumb approach [49,59], when an object is returned along the requesting path, in addition to caching the object en-route, this scheme also leaves a trail that records the forwarding direction of the object and the time when the object passes. When a new request arrives, if a cache hit occurs, then the object is returned and the process is finished. Otherwise, depending on the trail information, it makes a choice between forwarding the request downstream along the trail, or forwarding the request upstream towards the source. If forwarding downstream along the trail fails to find the desired object, it resorts to the standard forwarding as its fallback. In this sense, the routing hint is only used as an optimization. This is why this approach is also termed BECONS (Best effort CONtent Search). CATT [24] proposes a slightly different approach, which depends on potential based routing. For any file c, node n and node n_c that caches c, the potential value at node n is proportional to the expected quality of content c at n_c,[7] and inversely

[7] The quality of content can have several interpretations, for example, a caching node with high capacity outgoing links or high processing power can be assumed to provide high quality content.

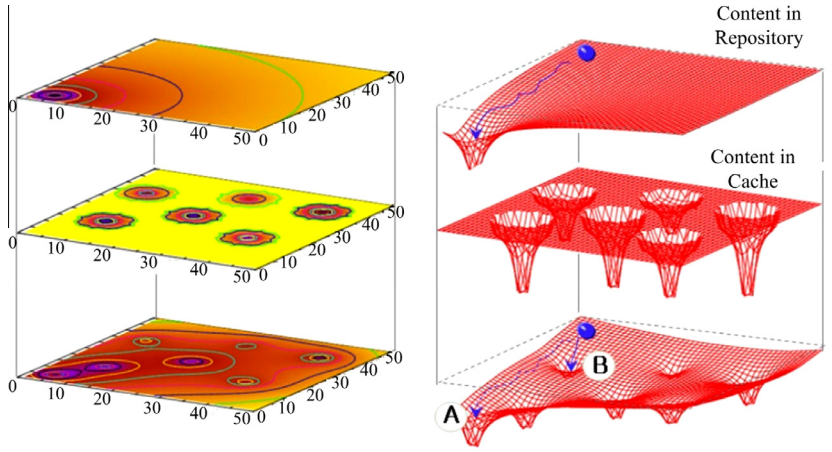


Fig. 6. Three types of potential fields. The one on the top is permanent potential field (PPF), the one in the middle is volatile potential field (VPF), and the one at the bottom is the combination of PPF and VPF. The blue ball represents a request for a content *c*, which, depending on the potential value of each node for *c*, could be routed to *c*'s original repository, or to a cache node that temporarily caches *c*.

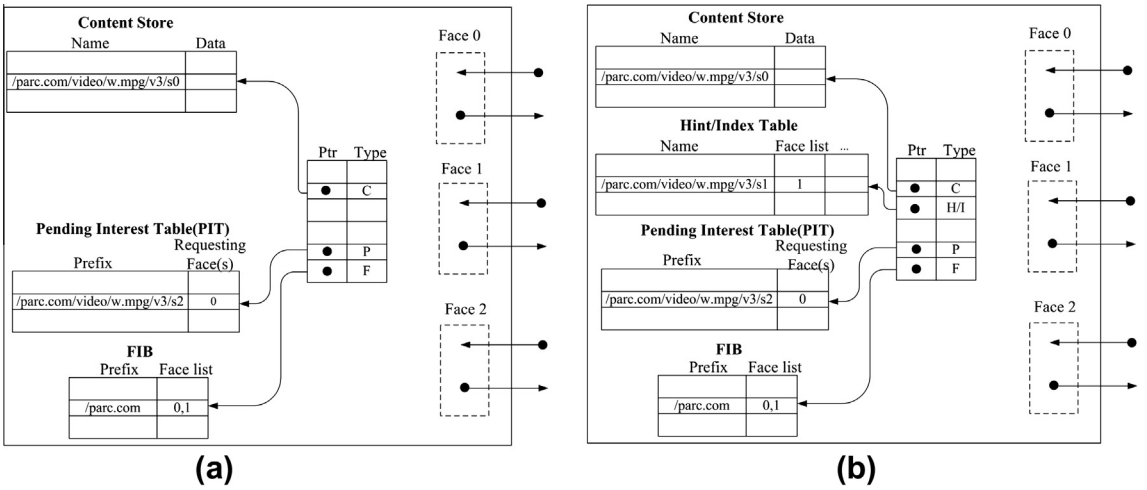


Fig. 7. An example showing how to modify the CCN forwarding engine to accommodate the content index or routing hints table: (a) original CCN forwarding engine; (b) modified CCN forwarding engine with content index or routing hints table.

