공학박사 학위논문

Routing and Caching in Information-Centric Networking

정보 중심 네트워크에서의 라우팅 및 캐싱

2015년 2월

서울대학교 대학원
전기 · 컴퓨터공학부
최훈규

Routing and Caching in Information-Centric Networking

지도교수 권태경

이 논문을 공학박사 학위논문으로 제출함

2014년 11월

서울대학교 대학원

전기 · 컴퓨터공학부

최훈규

최훈규의 박사 학위논문을 인준함

2014년 12월

위 원 장	김 종 권	(인)
부위원장	권 태 경	(인)
위 원	엄 현 상	(인)
위 원	전 병 곤	(인)
위 원	백 상 헌	(인)

Abstract

Routing and Caching in Information-Centric Networking

Hoon-gyu Choi
School of Computer Science & Engineering
The Graduate School
Seoul National University

When the Internet was designed decades ago, main applications are resource sharing such as remote login and file transfer. To support such applications, the key principle in the Internet architecture is point-to-point communications, and the key element is an IP address that identifies a host. Due to the flexible design of the Internet, a wide range of new applications and services have been introduced over the decades. The recent surge of Internet traffic is mainly attributed to applications such as web, P2P file sharing, and video streaming. In such applications, an end user is mostly interested in content itself, not in a particular host or its location.

Over the past few years, there have been many efforts to address the above issues from a content centric perspective. Those proposals are collectively called Information Centric Networking (ICN), which is largely deemed as a clean-slate approach. Most of the ICN studies think of content as a key element and hence assume a new paradigm by shifting from host-oriented communications to content-oriented

communications. Consequently, instead of locator-based routing, most ICN proposals consider name-based routing, which decouples content production and consumption in time and space domains. The decoupling enhances content availability and naming persistency, and supports in-network caching, multicast and mobility.

Most of ICN proposals use content names as routing entries, and thus the routing scalability is primary concern. ICN allows in-network caching as a built-in functionality. However, if network nodes make caching decisions individually, duplicate copies of the same content may exist among nearby nodes. To address these problems, this dissertation proposes a unified framework named Coordinated Routing and Caching (CoRC) that mitigates routing scalability and enhances the efficiency of the in-network storage.

Keywords : Information-Centric Networking, Content-Centric Networking, Routing, Caching

Student Number : 2008-20999

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Due to the flexible design of the TCP/IP protocols, a wide range of new applications and services has been proliferated on the Internet over the decades. According to the Cisco report [1], the recent surge of Internet traffic is mainly attributed to applications such as web, P2P file sharing, and video streaming. In such applications, an end user is mostly interested in content itself, not in a particular host or its location.

The gap between the original host-to-host Internet design and the current content-oriented usage patterns causes many problems such as inefficient content delivery and flash crowds. For example, when thousands of people request the same content, it might be forwarded over the same link thousands of times. This is because ordinary network nodes are not aware of the content due to the host-based IP routing. Content Delivery Networks (CDNs) [2] mitigate this inefficiency by locating popular contents to nearby storages. However, relying on CDN providers is not considered a fundamental and general solution, as (i) it requires substantial monetary cost to content providers of various sizes, (ii) it may burden Internet service providers (ISPs) depending on the locations of CDN storages, and (iii) it may not adapt to the time-varying content popularity (e.g., flash crowds). Also, the host-based TCP/IP protocol suite cannot handle mobility and security properly.

Over the past few years, there have been many efforts to address the above issues from a content centric perspective. Those proposals are collectively called

Information-Centric Networking (ICN), which is largely deemed as a clean-slate approach. Most of the ICN studies think of content as a key element and hence assume a new paradigm by shifting from host-oriented communications to content-oriented communications [3–7]. One of the key advantages in ICN comes from in-network caching; when a request encounters a network node that caches the content of interest, the node sends back the content immediately without contacting the original server. The more frequently an item is requested, the more likely the item is to be retrieved from a close in-network cache, not from the original publisher. By decoupling content production and consumption in the time and space domains, ICN enhances content availability and naming persistency, and naturally supports mobility, security and multicast.

This dissertation proposes a framework of Coordinated Routing and Caching (CoRC) to address the following challenges in ICN. (i) How to maintain Forwarding Information Bases (FIBs) scalable?, (ii) Which content will be cached by which node in order to efficiently utilize the network-wide storage?, and (iii) How to reach a nearby cached item without additional signalling overhead? By partitioning the whole content namespace and assigning each partition to a dedicated node, CoRC mitigates routing scalability and enhances caching efficiency with no control message exchanges. We combine routing and caching into a unified framework, while many prior proposals deal with routing and caching separately.

We have evaluated the CoRC framework in terms of routing scalability and cache efficiency. The first aspect is routing scalability, which is to figure out how many FIB entries are stored by a CoRC router. Based on the previous data, a FIB size in a CoRC router required for Inter/Intra-domain routing was calculated by projecting

the number of ISP/AS and the number of publishers to the year of 2030. Our results showed that a CoRC router can have a feasible size of a FIB. The second aspect is a comprehensive performance considering caching efficiency. Since routing and caching were designed in one framework in CORC, it is required to explore strengths and weaknesses for each of them. Therefore, we defined four variants in total including our CoRC framework by giving an option whether each of routing and caching is partitioned or not respectively. We implemented four variants of software router prototypes, and configured 4 ASes with 60 nodes in the Amazon EC2 service [8] to measure downloading time, hop count, cache hit ratio, and inter-AS traffic. Finally, packet processing time in a router is evaluated to show the feasibility of CoRC.

The rest of this dissertation is organized as follows. In chapter 2, we describe design principles behind ICN routing and caching. Related studies that seek to solve the ICN routing and caching issues are reviewed in chapter 3. The routing and caching mechanisms in the CoRC framework are detailed in chapter 4. Optimization methods are discussed to reduce the route stretch of CoRC in chapter 5. The FIB size of a router in CoRC is analyzed in chapter 6, and the network performance of the CoRC framework is evaluated in chapter 7. Packet processing time of CoRC forwarding in Linux machines is shown in chapter 8. Additional design issues to accommodate realistic conditions are discussed in chapter 9. The concluding remarks are given in chapter 10.

Chapter 2

Design Principles

2.1 How to Make FIBs Scalable?

Each router in ICN should be able to interpret a content name (instead of an IP address) in its FIB. Though several ICN proposals take a hierarchical name structure that can be aggregatable for routing scalability, making FIBs scalable is still a problem. For example, two routing entries in an FIB for content items whose names are *cnn.com/us/news* and *cnn.com/eu/news* can be aggregated to *cnn.com*. Assuming that content names are aggregated to their publisher names (e.g., *cnn.com*), a FIB needs to contain as many entries as the number of domain names. According to [9], the number of domain names currently registered with the Domain Name System (DNS) is approximately 10^9, which means a router should have 10^9 entries in its FIB. Thus, even if we assume only publisher names in a FIB, its size is order of magnitude higher than that of a current IP router in the default-free zone. Unfortunately, the current hardware capability of network nodes can hardly meet this requirement [10]. To reduce the FIB size of an ICN router to the level of that of an IP router, CoRC is designed based on the following two principles: (i) exploiting the hierarchy of current Internet and (ii) partitioning the FIB space (e.g., the whole content namespace) among ICN routers.

2.2 Where to Place the Cached Item?

Cache storage in a router is a limited resource, hence it is desirable to avoid caching duplicate copies of the same content across ICN routers for cache utilization. When data is sent back in ICN, each router on the path may cache the item in its local storage. If routers (with in-network storage modules) make caching decisions individually, caching redundant copies of the same content will happen frequently. In this case, the cache hit ratio for popular items will be high, and a small number of selected items can be fetched from a close router. However, a large number of non-cached items (which correspond to the tail part of the Zipf distribution) may have to be downloaded from a distant place, and its delivery cost is not marginal. Moreover, most of the cache hits occur at the network edge (the so-called filter effect) [11], thus the storage modules in upstream routers may not be efficiently utilized. CoRC seeks to make routers cache as diverse content items as possible, so that the network-wide in-network storages are efficiently utilized.

2.3 How to Coordinate between Routing and Caching?

The so-called on-path caching mechanism (which is adopted in most ICN studies) causes inefficient cache utilization because only the cache space in the nodes en route to the publisher of the requested item is checked for cache hits. If a router caches an item and wishes to allow other routers (i.e., off-path routers) to access the cached item, the off-path routers should populate a routing entry for the cached item, which worsens the above FIB scalability issue. What is worse, whenever items are cached and replaced, the change in the cache repository of a router should be adver-

tised to other routers, which may result in substantial signaling traffic, not to mention the update overhead of FIBs. To minimize such overhead to utilize cached items by off-path routers, CoRC makes each and every router know which router is to cache the item of interest.

2.4 How to Reflect the Current Internet Infrastructure and Business?

The current Internet consists of a number of independent networks, which are called Autonomous Systems (ASes). The current routing infrastructure and AS relations should be considered for practical deployment of CoRC. Also, the relations and businesses among Internet stakeholders like ISPs, CPs, and users should be taken into account. We design CoRC based on the following principles. First, we reflect the separation of inter-domain routing and intra-domain routing in the current IP routing. Second, we should design CoRC in such a way that the business models like CDNs can be easily accommodated. Because, ISPs now try to offer CDN-like services. We will discuss how ISPs can leverage the CoRC framework to offer such content business later. Without such economic incentives, the deployment of ICN technologies may not be realized in the foreseeable future. Also, it should be considered to minimize the inter-AS traffic because the transit cost across ASes is typically charged based on the traffic volume [12].

