Studies on Traffic Control Schemes in Information Centric Networks

その他のタイトル	情報指向ネットワークにおけるトラヒック制御方式に関する研究
著者	速水 祐作
year	2019-03-31
学位授与機関	関西大学
学位授与番号	34416甲第719号
URL	http://doi.org/10.32286/00018635

課程博士

2019年3月

関西大学審査学位論文

Studies on Traffic Control Schemes in Information Centric Networks

情報指向ネットワークにおけるトラヒック制御方式に関する研究

Yusaku Hayamizu

関西大学大学院 理工学研究科 総合理工学専攻

電気電子情報工学分野 情報通信工学領域

16D6001 速水祐作

博士論文要旨

理工学研究科 総合理工学専攻
電気電子情報工学分野 情報通信工学領域
16D6001
速水祐作

≪論題≫
Studies on Traffic Control Schemes in Information Centric Networks
情報指向ネットワークにおけるトラヒック制御方式に関する研究

≪概要≫
インターネットに流通する総トラヒック量は 2021 年までに 3.3 ゼタバイトに迫り，その中でもコンテンツ配信に関するトラヒックの割合は 82%になるとされている．この背景には，YouTube や Netflix に代表されるコンテンツプロバイダにより提供されるマルチメディアサービスの普及がある．これらのサービスに対して，我々ユーザの要望はコンテンツ自体のダウンロードであり，そのコンテンツがどこに存在するかは全く気に留めていない．いわば，ユーザがネットワークに求める要件はコンテンツ指向である．しかしながら，現在のインターネットは開発当初の設計理念を継承しており，IP (Internet Protocol) に基づいたローケーション指向なアーキテクチャである．つまりは，コンテンツをどこから取得するかをユーザは明示的に指定しなければならない．この乖離を無くし，効率的で安定したコンテンツ配信を実現するため，情報指向ネットワーク（ICN: Information Centric Networks）が提案された．この ICN の実現に向けて様々な技術課題が存在し，それらの課題を解決するため数々の研究者が日々研鑽を積んでいる．

本論文では，ICN におけるトラヒック制御に関する技術課題について検討する．本論文におけるトラヒック制御は，トラヒックが伝送される経路を制御する経路制御，ルータに一時的に保持されるコンテンツを取捨選択するためのキャッシュ制御，輻輳の発生を抑制してネットワーク資源を有効利用するための輻輳制御を含む．それに加え，ICN の災害時適用のための経路制御やキャッシュ制御，IoT (Internet of Thing) 環境におけるデータ処理への応用に向けた経路制御について着目し，一貫してトラヒック制御という見地から包括的な検討を行う．

≪各章の要旨≫
第一章は緒論であり，インターネットに代表される今日の IP（Internet Protocol）ネットワークが直面する課題と，それを解決するために提案されたネットワークアー

キテクチャである情報指向ネットワーク（ICN: Information Centric Networks）について概説する．その後，情報指向ネットワークにおける技術課題について述べ，既存の関連技術について説明し，本研究の工学的な意義と目的を明確化する．

第二，三，四章は ICN のトラヒック制御における基礎技術を対象とする．

第二章では ICN 固有の特徴であるネットワーク内キャッシュの利用効率を向上するため，経路制御とキャッシュ制御の統合効果に関する検討を行い，その効果を明らかにする．通過ルータすべてにキャッシュするという単純な TERC(Transparent En-Route Cache) の問題点を改善する形で提案されている既存キャッシュ制御は，経路制御として最短経路選択（SPR: Shortest Path Routing）を用いることを前提としている．本章では，最短経路以外に存在するキャッシュコンテンツを有効利用できるネットワーク内誘導方式（Breadcrumbs）を経路制御に用いた場合の性能を評価し，従来提案されているキャッシュ制御は Breadcrumbs 方式と組合せると十分効果を発揮できない事，SPR と組合せ性能が悪いとされていた TERC が Breadcrumbs 方式とは良好な組合せ効果を発揮する事を示す．それにより，ICN におけるキャッシュ利用において経路制御とキャッシュ制御の組合せが重要な性能決定因子であるという事を示す．

第三章では，第二章で明らかにされた知見を受け，経路制御とキャッシュ制御の統合効果の観点を踏まえつつ TERC の問題点であるネットワークキャッシュの安定性について述べる．そして，その問題を解決する Breadcrumbs に適した新しいキャッシュ制御方式を提案し，その有効性を示す．本方式では，コンテンツの人気度とネットワークトポロジにおけるネットワーク中心性に着目し，ユーザに要求されやすい高人気コンテンツをネットワークエッジに配置し，中程度の人気コンテンツをコア付近に配置する．これにより，キャッシュコンテンツを安定的にネットワーク内に滞在させ，さらにこれら安定して存在するキャッシュコンテンツをネットワーク内誘導によって利用可能となり，無駄な誘導を低減する効果が確認された．

第四章では，ICN における輻輳制御について述べる．IP は単一送信者のみからコンテンツ取得を行う通信モデルであるのに対し，ICN はコンテンツが何処から提供されても良いというコンテンツ指向な設計理念により複数送信者からのコンテンツ取得を許容している．そのため，IP と ICN は通信形態が大きく異なり，新たな複数送信者に対応した輻輳制御が必要となる．本章では複数送信者に対応した輻輳制御である MSC4N (MultiSource Congestion Control for Content Centric Networks) を対象に，ユーザ間の公平性を表す代表的概念である Resource Pooling の観点で評価を行い，IP における代表的なマルチパス輻輳制御である MPTCP (Multipath Transmission Control Protocol) と比較検討する．ICN を対象とする新しい輻輳制御方式 MSC4N が，IP における MPTCP と同様の設計理念に基づいており，複数送信者環境で有用な方式である事を性能評価により新たに明らかにしている．

第五，六章は，ICN のトラヒック制御における応用技術を対象とする．

第五章では，ICN の災害時応用について述べる．大規模な地震が発生するとインターネットは通信インフラ設備の破損により利用不可能となる．これは，IP が特定のサーバとセッションを構築した上で通信するセッションベースの形態を採用している事に起因する．それに対し，ICN はセッションレス通信であり特定のサーバとの接続性が通信性能に影響を及ぼさない．その上，ルータはキャッシュ機能を具備しているためコンテンツ配信性能も高く，災害時への ICN 応用によりネットワーク性能の向上が期待できる．しかし，ICN は有線ネットワークを対象としているため，災害時のような通信環境が劣悪な無線ネットワークへ単純に適用できない．災害時の無線ネットワークで通信リンクが断続的な通信状態をもち，リンク切断時に冗長なパケットの再送が頻発するという技術課題が存在する．本章ではこの課題に対し，動的な経路制御と通信リンクの状態に応じたパケット転送制御を提案し，リンクが切断状態にある場合における冗長な再送を抑制する方式を提案する．本提案により，災害時において貴重な無線資源とユーザ端末の電力消費を抑制した通信を実現する．

第六章では，ICN の IoT（Internet of Things）データ処理への適用について述べる．IoT の普及により膨大な量のセンサデータがクラウドにおいて集約・処理される．しかし，IoT においてはより厳しいコンピューティング要件が求められるようになり，エッジコンピューティングなどのネットワークエッジ付近におけるデータ処理が着目されている．エッジにおいて効率的なデータ処理を実現するためには，データ処理を柔軟に機能へ割り当てるような仕組みが重要となる．本章では，このような ICN の IoT への応用における処理要求の経路制御が，処理データならびに処理設備などのネットワーク資源を適切に制御するトラヒック制御の一部であるととらえ，ICN を利用してネットワーク内のデータ処理機能を発見するための経路制御手法を提案する．本手法ではフラッディングにより必要な機能を全探索し，発見した機能の組合せの中から最適なデータ処理経路を選択する．多くの機能の組合せに対し，ルーチングテーブルを事前に用意する手法では実現性が低いため，本章ではフラッディング探索によりデータと処理機能を適切に繋ぐ手法を提案した．性能評価により，提案方式がフラッディングによるオーバーヘッドを低く抑制しつつ，効率的にデータ処理を実現する方式であることを示した．

第七章は結論であり，本論文で得られた結果についてまとめるとともに今後の課題について述べる．

以　上

Abstract

The amount of Internet traffic reached 1.2 zettabytes by the end of 2016 and is expected to be 3.3 zettabytes by 2021. The ratio of content distribution traffic is reported to be 82% by 2021 due to rapid increase of multimedia services, e.g. YouTube and Netflix. In these services, almost users just would like to download the contents themselves and do not care about location of the contents. In other words, the users' requirements for networks are information-centric. However, the current Internet which transfers the traffic is still location-centric architecture. In order to fill this gap and support stable content distribution, ICN (Information Centric Networks) have been proposed, and a lot of effort has been devoted by many researchers in the ICN committees thus far to resolve a number of technical issues.

Chapter 1 gives research background of ICN and describes the CCN/NDN (Content-Centric Networking/Named-Data Networking) architecture which is one of the promising network architectures realizing the ICN concept. Then, we shed light on technical issues on traffic control schemes in CCN/NDN. In this thesis, we specialize only in the issues on route, cache, and congestion control as basic technologies, and those on applications to disaster communications and data processing in IoT (Internet of Things) as applied technologies. Finally, we clearly explain the outline of the thesis.

In Chapters 2, 3 and 4, we make studies on basic technologies in CCN/NDN traffic control.

Chapter 2 focuses on combination effects of routing and caching policies in ICN for leveraging in-network caches. Existing caching policies that improve cache hit performance compared to the simple caching policy, TERC (Transparent En-Route Cache), generally assumes shortest path routing (SPR) as routing policy. In this chapter, we firstly evaluate cache hit performance in the case that Breadcrumbs (BC) which is an in-network guidance method in ICN is applied to routing policy. Simulation results show that BC does not perform well with the previously proposed caching policies and surprisingly performs well with TERC which has low performance with SPR. From these results, we reveal that the combination of routing and caching policies is dominant factor for performance in ICN.

Chapter 3 gains deep insight from Chapter 2, and makes a study on stability of in-network caches which is a technical problem of TERC. In order to resolve this problem, we propose caching policy suitable for BC. Our proposed caching policy focuses on popularity of a required content and betweenness centrality of nodes on the download path of the content. With this policy, highly popular contents are likely to be located around edge areas of the network and moderately popular contents are stored in the core area. As a result, we can utilize these stabilized cached contents by BC guidance. This leads to reduction of redundant BC guidance and improvement of cache hit performance.

In Chapter 4, we make a study on multipath/multisource congestion control in CCN/NDN. Communication model of IP is one-to-one, i.e. one sender to one receiver. In contrast, that of CCN/NDN is many-to-one, i.e. multiple senders to one receiver, because CCN/NDN natively allows multisource information retrieval due to its architectural concept. MSC^4N (MultiSource Congestion Control for Content Centric Networks) has been proposed as a congestion control scheme for many-to-one multipath communications in CCN/NDN. In this chapter, from viewpoint of window control, we compare MSC^4N with MPTCP (Multipath Transmission Control Protocol) which is representative multipath congestion control in IP networks. Also, we evaluate performance of MSC^4N from viewpoint of resource

pooling which is the famous concept for fair sharing of network resources among users and compare its performance with MPTCP from resource pooling viewpoint. From these results, we reveal that MSC^{4}N is based on the design policies of MPTCP and effective congestion control for multipath/multisource communications in CCN/NDN.

Chapters 5 and 6 make studies on applied technologies in CCN/NDN traffic control.

In Chapter 5, we try to apply CCN/NDN to disaster communications in order to improve communication quality and reliability. When large disaster strikes such as earthquake have occurred, the Internet cannot be available due to damages of communication infrastructures. This problem stems from the location-centric feature of IP networks, i.e. session-based communication model with IP address. Users cannot communicate with content servers unless they know IP addresses of the content servers. In contrast, CCN/NDN is information-centric architecture and has session-less communication model. Connectivities to specific servers such as DNS do not affect communications quality between users and content servers. Furthermore, content distribution performance can be improved by applying CCN/NDN to disaster communications because routers temporarily store data packets in their cache storages as in-network cache. CCN/NDN has been, however, originally assumed to be used in well-connected wired networks, and it cannot be easily applied to disaster situations. In disaster situations, communication links are intermittent and redundant packet retransmissions are redundantly repeated while links are disconnected. In this chapter, we propose a dynamic routing protocol and packet forwarding schemes that adapt to changes of link conditions. In our proposed method, redundant packet retransmissions are suppressed during link disconnection. With our proposals, we can realize energy and bandwidth efficient information retrieval in disaster environment.

Chapter 6 explains an application of CCN/NDN to IoT data processing. Currently, tremendous amount of data are collected by sensor devices, transferred to distant clouds, and processed in the clouds. This computing model is not efficient because large amount of

traffic for data processing passes through the Internet. On one hand, in-network processing where the collected data are processed around edges of the network close to the sensor devices gains much attentions. In this environment, a network itself finds data processing functions in order to efficiently process data around the edge. In this chapter, we regard routing of data processing request as a part of traffic control schemes for appropriately utilizing network resources. Proactive approaches are not practical because they need to prepare large amount of routing entries in the forwarding table. Thus, we propose a reactive routing method to discover required data processing functions with ICN in a distributed manner. This method utilizes flooding search and selects the best route for data processing from all discovered combinations of data and functions. In the performance evaluation, we reveal that our proposed method can flexibly process data in a short time with low control overhead.

In Chapter 7, we conclude this thesis and describe the future works.

Acknowledgements

I would like to express my sincere appreciation to Prof. Miki Yamamoto, my advisor in Graduate School of Science and Engineering of Kansai University, for his innumerable help and enthusiasm in teaching and studying traffic control in information centric networks. An encounter with him has been great opportunity for attracting my heart to information centric networks. Thanks to him, my ambition has been kept high throughout my student life, and it will continue on my research career.

I am heartily grateful to Prof. Yoshinobu Kajikawa and Prof. Hiroyuki Yomo, readers of my thesis committee, for their insightful comments on my thesis. Their expertise have definitely polished this thesis.

I would like to extend my appreciation to Dr. Tadahiro Kitahashi, Emeritus Professor of Osaka University, Dr. Masaki Bandai, Professor of Sophia University, Dr. Tomohiko Yagyu of NEC Corporation, Dr. Akira Nagata of iD Corporation and Dr. Kouji Hirata of Kansai University, for their helpful advises and precious time for valuable discussions. Their kindness and efforts on my behalf are invaluable forever in my life.

I am truly thankful to my friends in the laboratory for their candid discussions, fellowship and continuous encouragement – special thanks to Mr. Kento Ikkaku and Mr. Daisuke Sugahara for their technical suggestions and kindness.

I dedicate this thesis to my parents who have loved, understood and supported me thus far.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

The amount of Internet traffic reached 1.2 zettabytes by the end of 2016 and is expected to be 3.3 zettabytes by 2021 [62]. The ratio of content distribution traffic is reported to be 82% by 2021 due to rapid increase of multimedia services, e.g. YouTube and Netflix. In these services, almost users just would like to download the contents themselves and do not care about location of the contents. In other words, the users' requirements are *information-centric*. However, the current Internet which transfers the traffic is still *location-centric* architecture. Communications are based on location ID, i.e. IP address, and packets for communications are transferred based on the IP addresses of endpoints.

ICN (Information Centric Networks) [18, 1] have been proposed to fill this gap and support stable content distribution. One of the most important goals of ICN is *space decoupling*. In other words, users can download their required contents anywhere from networks. When a user requires a content, the network itself flexibly discovers the content by *name* and transfers the content from an appropriate server to the user. ICN is just a general concept for efficient networking where communications are based on name of contents (information) and not a specific architecture.

In the early years, several network architectures that realize this ICN concept has been proposed [26, 54, 30]. *i3* (Internet Indirection Infrastructure) [54] is IP-based architecture that utilizes rendezvous-based communication model. Content discovery is conducted in an overlay network. Senders and receivers can communicate without knowing each other, thus this architecture realizes the ICN concept. DONA (Data-Oriented Network Architecture) [30] conducts name resolution in the application layer. While DNS name resolution is run outside the network, DONA conducts it inside the network by introducing Resolution Handler. In TRIAD [26], content routers cooperatively execute name resolution by advertising content name as in the case with route advertisement like BGP (Border Gateway Protocol). After finishing the name resolution, content itself is transferred based on IP addresses. These architectures are achieving the goal of the ICN concept, however, designed to run on the location-centric IP networks. Therefore, it cannot be said that these network architectures are completely information-centric ones.

The rise of CCN/NDN (Content-Centric Networking [28]/Named-Data Networking [59]) has made a significant impact on ICN research committees. CCN/NDN is the novel network architecture realizing the ICN concept in a distributed manner. Unlike IP, CCN/NDN does not require any location ID for communications and utilizes just a content name for communications between a content server and a user. In CCN/NDN, a user transmits a request packet to retrieve a content. The name of the content is written in the request and used for name-based routing at intermediate routers. When the request is transferred to a content server, the data packet is replied along the reverse path of the request packet. The intermediate routers temporarily store the data packet in their cache storages. Since the cached data packet can be reused by other users, effective utilization of the cache storage can reduce load of content servers, network traffic and latency for users. Therefore, CCN/NDN has attracted great attentions for its technological innovativeness as one of the promising network architectures that realize the ICN concept.

1.2 CCN/NDN

1.2.1 Overview of CCN/NDN

CCN/NDN has been proposed by V. Jacobson et al./L. Zhang et al. as the pure ICN architecture. There are some subtle differences of detailed components between CCN and NDN, however, the basic structures are the same. In this thesis, we would like to consistently use terminologies of CCN to avoid confusions.

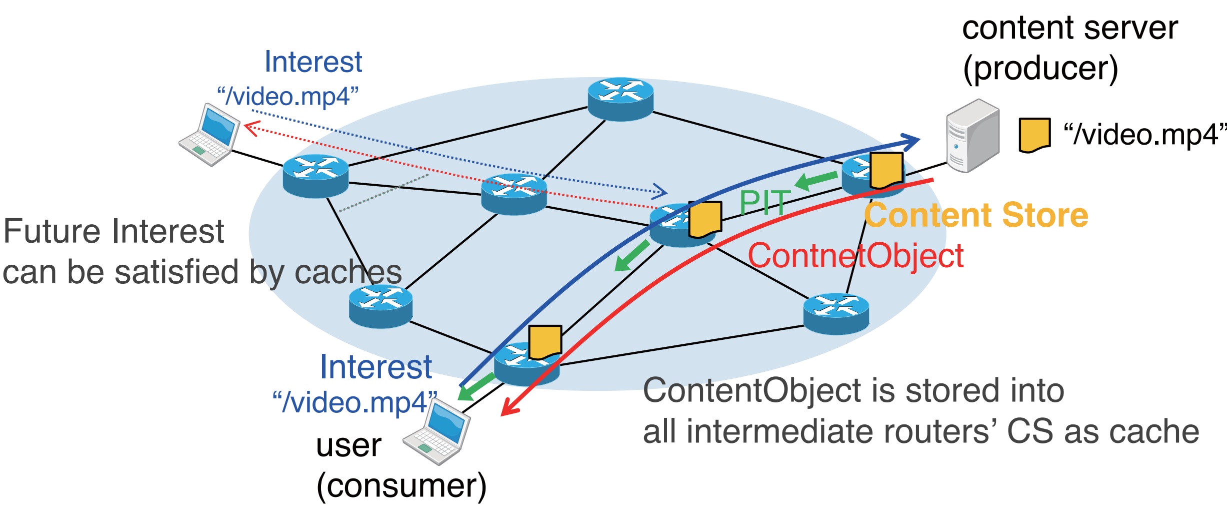


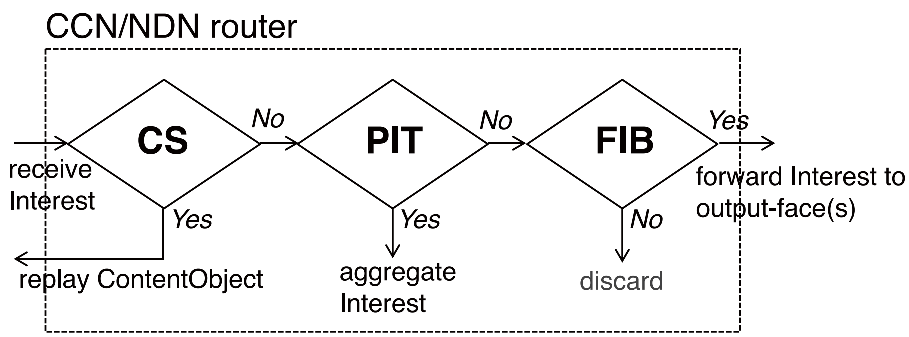
Figure 1.1: Example of Packet forwarding in CCN/NDN.

Figure 1.1 shows an example of packet forwarding in CCN/NDN. CCN/NDN has a receiver-driven architecture and a user (consumer) transmits an Interest packet to retrieve a ContentObject packet from content servers (producers). The Interest packet has a name of a content, e.g. "/video.mp4" and the ContentObject packet has the corresponding name of the Interest packet. An Interest packet is forwarded to a content server by looking up a name entry in FIB (Forwarding Information Base) in the router. During Interest transmission, each intermediate router temporarily records information of the incoming interface of the Interest as PIT (Pending Interest Table) to reply the ContentObject packet. A ContentObject packet is transmitted based on the PIT, i.e. reverse paths of the Interest packet, in a hop-by-hop manner. During ContentObject transmission, routers on the download path store the ContentObject packet in their CS (Content Store) based on their *caching policies*. When an Interest packet arrives at a router, the router looks up the CS. If the name of the Interest packet matches the name of a stored ContentObject packet, the router replies the ContentObject packet; otherwise, the router forwards the Interest packet to content servers based on its FIB.