proportional to the distance from n_c to n. When multiple caching nodes exist, the potential value of n is simply defined as the linear summation of the potential values which are influenced by individual caching nodes. Whenever a request for content c arrives at node n and n cannot satisfy the request, CATT forwards the request towards a neighbor of n that maximizes the potential difference between n and itself. In order to deal with the conflict between availability and network adaptability, two potential fields are defined. One is *permanent potential field (PPF)*, which is used for objects located in source repositories. Since these objects are relatively stable, the presence information can be advertised in global scope without having to be frequently updated, making these objects globally available. The other is *volatile potential field (VPF)*, which applies to objects in in-network caches. These objects are constantly getting replaced. So, to avoid the scalability issue caused by frequent updating, their presence information is only advertised within limited scope, making these cached objects only partially visible. A useful system can make use of combination of these two potential fields (CPF), which effectively balances between object availability and system scalability. Fig. 6 exemplifies the notion of these potential fields.

For most approaches that utilize partial object availability, an additional component should be introduced into the cache node. This additional component can be either a content index table or a routing hints table. Furthermore, an additional search step should be inserted into the request processing procedure to make use of partially visible objects in networked caches. When a request comes, at least three steps should now be involved: (1) look up its own data/content store; (2) if not found, then locate or search the object according to the content index table or routing hints; (3) if no object can be found, then resort to the routing mechanism to forward the request towards the data source. This is different from P2P request processing, as

Table 2
Object availability for different schemes. Here neighborhood visible means a cached object is visible within a vicinity ball of direct or indirect neighbors, whereas a cached object is partially visible means the object is visible to a subset (not necessarily in the node's neighborhood) of nodes.

	Visibility	Searching policy	Availability	Additional information to be stored
DONA	Node	Local	Path	No
CCN	Node	Local	Path	No
CERWC	Node	Local	Path	No
CONIC	Neighborhood	Query forwarding by index table	Neighborhood available	Object index table
CINC	Neighborhood	Query forwarding by hash function	Neighborhood available	Labels of neighborhood nodes
RIs	Globally visible	Query forwarding by routing hints table	Globally available	Routing hints table, proportional to the number of semantic subjects
Freenet	Partially visible	Query forwarding based on the mapping table between objects and source established by historic forwarding; if it fails, forward the query based on the object identifier	Partially available	Mapping between object identifiers and node identifiers, as well as labels of neighboring nodes
Breadcrumb	Partially visible	Forwarding the query downstream according to routing hints established by historic forwarding	Partially available	The five tuple associated with the objects that pass through this node within a time interval
CATT	Neighborhood	query forwarding based on potential value difference	Neighborhood available	Each object's potential value, proportional to the number of objects

there is no fallback in P2P if the search fails. Fig. 7 presents an example showing how to modify the CCN forwarding engine to achieve this purpose.

3.5.4. Comparison

In ICN, object availability is primarily a decision issue for volatile cached objects. For stable objects in source repositories, the global routing or resolution system can be used to ensure their global availability. So, here, we only compare the availability of non-persistent objects for different cache schemes.[8] Table 2 summarizes and compares these different schemes.

4. Challenges and future research directions

Although caching has made markedly notable progress since the birth of information/content-centric networking, it is still a new research area full of challenges. There are a lot of issues needing to be addressed. We list some important yet challenging problems here.

4.1. Chunk-level object popularity

Object popularity is one of the major properties that affect cache efficiency. In ICN, we should consider chunk-level object popularity, rather than file-level object popularity. However, till this writing, to the best of our knowledge there is neither analytical nor experimental study on chunk-level object popularity.

This study can be carried out in two directions. From the analytical point of view, one can establish the chunk-level object popularity model from prior knowledge. These include established knowledge about file-level object popularity, distribution of object size and reasonable assumption on users' access behavior for chunks. From the experimental point of view, since presently there is no large-scale operational ICN network infrastructure and applications, we have no way to measure the chunk-level object popularity directly. Instead, we can resort to P2P systems such as PPLive [68] to collect statistics about block-level object popularity. Certainly, the size of a block in P2P systems is different from the size of a chunk in ICN. However, their requesting behavior are similar, so the results are analogous, and can be used at least as a reference for chunk-level object popularity in ICN.