Chapter 3

Related Work

Routing scalability: Various solutions are proposed to reduce the FIB size in the ICN environment. Greedy routing [13] and an ISP-based name aggregation are suggested in the Named Data Networking (NDN) project [14] but they did not detail how actually to be adapted in NDN. A local FIB aggregation technique is proposed to scale IP forwarding tables [15] by aggregating entries with the same next-hop. However, this reduction is not enough for larger-scale ICN FIBs. In [16], the Lookup-and-Cache solution is proposed, where routers cache the fixed number of router records as the FIB. When a router has no FIB record for the given request, it sends a query to a centralized Routing Information Base (RIB), whose response is then cached in its FIB. However, there is no consideration how to disseminate such huge RIBs among ISPs. DHT-based routing [17, 18] constructs a virtual DHT to fully exploit the underlying hierarchical structure of the Internet. However, [17] requires multiple resolution stages to retrieve an item and αRoute [18] did not consider routing scalability within an AS.

Caching efficiency: In-network caching approaches are classified into two categories: explicit and implicit. Generally, explicit solutions advertise or explore cached items over the network within a limited range by exploiting the knowledge of a network topology, a storage capacity, and even a content popularity. They make it easy for requesters retrieve interesting items from nearby nodes. However, they may incur

(i) considerable advertisement traffic to share the information of the cache repositories [19, 20], or (ii) probing delay/traffic whenever a content request arrives [21]. Meanwhile, implicit solutions aim to realize efficient content placement without any additional signaling overhead. However, they typically have difficulties in locating items because each of items is cached in an independent and distributed manner [22, 23]. Similar cooperative caching techniques have been proposed in the web cache area whose characteristics are significantly different because they are based on an overlay environment [24].

There exist two recent proposals to map the dedicated node to a non-overlapping cache space [25, 26] using hash function to maximize the aggregated cache capacity. However, they did not consider routing scalability in both intra- and inter-domain routing, and restrictively investigated the impact of route stretch. CoRC is designed to perform routing and caching in an aligned fashion, which improves routing scalability and cache efficiency at the same time.

Chapter 4

CoRC: Coordinated Routing and Caching

Based on the principles in chapter 2, we detail the CoRC framework.

4.1 Name Resolution

In addition to hierarchical naming which is adapted in NDN, we make the following assumptions: (i) a content name always contains a publisher part, (e.g., the domain name in the URL), (ii) every AS has its own unique name, (iii) each publisher is a customer of a single AS. (Later, we will discuss site multi-homing cases.) Figure 4.1 illustrates the case in which two publishers (*abc.com* and *cnn.com*) are connected to two ASes (sprint and att), respectively. For the sake of simplicity, an ISP is assumed to be an AS throughout this dissertation.

The name of the AS (e.g. *att*) that provides the Internet connectivity to a given publisher (e.g. *cnn.com*) can be retrieved by looking up a name resolution system. The DNS is a good candidate to provide this functionality due to its flexibility, which is adopted in the CoRC framework. To obtain the AS name of a publisher (of a content item to be requested), a host sends an interest packet (i.e., a query) to its local resolution server (i.e., the local DNS server). The local DNS server will obtain the AS name of the publisher iteratively by exchanging interest and data packets with the corresponding mapping servers along the DNS hierarchy. After obtaining the AS

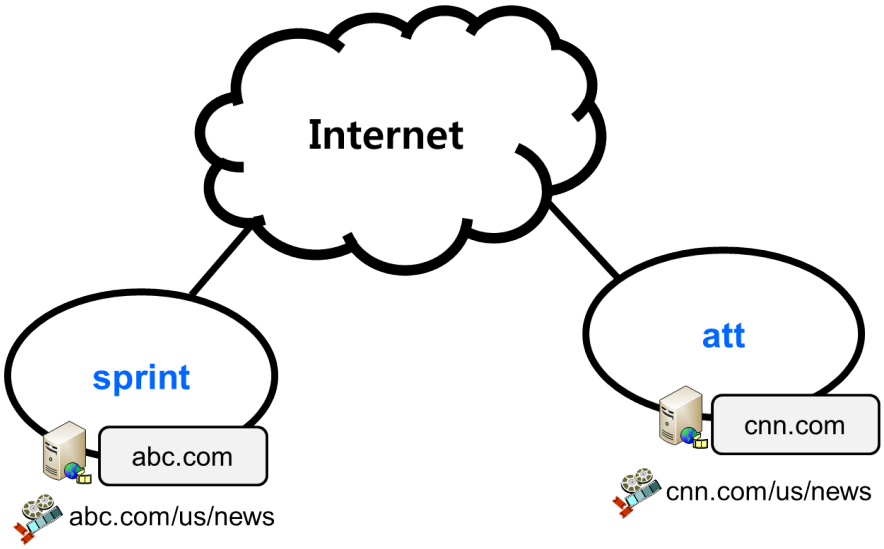

Fig. 4.1. Each publisher is connected to its AS. For a given publisher name, its AS name can be retrieved by extending DNS.

name for the item, the host issues an interest for the item; the AS name is also included in the interest packet for inter-domain routing (to be detailed below).

4.2 Routing

CoRC routing has two parts: intra-domain routing and inter-domain routing. For a given interest, intra-domain routing is performed to find the requested item if the publisher is within the local AS. The item can then be retrieved from the publisher (i.e., its server) located in the AS. Otherwise, the interest will be forwarded to the AS of the publisher by inter-domain routing.

4.2.1 Intra-domain Routing

For (the items of) a given publisher in an AS, all the routers in the AS should know how to forward interests to the publisher for intra-domain routing. One naive option is to populate as many entries in the FIB of every router as the number of local publishers. According to our estimation (to be detailed later), a few hundreds of millions of publishers are expected to be connected to the largest AS in year 2030, which is not scalable. To make FIBs scalable, we adopt the ViAggre approach [27], where the entire namespace is split into partitions, and each router contains only its partition in its FIB.

A publisher name in an interest is hashed to a fixed-length value (say, 128 bits), which is called a publisher identifier (PID). Then the PID space becomes the namespace to be covered by FIBs of routers. Let a partition of the PID space be represented by a PID prefix (like an IP prefix in BGP routing). A set of routers in an AS is selected as responsible routers (RRs), each of which is to maintain its own PID prefix. Each RR advertises its PID prefix throughout its AS, so that other routers populate the corresponding FIB entry (in PAR-FIB) for intra-domain routing. A PAR-FIB entry contains <PID prefix, next hop interface>. Each publisher calculates its PID and registers the PID prefix of its access router with the RR whose partition includes the PID. Each RR also maintains a Publisher Information Base (PIB), which is a mapping table whose entry contains <PID of a publisher, PID prefix of its access router>.

In Figure 4.2, let us illustrate how CoRC intra-domain routing operates. Suppose that four routers are RRs whose PID prefixes are 0b00, 0b01, 0b10, and 0b11, respectively. R01 is responsible for all the PIDs starting with prefix 0b01; that is, R01 should know the locations of all the publishers whose PIDs start with 0b01. Here, the

location of a publisher means the PID prefix of the access router of the publisher. Likewise, R00 knows the locations of publishers whose PIDs start with 0b00, and so on.

The interest to retrieve *cnn.com/us/news* will be first forwarded to R01 because the PID of cnn.com starts with 0b01 in this example. On the path from access router to R01, there can be multiple non-RRs which use PAR-FIB to forward the interest to R01.[1] By looking up the PIB, R01 tunnels the interest to the access router (R11) of the publisher *cnn.com*. In Figure 4.2, the PID prefix of the access router of *cnn.com* is 0b11. It is not straightforward to perform tunneling as there is no locator in ICN. We slightly abuse the PID prefix of a router as the router's identifier for this purpose. That is, the RR will tunnel an interest toward the access router of the corresponding publisher by adding the PID prefix of the access router into the interest. The routers forward the interest by looking up their PAR-FIBs with the PID prefix (added for tunneling).

4.2.2 Inter-domain Routing

When the interest packet arrives at the RR, the router knows whether the publisher (of the requested item) belongs to its local AS or not. If the router finds that the item is not in its cache and the publisher belongs to another AS, it will forward the interest to the neighbor AS toward the AS of the publisher. Every router also maintains an AS-FIB whose entry contains <AS name, next-hop interface> for inter-domain routing. Each entry in the AS-FIB is assumed to be populated by receiving an advertisement message from each AS by an inter-domain routing protocol such as BGP.

[1] Non-RRs are not visible in Figure 4.2 for the sake of simplicity.

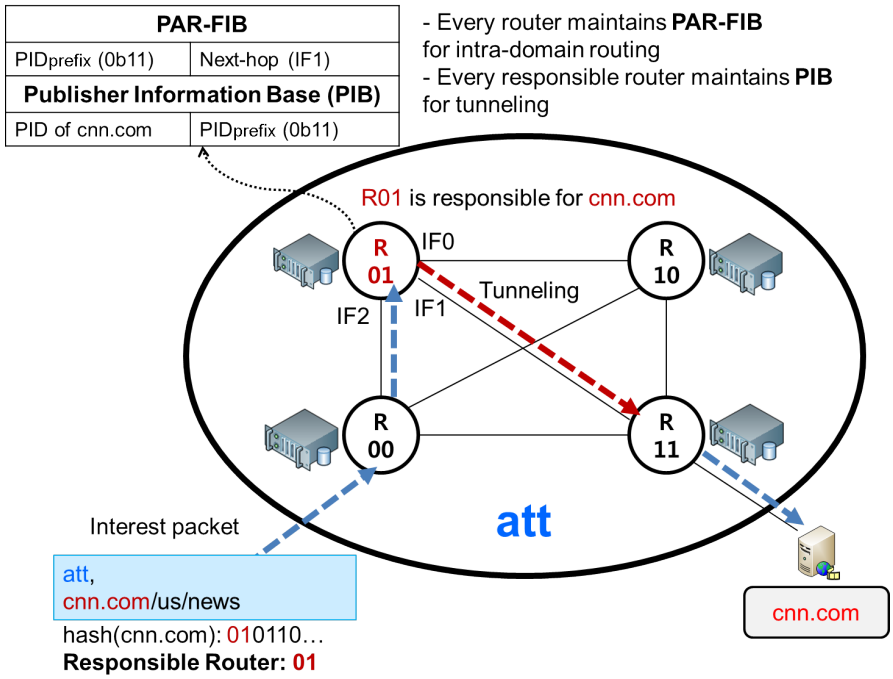

Fig. 4.2. Intra-domain routing of CoRC is illustrated.