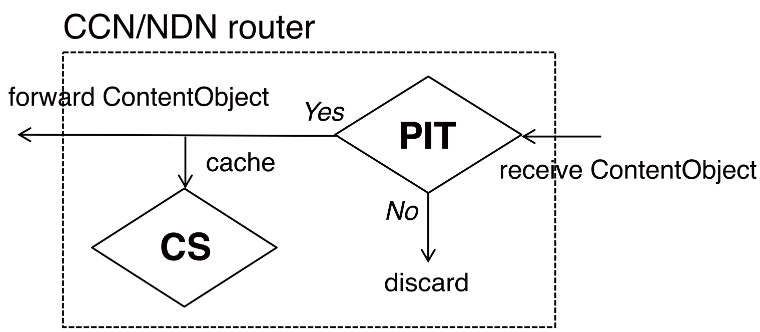
These communications are executed by CCN/NDN routers that have a name-based forwarding mechanism. Figure 1.2 shows the packet forwarding mechanism in a CCN/NDN router. As shown in Fig. 1.2(a), when a router receive an Interest packet, the router lookup CS based on the name of the Interest packet. If there is the corresponding ContentObject packet, the router replies the ContentObject packet; otherwise, the router lookup PIT with exact match. If there is the corresponding PIT entry, the Interest packet is aggregated and not forwarded. When there are no corresponding entries in CS and PIT, the Interest packet is forwarded to the upward routers by looking up FIB with longest prefix match. If there is no FIB entry, the Interest packet is discarded. When a router receive a ContentObject packet as in Fig. 1.2(b), the router lookup its PIT entry. If there is the corresponding PIT entry, the ContentObject is copied to the router's CS and forwarded to downward router based on PIT entry; otherwise, the ContentObject is just discarded.

1.2.2 Technical Issues in CCN/NDN

Toward efficient information retrieval, CCN/NDN has been developed, and a lot of effort has been devoted by many researchers in the ICN committees thus far. However, there are a number of technical issues to be resolved for spread of CCN/NDN. For example, from



(a) Interest forwarding mechanisms.



(b) ContentObject forwarding mechanisms.

Figure 1.2: Packet forwarding mechanisms in CCN/NDN router.

the implementation aspect, CCN/NDN routers which have high-speed packet forwarding function are required for realizing stateful forwarding. From the security aspect, there are problems for certificating content consumers and protecting from attacks of malicious users. However, the most important benefit of CCN/NDN is *in-network cache*, which is the inherent feature of CCN/NDN. In order to utilize this effective resource, traffic control that steers data traffic where to transport is a key research topic.

In this section, we explain technical issues on traffic control schemes in CCN/NDN. We categorize these issues into two technologies: *basic technology* and *applied technology*. For the basic technology, there are three main research topics: route control, cache control and congestion control. Route control (Routing) is one of the most important topics because the best route selection brings reduction of traffic load and users' communication latency. By developing cache control schemes, we can effectively utilize in-network caches and reduce traffic load and users' latency as we mentioned before. Developments of congestion control schemes are essential for efficient communications fully utilizing network resources and avoiding occurrences of network congestion. For the applied technology, we focus on applications of CCN/NDN to disaster communications and IoT data processing. By adopting CCN/NDN as the network architecture in these applications, we can effectively connect network applications and realize application-oriented communications.

Utilization of In-Network Caches

Lots of studies on improving cache utilization have been published [32, 5, 33, 29, 16, 22, 44, 43]. These works can be categorized into two approaches: *cache decision policy* and *cache replacement policy*. The cache decision policy of a router determines whether it stores incoming data packets. When an incoming data packet is decided to be stored by a router's cache decision policy and the router's cache storage is fully utilized, the router must decide which cached content(s) to be evicted from the cache storage in order to make space available, which is called cache replacement policy. Routers in ICN manage their cached contents based on these two cache management schemes. Most proposals that use these two policies generally assume shortest path routing (SPR) for content request routing. In SPR, only cached contents on the on-path, i.e. the shortest path from a consumer to a content server, can be utilized. This means a cached content on the off-path, i.e. outside the on-path, cannot be used even though it happens to be located close to the consumer.

Cache hit is caused by arrival of a request packet at a router. Content request routing is another dominant factor for cache performance in ICN. Several forwarding policies of request have been proposed to expand range of cache utilization to off-path, i.e. outside on-paths in SPR. Flooding [19] floods request packets and aggressively retrieves cached contents from off-paths. INFORM [20] forwards request packets to off-path caches with reinforcement learning based on content retrieval latency. CATT [21] calculates potential for cache discovery based on distance between a transmitted request packet and cached contents, and utilizes off-path caches. These forwarding policies, however, are not practical in terms of overhead such as control packet transmission and route calculation.

G. Rossini et al. [48] revealed that the combination of request forwarding policy "$\mathcal{F}$", cache decision policy "$\mathcal{D}$", and cache replacement policy "$\mathcal{R}$" plays an important role for performance of cache networks and the combination of $\mathcal{F}$ and $\mathcal{D}$, i.e. $<\mathcal{F}, \mathcal{D}>$, is especially a dominant factor. Also, they proposed Nearest Replica Routing (NRR), the off-path cache forwarding policy. In NRR, a user firstly floods a request packet and probes the nearest replica (cache). After that, the user forwards subsequent request packets to the cache node. This is the first work that evaluated the combination of $\mathcal{F}$ and $\mathcal{D}$, and used not only SPR but also off-path routing for $\mathcal{F}$. However, NRR is not practical in terms of network load because this method uses flooding for content discovery.

Breadcrumbs (BC)[47] has been proposed to probe off-path caches with low network load. When a user downloads a content from an original content server, each intermediate router sets a pointer indicating the download direction called breadcrumbs (bc). The chain of these bc is called bc-trail. If a request packet of another user occasionally encounters a breadcrumb pointer stored at a router along its on-path, its forwarding is switched from FIB-based forwarding to hop-by-hop forwarding along bc-trail. Because content download forms routing table of BC, i.e. bc pointers, BC does not require any control overhead for off-path routing. In this sense, BC is the best-effort approach with low control overhead. For cache decision policy $\mathcal{D}$ in BC, TERC (Transparent En-Route Cache), which is well-known to have low performance with widely-used SPR, has been assumed [47]. As G. Rossini revealed, cache performance crucially depends on the combination of cache decision policy $\mathcal{D}$ and request forwarding policy $\mathcal{F}$, and these characteristics can be also seen in the case for the forwarding policy of BC.

Controlling Network Congestion

Resouce Pooling is a key concept for efficient utilization of communication resources in a network. D. Wischik et al. [57] define resource pooling as "making a collection of resources behave like a single pooled resource". Conventional fairness concept, i.e. fair share of a bottleneck link is microscopic fairness as represented by TCP (Transmission Control Protocol) fairness [17]. Fairness in resource pooling is *macroscale*, i.e. macroscopic fair share that users fairly share common network resources. MPTCP (Multipath Transmission Control Protocol) [46, 24] has achieved this concept by cooperatively controlling congestion windows for multiple paths in IP (Internet Protocol) networks.

In CCN/NDN, communication is initiated by a user's content request. The communication model in CCN/NDN is a pull-based one. A content request has the name of a required content file and is transferred based on name-based routing tables, i.e. FIB in routers. Since each FIB entry is created by a routing advertisement from the content server and replicated servers that provide the same content, advertisements from these servers might create multipath information for each content in FIB of each router. In other words, CCN/NDN natively allows multipath/multisource communications as its architectural concept. Congestion control, therefore, should be also adaptive to multipath/multisource communications and achieve resource pooling for macroscopic fairness among users.

CCN congestion control methods [11, 50, 13, 49, 12, 41, 45] which have been proposed thus far, have not conducted an adequate study in the aspect of resource pooling. MSC^{4}N (MultiSouce Congestion Control for Content Centric Networks) [38] has been proposed for multipath communications in CCN/NDN environment. This method adaptively controls data traffic transferred in a bottleneck link among multiple paths, and does not require any location information to adequately regulate congestion at the bottleneck link. As we will carefully discuss later in Chapter 4, this method would satisfy MPTCP's design goals and has a possibility to accomplish resource pooling in CCN/NDN.

Application to Disaster Communications

Information network is one of the most essential social infrastructures. Survivability of the information network in disaster stricken areas is very important because accurate information are vital for prompt recovery. In the current Internet, connectivity between users and servers depends on not only a transmission path between them but also reachability to the DNS (Domain Name System) servers. DNS resolves IP address of information servers in response to the users' queries for URL (Uniform Resource Locator). If DNS becomes unavailable, all users always fail to reach the information servers with URL. When a large-scale disaster occurs, victims are more likely to obtain the information of disaster damage and safety of relatives. In such information retrievals, users do not care about the place where the information is obtained if the information is authenticated. From this information-centric viewpoint, information-centric architectures [1, 18], e.g. CCN/NDN, are suitable for information retrieval in the disaster situation. In CCN/NDN, contents can be obtained not only from the original servers, but also from caches (copies) stored in the routers. This information-centric feature leads to the robustness in terms of the content retrieval.

According to the report by the Ministry of Internal Affairs and Communications of Japan [64], after the great East Japan Earthquake, about 5 million people became *returner refugees*, who could not reach their home because they had no means of public transportation. There were more than 2000 shelters. The communication components such as base stations were damaged and it was difficult to obtain accurate information. The police and the Self-Defense Force went round among shelters to gather information. When a disaster occurs, information network infrastructures might be fragmented into disconnected local networks due to the physical damage and outage. We assume that most nodes such as smartphones in each local network have the connection to a Wi-Fi access point in a shelter. Among these fragmented local networks, vehicles called data mules go around in order to make short-term connectivity to each of these fragmented networks. The short-term connectivity held by a data mule enables nodes to exchange information among the fragmented networks. This network environment is known as DTN (Delay Tolerant Network). RFC4838 (Delay-Tolerant Networking Architecture)[14] describes DTN architecture which embraces the concept of occasionally-connected networks.

In this thesis, we try to apply CCN/NDN in the fragmented networks which often suffer extremely long-term disconnection by *intermittent links*, such as several hours or a few days. Since CCN/NDN architecture is originally designed for the well-connected networks, there have not been any proposed routing protocols and retransmission control schemes suitable for DTN. When CCN/NDN is simply applied to the fragmented networks, retransmission of the packets frequently occurs due to the short timeout interval. Such frequent retransmissions cause severe degradation of wireless network performance and reliability of information retrieval. The intermittent links has bad effects on not only wireless network performance but also cache behavior in routers. Because the original caching policy of CCN/NDN does not take account of popularity of content, unpopular contents can be stored in routers wastefully.

Application to Data Processing in IoT

In IoT (Internet of Things), large amount of data requires data processing. One of the most promising approaches for this requirement is edge computing [51, 63, 8]. Although moving computation resources from clouds to network edges reduces traffic load in the Internet, deploying the resources to all network edges is not practical in terms of implementation cost. Alternatively, *in-network processing* where intermediate nodes between the clouds and the network edges process data can be another approach for IoT data processing.

CCN/NDN is the promising network architecture that realizes ICN concept. CCN/NDN has been proposed originally for efficient content distribution, yet it can be applied to in-network processing environment. NFN (Named-Function Networking) [56, 53] is a novel architecture that enables in-network processing with the CCN/NDN framework. However, NFN passively locates downloaded functions[1] inside a network, which might cause ineffective distribution of functions.

NFaaS (Named Function as a Service) [31] has been proposed as an framework for ICN-based in-network function execution and addressed especially for a location problem of functions according to demand of IoT applications. NFaaS, however, only assumes a single function, e.g. delay-sensitive or bandwidth-hungry function, and types of data processing are basically limited to simple tasks. Another approach where multiple functions are prepared in a whole network and chain of these functions provides required data processing, is expected to perform flexible data processing for IoT. In this environment, routing methods that efficiently chain the multiple functions are highly required.

L. Liu et al. [34, 35] proposed function chaining that takes all information of required functions into account at hop-by-hop forwarding decision of request packets. This approach, however, requires proactive routing table, FIB, for all data and multiple functions. In spite of the fact that scalability issues of FIB is still controversial in CCN/NDN, this approach seriously exacerbates those issues. In ICN-based function chaining, we need to rethink a routing method in terms of scalability of FIB.

FIB is preliminarily constructed by proactive routing protocol at each router for advertised destinations. In ICN-based function chaining, each FIB entry needs to be constructed for each combination of data and functions to efficiently satisfy users' various requests for data processing. Consequently, the number of required FIB entries significantly increases.

[1]In this chapter, one of the elements of a data processing is called a function.

Also, in the proactive routing, when routes for function chaining have high temporal locality, some FIB entries might be rarely accessed and needlessly retained for a long time. It is wasteful use of memory resources in routers and is a significant technical problem from the viewpoint of scalability.

1.3 Outline of the Thesis

This thesis makes studies on traffic control schemes in ICN from the viewpoint of the aforementioned basic technologies and applied technologies. Figure 1.3 shows the bird's-eye view of this thesis. In Chapter 2, we study combination effects of route control schemes and cache control schemes in ICN. Chapter 3 obtains knowledge from Chapter 2, and proposes a cache control scheme which is suitable for a route control scheme with in-network guidance. Chapter 4 handles congestion control schemes in CCN/NDN. Chapter 5 focuses on an application of CCN/NDN to disaster communications and proposes route and cache control schemes. In Chapter 6, we make a study on application of CCN/NDN to IoT data processing in the aspect of route control.

In Chapter 2, we evaluate combinations of the off-path routing policy, BC, with several representative cache decision policies both in IP and CCN/NDN and show combination effects of routing and caching policies. Firstly, we evaluate BC with several cache decision policies [5, 33, 29] which are reported to have high performance with a generally used request forwarding policy, SPR. We select CCN/NDN for the evaluated network architecture. Also, we assume the IP for evaluated network because BC is originally proposed for IP network [47]. One of the most important difference between IP and CCN/NDN is content download path. In CCN/NDN, content download path is the completely reverse path of the request forwarding path (in [47] this policy is called DFQ: Download Follows Query).

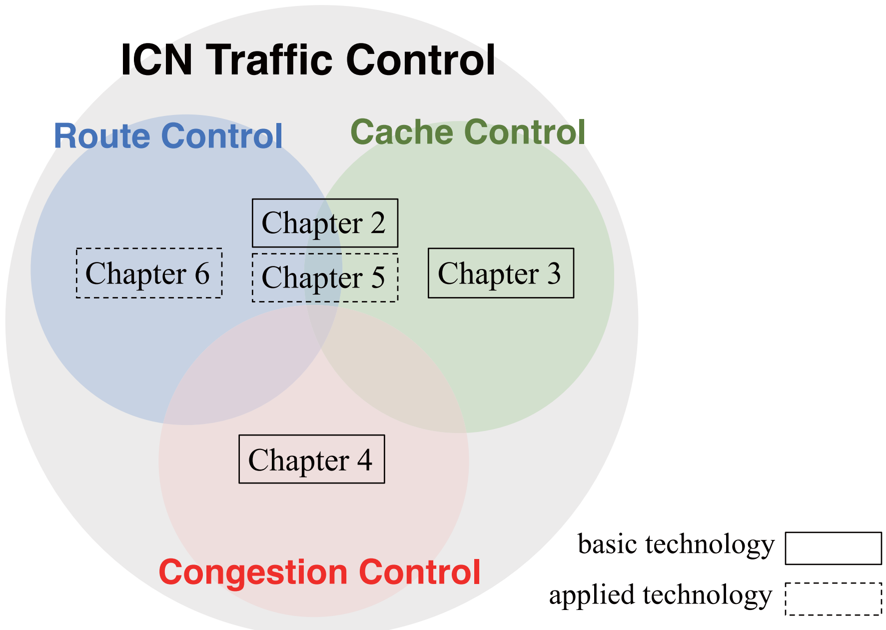


Figure 1.3: The bird's-eye view of this thesis.

However, in IP, content download path is basically[2] the shortest path between a content (stored in the server or a cache) and the requested user (in [47] this policy is called DFSP: Download Follows Shortest Path).

As we will describe in Chapter 2, the combination of BC and TERC is efficient because this combination brings diversity of cached contents. However, in TERC (also called CEE, Cache Everything Everywhere), each intermediate router on a download path always caches incoming data packets. This cache decision, TERC, is well known to make cached contents in a network highly dynamic, which means they are frequently replaced. Frequent replacement of cached content might degrade BC performance due to trail invalidation of BC, i.e. failure of request guidance. In Chapter 3, we propose a cache decision policy suitable for BC. Our proposed cache decision policy decides whether to store incoming data

[2]In fact, download path of IP is not limited only to the shortest path and can be even multiple paths. However, in this chapter, we especially focus on the case of shortest path routing in the IP environment to reveal the basic performance.

packets at a router based on betweenness centrality and content popularity. Our proposed policy is similar to "less cache" approach [15] to avoid duplicated contents from redundantly cached in a network. With our proposal, popular contents are stably cached at the edge of the network and unpopular contents are not likely to be cached in the network. As a result, frequency of cache replacement decreases and cache dynamics of the network is stabilized. Accordingly cached contents in a network is effectively diversified. These effects totally improve forwarding performance of BC, and more in-network caches are utilized.

In Chapter 4, we carefully investigate whether MSC^{4}N can achieve resource pooling in CCN/NDN. First, we explain mechanisms of MPTCP and how MPTCP achieves resource pooling by controlling multipath window sizes. Next, we discuss similarity of window size control between MPTCP and our previously proposed MSC^{4}N, which shows a possibility of resource pooling of MSC^{4}N. MPTCP takes account of resource pooling concept for multipath between a single source and a single receiver. Resource pooling concept discussed in this chapter is a quite new fairness concept because multipath in our MSC^{4}N composes of paths from multiple sources to a single receiver.

In Chapter 5, we propose the method for efficient information retrieval and cache control scheme in disaster situations. The former method consists of two parts, the dynamic routing protocol (NDRP: Name-based Distance-vector Routing Protocol) and the packet forwarding method (ICOQ: Interest and ContentObject Queueing). The combination of these proposals adapts to the intermittent links and achieves energy and bandwidth efficient information retrieval in disaster situations. The latter cache control scheme (POP: Popularity-based Caching) enables routers to stably store popular contents in their storages and improves performance of information retrieval.

From the viewpoint of scalability, reactive routing is another way to reduce such wasteful resource usage. With a reactive approach, popular route information is widely spread and unpopular route information is rarely distributed, which means effective distribution of route information is realized. In Chapter 6, we focus on popularity of route information and introduce this concept into routing methods for ICN-based function chaining. The proposal consists of two elements, RR (Route Records) and OR^3 (On-demand Routing for Responsive Route). RR is forwarding table to reactively manage routing information. OR^3 is a method for discovering the best route which is to be cached in RR. The combination of RR and OR^3 realizes scalable routing for ICN-based function chaining.

Finally, Chapter 7 concludes this thesis and describes the future works.

Chapter 2

Combination Effects of Routing and Caching in Information Centric Networks

In ICN, routers in a network are generally equipped with local cache storages and store incoming contents temporarily. Efficient utilization of total cache storage in networks is one of the most important technical issues in ICN, as it can reduce content server load, content download latency and network traffic. Performance of networked cache is reported to strongly depend on both cache decision policy and content request routing policy [48]. In this chapter, we evaluate several combinations of these two policies. Especially for routing, we take up off-path cache routing, Breadcrumbs, as one of the content request routing proposals. Our performance evaluation results show that off-path cache routing, Breadcrumbs, suffers low performance with cache decision policies which generally has high performance with shortest path routing (SPR), and obtains excellent performance with TERC which is well-known cache decision policy to have low performance with widely used SPR. Our detailed evaluation results in two network environments, emerging ICN and

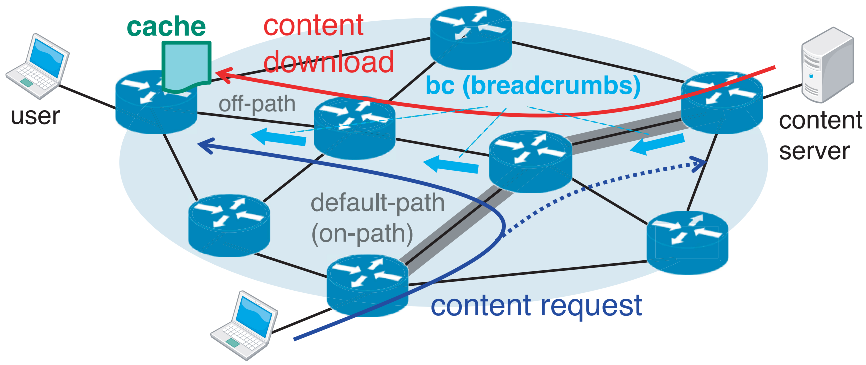


Figure 2.1: The behavior of Breadcrumbs

conventional IP, show these insights hold in both of these two network environments.

2.1 Breadcrumbs (BC)

In ICN, BC [47] has been proposed as a routing method for effective off-path cache utilization. Figure 2.1 shows a behavior of the BC method. During a data packet download, each router stores the data packet and sets pointers called breadcrumbs (bc) that indicate the download direction. When a request packet accidentally arrives at a router holding a breadcrumbs corresponding to the request packet, *BC forwarding* is started, and the request packet is forwarded towards the bc's direction in a hop-by-hop manner. By using bc effectively, off-path caches can be effectively utilized.

Request packets guided by bc may not be able to find cached content. This guidance failure occurs when cached content in CS of all routers along the bc-trail have been evicted (replaced). In order to avoid such redundant guidance, the BC method uses the following procedures.

- Time Out

  To reduce redundantly remaining bc, each bc has lifetime and is controlled by timer. If there are no accesses to bc during lifetime, the bc is removed by time-out. A router records the latest time-stamp of a request packet T_q and a data packet T_f on the corresponding bc. The time interval of time-out, i.e. lifetime of the bc is L_q for the request packet and L_f for the data packet. When the current time is later than $T_q + L_q$ and $T_f + L_f$, the bc is removed from the router.

- Trail Invalidation

  When a request packet directed by bc cannot find the cached content on routers along the bc-trail, trail invalidation occurs and the bc-trail is removed. Trail invalidation also occurs when a request packet arrives from an interface to which the corresponding bc is directed. In this case, bc pointer is directed to a router which has no corresponding data packet. Trail invalidation is executed by sending back a request packet along the reverse bc-trail (bc also has a reverse pointer [47]). When this returned request packet arrives at the router which initiated BC-forwarding, i.e. when a bc-trail is completely removed, the request packet is forwarded to a content server by FIB-based forwarding. This means after bc-trail invalidation, content request forwarding is switched back to FIB-based forwarding.

BC has been originally proposed for IP networks, and its partial deployment in IP networks has been studied [40]. On the other hand, BC can be also applied to CCN/NDN. Figure 2.2 shows the structure of a BC-enabled CCN router proposed in [39]. In this modification, BC table that records information of pointers (breadcrumbs) is inserted in addition to default components of CCN routers, i.e. FIB, PIT, and CS. In Interest processing, table lookup of BC table is before that of FIB and after that of PIT. By inserting the BC table at this position, we can utilize both PIT aggregation and off-path cache probing. In

ContentObject processing, when a router receives a ContentObject, the router creates a new BC entry in the BC table and forwards the ContentObject with PIT. If there are multiple output interfaces in the PIT entry, one of them is randomly selected and added to the BC table.