4.2. Correlation between requests and correlation-based cache decision

Caching on chunks invalidates the assumption that requests follow the so-called independent reference model. However, present caching theories and techniques are still based on this invalidated assumption. What is the inherent correlation between different requests, how to model this correlation, and how to optimize the cache decision policy based on the request correlation, is a potential way to improve the performance of the whole cache system. An intuitive recognition is that requests for different chunks of the same file are made in a sequential order, however, there are neither rigid mathematical models nor experimental studies on this subject. WAVE [16] has verified that the cache decision policy leveraging the sequential correlation between requests can indeed improve cache performance, which opens vast space for future research.

4.3. ICN friendly network topology

In the traditional end-to-end transmission paradigm, HOT model [4,36] is regarded as the optimal network topology because it can maximize the throughput of the whole network. However, ICN fundamentally changes the

[8] Although RIs and Freenet are object caching and searching schemes for P2P networks, their application scenario and underlying ideas are similar to the ICN cache network. They are instructive for ICN cache design, so we also include them as candidates for comparison. Here, each node in a P2P network is analogous to a cache node in an ICN's cache network.

end-to-end transmission mode. Hence, a problem worth of exploration is what kind of network topology is suitable for ICN network. Presently, the research on this subject is still in the vacuum state.

4.4. Low complexity cache decision and object query mechanisms

Intelligent cache decision policies can reduce cache redundancy and increase the diversity of cached contents. However, making full use of the content diversity needs complementary cache location mechanisms. How to devise and bundle low complexity implicit cache decision policies and the corresponding intelligent cache location schemes in the face of high dynamic in-network cache environment, still remains an active research direction.

4.5. Analytical modeling of an ICN cache network

Just as queuing theory is the foundation for understanding the behavior of packet switching networks, analytical modeling and theories of cache networks are also indispensable for the understanding of the behavior of cache networks. Early studies of cache theory are for a single cache node [22] or for special cache network topologies, e.g., linear topology or hierarchical tree topology [47,48,14]. These special network topologies simplify the interaction between cache nodes, and hence simplify the establishment and analysis of cache network models. However, the topology of ICN cache networks can no longer be represented as hierarchical trees, but should be represented by arbitrary graphs. Though recently several studies have been devoted to the ICN cache network modeling, they are still for hierarchical network topologies [10,11,45,6] except for the work carried out by Rosenswei et al [50,51].

The milestone work by Rosenswei has proposed approximate models for general cache networks and presented several sufficient conditions for the network to achieve steady state. Their work lays the foundation for future work. However, many issues remain untouched. Firstly, the sufficient and necessary conditions to achieve steady state in the proposed model still need to be explored. Secondly, the model adopts many assumptions used in hierarchical network modeling, which are no longer valid or need to be reexamined in ICN cache networks (e.g., invalidation of independent reference model, unknown chunk-level object popularity), so cache network modeling should be based on more appropriate assumptions. Finally, present cache network modeling is based on the LCE cache decision policy, not considering other cache decision and object search policies. However, as we see in the previous section, the emphasis of present techniques for ICN cache performance improvement is around optimizing cache decision policy and increasing object availability. So, modeling the cache network based on more efficient cache decision and object search policies can have more practical significance. Until now, there is no work in this direction.

5. Conclusion

As the primary communication paradigm gradually shifts from host-centric end-to-end communication to receiver-driven content retrieval, a number of innovative information/content centric networking architectures have recently been proposed. These architectures intend to provide native architectural support for highly efficient and scalable content retrieval applications and to solve the traffic explosion problem from bottom up. In these network architectures, transparent and ubiquitous in-network caches are the fundamental building block that guarantees efficient content retrieval. As a result, caching in ICN emerged as a hot research topic in recent years.

In this paper, we first investigated the new features of ICN caches and their potential influences and challenges for ICN caching techniques. Then we explored various aspects affecting ICN caching performance, including cache dimensioning, application-independent cache space sharing in a single cache node, cache decision policies, cache replacement policies and availability of cached contents. Finally, we pointed out several research directions worthy of further exploration. Generally speaking, research on ICN caching is still in its early stage. There are many theoretical as well as technical issues remaining to be addressed. The solution of these problems will have significant impacts on the dimensioning, designing and operation of real ICN networks.