Thus, the number of AS-FIB entries is the same as the number of ASes in the Internet. Note that the number of ASes currently is around a few tens of thousands and increases relatively slowly. Thus, the scalability of the AS-FIB is sustainable in the foreseeable future. In chapter 6, we will estimate the number of AS-FIB entries until the year 2030.

4.3 Caching

In the process of routing, an interest packet will visit the corresponding RR first. That is, requests with different PID prefixes are forwarded to the corresponding (and

hence different) RRs whose PID partitions are non-overlapping. In CoRC, each RR caches only items whose PID belongs to its PID partition. We align the caching and routing functionalities into the same router, so that interests for the items and their corresponding data packets (which may be cached) are all handled by the same router. As shown in Figure 4.2, an item whose content name starts with *cnn.com* is cached only at router R01 as R01 is in charge of routing all the interests whose PIDs start with 0b01.

Chapter 5

Optimization

CoRC coordinates each RR to perform routing and caching together for the same partition of the PID namespace. The advantages of this alignment are two-fold: (i) there is no advertisement signaling overhead to announce cached items in in-network storages, and (ii) there is no duplicate copy of the same item among RRs from a network-wide perspective. However, this kind of indirect routing (i.e., stopping by RRs) causes a longer delivery path. Hence, we propose two optimization methods which mitigate this stretch issue.

5.1 Assigning PID prefix to RR

The route stretch strongly depends on the location of the RR in charge of each request. Therefore, we need to carefully assign PID prefixes to RRs. We propose a simple but powerful prefix assignment algorithm with the following assumptions. First, the popularity distribution of prefixes is given. Second, requests are uniformly distributed among all edge routers. Last, RRs cache popular items depending on the content popularity. Unpopular items not cached in RRs represent the long-tailed part of the Zipf distribution, which can be assumed to be uniformly distributed to all edge routers of publishers.

Let $R = \{r_1, r_2, ..., r_k\}$ and $P = \{p_1, p_2, ..., p_k\}$[1] be the sets of RRs and PID pre-

[1] P is sorted in a popularity order of PID prefixes.

fixes, respectively. The function $f(k)$ represents the request frequency of each PID prefix, which satisfies $f(p_1) \geq f(p_2) \geq ... \geq f(p_k)$. In addition, if hop_n is defined as the number of hops traversed by the nth request, the optimal prefix assignment problem is to map P to R such that $\sum_{x=1}^{n} hop_x$ is minimized for n requests. Algorithm 5.1 first sorts R by the sum of distances to all edge routers in non-decreasing order and then iteratively assigns the most to least popular PID prefixes to the sorted R.

Algorithm 5.1 Prefix assignment algorithm

Input : Set of edge routers $E = \{e_1, e_2, ..., e_m\}$; Set of responsible routers $R = \{r_1, r_2, ...r_k\}$;

1: **for** all $r_k \in R$ **do**
2: $sum = \sum_{i=1}^{m} distance(e_i, r_k)$
3: $Array[k].r \leftarrow r_k$
4: $Array[k].s \leftarrow sum$
5: **end for**
6: Sort $Array[k]$ by s in a non-decreasing order
7: **for** all $p_k \in P$ **do**
8: Assign p_k to $Array[k].r$
9: **end for**

Proof of optimality: each of n requests is hashed into each of k PID prefixes. By the assumption, $n = n_{p_1} + n_{p_2} + ... + n_{p_k}$ where $n_{p_1} \geq n_{p_2} \geq ... \geq n_{p_k}$. If c_k items are cached at r_k and n_{c_k} is the number of requests to c_k among n_{p_k} requests, n_{c_k} requests are satisfied by r_k while $n_{p_k} - n_{c_k}$ requests are forwarded to the original publisher. Therefore, the expected number of hops for a request whose PID prefix is p_k is defined as follows.

$$AvgHop_{p_k} = \frac{\sum_{i=1}^{n_{c_k}} d(e_r(i), r_k) + \sum_{j=1}^{n_{p_k}-n_{c_k}} d(e_r(j), r_k, e_o(j))}{n_{p_k}} \quad (5.1)$$

$$(d(u,v,w) = d(u,v) + d(v,w))$$

Here, $e_r(i)$ and $e_o(i)$ represent edge routers attached to the requester and the original publisher for the ith request, respectively. The function $d(u,v)$ is defined as the shortest distance between u and v. Accordingly, the total number of hops for n requests is defined as follows.

$$\sum_{x=1}^{n} hop_x = \sum_{y=1}^{k} n_{p_y} \cdot AvgHop_{p_y} \quad (5.2)$$

By the inequality of rearrangement, $\sum_{x=1}^{n} hop_x$ is minimal. ($\because n_{p_1} \geq ... \geq n_{p_k}$ and $AvgHop_{p_1} \leq ... \leq AvgHop_{p_k}$.)

5.2 Hybrid Approach

Although CoRC improves the total cache hit ratio due to the network-wide aggregated cache utilization, independent caching may be more beneficial for popular items. In practice, it is reported that the popularity of items follows the Zipf distribution [28]. With independent caching, the popular items can be retrieved from any intermediate routers that their interests encounter. To leverage both independent and coordinated caching, we propose a hybrid approach, dubbed CoRC-HBD.

In CoRC-HBD, edge routers (access routers attached to requesters) perform in-

dividual and independent caching and only non-edge routers collaborate for coordinated caching. Edge routers cache every item being forwarded based on their own replace strategies, e.g. Least Recently Used (LRU). As most of the popular items accouting for the majority of traffic are likely to be retrieved from edge routers, the route stretch problem can be alleviated. Non-edge routers cache only items whose PIDs belong to individual partitions, and thus the network-wide cache diversity is somewhat retained. Note that a unique tag like the RR's PID prefix should also be assigned to each edge router and populated in a PAR-FIB (of every RR) in order to tunnel interests to non-RRs (i.e., edge routers).

Chapter 6

Routing Scalability

A router in the CoRC framework has three tables for request routing: (i) AS-FIB, (ii) PAR-FIB, and (iii) PIB. In this chapter, we estimate the number of ASes and domain names expected in year 2030 and then analyze how large each table will be in year 2030 to investigate a routing scalability.

6.1 AS-FIB

A router maintains AS-FIB to forward interests toward any ASes, so its size depends on the number of ASes. According to [29], there are currently more than 45,000 active ASes advertised in the Internet as of July 2013. Figure 6.1 shows that the number of ASes will be expected to reach around 120,000 by the year 2030, which is projected based on the recent history of AS numbers. Accordingly, we claim that AS-FIB will contain around 120K routing entries by the year 2030.

6.2 PAR-FIB and PIB

To estimate the size of a PAR-FIB, the total number of domain names should be analyzed at first. As of July 2012, around 0.9 billion domain names are registered according to Internet systems consortium [9]. Figure 6.2 shows that the total number of domain names is expected to reach almost 4B (or 2^{32}) by the year 2030. Note that

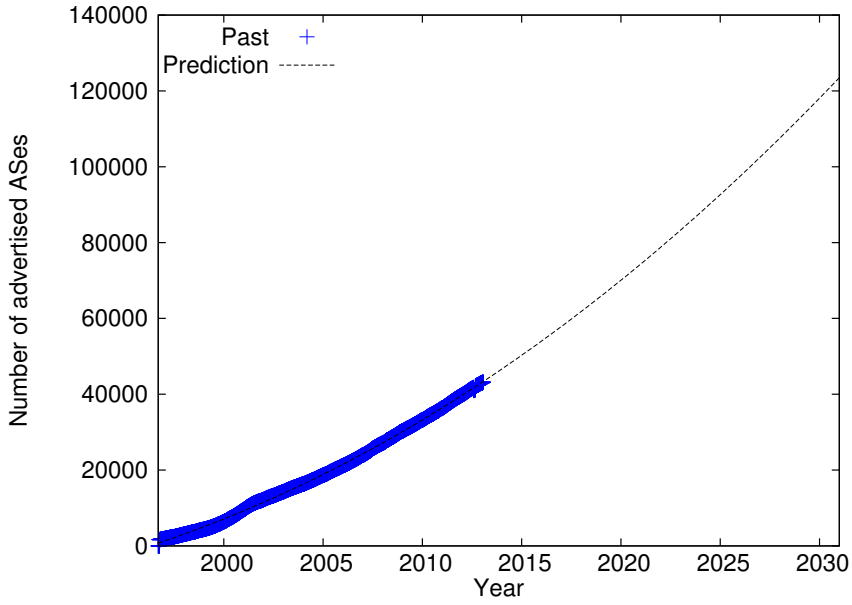

Fig. 6.1. The number of ISPs is predicted to reach around 120,000 by the year 2030.

we have to consider the total numbers of both domain names and ASes to calculate the number of domain names per AS. Definitely, larger ASes would have more domain names than smaller ASes.

It is reported that the number of connections among ASes (or AS degree) follows a Zipf distribution [28, 30]. In addition, according to [31], the number of routers per AS is positively correlated to the AS degree. If we make a similar conjecture with respect to the distribution of the numbers of domain names among ASes, we can assume that the number of domain names per AS also follows a Zipf distribution. We apply a Zipf distribution (its exponent is 1.0) to 4 billion domain names among 120K ASes. The maximum number of domain names per AS is estimated to be around 330M domain names in year 2030. Thus, we expect that the maximum number of

domain names per AS will be less than 2^{29} until 2030.