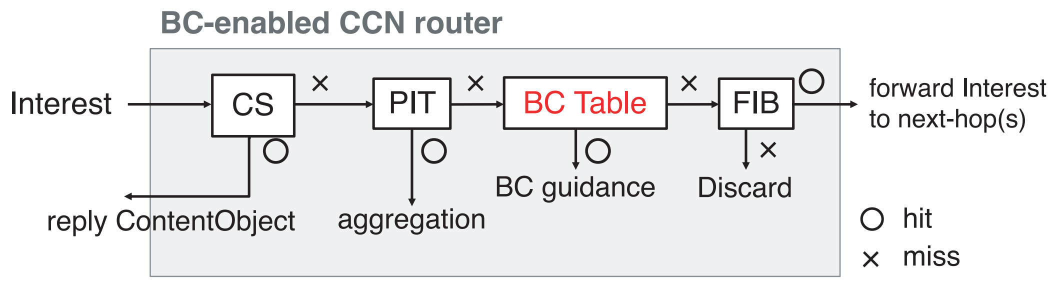


Figure 2.2: The structure of a BC-enabled CCN router

2.2 Cache Decision Policies

In this section, we explain some cache decision policies that are commonly used in CCN/NDN [58].

- TERC (Transparent En-Route Cache)

  In TERC [29], a data packet is always cached at all routers along a download path. This is also called LCE (Leave Copy Everywhere) or CEE (Cache Everything Everywhere) and used in the original CCN/NDN [28, 59]. This policy is the most popular cache decision policy used in many studies [19, 20, 21, 48, 47, 29, 22].

- Fix(p)

  Fix(p) [5, 32] is a simple cache decision policy that each router on a download path stores incoming data packets uniformly at random with a fixed probability

p ($0 \leq p \leq 1$). Rate of caching is reduced according to decrease of the parameter p, and also frequency of cache replacement decreases. As a result, relatively-popular contents tend to be stably cached in the network. As a special case, Fix($p = 1$) is identical to TERC.

- LCD (Leave Copy Down)

  In LCD[32, 33], a data packet is cached at the one-hop downward router from a cache hit router. Figure 2.3 shows an example of LCD behavior. The user downloads data packets from the server. In the first download, the cache is only generated in the router that is one-hop downward of the server. In the second download, the cache is copied into the next downward router. The same procedure is performed thereafter. Since this behavior delivers popular contents to one-hop closer to each generated Interest, this method is classified to popularity-based policy [58].

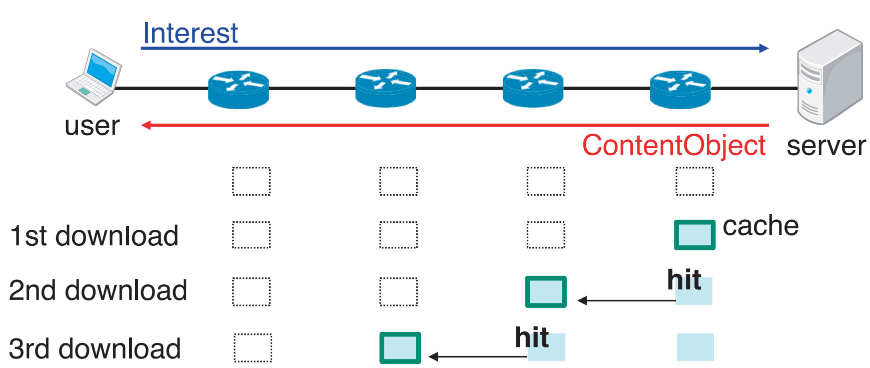


Figure 2.3: Example of LCD

- ProbCache

  ProbCache [44] is a probabilistic cache decision policy based on estimated cache capacity of routers on download path and hop count of data packets.

Each router x (the x_{th} router from content server) on a download path caches a data packet based on the following probability,

$$ProbCache(x) = \frac{\sum_{i=1}^{c-(x-1)} N_i}{T_{tw} \cdot N_x} \cdot \frac{x}{c}. \tag{2.1}$$

N_i is the size of cache memory of the i_{th} router from the content server on a download path. c is the total path length from the server to the user in terms of data hop count. With this probabilistic approach, ProbCache suppresses redundant caching and leaves caching space for other flows sharing a same path. Also, caching probability increases with getting close to users, and popular contents tend to stay near the users.

The original CCN/NDN [28] utilizes the combination <SPR, TERC>. When Fix(p), LCD or ProbCache is used for cache decision policy, i.e. <SPR, Fix(p)>, <SPR, LCD>, or <SPR, ProbCache>, cache hit performance increases compared to <SPR, TERC> [48]. Fix(p), LCD or ProbCache is frequently used as the effective cache decision policies in CCN/NDN. However, there are no confirmation that these cache decisions have high performance when they are combined with BC. In this chapter, we evaluate cache performance for the combinations of $\mathcal{F}$={SPR or BC} and $\mathcal{D}$={TERC, Fix(p), LCD or ProbCache} both in CCN and IP, and find out which cache decision policy is suitable for BC.

2.3 Performance Evaluation

2.3.1 Simulation model

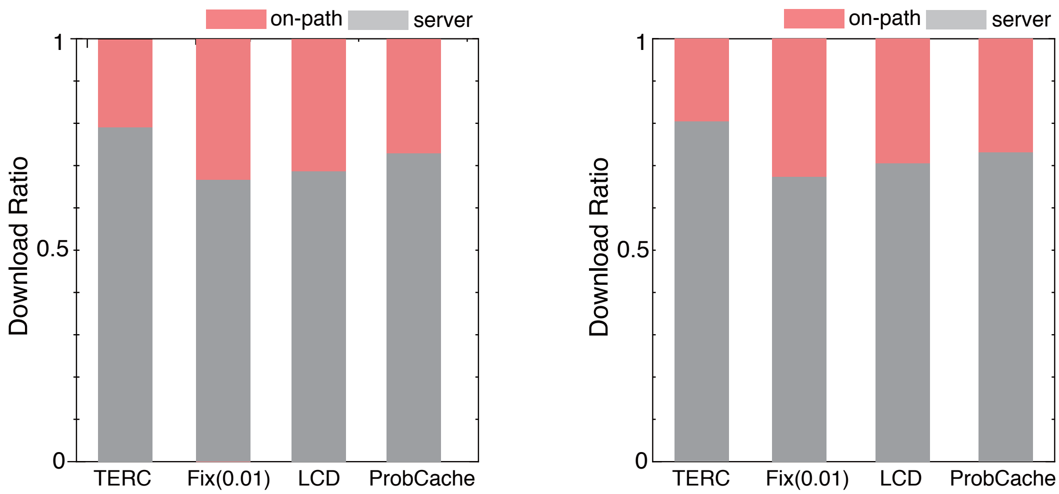
In this section, we evaluate cache hit performance of 8 combinations of the following policies, $\mathcal{F}$={SPR, BC}, $\mathcal{D}$={TERC, Fix(p), LCD, ProbCache}. The parameter p in

Fix(p) is set to 0.01 throughout our evaluation because this setting is reported to have better performance [48]. Even though BC is originally proposed for cache-enabled IP networks [47], it assumes two download path policies, DFQ and DFSP. DFQ and DFSP are corresponding to the download policies of emerging CCN/NDN and the current IP network, respectively. In this sense, we would like to evaluate the above 8 combinations in these two network environments, CCN and IP. We utilize ndnSIM1.0 [66] for CCN/NDN, and a self-made simulator used in [40] for IP.

The simulation topology is the Barabasi-Albert (BA) model [2] created by BRITE [60]. This model is a scale-free network topology that has power-law degree distribution. The number of routers is 100. Each router stores 5 contents in its cache storage and is connected to one user. All communication links are bi-directional. Request generation rate of each user is 1 packet per second. The number of servers is 5 and each server is connected to a randomly selected router. The number of contents is 10000. Each content is stored at a randomly selected server. Popularity of the contents follows Zipf distribution ($\alpha = 1.0$) [10]. We use LRU (Least Recently Used) for cache replacement policy ($\mathcal{R}$) for all combinations of forwarding ($\mathcal{F}$) and cache decision ($\mathcal{D}$).

2.3.2 Cache hit characteristics of SPR

Figure 2.4 shows content download ratio for each cache decision policy in SPR. Figure 2.4(a) shows performance of CCN and Fig. 2.4(b) shows that of IP. Cache hit ratio of <SPR, Fix(0.01)> is about 33%, and this combination has the highest performance in both network environments. The combination <SPR, TERC> has the lowest cache hit ratio ($\simeq$ 20%). When we use SPR for request routing policy, data transmission path of CCN and that of IP are the same because transmission paths of requests and data follow the shortest path both in CCN and IP. Cache hit characteristics would show the same results, and the slight differences between CCN and IP are caused by fluctuation of simulations.



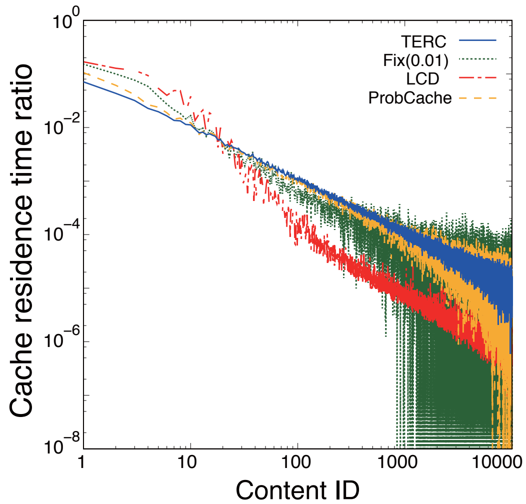
(a) CCN. (b) IP.

Figure 2.4: Download ratio in SPR.

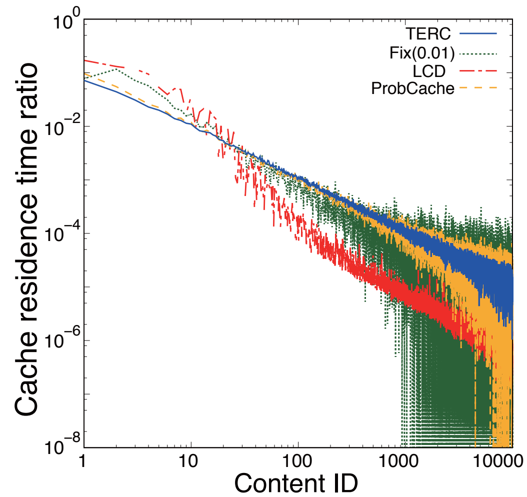
Figure 2.5 shows *cache residence time ratio* characteristics (cache distribution) with SPR in CCN and IP. Cache residence time ratio for content k is defined as follows,

$$\frac{\sum_{i \in \mathcal{V}} t_i^k}{\sum_{i \in \mathcal{V}} \sum_{k \in \mathcal{K}} t_i^k}.$$

$\mathcal{V}$ is the set of router in the network, and $\mathcal{K}$ denotes the set of contents. t_i^k is cache residence time for content k in node i. The horizontal axis shows content IDs that are sorted in descending order of content popularity. We call contents with IDs from 1 to 10 as *high-popular* contents, contents whose IDs are from 11 to 100 as *middle-popular* contents, and the others as *low-popular* contents. Figure 2.5 shows Fix(p), LCD or ProbCache tends to cache high-popular contents. Since TERC always caches all transferred content files regardless of their popularity, this policy tends to cache relatively unpopular contents compared to other two decision policies. This characteristic is observed in both network environments, CCN and IP. The reason is that content download follows similar path in CCN and IP. In CCN, download path of a content file is the reverse path of request packets.



(a) CCN.



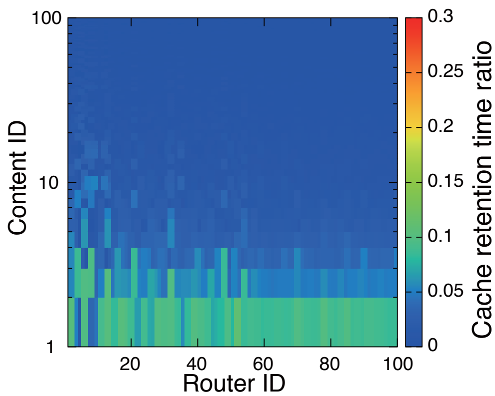
(b) IP.

Figure 2.5: Cache residence time ratio with SPR.

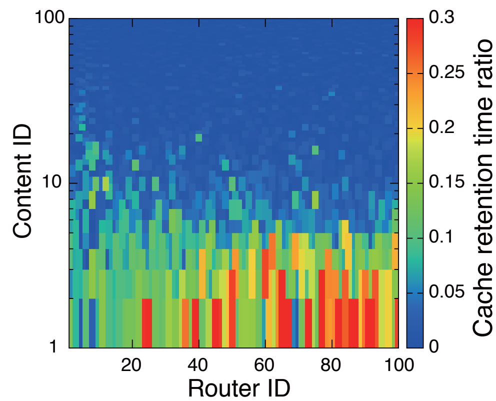
In other words, the content file is reversely transferred along the transmission path of the request packets (the shortest path from a user to a server). Also in IP network, download path is the shortest path from a server to a user. Therefore, when SPR is used for request forwarding policy, users download content files with almost the same shortest paths in both network environments.

Figure 2.6[1] reveals spatial aspect of cache distribution with SPR at each router. The horizontal axis shows router ID that are sorted in descending order of betweenness centrality. Routers which have small IDs are located at core area of the network, and routers which have large IDs are located at edge area. The colors of right-bar on each figure show *cache retention time ratio* for content k in node i, defined as $t_i^k / \sum_{k \in \mathcal{K}} t_i^k$. With TERC, unpopular contents tend to replace popular contents, and the popular contents are not stably retained to the edge of the network as in Fig. 2.6(a). With Fix(0.01), popular contents are randomly cached to specific locations. Once a popular content is stored to a specific node, the cached

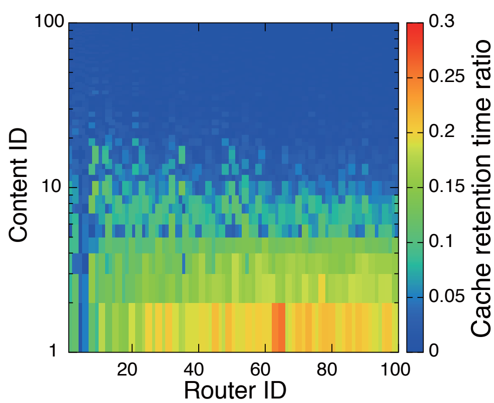
[1]Due to space limitations, Figs. 2.7, 2.6, 2.9, 2.10, and 2.11 show performance only for CCN because there were no large differences between CCN and IP in our evaluation.



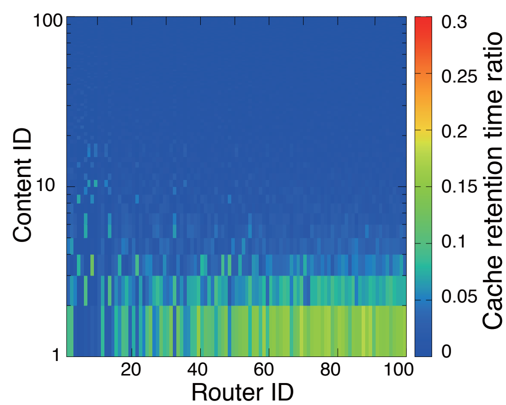
(a) TERC.



(b) Fix(0.01).



(c) LCD.



(d) ProbCache.

Figure 2.6: Spatial cache distribution in SPR (CCN).

content is frequently reused by other users and stably stays there. Thus, popular contents are sparsely stored in the network as in Fig. 2.6(b). With LCD, popular contents move to the edge of the network by repeated content downloads by users, and the popular contents are stably exist in the edge close to the users as shown in Fig. 2.6(c). ProbCache also moves popular contents to the edge of the network by its popularity-based caching as shown in Fig. 2.6(d). Although there are slight differences compared with LCD, popular contents tend to be stably located close to the users.

Figure 2.7† shows cache hit ratio for every content ID. As shown in this figure, <SPR,

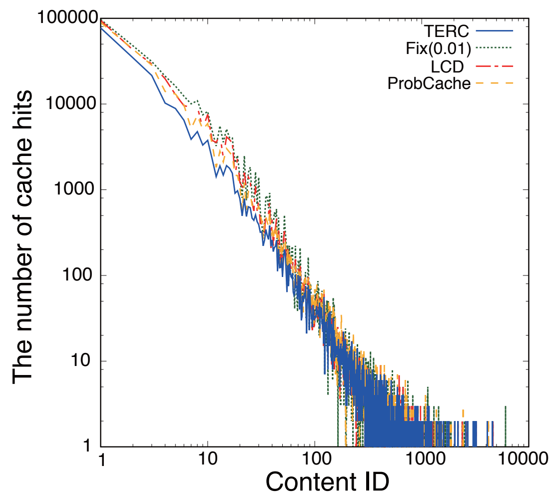
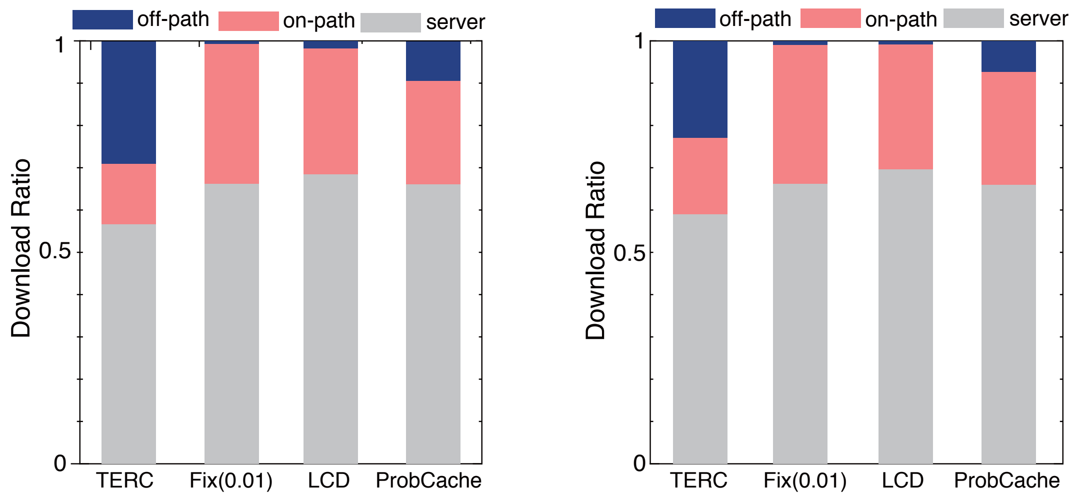


Figure 2.7: The number of cache hits with SPR (CCN)

Fix(p)/LCD/ProbCache> increases the number of cache hits for high-popular contents compared to TERC. The total number is <SPR, Fix(p)> = 330,942, <SPR, LCD> = 310,749, <SPR, ProbCache> = 268,375, and <SPR, TERC> = 206,730. Fix(p), LCD, or ProbCache increases cache hit performance for high-popular contents and effectively utilizes cached contents in the network. With $\mathcal{F}$=SPR, cache utilization is limited to only the on-path and search of cached content file by request forwarding is restricted. It is quite a natural results that cache decision policy which tends to store high-popular contents, such as Fix(p), LCD, and ProbCache, shows high performance.

2.3.3 Cache hit characteristics of BC

In this section, we evaluate cache hit performance for a content request forwarding policy BC. Figure 2.8 shows content download ratio for each cache decision policy in BC. As is the case for SPR, we evaluated the performance in two network environments, CCN and IP. As shown in Fig. 2.8(a), <BC, TERC> has about 43% cache hit ratio and the highest performance, although TERC has the lowest performance with SPR. The other



(a) CCN.

(b) IP.

Figure 2.8: Download ratio in BC.

cache decision policies that has high performance with SPR are not effective with BC. These characteristics are observed both in CCN and IP.

Firstly, we investigate the reason why the combination <BC, TERC> has excellent performance. Table 2.1 shows the average *invalidation ratio* of BC forwarding for each combination. The definition of invalidation ratio (failure rate of BC forwarding) is defined as follows,

$$\frac{\text{The number of failed BC forwarding}}{\text{The total number of BC forwarding}}. \tag{2.2}$$

<BC, Fix(p)> has high invalidation ratio both in CCN and IP. With Fix(p), downloaded contents are stored in routers on download paths according to the parameter p. When the value of p is less than 1, there is a possibility that no contents are cached on any router on a download path. To confirm this point, we change the value of parameter p and evaluate invalidation characteristics. Table 2.2 shows invalidation ratio (failure rate of BC forwarding) for the parameter p of Fix(p). These results show that invalidation ratio has the larger value with the smaller value of p. When p is extremely small e.g. $p = 0.001$,

no contents are cached on any router during download with high probability. As a result, invalidation ratio becomes significantly large value e.g. 0.998, thus the parameter p is dominant performance factor of trail invalidation in BC. The trail invalidation frequently occurs both in CCN and IP even when the value is 0.1, thus Fix (p) is not suitable for BC except for the special case of Fix(p = 1), which is identical to TERC.

<BC, ProbCache> also has high invalidation ratio because it is probability-based caching as well as Fix(p). With probabilistic approach, as previously described, there is a possibility that no contents are copied onto any routers during download. In this case, BC does not perform well, invalidation ratio increases, and BC forwarding performance is significantly degraded. Thus, ProbCache is not suitable for BC.

Table 2.1: Invalidation ratio for each combination.

	TERC	Fix (0.01)	LCD	ProbCache
CCN	0.337	0.979	0.297	0.775
IP	0.358	0.964	0.125	0.764

Table 2.2: Invalidation ratio of Fix(p).

p	0.1	0.01	0.001
CCN	0.778	0.964	0.986
IP	0.828	0.979	0.998

Table 2.3 shows the number of BC forwarding. In <BC, LCD>, there is fewer BC forwarding compared to other two policies, Fix(p) and TERC. In LCD, when cache hit occurs at a router, the content file is copied at the one-hop downward router of the cache hit router. At the same time, bc pointer is also generated at the cache hit router and the bc indicates the one-hop downward router. In other words, only one bc pointer is created with each content download with LCD. The number of BC forwarding considerably decreases

as shown in Table 2.3. For these reasons, the cache decision policies Fix(p), LCD and ProbCache are not suitable for BC.