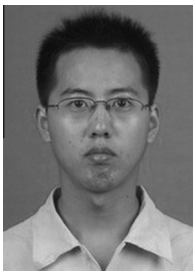
Acknowledgements

This work is partly supported by the National Natural Science Foundation of China under Grant Nos. 61100178, 61174152, 61303243 and 61202419, the Hi-Tech Research and Development Program of China under Grant No. 2013AA013503, and Strategic Pilot Project of Chinese Academy of Sciences under Grant No. XDA06010302.

References

[1] B. Ahlgren et al., Second NetInf Architecture Description, 4WARD EU FP7 Project, Deliverable D-6.2 v2.0, April 2010, FP7-ICT-2007-1-216041-4WARD/D-6.2, <http://www.4ward-project.eu/>.
[2] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, B. Ohlman, A survey of information-centric networking, IEEE Communications Magazine (2012).
[3] M. Ain et al., D2.3-Architecture Definition, Component Descriptions, and Requirements, Deliverable, PSIRP 7th FP EU-funded Project, February 2009.
[4] D. Alderson, L. Li, W. Willinger, J.C. Doyle, Understanding Internet topology: principles, models, and validation, IEEE/ACM Transactions on Networking 13 (6) (2005) 1205–1218.
[5] A. Anand, F. Dogar, D. Han, et al., XIA: an architecture for an evolvable and trustworthy Internet, in: Hotnets, 2011.
[6] J. Ardelius, B. Gr?nvall, L. Westberg, A. Arvidsson, On the effects of caching in access aggregation networks, ICN (2012).
[7] S. Arianfar, P. Nikander, Packet-level caching for information-centric networking, in: ACM SIGCOMM, ReArch Workshop, 2010.
[8] S. Borst, V. Gupta, A. Walid, Distributed caching algorithms for content distribution networks, IEEE INFOCOM (2010).
[9] L. Breslau, P. Cao, L. Fan, G. Phillips, S. Shenker, Web caching and zipf-like distributions: evidence and implications, IEEE INFOCOM'99 1 (1999) 126–134.