The sizes of a PAR-FIB and a PIB also depend on the number of routers (of an AS) that can serve as RRs. According to [32], some large ASes currently contain around 2^{20} routers. Let us make the following assumptions to estimate the sizes of a PAR-FIB and a PIB in year 2030: (i) the largest AS (in terms of number of domain names) has up to 2^{20} routers, (ii) a quarter of all routers in the AS serve as RRs in CoRC, and (iii) one eighth of all routers serve as RRs and another eighth as edge routers in CoRC-HBD. Also, 2^{29} domain names (actually PIDs) are estimated to be evenly distributed among RRs, which determines the size of a PIB.

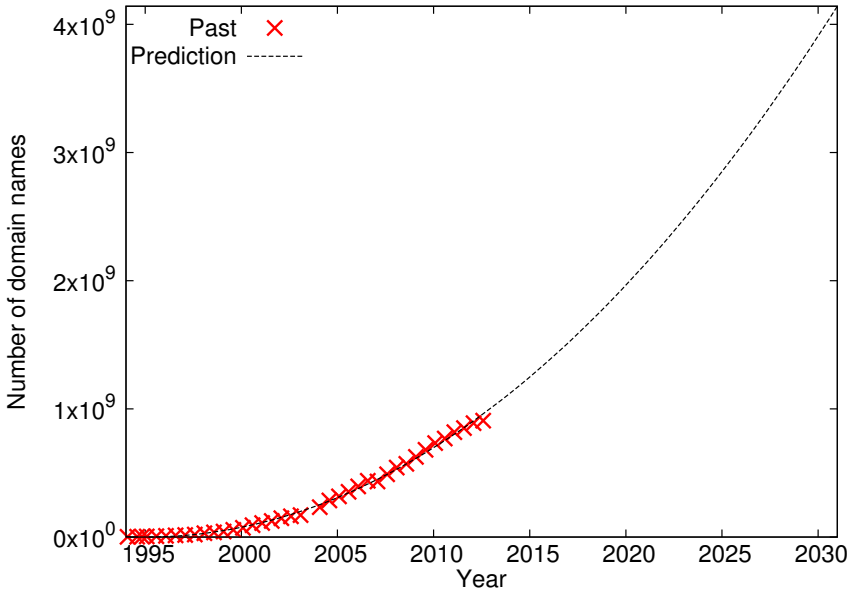

Fig. 6.2. The number of domain names is predicted to reach around 4 billion by the year 2030.

6.3 Numbers of Entries of Three Tables

Figure 6.3 shows the total number of entries of AS-FIB, PAR-FIB, and PIB depending on n routers in a extremely large AS with 2^{29} domain names, varying n from 2^{13} to 2^{20}. The total number of routing table entries of a CoRC router in year 2030 is bounded to the same level of the current number of IP prefixes (i.e., the FIB size of a DFZ router). However, a single PID requires at least a 128-bit hash value to guarantee the probability of hash collision negligible. Thus, the overall memory space for a CoRC router may be greater than that of the current IP router. There should be a trade-off between the size of PAR-FIB and that of PIB. If more routers participate in CoRC as RRs, each router has the larger PAR-FIB and the smaller PIB. Each AS can adjust the number of RRs in accordance with the network size.

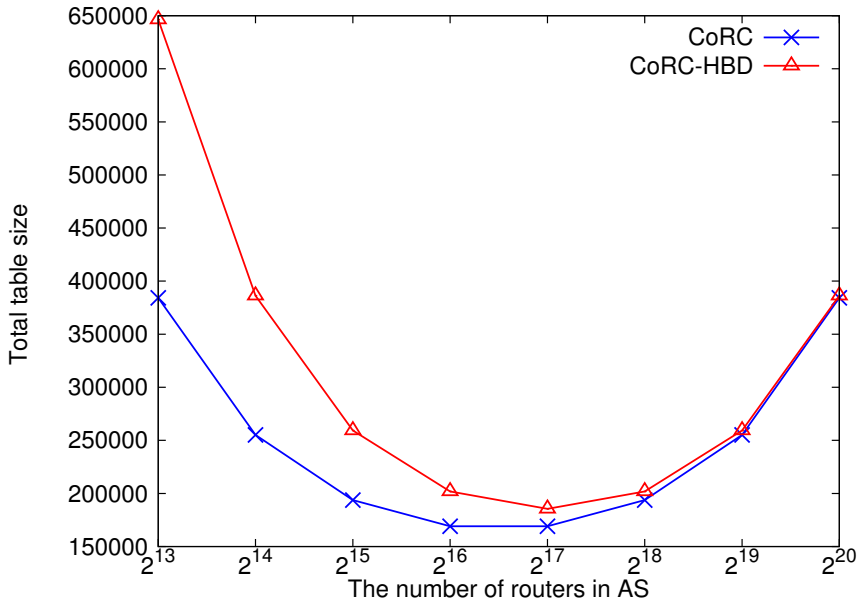

Fig. 6.3. The total number of entries in each of CoRC and CoRC-HBD routers is estimated.

Chapter 7

Network Performance

Splitting the whole namespace into the FIBs (of RRs) achieves the scalable FIB size at the cost of route stretch as explained earlier. In addition, it can efficiently exploit in-network storages across the given network by dividing the PID namespace among caches. To fully understand CoRC, we first conduct comprehensive emulations with 4 stub topologies. Then, we investigate the route stretch within a single AS.

7.1 Performance Metrics

- **Cache hit ratio:** Assigning partitions of the whole caching space to individual RRs enables more items to be cached at in-network storages (in RRs).
- **Content delivery latency:** Two specific metrics are considered: (i) the average delivery hop and (ii) the average content retrieval time, which are directly impacted by route stretch.
- **Traffic load:** We focus on inter-AS traffic because stub ASes should generally pay transit fees which is normally proportional to the cross-AS traffic volume. We also investigate total traffic to study how much traffic is stretched.

7.2 Compared Schemes

We consider five different schemes based on routing and caching strategies, which are to be compared in terms of above metrics.

- **Vanilla ICN:** In Vanilla ICN, routers forward interest packets to the publisher of the content along the shortest path by looking up flat FIB entries. We assume that each router makes independent caching decisions with LRU replace strategy. Vanilla ICN has the routing scalability problem as every router maintains the entire FIB, and is deemed as a baseline.

- **Coordinated Routing/Individual Caching (CRIC):** CRIC is a modified Vanilla ICN scheme where splitting FIB is used to address the routing scalability issue. In CRIC, interest packets are first sent to RRs, and then delivered to publishers via tunneling. When data packets are relayed back to the requesters, the routers on the path cache content individually. CRIC is meaningful in examining the impact of the lengthened path and the limited cache diversity.

- **CoRC and CoRC-HBD:** CoRC achieves the cache diversity by performing coordinated caching at the cost of longer route stretch. CoRC-HBD mitigates the route stretch issue while the network-wide cache diversity is somewhat degraded by combining both individual and coordinated caching strategies.

- **Oracle:** As another reference, the oracle version of caching is also evaluated. A border router in each AS has its own storage whose size is the sum of all the cache sizes within the AS in other schemes. The gateway router's storage is filled with the most popular items based on the knowledge of the popularity

distribution. Oracle is deemed to achieve the best performance since flat routing (i.e., the shortest path routing) is used, and the cache utilization is the best.

Note that we do not directly compare our CoRC with previous proposals. We highlight that CoRC outperforms the reference schemes in terms of above metrics, even though CoRC uses indirection to address the routing scalability.

7.3 Experimental Setting

We implemented software routers running on Amazon Elastic Compute Cloud (EC2) [8] to emulate CoRC and the other schemes. In our experiments, 60 routers are installed as 60 virtual machines in Amazon data centers in five different regions: Oregon, North California, North Virginia, Sao Paulo, and Ireland. We assume that each of four regions represents a stub AS and the other region (North Virginia) represents a transit AS that interconnects the four stub ASes. We constructed a tree topology whose mean degree is 2.42 at the router level in each stub AS by referring to [33]. Every stub AS consists of 14 routers: 8 edge, 4 backbone, and 2 border routers. The transit AS has 4 routers.

Content items are uniformly distributed over four ASes, each of which is configured with a total cache space equal to 0.5%, 1.0% and 1.5% of the total content volume[1]. In Oracle, the whole cache space is allocated to a border router. In the other schemes, by taking into account the difference of router capability, each of backbone, border, and edge routers has one eighth, one eighth, and one thirty-second of a total cache space, respectively. Note that the FIB is also split with the same ratio. PID

[1]We omit the cases of 0.5% and 1.5% due to the marginal difference.

prefixes are generated by modular hashing and assigned to the corresponding RRs by the Algorithm 5.1. A request for 10 MB data is generated at each edge router with a poisson arrival rate 0.1 (per second), which means 32 data items are requested per every ten seconds on average. Users (generating interests) belonging to an edge router are running in the same virtual machine as their edge router. The skewness of the popularity distribution is determined by the Zipf exponent α: 0.6, 0.8, 1.0, 1.2, and 1.4.

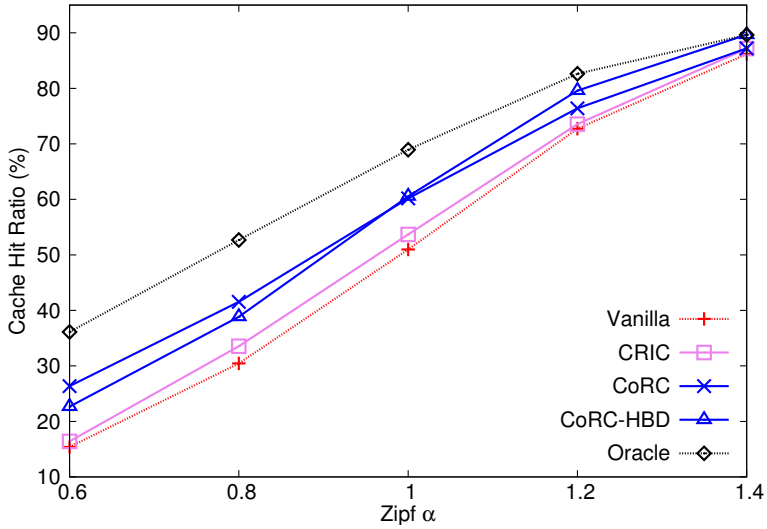

Fig. 7.1. The comparison of cache hit ratio of CoRC and other schemes is shown.