Table 2.3: The number of BC forwarding.

	TERC	Fix (0.01)	LCD	ProbCache
CCN	415,958	278,132	26,076	335,867
IP	359,993	249,819	31,099	308,172

Figure 2.9[†] shows cache residence time ratio in BC[†2]. As described previously, cache residence time ratio of TERC shows long-tail distribution compared to other cache decision policies. This means unpopular contents can be stored comparatively more frequently in the networks. In other words, TERC's cache distribution has "diversity" and middle-popular contents are also likely to be stored in the network. As a result, the number of cache hits for the middle-popular contents increases and <BC, TERC> totally has the excellent performance as shown in Fig. 2.10[†].

Figure 2.11[†] classifies cache hits according to cache hit location. Figure 2.11(a) shows on-path cache hit ratio and Fig. 2.11(b) shows off-path one. With respect to on-path cache hit, Fix(p), LCD and ProbCache have high performance, because these policies store high-popular contents in the network. These policies, however, cannot utilize bc effectively and have low performance in off-paths. TERC comparatively has the lowest cache hit performance in on-paths, however, this policy gains significant cache hits in off-paths. The improvement especially for middle-popular contents is observed as shown in Fig. 2.11(b). With TERC, the number of BC forwarding is large and invalidation ratio is low as in Table 2.3 and Table 2.1, respectively. This is because cache distribution in the network has diversity compared to other two decision policies as in Fig. 2.9. BC can find the diverse off-path caches by in-network guidance and retrieve them. Consequently, <BC, TERC> is

[2†]There is no large difference between BC and SPR with respect to spatial cache distribution as described in Fig. 2.6. Thus, we would like to omit the results due to space limitations.

the most efficient combination in terms of total cache hit performance both in two network environments, CCN and IP.

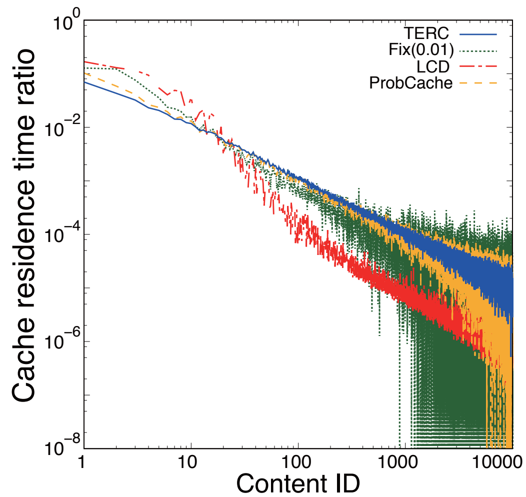


Figure 2.9: Cache residence time ratio with BC (CCN).

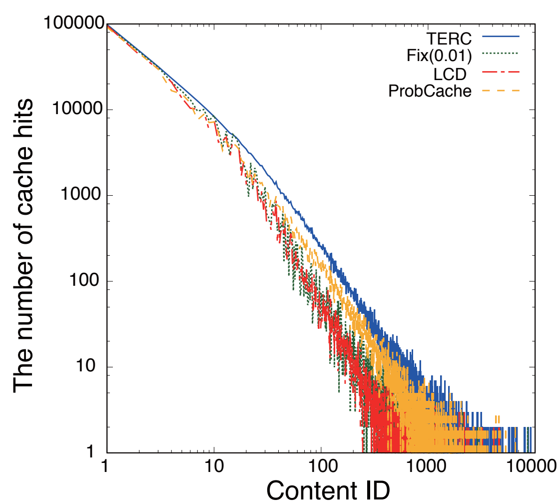
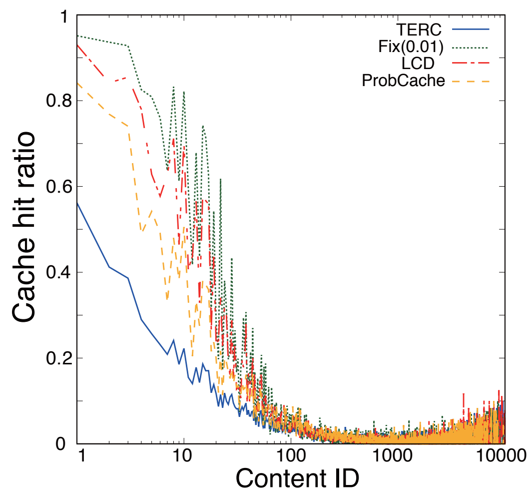
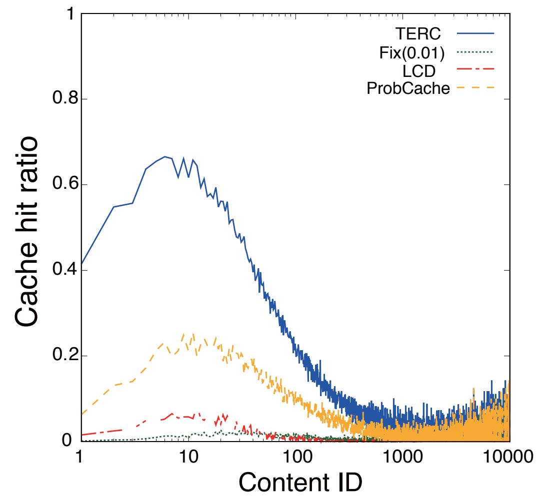


Figure 2.10: The number of cache hits with BC (CCN).



(a) On-path.



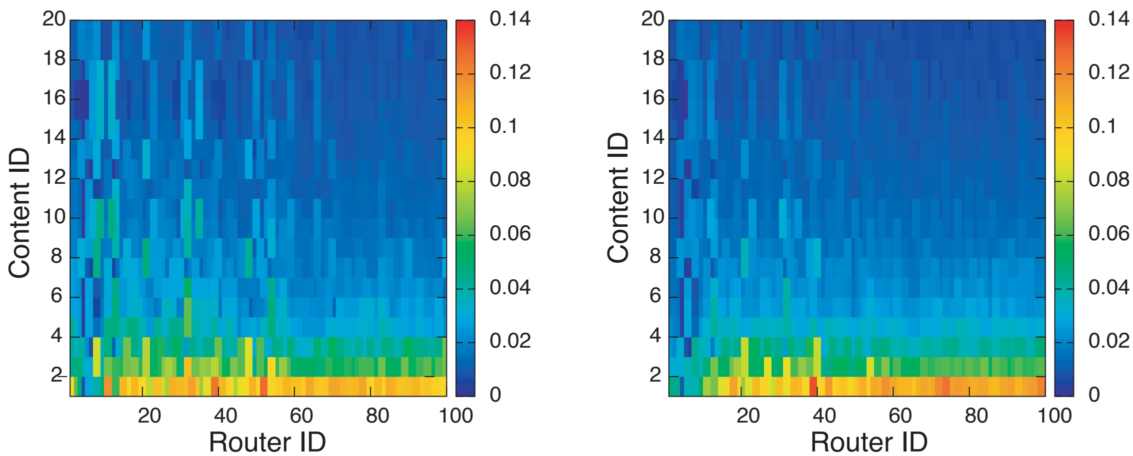
(b) Off-path.

Figure 2.11: Classification of cache hit (CCN).

2.3.4 CCN vs. IP

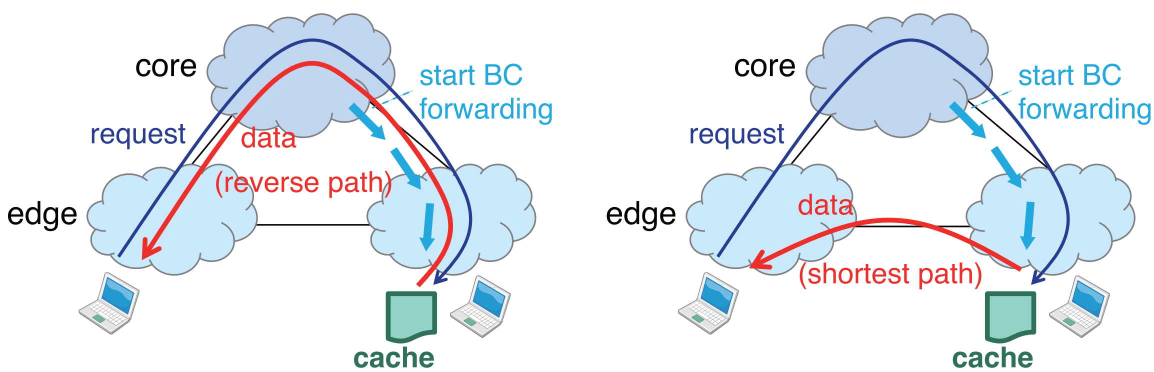
In this section, we investigate the reason for a difference between Figs. 2.8(a) and (b). In these figures, on-path/off-path cache hit ratio between CCN and IP slightly ($\simeq$ 5%) differs for the combination <BC, TERC>. CCN's off-path cache hit by BC forwarding is higher than IP's one. Figure 2.12 shows the normalized number of data transmissions, i.e. spatial distribution of data traffic. In IP, data transmission of popular contents relatively concentrate on the edge of the network as shown in Fig. 2.12(b). This means data traffic transferred in the core area is filtered (reduced) by cache hit around the edge. In CCN, popular contents tend to be transmitted not only in the edge area but also the core area when compared to IP as shown in Fig. 2.12(a).

Figure 2.13 briefly explains the difference of data transmission path between CCN and IP. With CCN, data packets are transferred on the reverse path of request packets, thus, the data packets are likely to be routed to the core of the network and replace popular cached contents frequently as shown in Fig. 2.13(a). In contrast, IP's data transmission path is the



(a) CCN. (b) IP.

Figure 2.12: Spatial distribution of data traffic.

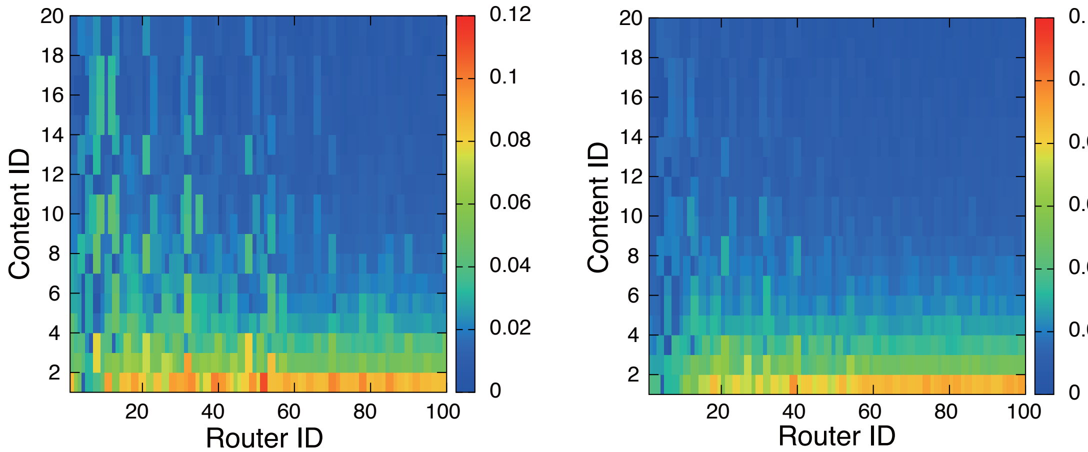


(a) CCN. (b) IP.

Figure 2.13: Difference of data transmission originated in BC forwarding between CCN and IP.

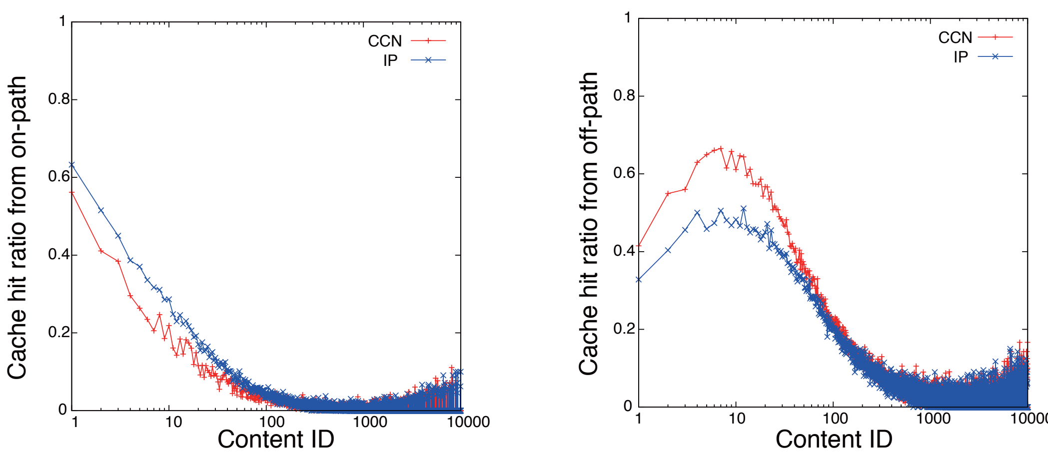
shortest path and content download can be completed around edges even when a request packet has been passed through the core of the network as shown in Fig. 2.13(b). Hence, IP's download traffic tends not to be transferred in the core of the network as compared with CCN, and popular contents are stably located at the edges as shown in Fig. 2.14

Figure 2.15 shows a comparison of on-path/off-path cache hit ratio of CCN and IP. As shown in Fig. 2.15(a), IP's on-path cache hit ratio is higher than CCN especially for



(a) CCN. (b) IP.

Figure 2.14: Spatial cache distribution.



(a) On-path. (b) Off-path.

Figure 2.15: Difference of cache hit ratio between CCN and IP.

popular contents whose content IDs are 1-100. This is because the popular contents are stably cached in the network in IP, and cache hit occurs especially around the edge of the network close to users. In CCN, content download passes through the core of the network, and cached popular contents can be frequently replaced by other content downloads as compared with IP. Even if cache hit has not occurred in on-path, users can successfully

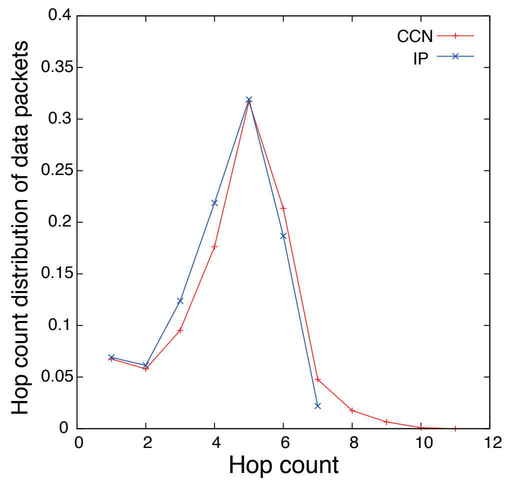


Figure 2.16: Hop count distribution of data packets in CCN and IP.

download contents from off-path cache as shown in Fig. 2.15(b). This feature is one of benefits of the BC method, but sometimes has a negative effect on network performance.

Figure 2.16 shows hop count distribution of data packets. The maximum hop count of IP, which is equal to network diameter, is 7 because IP's download path of data packets is the shortest path. The maximum hop count of CCN is 11 and increased by 57%. For average performance, the average data hop count of CCN is 4.58, and that of IP is 4.31. IP's download path is the shortest path and relatively shorter than CCN's one. CCN's download path is the reverse path of Interest packets and tends to be routed to the core of the network. As a result, traffic load increases especially in the core area in the case of CCN. IP can avoid such traffic concentration by shortest path routing of data traffic and filtering effect, i.e. stably-cached popular contents around edges. Traffic load of IP is totally reduced by ≃8% compared with CCN in our evaluation. (Definition of traffic load is the number of transmitted data packets in the network.) Therefore, IP has better performance compared to CCN in terms of traffic load by mitigating traffic concentration in the core area.

2.4 Summary

In this chapter, we evaluated the performance of an in-network guidance method, Breadcrumbs, with several cache decision policies in comparisons with shortest path routing. When shortest path routing is adopted as the forwarding policy, cache decision policies, Fix(p), LCD, and ProbCache that distribute high-popular contents in a whole network show high performance because content search for cached contents is limited to the on-path. However, when we use Breadcrumbs, the range of cache discovery is expanded to off-path caches. In TERC, not only high popular contents but also middle and low popular contents are also cached. TERC makes effective bc-trail because all routers on this trail store the corresponding content. Therefore, in this chapter we clearly reveal that combination of BC and TERC brings effective usage of off-path cached contents, which has not been deeply investigated thus far. These characteristics were observed both in two network environments, i.e. emerging CCN/NDN and conventional IP networks.

In the next chapter, we explain a problem of TERC combined with BC and propose a cache decision policy suitable for BC.

Chapter 3

Effective Caching Policy for Breadcrumbs in Content-Centric Networking

As described in the previous chapter, the combination of BC and TERC is effective because this combination brings diversity of cached contents. However, TERC is also called CEE (Cache Everything Everywhere), and each intermediate router on a download path always caches incoming data packets. This cache decision, TERC, is well known to make cached contents in a network highly dynamic. Frequent replacement of cached contents degrades guidance performance of BC due to increase of guidance failures by BC.

In this chapter, we propose a cache decision policy based on betweenness centrality and content popularity, which is suitable for Breadcrumbs. Our proposed cache decision policy makes popular contents located at edge area of a network, which stabilizes popular cached contents. Moderate popular contents tend to be stored in core area, which induces in-network guided requests to encounter cached contents more frequently. Our performance evaluation results reveal that the combination of Breadcrumbs and our proposed cache decision policy improves cache hit performance compared to existing cache decision policies.

3.1 Less Cache Policy based on Betweenness Centrality

Reducing frequency of cache replacement is effective way to increase cache hit performance in ICN. W. Chai, et al. [15] have proposed a centrality-based cache decision policy to decrease frequency of in-network cache replacement. Betweenness centrality [9] is defined as the following equation.

$$c_v^B = \sum_{i \neq v \neq j \in \mathcal{V}} \frac{\sigma_{i,j,v}}{\sigma_{i,j}}, \ \forall v \in \mathcal{V}, \tag{3.1}$$

where $\sigma_{i,j}$ is the number of content delivery paths from user node i to content server node j, and $\sigma_{i,j,v}$ is the number of content delivery paths from node i to node j that pass through node v. The content delivery path means the shortest path in this work. These values $\{c_v^B\}$ are configured on each node $v \in \mathcal{V}$ and used for cache decision making.

When a user transmits a request packet, the maximum value of betweenness centrality among nodes $\mathcal{V}_{\mathcal{SP}}$ on the shortest path $\mathcal{SP}$, i.e., $\max_{\forall v \in \mathcal{V}_{\mathcal{SP}}} \{c_v^B\}$ is copied onto the request packet. When the request packet finds a corresponding data packet, the value recorded on the request packet is also copied onto the corresponding data packet. During the data packet transmission, each node compares the value of betweenness centrality recorded on the data packet with its own one. If the recorded value matches a node's one, the node decides to cache the incoming data packet. Otherwise, the node does not cache the data packet. In such a way, cache replacement in a network is reduced and high-centrality nodes tend to possess popular contents because popular contents are frequently downloaded by users. These cached contents are also reused by the users' subsequent requests. This behavior promotes the network to possess popular contents, and cache hit performance is improved compared to existing cache decision policies. However, this proposal assumes to use SPR for forwarding strategy and does not take account of other forwarding strategies. Also, diversity of cached contents is low because only popular contents tend to be cached in the network.

3.2 Cache Decision Policy based on Betweenness Centrality and Content Popularity

In this section, we propose a cache decision policy based on betweenness centrality and content popularity. Betweenness centrality is used for suppressing redundant cached contents in a network and placing popular contents to the edge of the network. Cache hits for the popular contents are frequently caused by the edge routers close to users, and content downloads are likely to be completed around the edge area. Cache replacement frequency is reduced in the core area, and in-network caches are stabilized. As a result, BC can successfully find these stabilized caches, and BC forwarding performs well. This is the first purpose of our proposal. Secondly, by taking content popularity into account, the cached contents are diversified. Diversity of the cached contents enables the BC method to retrieve various contents from in-network caches.

Eq. (3.2) is *decision factor* for content k at node v, which is used for our proposed cache decision.

$$df_v^k = \frac{1}{c_v^B} \cdot p_k, \tag{3.2}$$

where c_v^B is betweenness centrality of node v defined in Eq. (3.1), and p_k is popularity of content k. In this chapter, we assume that distribution of content popularity follows the Zipf distribution [10], i.e. $p_k = \frac{1/k^\alpha}{\sum_{l=1}^{L} 1/l^\alpha}$, where the total number of contents is L. Popularity factor α is set to 1.0. Generally popularity of contents is not easily obtained at each router. Measurement of traversing contents [6] or some authorized measurement site offering popularity of contents enables each router to obtain these information. In our definition of the decision factor, the inverse number of betweenness centrality is multiplied by content popularity. This decision factor lets popular contents tend to be cached at the edge nodes, which have low centrality. Unpopular contents are not likely to be cached. This encourages reduction of redundant cached contents and cache replacement frequency. A downloaded data packet is just stored in the router(s) which gives the maximum along the shortest path from an end user to a content server.

Algorithm 1 shows cache decision algorithm of our proposal. This algorithm achieves our two goals described above. In the initial state, betweenness centrality $\{c_v^B\}$ and content popularity $\{p_k\}$ are given at each node, and the value of $df_{interest}^k$ is initialized. When node v receives an Interest packet for content k, node v calculates decision factor with Eq. 3.2. Also, node v looks up its Content Store CS_v, and compares df_v^k with the minimum decision factor of cached contents, i.e. $\min_{\forall k_n \in \mathcal{K}_{CS_v}} \{df_{data}^{k_n}\}$. $\mathcal{K}_{CS_v}$ is the set of cached contents in CS_v. If df_v^k is larger than the minimum decision factor, the node v updates $df_{interest}^k$ stored in the Interest packet. When an Interest finds the corresponding ContentObject packet for content k, the value stored in the Interest is copied onto the ContentObject packet as df_{data}^k. During transmission of the ContentObject packet k, each node v decides whether to store the transmitted ContentObject packet. If the value of df_v^k matches df_{data}^k, node v caches the ContentObject packet. Otherwise, node v does not cache the ContentObject packet and only forwards it.