[10] G. Carofiglio, M. Gallo, L. Muscariello, Bandwidth and storage sharing performance in information centric networking, ICN (2011).
[11] G. Carofiglio, M. Gallo, L. Muscariello, D. Perino, Modeling data transfer in content-centric networking, France Telec., Technical Report, 2011.
[12] G. Carofiglio, V. Gehlen, D. Perino, Experimental evaluation of memory management in content-centric networking, ICC (2011).
[13] Y. Chawathe, S. Ratnasamy, L. Breslau, Making Gnutella-like P2P Systems Scalable, in: ACM SIGCOMM, 2003. (1084).
[14] H. Che, Y. Tung, Z. Wang, Hierarchical web caching systems: modeling, design and experimental results, IEEE Journal on Selected Areas in Communications 20 (7) (2002) 1305–1314.
[15] D. Cheriton, M. Gritter, TRIAD: A New Next-Generation Internet Architecture, January 2000.
[16] K. Cho, M. Lee, K. Park, T.T. Kwon, Y. Choi, S. Pack, WAVE: popularity-based and collaborative in-network caching for content-oriented networks, in: IEEE INFOCOM Workshop on NOMEN, 2012.
[17] J. Choi, J. Han, E. Cho, et al., A survey on content-oriented networking for efficient content delivery, IEEE Communications Magazine 49 (3) (2011) 121–127.
[18] Cisco visual networking index: forecast and methodology: 2011–2015, May 2012.
[19] I. Clarke, O. Sandberg, B. Wiley, T.W. Hong, Freenet: a distributed anonymous information storage and retrieval system, in: Workshop on Design Issues in Anonymity and Unobservability, 2000.
[20] E. Cohen, S. Shenker, Replication strategies in unstructured peer-to-peer networks, in: SIGCOMM, 2002. (650).
[21] A. Crespo, H.G. Molina, Routing indices for peer-to-peer systems, in: ICDCS, 2002. (902).
[22] A. Dan, D.F. Towsley, An approximate analysis of the LRU and FIFO buffer replacement schemes, in: SIGMETRICS, 1990, pp. 143–152.
[23] DECADE working group, 2010, <https://datatracker.ietf.org/wg/decade>.
[24] S. Eum, K. Nakauchi, M. Murata, Y. Shoji, N. Nishinaga, CATT: potential based routing with content caching for ICN, ICN (2012).
[25] C. Fricker, P. Robert, J. Roberts, N. Sbihi, Impact of traffic mix on caching performance in a content-centric network, in: IEEE INFOCOM Workshop on NOMEN, 2012.
[26] A. Ghodsi, T. Koponen, J. Rajahalme, P. Sarolahti, S. Shenker, Naming in content-oriented architectures, in: ICN 2011, August, Tronto, Ontario, Canada, 2011.
[27] M. Hefeeda, O. Saleh, Traffic modeling and proportional partial caching for peer-to-peer systems, IEEE/ACM Transactions on Networking 16 (2006) 1447–1460.
[28] V. Jacobson, D.K. Smetters, J.D. Thornton, et al., Networking named content, in: CoNEXT'09, 2009.
[29] A.X. Jiang, J. Bruck, Optimal content placement for en-route web caching, in: Second IEEE International Symposium on Network Computing and Applications, 2003, pp. 9–16.
[30] K. Katsaros, G. Xylomenos, G.C. Polyzos, MultiCache: an incrementally deployable overlay architecture for information-centric networking, IEEE INFOCOM (2010).
[31] K. Katsaros, G. Xylomenos, G.C. Polyzos, MultiCache: an overlay architecture for information-centric networking, Computer Networks 55 (4) (2011) 936–947.
[32] T. Koponen, M. Chawla, B.G. Chun, A. Ermolinskiy, K.H. Kim, S. Shenker, I. Stoica, A data-oriented (and beyond) network architecture, in: ACM SIGCOMM, 2007.
[33] M.R. Koruolu, M. Dahlin, Coordinated placement and replacement for large-scale distributed caches, IEEE Transactions on Knowledge and Data Engineering 14 (6) (2002) 1317–1329 (142).
[34] N. Laoutaris, S. Syntila, I. Stavrakakis, Meta algorithms for hierarchical web caches, in: Proceedings of the 2004 IEEE International Performance, Computing and Communications Conference, 2004, pp. 445–452.
[35] N. Laoutaris, H. Che, I. Stavrakakis, The LCD interconnection of LRU caches and its analysis, Performance Evaluation 63 (7) (2006) 609–634.
[36] L. Li, D. Alderson, J.C. Doyle, W. Willinger, Towards a theory of scale-free graphs: definition, properties, and implications, Internet Mathematics 2 (4) (2006) 431–523.
[37] Z. Li, G. Simon, Time-shifted TV in content centric networks: the case for cooperative in-network caching, in: Proceedings of ICC, 2011.
[38] Y. Li, T. Lin, H. Tang, P. Sun, A chunk caching location and searching scheme in content-centric networking, ICC (2012).
[39] Y. Lu, T.F. Abdelzaher, A. Saxena, Design, implementation and evaluation of differentiated caching services, IEEE Transactions on Parallel and Distributed Systems (2004).
[40] Q. Lv, P. Cao, E. Cohen, K. Li, S. Shenker, Search and replication in unstructured peer-to-peer networks, in: ICS'02, 2002. (1894).
[41] Z. Ming, M. Xu, D. Wang, Age-based cooperative caching in information-centric network, in: IEEE INFOCOM Workshop on NOMEN, 2012.
[42] S. Paul, J. Pan, R. Jain, Architectures for the future networks and the next generation Internet: a survey, Computer Communications 34 (2011) 2–42.
[43] D. Perino, M. Varvello, A reality check for content centric networking, in: ACM SIGCOMM ICN Workshop, 2011.