7.4 Average Cache Hit Ratio

Figure 7.1 shows the average cache hit ratio with a cache space ratio of 1.0%, varying α. Obviously, splitting the whole cache space into non-overlapping partitions in CoRC allows more items to be stored in the caches, resulting in a higher cache

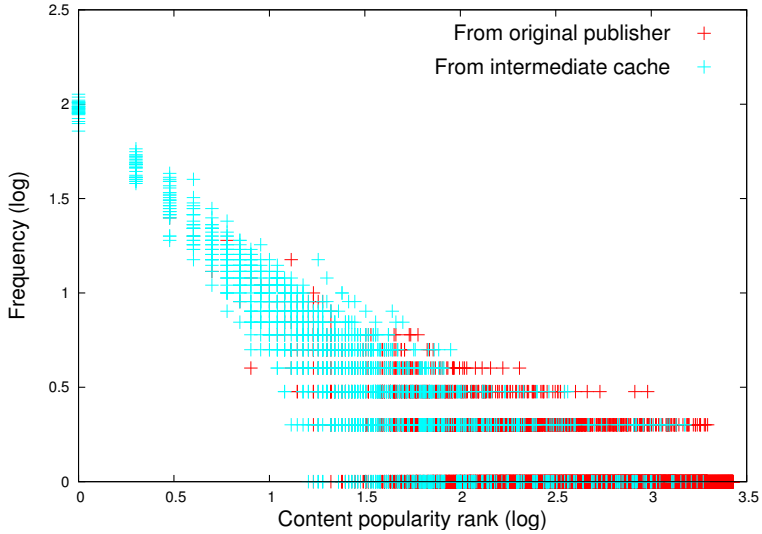

Fig. 7.2. The cache utilization of Vanilla is illustrated.

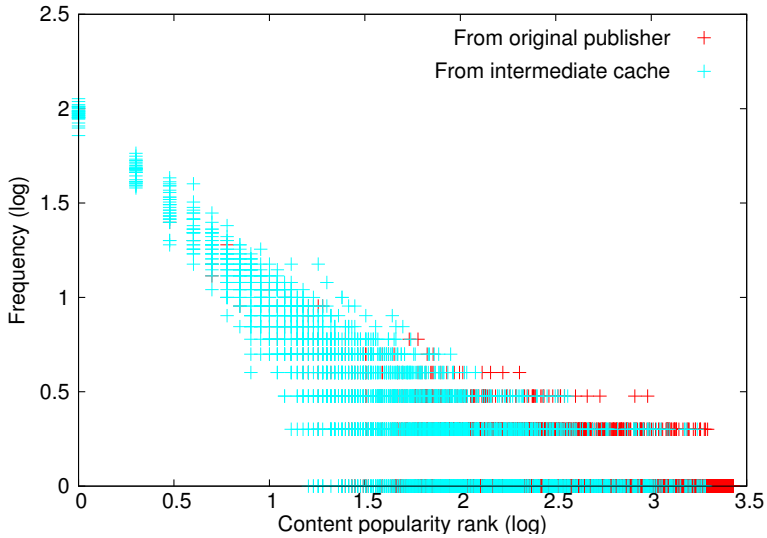

Fig. 7.3. The cache utilization of CoRC is illustrated. CoRC can serve more diverse items than Vanilla, which results in higher cache hit ratio.

hit ratio. There are two interesting observations. First, CoRC-HBD achieves almost similar performance to Oracle as α increases from 1.2 to 1.4. Note that the cache hit ratio of CoRC is higher than that of CoRC-HBD when α is 0.8 but the performance gain is reversed as α increases to 1.2. It is because with highly skewed requests (e.g., α is 1.2 or 1.4), popular items have more requests while non-popular items have less requests. Accordingly, the effect of partitioning a cache space is mitigated. Second, CRIC exhibits the higher hit ratio than Vanilla because CRIC routes interests via RRs, which gives a higher cache hit chances at RRs.

To better understand the splitting effect, we investigated the frequency of content requests which arrive at each router, which is plotted according to their popularity in a log-log scale. We distinguished items which are retrieved from intermediate caches and original publishers by color. Figure 7.2 and 7.3 reveal that only popular items are mostly cached in Vanilla while CoRC can cache much more items than Vanilla.

7.5 Content Delivery Latency

Figure 7.4 shows the ratio of the average hop count of each scheme to that of Vanilla, and Figure 7.5 is the average route stretch which is the (hop count) ratio of a path taken by each scheme to retrieve an item to the shortest path to its publisher, as α varies. Vanilla has a fairly low hop count throughout α since it takes the shortest path to the original servers (in cases of cache misses). With not-so-skewed popularity distributions (e.g., α is 0.6 or 0.8), Oracle achieves the lowest hop count since it caches most popular items while the other schemes may suffer from cache misses due to relatively small popularity differences among items. CoRC and CoRC-HBD show

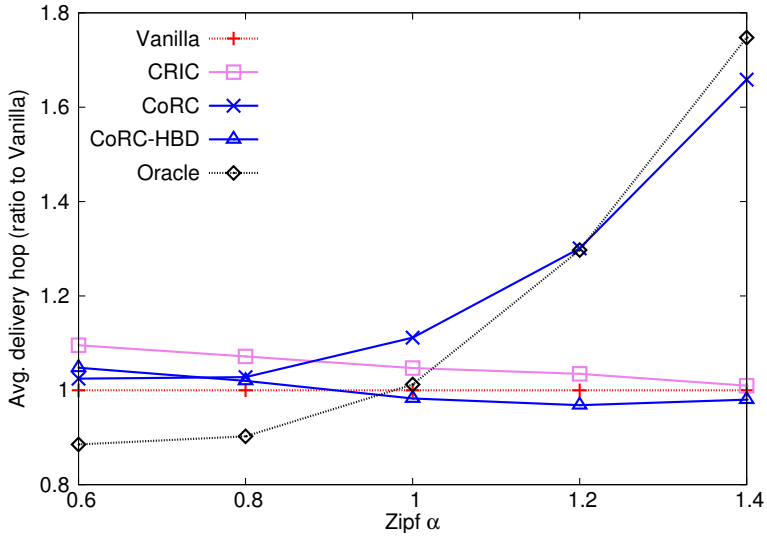

Fig. 7.4. The average hop count of each scheme is plotted. CoRC-HBD achieves the near-optimal performance.

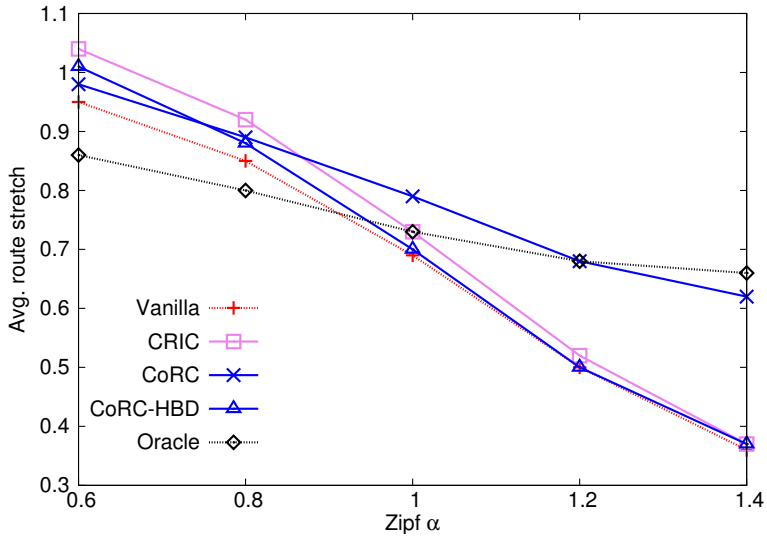

Fig. 7.5. The relative path length of each scheme is plotted. CoRC-HBD achieves the near-optimal performance.

a slightly longer hop count than Vanilla due to route stretch. As α increases, however, CoRC-HBD outperforms the other schemes since (i) popular items get more requests and are cached at edge routers, and (ii) cache diversity is somewhat retained. Note that Oracle performs poorly with the highly skewed popularity distribution (e.g., α is 1.2 and 1.4), since a single cache is located at the border router; popular items in the other schemes are likely to be cached at edge and backbone routers.

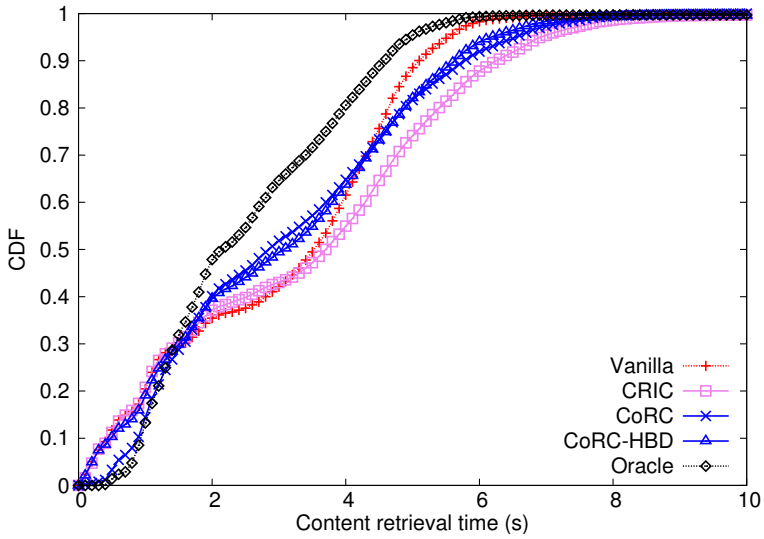

Fig. 7.6. The content retrieval time (when α is 0.8) is illustrated.