This algorithm is based on the less cache policy of the existing approach as described in the section 3.1. Additionally, our proposal uses information of cached contents on a content delivery path by looking up them during Interest transmission. This behavior limits

Algorithm 1 *Centrality- and popularity-based caching.*

Input: $\{c_v^B\}, \{p_k\}, \ \forall v \in \mathcal{V}, \ \forall k \in \mathcal{K}$
Initialize: $df_{interest}^k \leftarrow 0$

<u>When node v received Interest for content k</u>
calculate df_v^k
look up cached data $\mathcal{K}_{CS_v}$ and get the minimum decision factor
if $df_v^k > \min_{\forall k_n \in \mathcal{K}_{CS_v}} \{df_{data}^{k_n}\} \ \wedge \ df_v^k > df_{interest}^k$ **then**
    $df_{interest}^k \leftarrow df_v^k$
else
    do nothing
end if

<u>When Interest k find a ContentObject packet</u>
$df_{data}^k \leftarrow df_{interest}^k$
send back the ContentObject

<u>When node v received ContentObject for content k</u>
if $df_v^k = df_{data}^k$ **then**
    cache ContentObject k and forward it
else
    do not cache ContentObject k and only forward it
end if

the range of caching nodes on the content delivery path, and each content is cached at appropriate location, i.e. highly popular contents are at the edge and moderately popular contents are at the core of a network. Extremely unpopular contents are not likely to be cached.

Figure 3.1 briefly shows a benefit of our proposed caching algorithm. The horizontal axis shows router locations of a content delivery path, a shortest path between an end user and a content server. v_1 is the one-hop upward router from the end user, and v_m is the one-hop downward router fro the content server. Generally, routers close to edge (end hosts) have low betweenness centrality and core routers, generally located around center of the content delivery path, have high betweenness centrality. The vertical axis shows content popularity. The curved line shows the minimum popularity of cached contents at

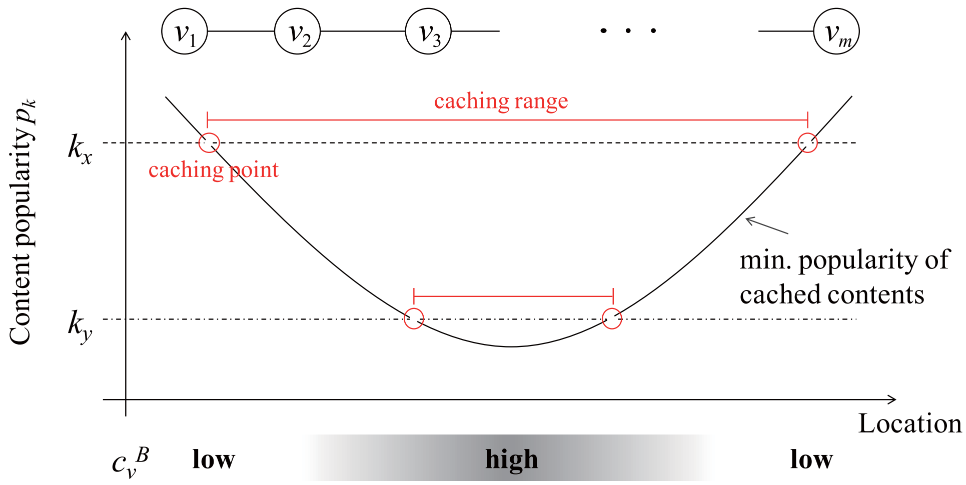


Figure 3.1: Benefit of our proposed caching algorithm.

each router when our proposal is applied. When a new request for content k_x is sent from v_1 to v_m, content popularity of k_x is compared at each router with the minimum popularity of cached contents. If popularity of k_x is lower than the minimum popularity of cached contents, this router is not selected as a candidate for caching node. Thus, routers within *caching range* depicted on content popularity line k_x in the figure, are candidates for caching node for this content. Along these candidates, the router(s) with the lowest betweenness centrality is selected as the router for caching this content in our algorithm. Thus, *caching points* denoted in the figure (two red circles on the content popularity line k_x) are selected as caching nodes. When popularity of content k_y, p_{k_y}, is lower than p_{k_x}, this caching range is narrower than that of k_x as shown in this figure. Therefore, low popular contents tend to be stored in core side of the network by our proposed algorithm. High popular contents are to be stored at edge side of the network.

3.3 Performance Evaluation

3.3.1 Evaluation Model

In this section, our proposed cache decision policy is evaluated. Throughout the simulations, we use BC and LRU for the forwarding strategy and the cache replacement policy, respectively. The following two cache decision policies are evaluated.

- TERC: The original policy of CCN/NDN [28, 59]
- Proposal: Our proposed policy

Simulation tool is ndnSIM 1.0 [66]. The network model for evaluation is the Barabási-Albert model [2], which is created with topology generator BRITE[60]. The number of routers is 100, and one user is connected to each router. Total number of contents is 10000. These contents are placed to randomly selected 5 content servers. Capacity of each router's CS is 5 packets. Popularity of users' content request follows the Zipf distribution ($\alpha = 1.0$) [10], and Interest generation rate at each user is 1.0 packet per second.

3.3.2 Effect on Cached Contents

Firstly, we investigate an effect of our proposal on cached contents. Figure 3.2 shows normalized cache residence time ratio of our proposal. Cache residence time ratio for content k is defined as follows,

$$\frac{\sum_{v \in \mathcal{V}} l_{v,k}}{\sum_{v \in \mathcal{V}} \sum_{k \in \mathcal{K}} l_{v,k}}, \tag{3.3}$$

where $l_{v,k}$ denotes cache lifetime for content k at node v. Cache lifetime is defined as time interval from generation time period of a stored content to its evicted time period. In this figure, cache residence time ratio of our proposal is normalized with results obtained for

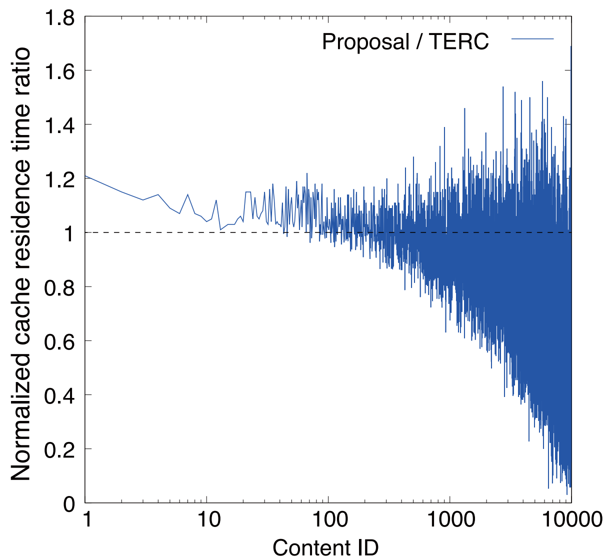
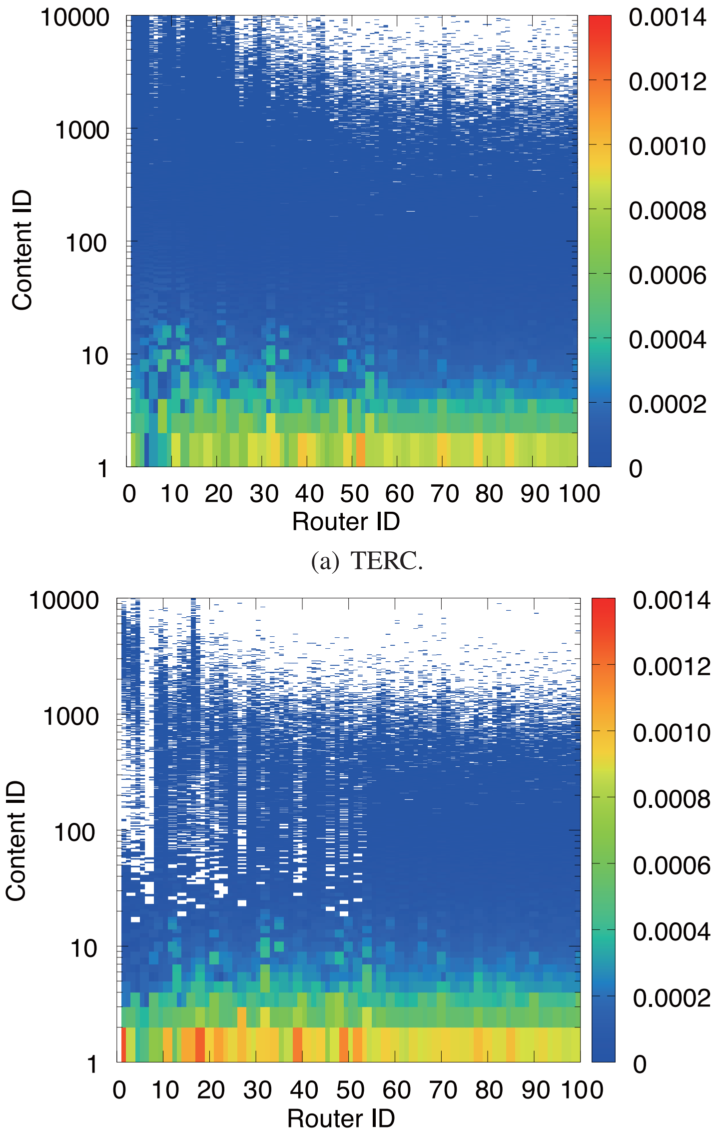


Figure 3.2: Normalized cache residence time ratio vs content ID.

TERC. The horizontal axis shows content IDs that are sorted in descending order of content popularity. In this chapter, we define the contents whose IDs are from 1 to 10 as *high-popular* contents, the contents whose IDs are from 11 to 1000 as *middle-popular* contents, and the others as *low-popular* contents. As shown in Fig. 3.2, high-popular contents are stably cached in the network, and low-popular contents are not likely to be cached with our proposal. In this figure, effect of our cache decision policy is briefly confirmed.

In Fig. 3.3, we focus on topological characteristics. This figure shows geographical distribution of cached contents. The x-axis denotes router IDs that are sorted in descending order of betweenness centrality. Routers that have small IDs have high-centrality. Cache residence time ratios for all contents are depicted as heat map. As shown in the figures, high-popular contents are more stably cached at the edge of network, i.e., the routers whose IDs are from 11 to 100. Our proposal enables the edge routers to cache high-popular contents. The middle- and low-popular contents are rarely cached especially at the core of the network, i.e., routers whose IDs are from 1 to 10. The white dots in the Fig. 3.3(b)

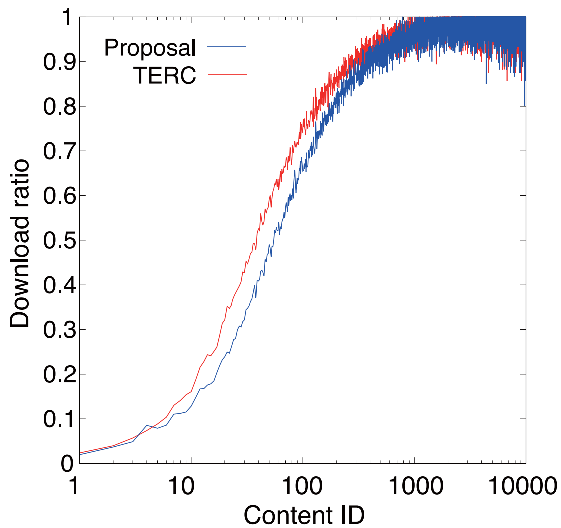


(a) TERC.

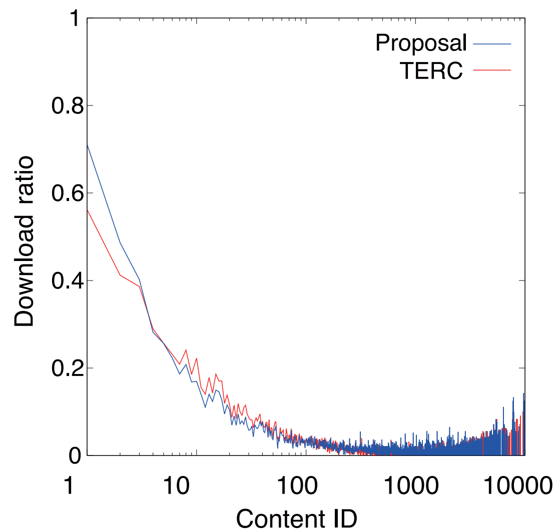
(b) Propsal.

Figure 3.3: Geographical distribution of cached contents.

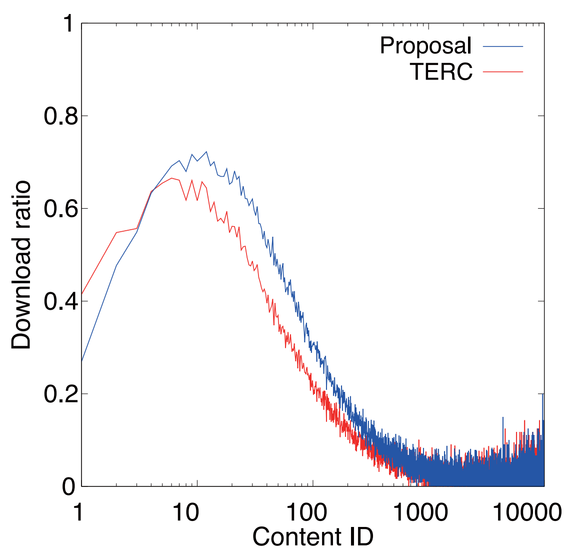
means that no caches are created in the network throughout the simulation. In our proposal, decision factor is defined as multiplication of the inverse number of betweenness centrality and content popularity. Therefore, the decision factor for middle- and low-popular contents is enormously low at high-centrality routers. In other words, these routers prepare a space



(a) Download from servers.



(b) Download from on-path caches.



(c) Download from off-path caches.

Figure 3.4: Classification of download ratio.

to cache for moderately popular contents.

Table 3.1: Content download ratio.

	server	on-path	off-path
TERC	0.567	0.143	0.289
Proposal	0.534	0.158	0.307

3.3.3 Cache Hit Performance

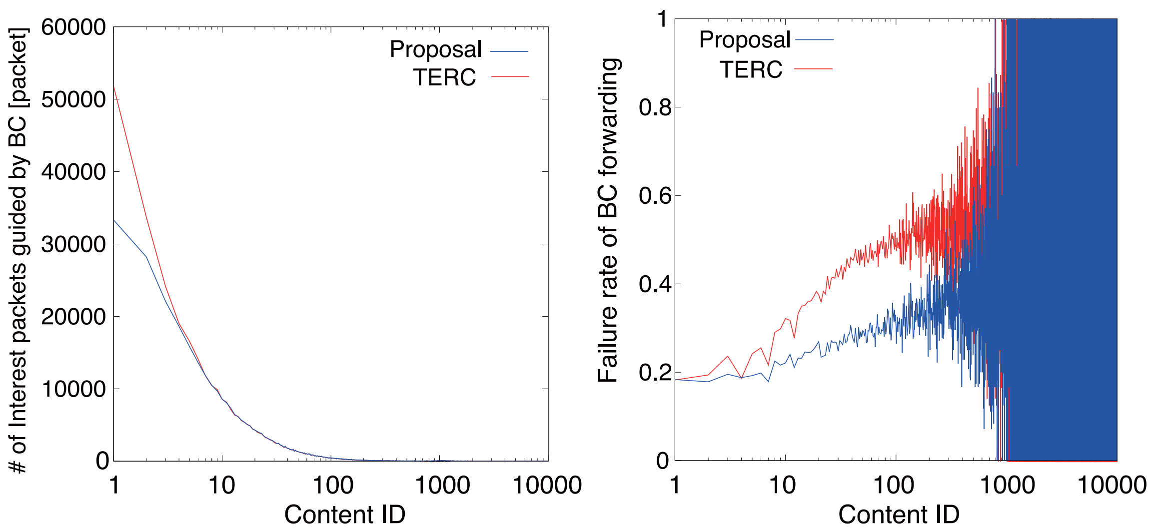
In this section, we evaluate cache hit performance. Table 3.1 shows download ratio for each method. Download ratio is defined as the ratios of the number of Interest packets forwarded to the servers, on-path caches, or off-path caches divided by the total number of transmitted Interest packets. Our proposal slightly improves cache hit performance for both on-path and off-path compared to TERC. In Fig. 3.4, we focus on content popularity of the downloaded contents. As shown in Fig. 3.4(a), our proposal reduces downloads from original content servers for all contents compared to TERC. There are two reasons for the performance improvement. The first reason is that increase of on-path cache hit for the high-popular contents. Our proposal encourages the edge routers to store the high-popular contents. As shown in Fig. 3.4(b), this situation promotes on-path cache hits and reduces download traffic for popular contents from the core of the network. Secondly, this situation also affects cache hit performance in off-path as in Fig. 3.4(c). Traffic demand for high-popular contents is satisfied and filtered by on-path caches around the edge routers, and traffic load of the high-popular contents are alleviated. Cached contents around the network core are stabilized and cache hit performance for the middle-popular contents is totally improved by BC forwarding. As a result, cache hit ratio for the middle-popular contents, whose IDs are 11 to 1000, especially increases as shown in Table 3.2.

Figure 3.5 shows forwarding performance of BC. In our proposal, the number of Interest packets guided by BC is less than TERC as in Fig. 3.5(a). As described in Fig. 3.5(b), however, failure rate of BC forwarding is much lower than TERC especially for the high-

Table 3.2: Cache hit ratio.

Content ID	1-10	11-1000	1001-10000
TERC	0.933	0.323	0.012
Proposal	0.942	0.386	0.015

and middle-popular contents. In our proposal, the high-popular contents are downloaded from on-path caches. Download traffic for the high-popular contents decreases and cache replacement frequency in the network is reduced. This makes in-network caches more stable. Figure 3.6 shows cache replacement characteristics. Our proposal significantly decreases the number of cache replacements for all contents by suppressing redundant cache generation. As a result, users successfully utilize stabilized in-network caches with BC forwarding.



(a) The number of guided Interest packets by BC.

(b) Failure rate of BC forwarding.

Figure 3.5: BC forwarding characteristics.

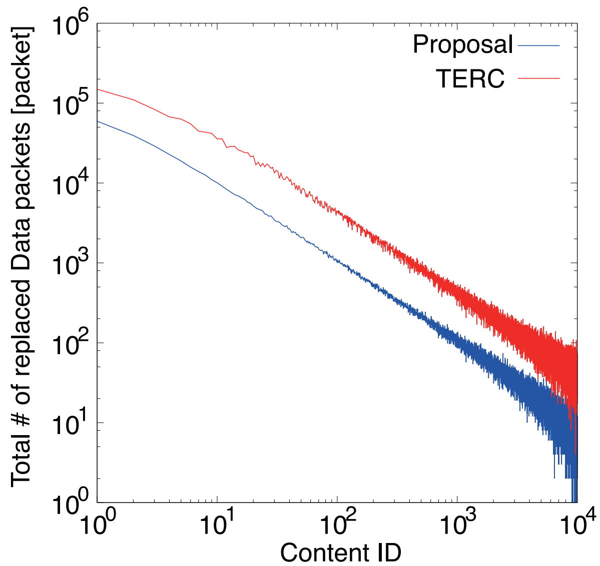


Figure 3.6: Cache replacement characteristics.

3.4 Summary

In this chapter, we proposed a cache decision policy suitable for BC in CCN/NDN. The policy controls location to be cached based on betweenness centrality on a content delivery path and content popularity of a requested content. Our proposal enables popular contents to be stably cached at the edge of the network. Unpopular contents are not likely to be cached in the network. As a result, frequency of cache replacement decreases and cache hit performance is improved due to the off-path cache guidance by BC forwarding.

Chapter 4

Resource Pooling in Multipath Congestion Control for Content-Centric Networking

Cache control schemes studied in Chapters 2 and 3 are essential for efficiently utilizing in-network cache and improving network performance. There is, however, another dominant factor of the network performance, *congestion*. When packet losses caused by congestion, the performance is significantly degraded. In order to handle this problem, congestion control is an important technique for mitigating congestion and preventing the packet losses in the network.

In information networks, many users concurrently utilize network resources. Resource Pooling concept has been proposed as fair sharing of total network resources among all users sharing a network so that a whole network is treated as a single pooled resource. MPTCP has been proposed as one of the most promising congestion controls which realize the resource pooling in the situation that each user can utilize multiple paths. In CCN/NDN, several multipath/multisource congestion control methods have been proposed. A fundamental difference between CCN congestion control and MPTCP where single source is assumed, is multisource and the resource pooling in multipath/multisource situation is quite a new research issue. In this chapter, we discuss congestion control for CCN from the viewpoint of the resource pooling and show that the previously proposed congestion control satisfies the resource pooling concept. Our simulation results also show that the proposed congestion control achieves macroscopic fairness among users, which means the resource pooling concept is satisfied.

4.1 Resource Pooling in Multipath Congestion Control

This section gives detailed explanation of existing multipath congestion control both in IP and CCN from the viewpoint of resource pooling.

4.1.1 Multipath Congestion Control in IP

MPTCP [46, 24], the most commonly used multipath congestion control in the current Internet, sets the following three design goals to realize resource pooling.

1. *Improve throughput*

   A multipath flow should perform at least as well as the single-path flow which would have obtained the best throughput. This ensures that a user has an incentive to deploy multipath.

2. *Do no harm*

   A multipath flow should not obtain any more capacity on each path than if it would have obtained in a single path case. This guarantees that multipath does not excessively harm other flows.

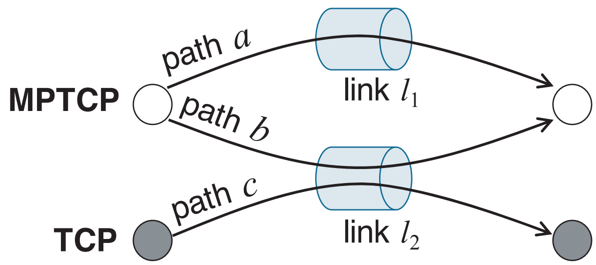


Figure 4.1: Example of Multipath TCP's design goals.

3. *Balance congestion*

   A multipath flow should move as much traffic as possible from its most-congested paths, subject to meeting the first two goals.