[44] S. Podlipnig, L. B?SZ?rmenyi, A survey of web cache replacement strategies, ACM Computing Surveys 35 (4) (2003) 374–398.
[45] I. Psaras, R.G. Clegg, R. Landa, W.K. Chai, G. Pavlou, Modelling and evaluation of CCN-caching trees, Networking (2011) (22).
[46] I. Psaras, W.K. Chai, G. Pavlou, Probabilistic in-network caching for information-centric networks, ICN (2012).
[47] P. Rodriguez, C. Spanner, E.W. Biersack, Web caching architectures: hierarchical and distributed caching, in: 4th International Caching Workshop, 1999.
[48] P. Rodriguez, C. Spanner, E.W. Biersack, Analysis of web caching architectures: hierarchical and distributed caching, IEEE/ACM Transactions on Networking 9 (4) (2001) 404–418.
[49] E.J. Rosensweig, J. Kurose, Breadcrumbs: efficient, best-effort content location in cache networks, IEEE INFOCOM (2009).
[50] E.J. Rosensweig, J. Kurose, D. Towsley, Approximate models for general cache networks, IEEE INFOCOM (2010).
[51] E.J. Rosensweig, D.S. Menasche, J. Kurose, On the steady-state of cache networks, IEEE INFOCOM (2012).
[52] D. Rossi, G. Rossini, Caching performance of content centric networks under multi-path routing (and more), Technical Report, Telecom ParisTech, 2011.
[53] D. Rossi, G. Rossini, On sizing CCN content stores by exploiting topological information, in: IEEE INFOCOM Workshop on NOMEN, 2012.
[54] O. Saleh, M. Hefeeda, Modeling and caching of peer-to-peer traffic, in: Proc. 2006 Int. Conf. Network Protocols (ICNP'06), 2006, pp. 249–258.
[55] H. Shen, S. Xu, Coordinated en-route web caching in multi-server networks, IEEE Transactions on Computers 58 (5) (2009) 605–619.
[56] H. Song, N. Zhong, Y. Yang, et al., Decoupled application data ENROUTE (DECADE) problem statement, draft-ietf-decade-problem-statement-00.txt, August 2010.
[57] X. Tang, S.T. Chanson, Coordinated en-route web caching, IEEE Transactions on Comuters 51 (6) (2002) 595–607.
[58] R. Tewari, M. Dahlin, H.M. Vin, Design considerations for distributed caching on the Internet, in: Proceedings of 19th IEEE International Conference on Distributed Computing Systems, 1999, pp. 273–284.
[59] T. Tsutsui, H. Urabayashi, M. Yamamoto, E. Rosensweig, J.F. Kurose, Performance evaluation of partial deployment of breadcrumbs in content oriented networks, in: 5th International Workshop on the Network of the Future (in conjunction with ICC), 2012.
[60] G. Tyson, S. Kaune, S. Miles, Y. EI-Khatib, A. Mauthe, A. Taweel, A trace-driven analysis of caching in content-centric networks, in: ICCCN, 2012.
[61] D. Wessels, K. Claffy, Application of Internet cache protocol (ICP) version 2, IETF draft: draft-wessels-icp-v2-appl-00, May 1997.
[62] A. Wierzbicki, N. Leibowitz, M. Ripeanu, et al., Cache replacement policies revisited: the case of P2P traffic, in: Proceedings of the 4th International Workshop on Global and Peer-to-Peer Computing (GP2P'04), Chicago, 2004, pp. 182–189.
[63] T.M. Wong, J. Wilkes, My cache or yours? Making storage more exclusive, in: Usenix Association Proceedings of the General Track, 2002, pp. 161–175.
[64] Beverly Yang, Hector Garcia-Molina, Efficient search in peer-to-peer networks, in: ICDCS, 2002. (1068).
[65] L. Zhang, D. Estrin, J. Burke, et al., Named Data Networking (NDN) Project (2010).
[66] G. Zhang et al., P2P traffic optimization, Science China Information Sciences (2012).
[67] Y. Zhu, M. Chen, A. Nakao, Conic: content-oriented network with indexed caching, in: IEEE INFOCOM Workshops, 2010.
[68] http://www.ppltv.com.cn/.



Guoqiang Zhang was born in 1980. He is now an associate professor in the School of Computer Science and Technology, Nanjing Normal University, China. He received his Ph.D. from Institute of Computing Technology, Chinese Academy of Sciences in March 2008. His research interests include Future Networks, P2P traffic optimization, network designing, Internet topology mapping and modeling.

Dr. Yang Li received the BS degree in signal and information processing in Harbin Institute of Technology, Heilongjiang, China, in 2004, the PhD degree in Communication Networking in Kyungpook National University, Korean, in 2009. She is currently an assistant professor in Institute of Acoustics, Chinese Academy of Sciences. Her research interests include content distribution, future Internet, and broadband wireless communication.

Dr. Tao Lin is an associate professor of High Performance Network Lab, Institute of Acoustics, Chinese Academy of Sciences (CAS). His research interests include information-centric network, in-network caching management, server selection and request routing mechanism for future large-scale content distribution system. In earlier years, he also devoted himself on the research of mobility management and distributed multimedia communication. He has published more than 30 research papers in international journals and conferences in those areas.