Figure 7.6, 7.7, and 7.8 show the CDF of the content retrieval times of each scheme when α is 0.8, 1.0, and 1.2, respectively.

Regardless of the skewness of the popularity distribution, CoRC-HBD achieves mostly shorter retrieval time by taking advantage of both independent and coordinated caching. When α is 1.0, for example, about half of all requested items (that are more popular) are retrieved from (closer) edge routers in Vanilla, CRIC, and CoRC-

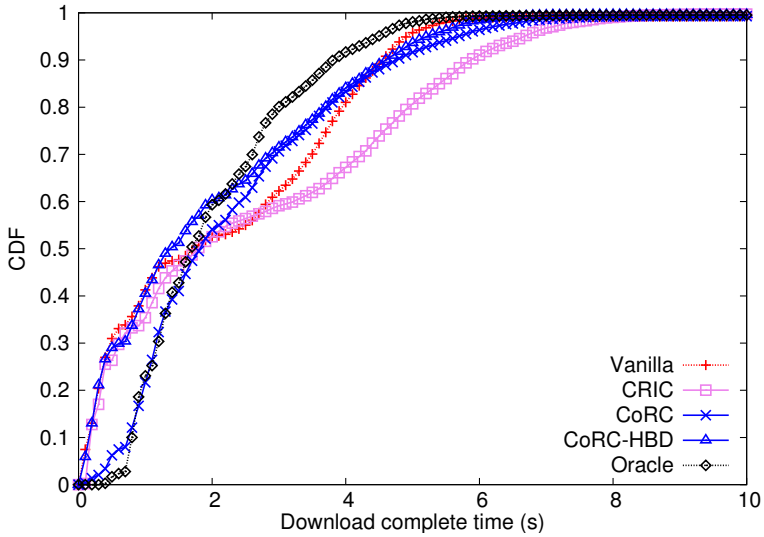

Fig. 7.7. The content retrieval time (when α is 1.0) is illustrated.

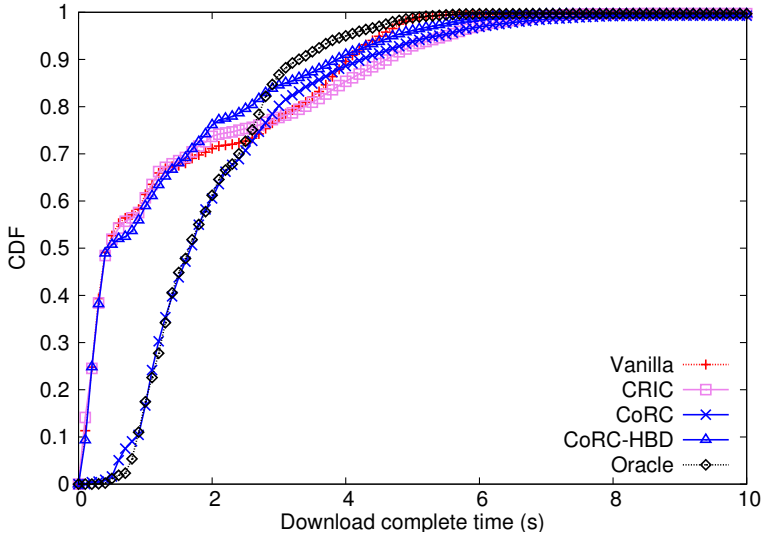

Fig. 7.8. The content retrieval time (when α is 1.2) is illustrated.

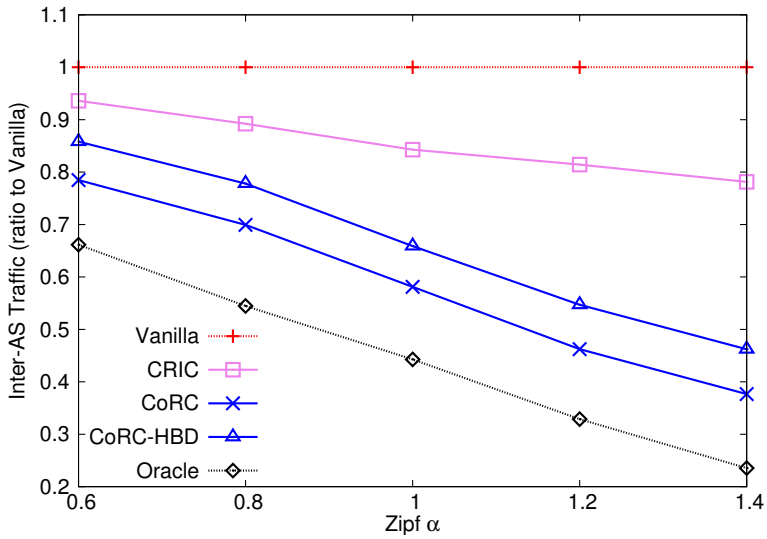

Fig. 7.9. Inter-AS traffic is plotted for each scheme. Traffic reduction comes from caching diversity.

HBD due to the independent caching, which shows shorter retrieval time. For the other half (i.e., items less popular), they are retrieved from caches in RRs in CoRC and CoRC-HBD due to the cooperative caching, compared to Vanilla and CRIC. While Oracle shows the shortest retrieval time for the less popular items (but located in the cache storage), it performs poor for the popular items. To summarize, CoRC-HBD achieves the low route stretch, and hence the low delivery latency while mitigating the routing scalability issue.

7.6 Traffic Load

Figures 7.9 shows the traffic load over inter-AS links, which is normalized to the Vanilla case. As expected, Oracle achieves the smallest inter-AS traffic due to the

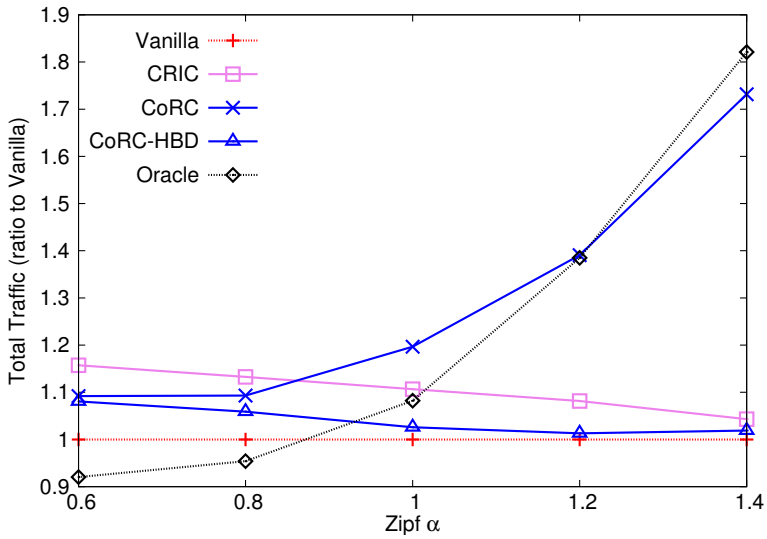

Fig. 7.10. Total traffic is plotted for each scheme. CoRC-HBD achieves the near-optimal performance.

omniscient knowledge of content popularity, while Vanilla shows the highest inter-AS traffic due to poor cache diversity and cache hit ratio. We have two observations: (i) CoRC performs better than CoRC-HBD, which is in turn better than CRIC, and (ii) as α increases, the reduction of inter-AS traffic also increases. It can be explained by the degree of cache diversity in these schemes.

Figure 7.10 shows the total amount of traffic, which is normalized to the Vanilla case. The total traffic is the sum of transferred traffic for all the links. Notice that Oracle and Vanilla ICN reveal interesting patterns. With not-so-skewed popularity distributions (e.g., α is 0.6 or 0.8), Oracle is better than Vanilla since the cache utilization is more important when content items exhibit relatively small popularity differences. However, with the highly skewed popularity distribution (e.g., α is 1.2 or 1.4), Vanilla ICN is better than Oracle since popular items have more requests, and

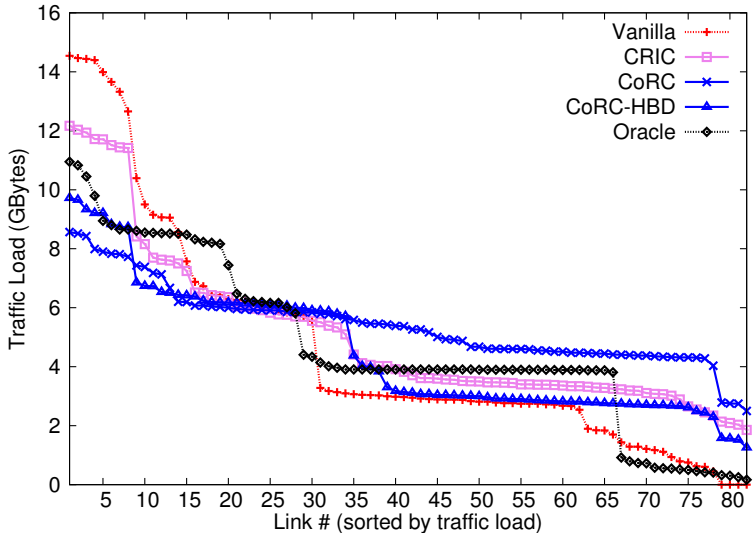

Fig. 7.11. Traffic load of each link is plotted for each scheme. Traffic load is balanced in CoRC and CoRC-HBD.

hence interests for popular items are more likely to be hit on cache. The similar pattern is observed between CoRC and CRIC. When α is 0.8, CoRC is better than CRIC since the cache diversity is more effective to reduce traffic. Meanwhile, when α is larger than 1.2, CoRC is worse than CRIC since every interest should visit its RR first. Overall, CoRC-HBD reduces the effect of route stretch caused by RRs.