Figure 4.1 shows the case where MPTCP shares its path with TCP and we would like to briefly explain these three goals with this case. In this example, only subflow f_a routed on path a of a multipath flow (MPTCP) utilizes link l_1, and the other subflow f_b on path b shares the bottleneck link l_2 with singlepath TCP flow f_c^{TCP}. MPTCP's total throughput t_{total} is equal to $\sum_{f_i \in \mathcal{F}} t_{f_i}$ where t_{f_i} is throughput of subflow f_i and $\mathcal{F}$ is the set of subflow of MPTCP, i.e. f_a, f_b in this example. A singlepath TCP flow f_j^{TCP} on path j gains $t_{f_j}^{TCP}$. The first goal, *improve throughput* can be expressed by the following equation,

$$t_{total} \geq \max\ t_f^{TCP}, \quad \forall f \in \mathcal{F} = \{f_a, f_b\}. \tag{4.1}$$

This condition is an incentive of multipath utilization for users. The second goal, *do no harm* means,

$$t_f \leq t_f^{TCP}, \quad \forall f \in \mathcal{F} = \{f_a, f_b\}. \tag{4.2}$$

Even when MPTCP is used by the upper host in the figure, throughput of regular TCP is guaranteed by this condition and users using conventional TCP are not negatively impacted by MPTCP. The third goal, *balance congestion* moves traffic from a congested path to an uncongested path as much as possible with satisfying the above two equations. In our simple example in Fig. 4.1, MPTCP's traffic on path b should be moved to path a as much as possible.

In order to conduct this traffic migration required for balance congestion, MPTCP utilizes Window Coupling (WC). In WC, congestion window for each path is coupled with other subflow(s) and cooperatively managed. WC utilizes the size of congestion window as a barometer of congestion on each path. In other words, when the size of window for a path is small, a sender assumes the path is congested, and vice versa.

In conventional TCP, window size of path r, w_r, additively increases per ACK (Acknowledgement) reception as shown in Eq. 4.3. When congestion has occurred on path r, window size is multiplicatively decreased as in Eq. 4.4,

$$w_r \leftarrow w_r + \frac{1}{w_r}, \tag{4.3}$$

$$w_r \leftarrow \frac{w_r}{2}. \tag{4.4}$$

In WC, w_r, window size of path r increases and decreases in a coupled manner as shown in Eq. 4.5 and 4.6, respectively.

$$w_r \leftarrow w_r + \frac{1}{w}, \tag{4.5}$$

$$w_r \leftarrow w_r - \frac{w}{b}, \tag{4.6}$$

where w is coupled window size, i.e. $w = \sum_{r \in \mathcal{R}} w_r$. $\mathcal{R}$ is the set of paths of a multipath flow.

In Eq. 4.6, b is a constant parameter. In TCP's uncoupled window control, w_r increases 1 segment per RTT, thus, the increased amount of traffic is independent of the current window size w_r. In MPTCP, because the number of ACKs received from path r per RTT is w_r, w_r increases w_r/w per RTT. This means the increased amount of each congestion window depends on its window size w_r. In other words, when congestion window of a path is relatively large compared to other paths, the congestion window increases more aggressively.

MPTCP originally introduced the *fully coupled* method where both increase and decrease of window size are coupled, and balance congestion is positively performed to achieve resource pooling. Fully coupled, however, is reported to cause a *flappy* problem. In fully coupled, when window size of a multipath flow is occasionally transferred to one path, this imbalanced situation is kept for a certain long time. Accidentally, window size is flip to another path. This behavior is repeated until the flow completion and called flappy. Because the *linked increase* where only window increase is coupled and decrease is uncoupled, can resolve this flappy problem, MPTCP uses linked increase as its window control algorithm. Although this method departs slightly from the complete resource pooling, it can avoid the flappy problem and stably utilize network resources.

4.1.2 Multipath Congestion Control in CCN

In CCN/NDN, a data packet (ContentObject) is transferred on the reverse path of the corresponding request packet (Interest). Users can control transmission rate of data packets by rate adaptation of its transmitting Interest packets [11, 50, 13, 49, 12, 41]. This kind of congestion control is categorized into two types, *end-to-end* approach and *hop-by-hop* one.

In the end-to-end approach, transmission rate of Interests is regulated by a receiver (user). ICP (Interest Control Protocol) [11], CCTCP (Content Centric TCP) [50], and RAAQM (Remote Adaptive Active Queue Management) [13] are categorized into this end-to-end approach. ICP and CCTCP are TCP-like window-based congestion control with AIMD (Additive Increase/Multiplicative Decrease) algorithm. In both methods, a receiver only manages one congestion window even if its download path has been branched to multiple sources. Thus, the receiver cannot adequately regulate its transmission rate according to congestion state of each branching path, which prevents it from achieving resource pooling. RAAQM is window-based multipath-aware congestion control that utilizes path identification with RTT measurement. There are two policies for Interest forwarding, one is a random forwarding policy and the other is a static weight-based forwarding policy. From viewpoint of resource pooling, the first policy cannot forward Interest to branches at appropriate data rate. The second policy cannot also adapt to dynamic change of congestion state.

HoBHIS (Hop-By-Hop Interest Shaping mechanism) [49], HR-ICP (Hop-by-hop and Receiver-driven Interest Control Protocol) [12], and Flow-aware [41] are categorized into the hop-by-hop approach, where each intermediate router independently regulates Interest transmission rate according to congestion state of neighboring communication links. With these methods, microscale rate control at link level is performed and link-level microscopic fairness would be achieved. However, macroscale resource pooling cannot be achieved because each router does not have the global view.

INRPP (In-Network Resource Pooling Principle) [45] has been proposed to address resource pooling in CCN/NDN. In INRPP, a router which detects congestion on a path discovers another path to offload data traffic. If offloading paths were not found, the router conducts back pressure and mitigate the congestion. Offloading traffic from a congested path to non-congested one accelerates resource pooling, however, INRPP does not assume "multisource" download from multiple content servers. Therefore INRPP cannot be applied to multisource case which is generally the most important and powerful advantage of content oriented networks, such as CCN/NDN.

4.2 MSC^{4}N (MultiSource Congestion Control for Content Centric Networks)

In this section, we discuss resource pooling of MSC^{4}N which has been proposed for multisource congestion control in CCN/NDN [38]. Section 4.2.1 gives an overview of MSC^{4}N. Section 4.2.2 gives consideration of MSC^{4}N from the technical perspective of resource pooling.

4.2.1 Overview of MSC^{4}N

In MSC^{4}N, routers on a download path actively intervene in receiver's congestion control because the routers can directly detect congestion by observing their buffers. This cooperation enables MSC^{4}N to adequately control congestion occurred at anywhere in networks without obtaining location information of a congested path.

MSC^{4}N has the following four technical features.

Router

1. Weight-based probabilistic Interest forwarding
2. Early-stage congestion detection and notification
3. Adequate data rate reduction

Receiver

4. Congestion-aware window decrease

Figure 4.2 shows an overview of MSC^{4}N. In MSC^{4}N, a receiver regulates Interest transmission rate based on AIMD algorithm, and (1) a router at a branching point forwards incoming Interest packets based on probability computed by *weight* for each branching path

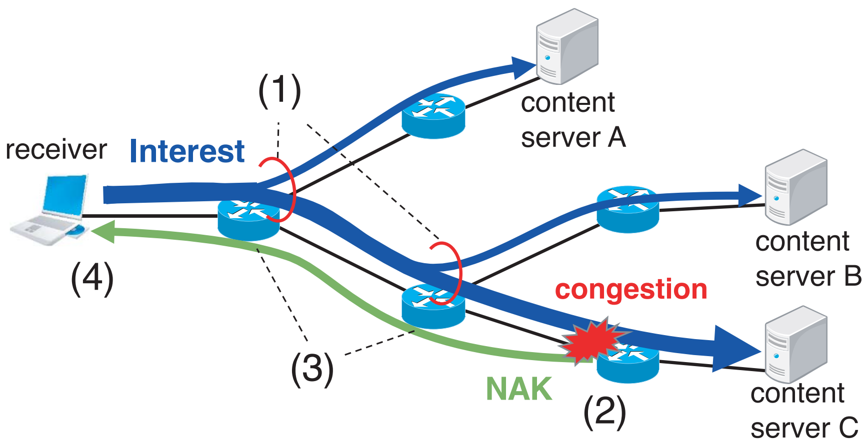


Figure 4.2: Overview of MSC^{4}N.

toward a content server. This weight is dynamically updated according to congestion state of the branching paths. (2) In order to prevent overflow of buffers, every router detects a sign of congestion in early stage and proactively notifies the receiver of the congestion with NAK (Negative Acknowledgement) packets. (3) The first branching router that receives congestion notification from a congested link decreases data traffic by half. The second and later branching routers adequately update their weight and reduce data traffic on the congested link. (4) The receiver also decreases its window size based on the weight of the congested link which is conveyed with NAK. With the above procedures, MSC^{4}N can adequately halve data traffic only of a congested link in multisource download. The other uncongested paths are not affected. Due to limitations of space, we would like to omit the detailed mechanism of MSC^{4}N, and please refer to [38] for more detailed information.

4.2.2 Resource Pooling and MSC^{4}N

In this section, we discuss about adaptability of the proposed MSC^{4}N to resource pooling, i.e. whether MSC^{4}N can achieve resource pooling by accomplishing the three design goals. MSC^{4}N assumes multisource download where a download path can have multiple branching points. In this section, we consider the case that there is one branching point in a download path for simplicity. A receiver increases its congestion window based on Eq. 4.7 per ContentObject reception and decreases based on Eq. 4.8 per NAK reception.

$$w \leftarrow w + \frac{1}{w}, \tag{4.7}$$

$$w \leftarrow w(1 - \frac{d}{2}), \tag{4.8}$$

where w is congestion window size at the receiver, and d is traffic ratio of a congested path to the total traffic from the receiver. d is defined as $d = \frac{g_i}{\sum_{j \in \mathcal{J}} g_j}$, where g_j is weight of interface j and $\mathcal{J}$ is set of output interfaces registered in an FIB entry. The value of d therefore denotes ratio of traffic routed to congested path i to all traffic transferred in multipath. This value is notified with NAK by a router which has detected congestion.

For the first goal (improve throughput), TCP-like congestion control increases w_r (window size of each path r) as in Eq. 4.3 and decreases by half as in Eq. 4.4. When MSC^{4}N is simply applied only to a single path, window increase and decrease behavior would be the same as AIMD behavior of TCP-like method as in Eqs. 4.3 and 4.4. Throughput of MSC^{4}N is equal to that of TCP-like approach. MSC^{4}N can utilize other multiple paths for content download in addition to the single path. This means improve throughput is definitely satisfied. For the second goal (do no harm), in TCP-like congestion control, window size for each path increases 1 segment per RTT. MSC^{4}N increases $\frac{g_i}{\sum_{j \in \mathcal{J}} g_j} < 1$ per RTT. With respect to window decrease, both methods decrease their congestion windows by half. MSC^{4}N's throughput never exceeds TCP-like congestion control's one on each path. For the third goal (balance congestion), in MSC^{4}N, the amount of window increase at a branching point for path i is equal to $\frac{g_i}{\sum_{j \in \mathcal{J}} g_j}$ per RTT. $\frac{g_i}{\sum_{j \in \mathcal{J}} g_j}$ means weight of probabilistic Interest forwarding for path i, i.e. barometer of congestion on path i. When the value of $\frac{g_i}{\sum_{j \in \mathcal{J}} g_j}$ is large, path i is not congested and more likely to be utilized for transferring traffic, and vice versa. Therefore, traffic transferred on a congested path is actively moved to uncongested paths and balance congestion would be accomplished.

Next, we will discuss an analogy of window control between MSC^{4}N and the linked increase method in MPTCP. In linked increase, WC is applied to window increase, and a path which has relatively-large window size tends to increase its congestion window. Window size for path r, i.e. w_r, increases $\frac{w_r}{w}$ per RTT. In MSC^{4}N, a receiver additively increases congestion window per RTT based on Eq. 4.7, and traffic amount which transferred on branching path i increases $1 \cdot \frac{g_i}{\sum_{j \in \mathcal{J}} g_j}$. Also $\frac{g_i}{\sum_{j \in \mathcal{J}} g_j}$ can be regarded as $\frac{w_i}{w}$ by window size w_i which is theoretical[1] window size of path i. Window increase is coupled in both methods, therefore, MSC^{4}N and linked increase in MPTCP have an analogy in terms of window increase. For window decrease phase, linked increase halves window size of a congested path based on Eq. 4.4. MSC^{4}N also decreases traffic transferred in a congested path by half with Eq. 4.8. As described above, MSC^{4}N has an analogy with the linked increase method in terms of window control mechanisms. Therefore, MSC^{4}N has a possibility to achieve resource pooling for multipath/multisource communications in CCN/NDN.

4.3 Performance Evaluation

Firstly, we evaluate the flappy problem, which is one of the technical problems of resource pooling in multipath communications, in Section 4.3.1. Next, we evaluate MSC^{4}N from the resource pooling viewpoint in Section 4.3.2. Throughout our evaluation, we use ndnSIM [66], which is the famous CCN/NDN simulator based on ns-3 [65].

[1]In MSC^{4}N, a router at a branching point probabilistically forwards incoming Interest packets to output interfaces. Precise amount of traffic that transferred in path i fluctuates and may be slightly different from w_i. In this chapter, we define theoretical window size w_i as the product of forwarding probability for path i and window size of receiver w.

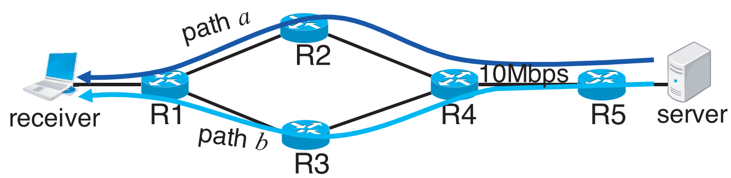
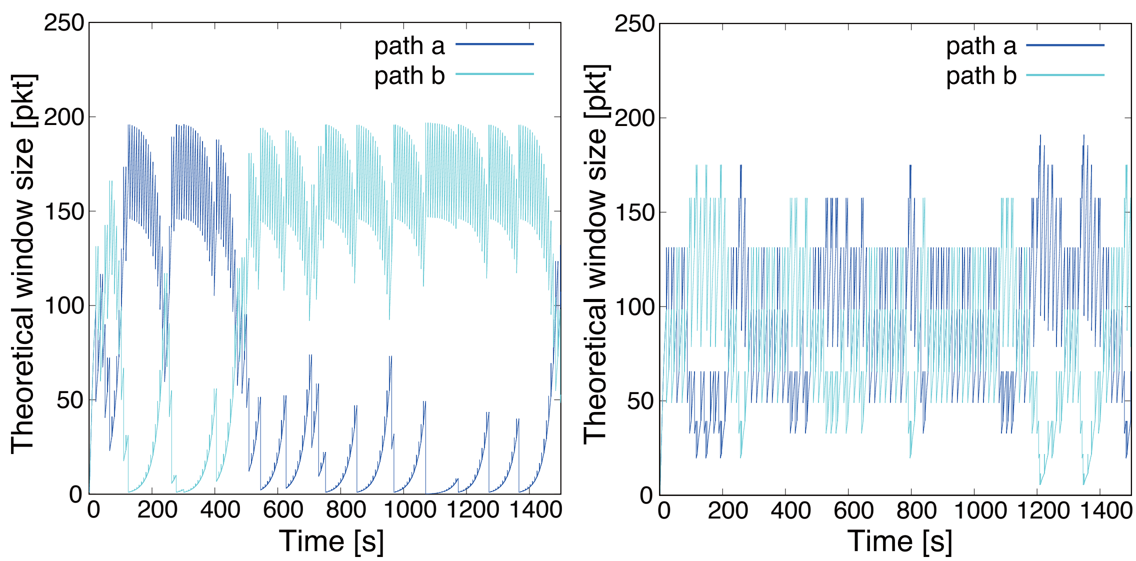


Figure 4.3: Evaluation topology (Shared bottleneck model).

4.3.1 Flappy

Flappy significantly degrades stability of data traffic transferred in multiple paths. In order to stably utilize resources in the multiple paths, MPTCP emphasized avoidance of the flappy above all and utilized linked increase algorithm to avoid the flappy. The flappy problem can also occur in CCN/NDN.



(a) FC^3. (b) MSC^{4}N.

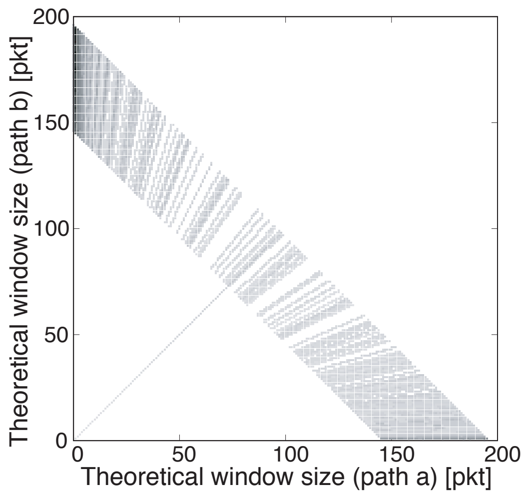
Figure 4.4: Theoretical window size.

Figure 4.3 shows an evaluation topology, which is used in [46] for observing the flappy problem. In this model, two flows of the receiver share the bottleneck link between R4 and R5, whose capacity is 10 [Mbps]. Capacity of the other links is 10 [Gbps]. RTT between the receiver and the server is 80 [ms]. Payload size of ContentObject packet is 1 [KB]. Simulation time is 1500 seconds. The router (R5) sends congestion notifications, i.e. NAK packets when its queue length reaches 100 packets. MSC^4N is compared with Fully Coupled-like Congestion Control (FC^3) where both of window increase and decrease are coupled in CCN/NDN.

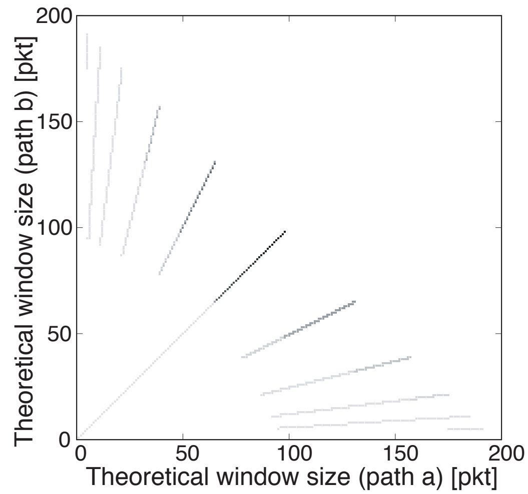
Figure 4.4 shows theoretical window size of FC^3 and MSC^4N on each path. As shown in Fig. 4.4(a), with FC^3, initially data traffic is concentrated on path a for 100-450 [sec], then, the traffic flips to path b around 500 [sec]. This is flappy, and the flappy causes unfair resource utilization of multiple paths. In contrast, theoretical window size of MSC^4N for each path is stabilized as shown in Fig. 4.4(b). Figure 4.5 shows dynamics of theoretical window size for each path. Window decrease lines are not depicted in this figure. Density of the plots means congestion windows are likely to be maintained at that point. Many plots of MSC^4N are concentrated around $w_1 = w_2$, while those of FC^3 are concentrated close to either $w_1 = 0$ or $w_2 = 0$ for a long time. From these results, MSC^4N can stably and fairly utilize multipath resources by avoiding flappy.

4.3.2 Resource Pooling

Next, we evaluate performance of MSC^4N from the resource pooling viewpoint. Figure 4.6 shows an evaluation topology based on the fence model in MPTCP [46, 24](our model is modified to be multi-source model). MPTCP applies RED (Radom Early Detection) [23] as active queue management in order to remove phase effects. We also apply RED for our evaluation to avoid the phase effects. The RED parameters, min_{th}, max_{th}, and max_p are set to 100 [pkt], 200 [pkt], and 0.02, respectively. There are four bottleneck links (link 1-4) whose bandwidth are 10 [Mbps], the other links' capacity is 10 [Gbps]. The other simulation parameters are the same with the previous evaluation.



(a) FC^3.



(b) MSC^4N.

Figure 4.5: Dynamics of theoretical window size.

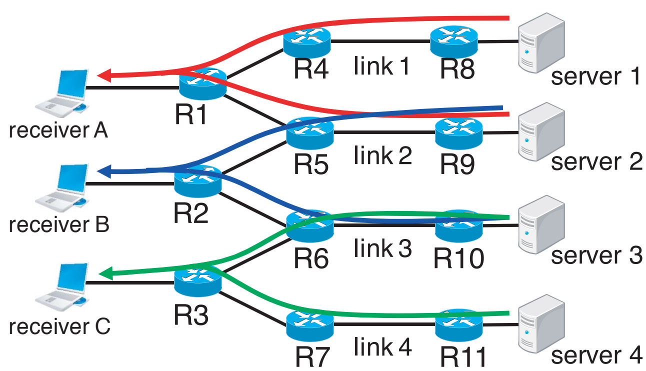
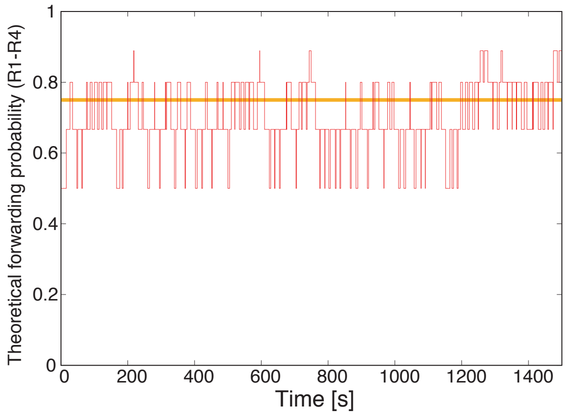
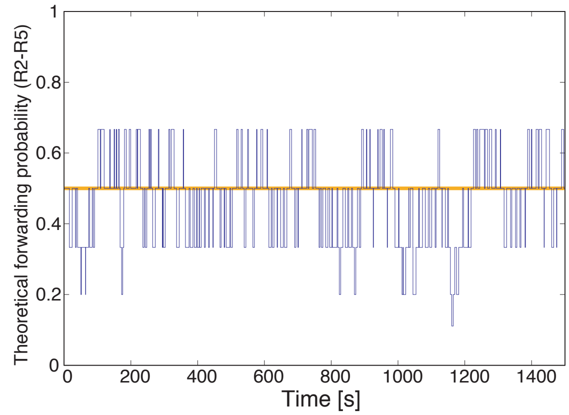


Figure 4.6: Evaluation topology (Fence model).