Figure 7.11 shows traffic load of each link when α is 1.0. Splitting the whole cache space to routers helps spread traffic over all links. It is another advantage for network management.

7.7 Route Stretch vs. Topology

According to the CAIDA's analysis [34], the number of routers and their degree distribution in each AS are different. To study how route stretch is affected by a topology in a single AS, we evaluate the above schemes with six different topologies (see Table 7.1). Four router-level topologies are generated by IGen [35] with 64 nodes. We vary the node degree distribution of each topology by changing the number of links between access and backbone routers and between clusters. In addition, two real topologies are considered: GEANT (pan-European research and education network) and GARR-X (Italian research and education network).

Table. 7.1. Topology properties

Name	Top_A	Top_B	Top_C	Top_D	GEANT	GARR-X
Nodes	64	64	64	64	40	34
Mean Degree	2.84	4.34	5.87	7.41	2.98	2.70
Std.dev	3.57	4.51	5.45	6.58	2.01	1.53
Min	1	2	3	4	1	1
Max	12	18	22	27	10	6

In this experiment, an interest packet is generated by a randomly selected node with a poisson arrival rate 0.1 (per second). In CoRC, all nodes in each topology are in charge of non-overlapping parts of the whole cache space while a half of them are selected as RRs in CoRC-HBD. Figure 7.12 shows the relative path length of CORC and CoRC-HBD with the cache space ratio of 1.0%. We made two observations. First, regardless of a topology, route stretch does not exceed a certain level, which gives an incentive to the AS that adopts CoRC. Second, route stretch affected by the skewness of content popularity can be alleviated by CoRC-HBD.

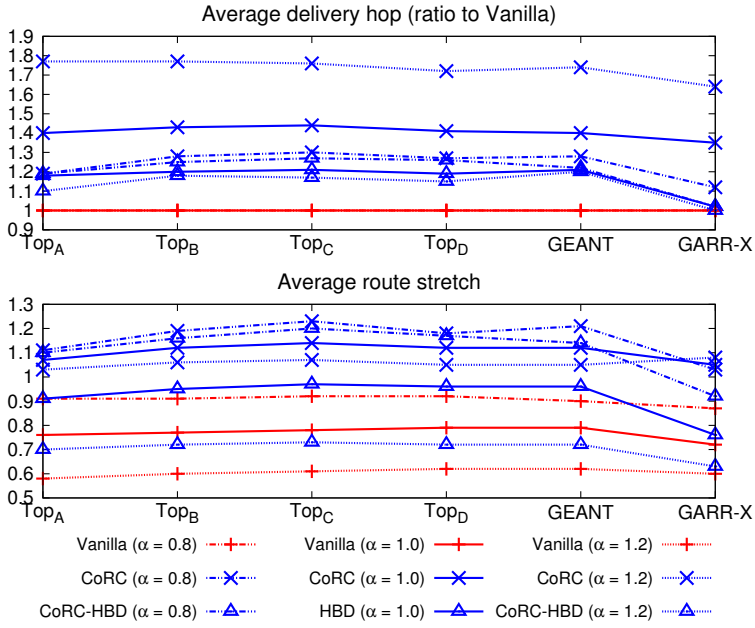

Fig. 7.12. Route stretch is almost irrespective of a topology.

Chapter 8

Packet Processing Time in a Router

In Vanilla ICN routing, an incoming interest packet can be forwarded by looking up the FIB once. Forwarding an interest packet in the CoRC framework may take longer time since there are more actions. In this chapter, we evaluate the interest packet processing time in a router with the CoRC framework.

8.1 Methodology

We implement Vanilla routing and CoRC in two different settings: (i) the CCNx framework over UDP/IP/Ethernet [36], and (ii) the Linux kernel over the Ethernet link layer. In both settings, we increase the number of incoming interest packets per time and measure the packet drop rate, which is the percentage of dropped interest packets among all the incoming interest packets.

Also, we use another PC to generate interest packets. Thus, depending on the settings, the interest packet generator is implemented in user space (i.e., in the CCNx framework), or in kernel space. The number of generated interest packets per time is varied to see the relation between the traffic rate (incoming interest packets per second) and the drop rate. As a data packet is forwarded by the same mechanism in CoRC and Vanilla ICN routing, the forwarding mechanism of data packets is not implemented, and data packets are not generated. Each PC has an Intel Core i3-2100

CPU running at 3.10 GHz with 3 MB L2 cache and 3 GB memory.

8.2 Drop Rate vs. Interest Packet Rate

The drop rate of interest packets in the CCNx framework (in user space) is shown in Figure 8.1. Interest packets start being dropped at 6000 packets per sec (pps) with Vanilla routing, while CoRC shows packet drops at 4000 pps. This result reveals that there is additional forwarding delay in CoRC with the first setting due to two actions: (i) checking the AS name, and (ii) looking up the PIB and performing tunneling.

In the second setting, both CoRC and Vanilla ICN routing schemes are implemented in Linux kernel space. The drop rates of both schemes start at significantly higher interest packet rate (40K pps), and there are marginal differences as shown in Figure 8.2. This result indicates two observations. First, by implementing routing schemes in kernel space, the processing overhead (e.g., context switching) for the additional actions in forwarding interest packets becomes marginal. While CoRC requires more actions in forwarding interests, the additional processing delay is not significant even with software implementation (in kernel). We believe that forwarding speed of interest packets on the hardware implementation of CoRC would be comparable to that of Vanilla routing on the hardware implementation.

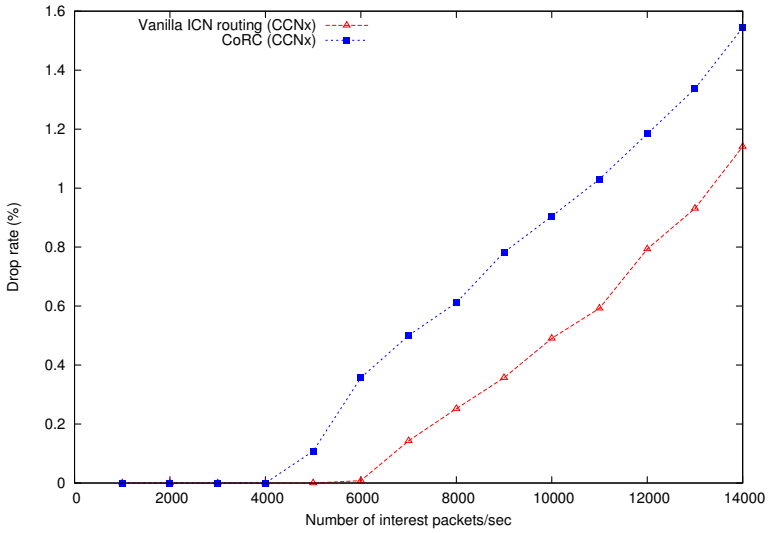

Fig. 8.1. Drop rate versus interest packet rate in CCNx is plotted.

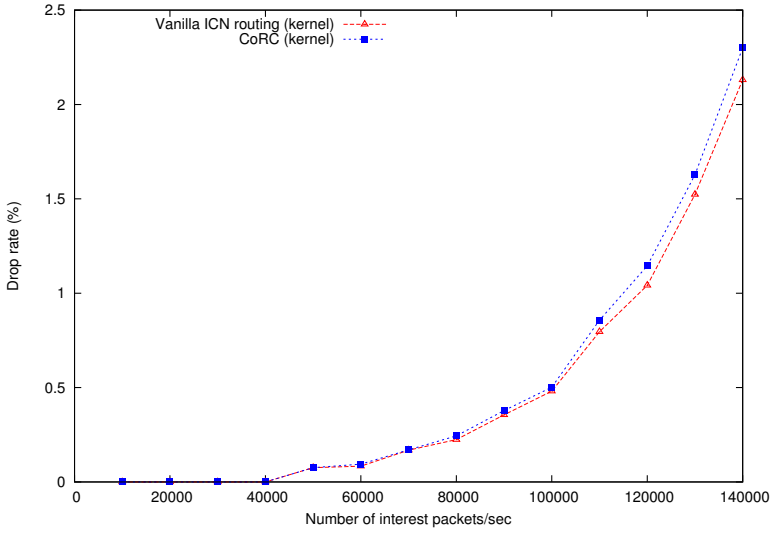

Fig. 8.2. Drop rate versus interest packet rate in kernel is plotted.

Chapter 9

Discussions and Future Work

9.1 Hashing by Publisher Name

CoRC generates a PID by hashing a publisher name, which has the following two advantages. First, routing scalability is enhanced by aggregation. Second, a particular node is in charge of all contents with the same publisher, which is advantageous in terms of network management and business with content providers. However, hashing can cause collisions, which may have a marginal effect on the routing and caching performance in CoRC. First of all, the PID collision probability is extremely low with long hash length (say, 128 bits). Suppose two different publisher names are hashed into the same PID. An interest packet for either of the two publishers is delivered to the same RR by intra-domain routing in CoRC. The RR will look up its PIB to find out the access router of the correct publisher. When the corresponding PIB entry were populated, the RR already knows there is a hash collision for the entry. To differentiate the two publisher names mapped to the same PID, we need another field (say, the string of the publisher name) in the PIB only for the collided entries.

9.2 Dealing with Router Failure

Like ViAggre [27], multiple replicas with the same PID prefix can be allocated for availability. When any RR is down, its backup can come to the front by advertising the PID prefix over the network. In this case, the caching procedure should be restarted. Note that caching is less crucial than routing in case of node failures, and eventually most of popular items will be cached in a relatively short time. There are two issues to investigate as a future work; (i) How to deploy the backup routers considering the AS topology? and (ii) How to share the cached item among backup routers?

9.3 Resolution System and Multihoming

Most of organizations are connected to multiple ASes for availability and traffic engineering purposes. Suppose *cnn.com* is connected to AS1 and AS2. The publisher may want to receive interests for *cnn.com/video* via AS1, and other interests for *cnn.com* via AS2. To support, the DNS should be able to map AS names not just at the level of publisher names but also at a finer granularity, e.g., directory under the same publisher. Hence, we can keep maintaining two mapping entries for *cnn.com* and *cnn.com/video* in the DNS. Moreover, multiple AS names may have to be dynamically selected, depending on the purposes such as load balancing and failover. For this purpose, a possible option for reliable communications is to send multiple AS names to a requester at once, and then the requester can control how to use multiple ASes to access the particular publisher, which requires a further study.

Chapter 10

Summary

This dissertation proposed the CoRC framework in Information-Centric Networking (ICN) for: (i) routing scalability, (ii) caching efficiency, and (iii) coordination between routing and caching. To shrink the FIB size, CoRC uses two kinds of routing (i.e., intra-domain and inter-domain routing), with two assumptions: (i) every content publisher is connected to an AS (Autonomous System) and (ii) every AS has its name, which is advertised across the Internet. With the AS name-based inter-domain routing, a router in an AS needs to handle routing packets only for the publishers belonging to the local AS. For the purpose of scalable intra-domain routing, CoRC partitions a forward information base (FIB) among routers so that every router maintains only a partition of the entire FIB. To maximize cache diversity, CoRC makes each router cache content items whose names belong to its own partition of the content namespace. An overlay testbed of 60 routers were constructed, which runs on Amazon EC2 in five regions. The results revealed that the proposed CoRC achieves the higher cache hit ratio, and hence reduce the inter-AS traffic, compared to Vanilla ICN. The route stretch caused by CoRC can be alleviated by applying optimizations. Several issues remain for future studies; (i) How to support multihoming cases. (ii) How to assign ideal prefix to RRs, and (iii) How to allocate the backup router and how to exchange the cached item in case that current RRs are down.

Bibliography

[1] Cisco Visual Networking Index: Forecast and Methodology, 2012-2017, May 2013.

[2] Andrea Passarella. Review: A Survey on Content-centric Technologies for the Current Internet: CDN and P2P Solutions. *Computer Communications*, 2012.

[3] Teemu Koponen *et al.* A Data-oriented (and Beyond) Network Architecture. In *Proceedings of ACM SIGCOMM*, 2007.

[4] Van Jacobson *et.al.* Networking Named Content. In *Proceedings of ACM CoNEXT*, 2009.

[5] PURSUIT. `http://www.fp7-pursuit.eu/`.

[6] SAIL. `http://www.sail-project.eu/`.

[7] 4WARD. `http://www.4ward-project.eu/`.

[8] Amazon Elastic Compute Cloud (Amazon EC2). `http://aws.amazon.com/ec2/`.

[9] The ISC domain survey. `http://www.isc.org/solutions/survey`.

[10] Diego Perino *et al.* A Reality Check for Content Centric Networking. In *Proceedings of the ACM SIGCOMM Workshop on Information-Centric Networking*, ICN, 2011.

[11] Carey Williamson *et al.* On Filter Effects in Web Caching Hierarchies. *ACM Transactions on Internet Technology*, 2002.

[12] Steven DiBenedetto *et al.* Routing Policies in Named Data Networking. In *Proceedings of the ACM SIGCOMM Workshop on Information-Centric Networking*, ICN, 2011.

[13] Fragkiskos Papadopoulos *et al.* Greedy Forwarding in Dynamic Scale-Free Networks Embedded in Hyperbolic Metric Spaces. In *Proceedings of IEEE INFOCOM*, 2010.

[14] Named Data Networking. `http://www.named-data.net/`.

[15] Xin Zhao *et al.* On the Aggregatability of Router Torwarding Tables. In *Proceedings of IEEE INFOCOM*, 2010.

[16] Andrea Detti *et al.* Supporting the Web with an Information Centric Network that Routes by Name. *Computer Networks*, 2012.

[17] Hang Liu *et al.* A Multi-Level DHT Routing Framework with Aggregation. In *Proceedings of the ACM SIGCOMM Workshop on Information-Centric Networking*, ICN, 2012.

[18] Reaz Ahmed *et al.* αRoute: A Name Based Routing Scheme for Information Centric Networks. In *Proceedings of IEEE INFOCOM*, 2013.

[19] Yaogong Wang *et al.* Advertising Cached Contents in the Control Plane: Necessity and feasibility. In *Proceedings of INFOCOM NOMEN Workshop*, 2012.

[20] Suyong Eum *et al.* CATT: potential based routing with content caching for ICN. In *Proceedings of the ACM SIGCOMM Workshop on Information-Centric Networking*, ICN, 2012.

[21] Munyoung Lee *et al.* SCAN: Scalable Content Routing for Content-Aware Networking. In *Proceedings of IEEE ICC*, 2011.

[22] Zhongxing Ming *et al.* Age-based cooperative caching in Information-Centric Networks. In *Proceedings of INFOCOM NOMEN Workshop*, 2012.

[23] Ioannis Psaras *et al.* Probabilistic In-Network Caching for Information-Centric Networks. In *Proceedings of the ACM SIGCOMM Workshop on Information-Centric Networking*, ICN, 2012.

[24] Jia Wang. A Survey of Web Caching Schemes for the Internet. *ACM Computer Communication Review*, 1999.

[25] Lorenzo Saino *et al.* Hash-routing Schemes for Information Centric Networking. In *Proceedings of the ACM SIGCOMM Workshop on Information-Centric Networking*, ICN, 2013.

[26] Sumanta Saha *et al.* Cooperative Caching through Routing Control in Information-Centric Networks. In *Proceedings of IEEE INFOCOM*, 2013.

[27] Hitesh Ballani *et al.* Making routers last longer with ViAggre. In *Proceedings of USENIX NSDI*, Boston, Massachusetts, 2009.

[28] Lada A. Adamic *et al.* Zipf's Law and the Internet. *Glottometrics*, 2002.

[29] AS Number Analysis Reports. `http://bgp.potaroo.net/`.

[30] Michalis Faloutsos *et al.* On power-law relationships of the Internet topology. *SIGCOMM Computer Communications Review*, 1999.

[31] Hongsuda Tangmunarunkit *et al.* Does AS size determine degree in as topology? *ACM SIGCOMM Computer Communications Review*, 2001.

[32] Bradley Huffaker *et al.* Toward Topology Dualism: Improving the Accuracy of AS Annotations for Routers. In *Proceedings of PAM*, 2010.

[33] Ingo Scholtes *et al.* TopGen - Internet Router-Level Topology Generation Based on Technology Constraints. In *Proceedings of SIMULTOOLS*, 2008.

[34] Internet topology at router- and AS-levels, and the dual router+AS Internet topology generator. `http://www.caida.org/research/topology/generator/`.

[35] Bruno Quoitin *et al.* IGen: Generation of Router-level Internet Topologies through Network Design Heuristics. In *Proceedings of ITC*, 2009.

[36] CCNx. `http://www.ccnx.org/`.

초 록

인터넷이 처음 발명되었을 때, 대부분의 애플리케이션은 원격 접속 혹은 파일 전송 등과 같이 자원을 공유하기 위한 목적으로 이용되었다. 그러한 애플리케이션들을 지원하기 위한 인터넷 아키텍처는 점 대 점 통신을 기반으로 하였으며, 핵심이 되는 요소는 특정 호스트를 식별하는 아이피 주소였다. 이후 수 십 년 간 다양한 애플리케이션들과 서비스들이 새로이 등장하면서 인터넷의 사용 패턴은 크게 변화하였다. 시스코에 의하면 인터넷 트래픽의 대다수는 웹, P2P, 파일 공유, 그리고 비디오 스트리밍이 점유하고 있다. 이러한 새로운 애플리케이션들에서 사용자는 특정 위치나 호스트가 아니라 콘텐트 자체에 관심을 두고 있으며, 점 대 점 통신에 기반한 인터넷 설계와 정보 중심적인 현재의 이용 패턴 사이의 괴리는 콘텐트 전송의 비효율성을 비롯해 많은 문제를 야기하였다. 아이피 주소에 기반해 전달을 하는 네트워크 노드들이 자신이 어떤 콘텐트를 전송하고 있는지 알 수 없기 때문이다.

이러한 문제를 해결하기 위해 네트워크를 정보 중심적인 관점에서 새롭게 설계하자는 목적 하에 정보 중심 네트워크 (Information-Centric Networking, ICN)가 등장하였다. 정보 중심 네트워크에서의 핵심은 (i) 콘텐트 이름에 기반해 라우팅을 하며, (ii) 네트워크 노드들이 스토리지를 가지고 있어 지나가는 콘텐트를 인식하고 캐싱할 수 있는 것이다. 이것은 콘텐트의 송신과 수신을 시간적, 공간적으로 분리함으로써 콘텐트의 가용성을 강화시키고 멀티캐스트 및 이동성을 구조적으로 지원하게 된다.

본 학위논문은 정보 중심 네트워크에서 발생하는 많은 이슈들 중에서 라우팅과 캐싱에 집중하였다. 정보 중심 네트워크가 콘텐트 이름을 이용하기 때문에 라우팅 테이블이 증가하는 문제를 해결하였으며, 각 네트워크 노드가 독립적으로

캐싱을 함으로써 캐싱 스토리지를 비효율적으로 해결하는 문제를 해결하였다. 이전의 연구들은 라우팅과 캐싱 각각이 풀어야 할 문제를 독립적으로 취급해 왔으나 본 학위 논문은 이들을 하나의 프레임워크로 묶어서 각각이 직면하고 있는 상이한 목적을 동시에 달성하고자 하였다.

주요어: 정보 중심 네트워킹, 콘텐트 중심 네트워킹, 라우팅, 캐싱
학번: 2008-20999