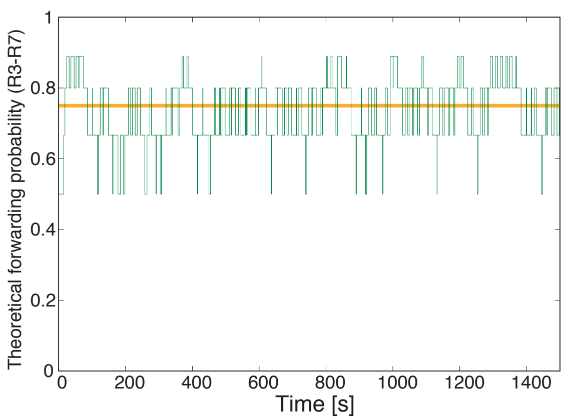
Figure 4.7 shows forwarding probability of Interest at each branching router. The horizontal yellow line on each graph shows the ideal value of forwarding probability of Interest so that the four bottleneck links are fairly shared based on the resource pooling concept. As shown in this figure, MSC^4N can stably forward Interest packets with the ideal probability. This means balance congestion is roughly achieved, and the receivers A and C can move traffic from congested paths (link 2, 3) to uncongested paths (link 1, 4).



(a) R1



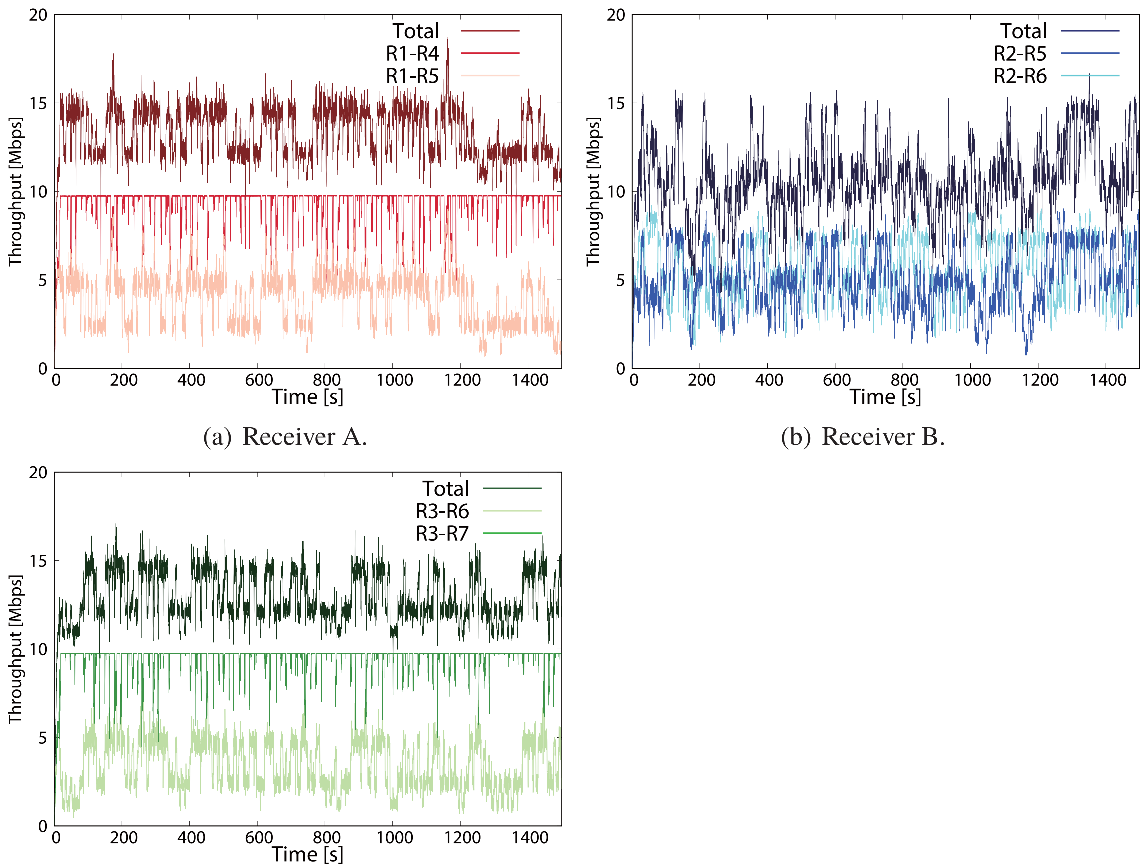
(b) R2



(c) R3

Figure 4.7: Forwarding probability of Interest for each branching path at routers.

Throughput characteristics of receivers are the most important performance metrics in resource pooling. Figure 4.8 shows total throughput characteristic and its breakdown of each branching path. In Figs. 4.8(a) and 4.8(c), the darkest-colored lines denote total throughput of receiver A and C. The lightest-colored lines show throughput of shared bottleneck links (link 2, 3). These results show that traffic on the congested paths (link 2, 3) of receiver A and C is moved to uncongested path (link 1, 4) and is stably regulated. As shown in Fig. 4.8(b), total throughput of receiver B is slightly less than that of receiver A/C because both links are relatively congested compared to link 1 or link 4. Receiver B, however, stably



(a) Receiver A.

(b) Receiver B.

(c) Receiver C.

Figure 4.8: Throughput characteristics.

utilizes multiple paths and gains a certain level of throughput on average. Table 4.1 shows the average throughput of each receiver and throughput normalized by receiver B. When uncoupled congestion control like TCP was used for multipath communications, bandwidth of link 1 and 4 would be exclusively used by receiver A and C, respectively. Bandwidth of link 2(3) is fairly shared by receiver A and B(B and C). As a result, total throughput of receiver A/C is approximately 1.5 times of receiver B's throughput, i.e. normalized throughput of receiver A/C is equal to 1.50. With MSC^{4}N, the normalized throughput is 1.25 at the highest, and fairness among receivers is improved. Therefore, MSC^{4}N can achieve resource pooling at a certain level in CCN/NDN multipath communications and stably utilize multipath resources without the flappy problem.

Table 4.1: The average performance of throughput.

	Rcv A	Rcv B	Rcv C
average [Mbps]	13.4	10.7	13.0
normalized	1.25	1	1.21

4.4 Summary

In this chapter, we introduced a key concept for multipath resource sharing, resource pooling, and MPTCP which is the most famous congestion control addressing the resource pooling in IP networks. We discussed an analogy of MPTCP and MSC^{4}N, the previously proposed multipath congestion control in CCN/NDN. Simulation results show that MSC^{4}N can avoid flappy problem and stably utilize multipath communication resources. Also, MSC^{4}N could achieve resource pooling at a certain level as well as MPTCP.

After we have made studies on basic technologies, i.e. route and cache control schemes in ICN traffic control, in this chapter, we studied congestion control to provide better network performance for users. By taking account of these control schemes, we can efficiently utilize network resources including cached contents and communication links in CCN/NDN.

In the later chapters, we make studies on applied technologies in ICN traffic control.

Chapter 5

Energy Efficient Information Retrieval for Content-Centric Networking in Disaster Environment

From this chapter, we shift our vision to applied technologies in CCN/NDN traffic control. Especially in this chapter, we conduct research on application of CCN/NDN to disaster communications, which is one of the most prospective use cases.

Communication infrastructures under the influence of the disaster strikes e.g. earthquake, will be partitioned due to the significant damage of network components such as base stations. The communication model of the Internet bases on a location-oriented ID, i.e. IP address, and depends on the DNS for name resolution. Therefore such damage remarkably deprives the reachability to the information. To achieve robustness of information retrieval in disaster situation, we try to apply CCN/NDN to information networks fragmented by the disaster strikes. However, existing retransmission control in CCN is not suitable for the fragmented networks with intermittent links due to the timer-based end-to-end behavior. Also, the intermittent links cause a problem for cache behavior. In order to resolve these technical issues, we propose a new packet forwarding scheme with the dynamic routing protocol which resolves retransmission control problem and cache control scheme suitable for the fragmented networks. Our simulation results reveal that the proposed caching scheme can stably store popular contents into cache storages of routers and improve cache hit ratio. And they also reveal that our proposed packet forwarding method significantly improves traffic load, energy consumption and content retrieval delay.

5.1 Fragmented Content Centric Networks

In this chapter, we apply CCN as information distribution platform in fragmented networks in the aftermath of disaster. Figure 5.1 shows an example of assumed network in our scenario. We assume that the network is fragmented due to damage of the components in disaster situations. In this environment, rescue cars and helicopters go around shelters which accommodate evacuees. Rescue teams, e.g., organization of the government: the police, the Self-Defense Forces, and the fire fighting, set their headquarters at the institutions like Government Office (GO) that have the accessibility to the Internet. They explore damage situations, manage transportation of relief goods and provide accurate information to evacuees. The vehicles equipped with CCN routers and wireless antennas to delivery messages are called Data Mules (DMs). Each shelter holds some Gateways (GWs). The GWs play two roles, a wireless access point for evacuees and a CCN router. The CCN routers in the DMs and the GWs have high capacity storages, and store contents such as safety messages. The DMs and the GWs communicate each other and all messages from/to User Devices (UDs) in the shelter are transferred through the GWs.

In this network, the communication links between the DMs and the GWs are intermittent. Long term disconnection and short term connection will occur. In order to recognize the intermittent links and realize reliable communications, a dynamic routing protocol that

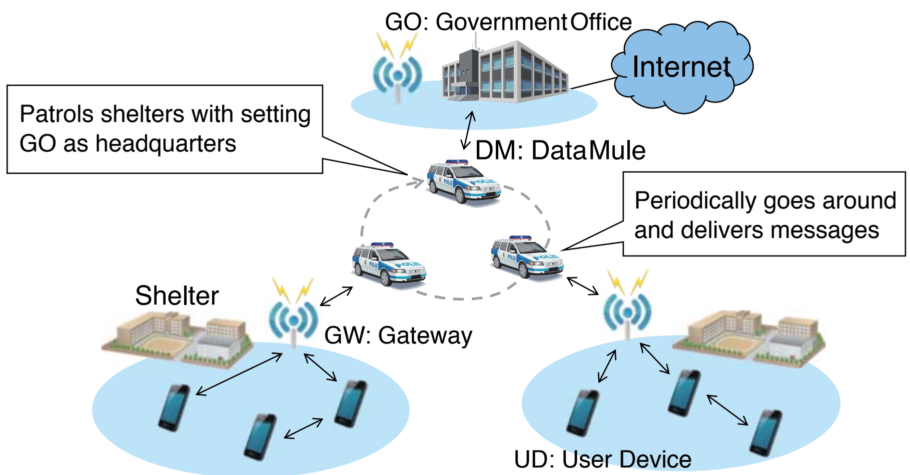


Figure 5.1: Fragmented Content-Centric Networks.

recognizes the intermittent links and a packet forwarding scheme that efficiently transfers packets are needed.

5.1.1 Impact of Intermittent Links

In this section, we explain the technical issues of intermittent links. First, we show a problem of CCN/NDN's retransmission control. We then describe the impact of intermittent links on cached contents.

CCN/NDN's Retransmission Control

This section introduces transport protocol in CCN/NDN and its problem in disaster situations. In CCN, there is no specifications for transport protocol [61]. In contrast, NDN implements reliable data transfer by applications' Interest retransmission [59]. Such end-to-end control imitates the retransmission mechanism of TCP, i.e., timer-base control with RTO (Retransmission TimeOut). In the initial state, RTO is set to the default value based on the estimated RTT. If the node receives a ContentObject during RTO, the node recalculates

RTO with measured RTT. If no ContentObject packets return during RTO, the node infers that the network is congested and doubles RTO value. With this RTO adjustment, the node can reduce the load caused by retransmission, and avoid congestion collapse.

However, we assume fragmented networks in the disaster case whose links are intermittent. If there are intermittent links on an end-to-end path, timer-based Interest retransmission is not appropriate. Even if a user retransmits Interest packets repeatedly, the user may eventually not be able to retrieve ContentObject packets due to the loss of the Interest packets. Because it is hard for the users to estimate the timing of intermittent links' connection and disconnection, timer-based end-to-end retransmission control does not work properly in the fragmented networks. Furthermore, if more than two links are intermittent and end-to-end path connectivity is always disconnected due to the intermittent links, end-to-end retransmission definitely fails. Thus, it is necessary to develop a novel retransmission control scheme for CCN, which is suitable for the fragmented networks.

Cached Contents

Intermittent links cause a considerable influence on cache behavior. Original CCN adopts TERC (Transparent En-Route Caching) and LRU (Least Recently Used) for caching policies. The former is a cache decision policy and the latter is a cache replacement policy. In the fragmented networks, when an intermittent link is recovered, a sending node transmits multiple ContentObject packets to a receiving node at one time. If the receiver uses TERC and LRU for its caching policies, cached ContentObject packets of the receiving node are replaced by the sending node's data transmission.

Figure 5.2 shows an example of the impact of intermittent links on cached contents. At t_1, Interest packets transmitted by node A reach node B. Some of the Interest packets hits on node B and the remaining Interest packets are transferred to node C. Between t_1 and t_2, node C moves to communication range of node D. At t_2, node C receives ContentObject packets

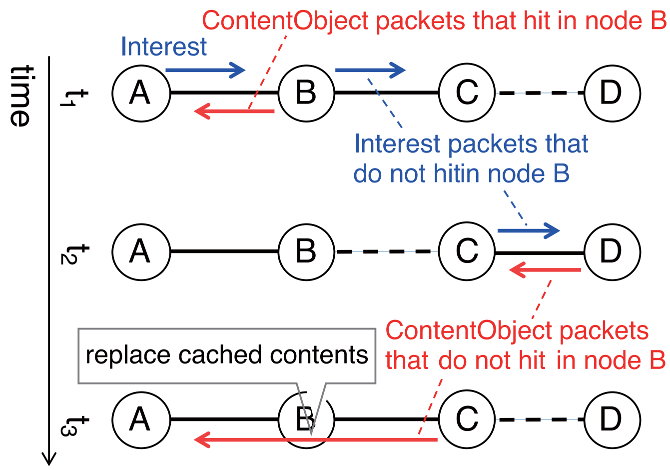


Figure 5.2: Impact of intermittent links on cached contents.

that do not hit in node B from node D. Between t_2 and t_3, node C moves to communication range of node B once again. At t_3, node C transmits the ContentObject packets to node B. Because node B uses TERC and LRU, cached contents in node B are replaced by the incoming ContentObject packets from node C. Since node C transmits ContentObject packets to node B during connection period of the link B-C, arrival interval of the incoming ContentObject packets from node C is much shorter than that of incoming Interest packets from node A. In this case, cached ContentObject packets in CS on node B are rarely accessed by the Interest packets from node A during the node C's data transmission. Unless the cache access occurs in the CS, access time of cached ContentObject packets never be updated by LRU. In such a situation, all of the cached contents stored in the CS will be replaced even if they are relatively popular contents.

To leverage in-network caches as much as possible, it is the better policy that routers cache popular contents rather than unpopular one. The combination of TERC and LRU is reported to store popular contents in networks. However, in fragmented networks, this combination is less effective. A caching policy that is suitable for the fragmented networks and stable store popular contents in the networks is required.

5.1.2 Reason for CCN

As described in the RFC 5050 [52], DTN has bundle applications closely related to its bundle layer protocol. Usual applications that have no tolerance for delay, e.g. video viewing, do not work well in the protocol suites defined in RFC 5050. In contrast, CCN has ability to accommodate file transfer and realtime applications. CCN abstracts interfaces to applications, and the applications do not need to be aware of underlying layers. In disaster situations, CCN forwarding is switched from ordinary one (e.g. OSPFN: Open Shortest Path First for Named-data, default strategy) to our proposals (NDRP, ICOQ) described in the next section. In other words, CCN can switch their forwarding protocol according to network conditions. Abstracted interfaces to applications are not to be changed even when disaster occurs. Decoupling applications from underlying protocol layers in CCN enables flexible usage of network applications both in a disaster case and usual case.

5.2 Information Retrieval Methods for Energy Efficient Disaster Communications

In order to cope with intermittent links in the fragmented networks, we propose following three schemes.

- NDRP: name-based routing protocol
- ICOQ : packet forwarding scheme
- POP : cache decision policy

Cooperation of NDRP and ICOQ improves energy and bandwidth efficiency to realize sophisticated packet transmission. POP improves cache utilization efficiency by stably storing popular contents in CS.

5.2.1 NDRP: Name-based Distance-vector Routing Protocol

NDRP is a name-based dynamic routing protocol that recognizes intermittent links. NDRP adopts periodical updates of route information and sequence numbering of updates referring to DSDV (Destination Sequenced Distance Vector Routing Protocol) [42] used in mobile ad hoc networks (MANET). NDRP advertises name prefixes for contents and allows nodes to calculate the routes to the contents including content replicas with distance-vector algorithm. With NDRP, every node can recognize whether there is any intermittent links on paths to a content or not.

NDRP distinguishes three states for the *route* and two states for the *link*. Route states are distinguished by VALID, STALE and INVALID. In the VALID state, all links on a path to destination name prefix are *Connected*. In the STALE state, there are *Disconnected* links on a path to destination name prefix. In the INVALID state, route timeout occurred while the state keeps STALE. Originally, routing protocols in MANET have two route states, VALID and INVALID. However, fragmented networks have intermittent links and the routing protocol needs to deal with cluster-to-cluster connection. Therefore, we introduce a medial state, STALE, for the temporary unavailable routes to retain disconnected routes among clusters.

There are two states for the link, i.e., connectivity to the next hop neighbors. One is *Connected*, the other is *Disconnected*. *Connected* means that the node keeps receiving routing updates from neighbors. *Disconnected* means that the node does not receive any updates for the link timeout. The link timeout is calculated with the following equation,

$$\begin{aligned} LinkTimeout \text{ [sec]} &= UpdateInterval \text{ [sec]} \\ &\quad \cdot RouteHoldTimes \text{ [count]}. \end{aligned}$$

Update Interval is the interval for broadcasting routing information. *Route Hold Times* means the number of counts of *Update Interval* that keeps link state *Connected*. With this equations, we only regard a link which continuously receives broadcasted control packets as connected. Route timeout is generally set much longer than link timeout (e.g. several hours, a few days). Route state is advertised in the routing information.

The update interval of NDRP advertisement affects bandwidth/energy overhead and content retrieval delay. As the interval of NDRP advertisement becomes long, traffic load decreases and energy is saved. However, our proposal, ICOQ introduced in next section is performed over NDRP. If the update interval is too long, delay performance of ICOQ deteriorates. In the light of the trade-off between bandwidth/energy and delay, we'll set the update interval of NDRP advertisement to 5-15 [sec] throughout our evaluation.

5.2.2 ICOQ: Interest and ContentObject Queueing

ICOQ is the packet forwarding scheme suitable for the fragmented networks. In cooperation with NDRP, ICOQ performs efficient Interest/ContentObject forwarding that saves bandwidth and energy. ICOQ consists of IQ (Interest Queueing) and COQ (ContentObject Queueing).

IQ: Interest Queueing

If there is an intermittent link on a path from a user to a destination name prefix, the route state is STALE and IQ is applied for upward Interest transmission. In IQ, the user and intermediate routers execute hop-by-hop Interest retransmission instead of end-to-end retransmission in original CCN[1]. The user and routers retransmit Interest packets at periodical interval by polling their PIT. They repeat periodical retransmission until

[1]If there are no intermittent links on the path and the route state is VALID, the user executes end-to-end retransmission control as usual.

receiving an Interest ACK (Acknowledgement) or ContentObject packet from upstream neighbors. When a node receives an Interest ACK, the node suspends hop-by-hop Interest retransmission for a constant time. With Interest ACK, the downstream nodes hand over the responsibility of retransmission to the next hop nodes in hop-by-hop manner. This handover continues until the Interest packet reaches the edge of the connected path. When the Interest packet is forwarded to the node that cannot transmit it to the next hop node due to link disconnection, the node stores the Interest packet in its Interest Queue. The stored Interest packet is forwarded when link connectivity is recovered.

Figure 5.3 shows an example of IQ's forwarding mechanism. We assume a situation that node A requests contents stored in node D. Since there is an intermittent link on the path from node A to D, node A, B and C perform hop-by-hop Interest retransmission. This retransmission is suspended by reception of Interest ACK packets. When node A receives an Interest ACK packet from node B, node A postpones its retransmission timeout until constant time later and retains its PIT for the constant time. This procedure is hand over of responsibility of Interest retransmission. Also, node B hands over the responsibility to node C. When the Interest packet reaches node C, node C stores the Interest packet into its Interest Queue because node C cannot transmit the Interest packet to node D due to its link disconnection to node D. Node C waits for transmission opportunity of the Interest packet. When node C moves and link connectivity between node C and node D is recovered, node C sends the stored Interest packet to node D.

COQ: ContentObject Queueing

COQ is applied for taking account of downward link connectivity in DTN environment. In COQ, when a ContentObject packet is forwarded to a router, the router looks up its PIT and checks the connectivity of the output-interface. If the interface is disconnected, the ContentObject packet is stored into the router's ContentObject Queue and waits for

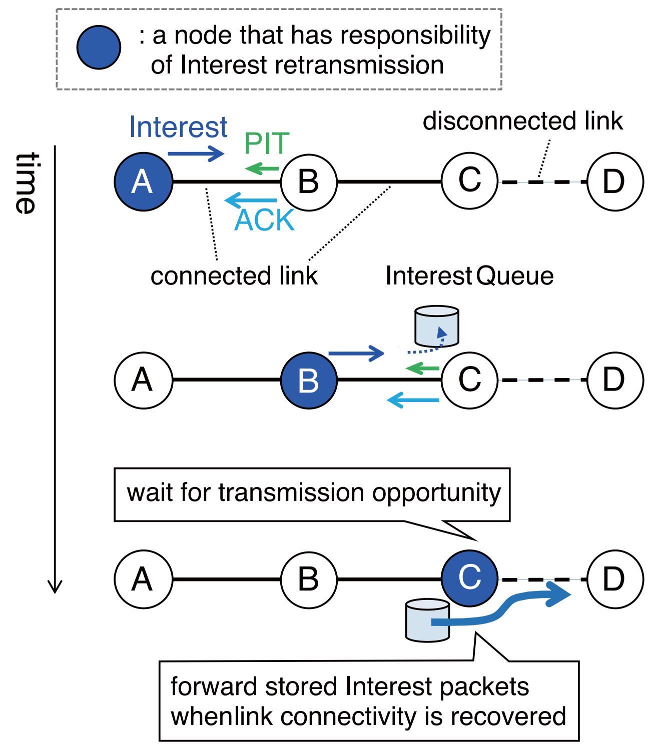


Figure 5.3: Interest forwarding in IQ.

transmission opportunity. The stored ContentObject is forwarded when link connectivity is recovered as well as IQ.

Figure 5.4 shows an example of COQ. This is the successive situation of Fig. 5.3. When node D receives the Interest packet from node C, node D replies a ContentObject packet to node C. Since the link between node B and node C is disconnected, node C stores the ContentObject packet into its ContentObject Queue. When node C moves to communication range of node B, node C dequeues the ContentObject packet and forwards it to node B. In such a way, we recognize intermittent links and perform reliable information retrieval for both upward and downward packet transmission. Also, we save bandwidth and energy by suppressing redundant packet transmissions.

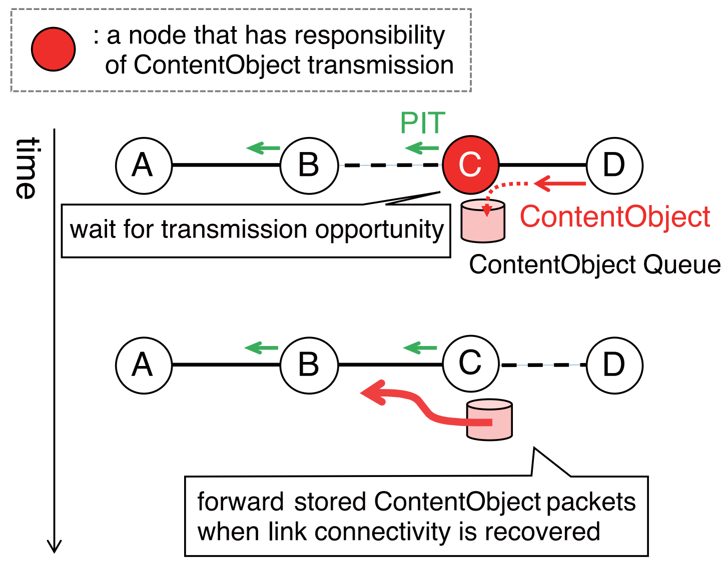


Figure 5.4: ContentObject forwarding in COQ.

Implementation issues of ICOQ in CCN

In order to implement ICOQ in CCN, some modifications of router operation are required. Interest Queue and ContentObject Queue is implemented with PIT and CS, respectively. CCN routers record arrival order of Interest/ContentObject packets in PIT. The routers dequeue the stored packets when corresponding link connectivity is recovered. This method retains routers' PIT entry for a long period compared to the original CCN. There is a possibility of PIT overflow. To resolve this problem, a PIT replacement policy, e.g. prioritized PIT replacement, is necessary, which is left for our future work.

Multi-source Information Retrieval

Our proposals, NDRP and ICOQ utilize multiple repositories, i.e., content replicas and multiple routes toward the repositories. If a router has multiple output-interfaces in a FIB entry, the router checks the state of the interfaces in Interest processing. If all interfaces are disconnected, the router stores the Interest packet into its Interest Queue. The stored packet is immediately forwarded to the corresponding interface when connectivity of an interface is recovered. If at least one interface is connected, the router forwards the Interest packet to an interface that has the least cost, e.g. the least hop count. If there are multiple connected interfaces and their costs are equal, the router randomly selects an interface to forward. In such an operation, we can utilize multiple repositories in the network. However, advertising and leveraging of the content replicas consumes bandwidth and energy. We leave advertisement scheme and utilization method of content replicas for future work.

5.2.3 Popularity-based Caching

In this section, we introduce a new cache decision policy, POP for CCN in the fragmented networks. This method counts the number of incoming Interest packets and ranks them based on the measured numbers. If the number of incoming Interest packets is larger, the rank has higher value. By measuring Interest packets in real-time, we can adapt to content locality or change of content popularity. Consequently, routers stably retain popular contents in their CS.

Algorithm 2 is the pseudo code of POP algorithm. $p_{i,k} \in P_i$ is the cumulative number of incoming Interest packets k in router i. When router i receives Interest packet k, it increments $p_{i,k}$ and sorts its P_i in descending order in advance. When router i receives ContentObject k, the router looks up P_i and gets the popularity rank of content k, $r_{i,k}$. If $r_{i,k}$ is less than S_i, router i stores ContentObject k into its CS and forward it. Otherwise, router i does not store ContentObject k and only forwards it.

To adapt to change of content popularity or locality, router i updates its values of P_i at fixed intervals. As shown in Algorithm 2, each element of P_i is exponentially weighted average. When a value of an element is small enough, it is removed from P_i, which reduces space complexity of this algorithm significantly. With the algorithm, routers decide whether to store an arrived ContentObject packet based on content popularity and perform effective

Algorithm 2 Pseudo code of POP algorithm

$p_{i,k}$: cumulative number of incoming Interest k in router i
$r_{i,k}$: popularity rank of ContentObject k in router i
S_i : cache capacity of router i
- -
When router i received Interest k
increments $p_{i,k}$
sorts P_i in descending order
When router i received ContentObject k
looks up P_i and gets the popularity rank $r_{i,k}$
if $r_{i,k} <= S_i$ **then**
  stores ContentObject k and forwards it
else
  does not store ContentObject k and forwards it
end if
At fixed intervals
$P_i \leftarrow \lambda \cdot P_i$; $0 < \lambda < 1.0$

content caching.

There are some popularity-based cache decision policies that are similar to POP [37][27][7]. However, in disaster situations, these policies do not take account of intermittent links and may not be able to realize our ideal situation that routers close to users store popular contents.

LCD (Leave Copy Down) [37] is a popularity-based approach inspired by the feature that popular contents are frequently downloaded by users. Interest packets, however, are aggregated at intermittent links in fragmented networks, and the number of content downloads for popular contents declines. Prob-PD [27] probabilistically decides whether to store ContentObject packets based on caching probability that considers content popularity and download hop count. However, when a user download a content with Prob-PD and ICOQ, the content may not be cached at the one-hop downward node of a server due to its "probabilistic" approach. In such a case, the content is not to be forwarded to the user thereafter. In MPC (Most Popular Content) [7], each router measures the number of incoming Interest packets and decides whether to store ContentObject packets based on a threshold value. In our assumed situation, RTT is extremely long compared to wired networks and highly fluctuates. This feature prevents routers from configuring an appropriate threshold value and caching popular contents.

In any of these cases, routers cannot store popular contents. In our proposed POP, each router measures incoming Interest packets and simply utilizes their relative popularity for caching. By using this policy, we can stably store relatively popular contents into routers close to users.

5.3 Performance Evaluation

This section gives simulation results for performance in terms of packet forwarding scheme in 5.3.2 and cache decision policy in 5.3.3. Also, overall performance for content size variation is revealed in 5.3.4. We use ndnSIM1.0 [66] as a simulation tool and run 10 different simulations for each evaluated result.

5.3.1 Simulation Model

In our evaluation, we assume that at most 100 evacuees take refuge to one shelter in reference to [64]. DMs also rarely connect to more than two GWs simultaneously. Figure 5.5 shows the simulation topology. There are four types of node, User Device (UD), Gateway (GW), Data Mule (DM) and Government Office (GO). The DM repeatedly goes back and forth between the GW and the GO at a constant speed of 36km/h. The cycle of the DM is 500 seconds. For wireless communication, we use 802.11g. Cache replacement policy is LRU. The critical factor to degrade the performance is wireless interference between UDs, GWs and DMs. However, with using Wi-Fi in a shelter, there are no interference among the GWs in different shelters because of the distance.

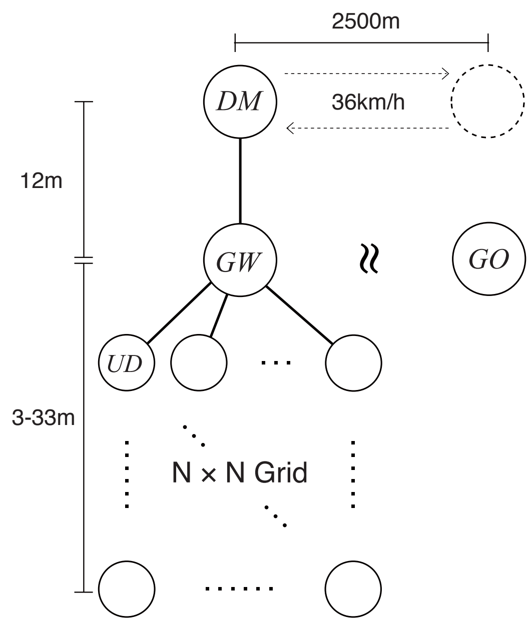


Figure 5.5: Simulation topology.

5.3.2 Effect of Packet Forwarding Scheme

To evaluate efficiency of ICOQ, we conduct computer simulation. Traffic pattern of users follows flash crowd model that all users transmit their requests within about 10 seconds. In this section, the size of a content is 50K bytes. A content is divided into 50 chunks. All contents are stored in GO. Each user requests a different content. In this evaluation, we would like to focus on evaluation of packet forwarding method and do not care the aspect of caching. In the disaster case, batteries of both user devices and network equipment are limited. Energy efficiency is, therefore, one of the most important aspects for network sustainability. We use the number of user devices as the simulation parameter and three performance metrics, i.e., traffic load, energy consumption of user devices and delay for content retrieval.

We compare the following four methods in our evaluations.

1. *NoSup: No Suppression*

   Lifetime of a PIT entry in CCN routers is assumed to be so long that it is not expired during disconnection of paths. Interest packets are repeatedly retransmitted by both the end user and intermediate routers until a ContentObject packet is successfully received.

2. *SupForDisconnect: Suppression for Disconnection*

   In addition to *"No Suppression"*, retransmission of Interest packets is suppressed when a next hop link is disconnected. Link disconnection is detected by NDRP.

3. *IQ: Interest Queueing*

   Our proposed IQ in 5.2.2 is only applied.

4. *ICOQ: Interest and ContentObject Queueing*

   In addition to IQ for upward interest transmission, COQ in 5.2.2 is also applied for downward data transmission.

Network Performance and Energy Consumption

Figure 5.6 shows traffic load characteristics for the number of users in the GW. Traffic load is defined as the total number of transmitted Interest by the users, the GW and the DM. In NoSup, which is a straw man approach, redundant retransmission of Interest cannot be prevented. In SupForDisconnect, redundant retransmissions slightly decrease compared with NoSup because the GW stops retransmission while the link with the DM is disconnected. Our proposed IQ and ICOQ use Interest ACK for suppression of retransmission and significantly decrease redundant Interest transmission.

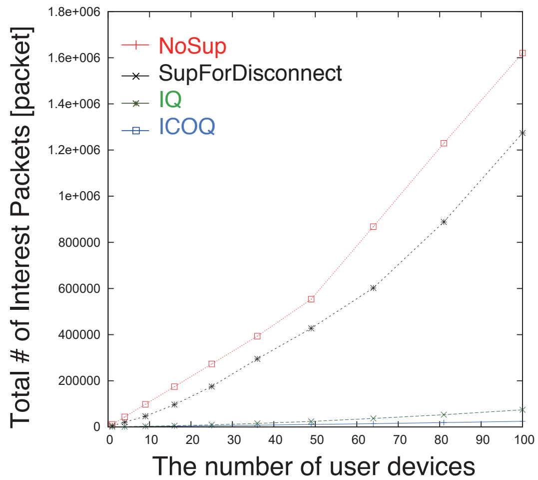


Figure 5.6: Traffic load characteristics.

Table 5.1 shows the total performance in terms of energy consumption of all users. The second column is the mean value for 100 users and the third column is the coefficient of variation. Energy consumption is defined as the total consumed energy until the completion of the content retrieval. This energy consumption is calculated with the energy model in ns-3 [65], which is the base simulator of ndnSIM. The wifi radio energy model in ns-3 calculates energy consumption for data transmission/reception, Clear Channel Assessment (CCA) busy/idle and switching. These parameters are set to the default value of ns-3. NoSup and SupForDisconnect consume plenty of energy due to the redundant Interest retransmission. In SupForDisconnect, the GW prevents redundant Interest retransmission because it can distinguish disconnection of the link between the GW and the DM with NDRP. However, since user nodes cannot find out disconnection of this link, all user devices continue redundant Interest retransmission. Our proposed ICOQ can achieve less energy consumption as a result of the decrease of redundant Interest retransmission. In contrast, nevertheless IQ restrains its redundant retransmission of Interest packets, it has high energy consumption. This is because IQ scores low performance in terms of content

Table 5.1: Average energy consumption of user devices.

forwarding method	mean [J]	coefficient of variation
NoSup	5.04	0.038
SupForDisconnect	11.7	0.046
IQ	7.94	0.074
ICOQ	1.88	0.176

retrieval delay as mentioned below. In the energy model we employed, nodes get more chances of CCA and receiving routing messages because of the increase of content retrieval delay. Therefore, the longer content retrieval delay causes higher energy consumption.

Content Retrieval Delay

Figure 5.7 shows the average time for content retrieval of all users. With increase of the number of users, content retrieval delay of IQ and SupForDisconnect increases. With NoSup, in case of over 49 users, data transmission from the DM to the GW cannot be completed during the connection period due to wireless channel contention among users especially around the GW. Content retrieval delay increases step-wise because data transmission is postponed to the next connection period between the DM and the GW.

IQ shows better performance from the viewpoint of the overhead of Interest transmission as shown in the previous section. It has the worst performance regarding the content retrieval delay. In IQ, the DM does not care about the link state with the GW. The DM starts data transmission just after it obtains ContentObjects from the GO. The GW obtain contents from the DM by retransmitting Interest packets because the DM has cached contents in its CS. Since retransmissions of Interest packets are regulated for a long period by Interest ACK, periodical retransmission phase in the GW is intentionally delayed. Therefore, IQ cannot adequately adjust retransmission suspension for circulation interval of the DM.

SupForDisconnect achieves slightly better content retrieval delay than IQ. This is because the GW periodically checks its PIT while the state of the link to the next hop is disconnected and starts Interest retransmission just after link state is changed to connected. This Interest retransmission behavior is similar to NoSup. However with SupForDisconnect, the DM starts data transmission with short delay. This is because the state of link in the GW changes after NDRP receives routing message from the DM. There is a time lag between the state of physical connection and logical link connection.

Eventually, content retrieval delay of three methods mentioned above is worse than our proposed ICOQ. Other three methods retrieve contents in pull-based fashion, that is, content retrieval is initiated by Interest retransmission. On the other hand, in ICOQ, since the DM pushes requested contents just after it established a link with the GW, it scores high performance. Content retrieval delay of our proposed ICOQ shows increase at 80-100 users. In this situation, all user traffic demands are not satisfied in one cycle of the DM. Compared with the other three methods, ICOQ moves this overload situation to the rightmost in horizontal axis, i.e., large number of users. ICOQ transfers packets with the most efficient bandwidth usage and has the best performance. Also, it has the least energy consumption as shown in the previous section. Simulation results conclude that ICOQ is the most suitable packet forwarding scheme for CCN in disaster situations.

5.3.3 Effect of Cache Decision Policy

In this section, we evaluate our proposal, POP compared with TERC and show the impact of cache decision policy on performance in terms of cache hit ratio and delay. In this evaluation, we adopt ICOQ for packet forwarding scheme. The number of contents is 100. The size of each content is 1K bytes. Interest is generated with 0.01 packet per second. Popularity of generated Interests follows the Zipf-like distribution ($\alpha = 1.0$) [10]. The DM and the GW are equipped with CS whose capacity is 20K bytes. Simulation time is 50000 seconds. The other assumptions are same as the previous evaluation.

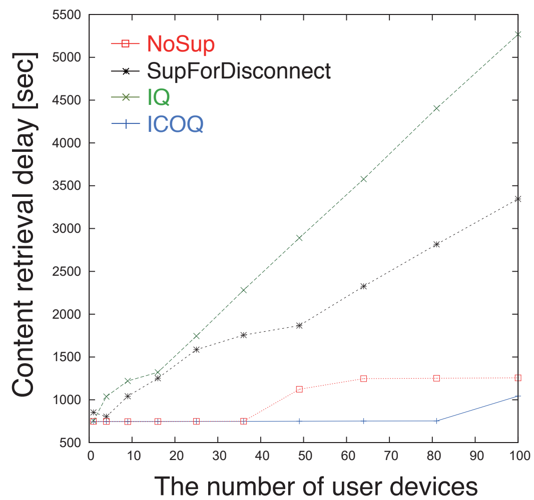


Figure 5.7: Content retrieval delay vs the number of user devices.

Cache Hit Performance

Figure 5.8 shows cache hit ratio in the GW. In the horizontal axis, the smaller content ID is, the more popular the content is. POP improves cache hit ratio of popular contents. In our model, the GW receives many Interest packets for popular contents. By using POP as cache decision policy, the popular contents are more likely to be stored in the GW. Users can retrieve the popular contents from the GW when compared to TERC. POP promotes the GW to cache the popular contents and the GW seldom cache the unpopular one. In TERC, there are some cache hits for unpopular contents compared to POP. However, the gain of them is not influential and the loss of popular contents is dominant for performance.

With TERC, since routers store all incoming contents, cached contents in the DM and the GW will be identical during one travel cycle of the DM. In contrast, in POP, the GW caches popular contents and due to filtering effect, the DM caches relatively unpopular contents. As a result, POP can efficiently use cache capacity in the network.

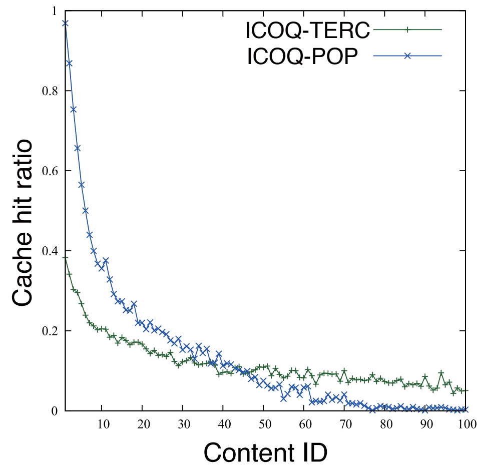


Figure 5.8: Cache hit ratio for each content.

Content Retrieval Delay

Next, we evaluate delay for content retrieval by all users. Figure 5.9 shows average content retrieval delay for each content. In POP, users retrieve popular contents from the GW and their delay become very short. Also, POP efficiently uses cache capacity of the DM and the GW compared to TERC. Average content retrieval delay for all contents of POP is 349.27 [sec] and that of TERC is 509.63 [sec]. Our proposed POP obtains approximately 30% performance improvement in content retrieval delay.

5.3.4 Effect of Content Size Variation

In the previous evaluation, we set each content size to 1K bytes. In this section, we vary content size from 1K (1 chunk) to 10K bytes (10 chunks). Also, we change cache capacity of CS on the DM and GW from 20K to 200K bytes according to the content size. We compare ICOQ with the other forwarding method, NoSup that does not recognize link connectivity. We used POP and TERC for cache decision policies as in the previous evaluation. Performance metrics are traffic load, energy consumption of user devices and

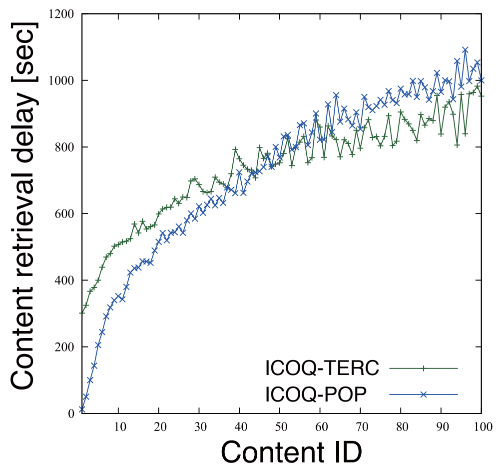


Figure 5.9: Content retrieval delay for each content.

content retrieval delay. For other assumptions and simulation parameters, we use same parameters as the previous evaluation in 5.3.3.

Traffic load and Energy Consumption

Figure 5.10 shows traffic load characteristics with various content sizes. Traffic load is defined as the number of transmitted Interest packets in the whole network. As content size increases, traffic load proportionally increases. In NoSup, traffic load significantly increases due to the redundant Interest retransmissions. Since NoSup does not care link connectivity, the users and routers redundantly retransmit Interest packets. The redundant Interest packets generate enormous traffic in the network. In contrast, ICOQ uses Interest ACK packets for suppression of such redundant retransmission and prevents much redundant traffic from being generated.

Figure 5.11 shows average energy consumption per user during the simulation. Energy consumption increases linearly with content size. ICOQ consumes much less energy than NoSup. By decreasing redundantInterest transmissions, ICOQ can save energy required for data transmission/reception. NDRP advertisement generates additional traffic load

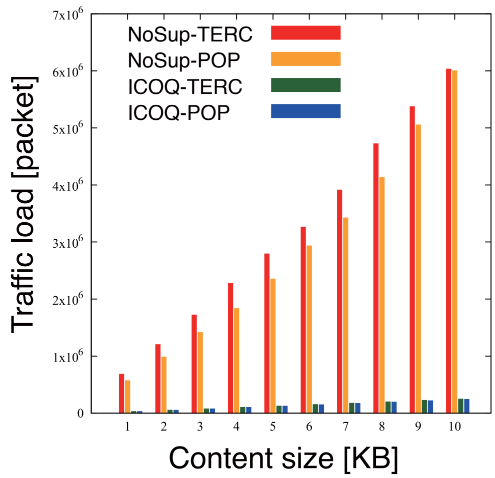


Figure 5.10: Traffic load vs content size.

and energy consumption. However, if we do not use routing protocol, each node cannot recognize connectivity to its neighboring nodes. In such a case, each node has to take not-reactive approaches, e.g. periodical flooding studied in [4] although the approach consumes considerable energy consumption as shown in NoSup of Fig. 5.11. ICOQ can recognize link connectivity thanks to NDRP and suppress redundant Interest transmission during link disconnection. As a result, ICOQ reduces energy consumption by suppressing redundant Interest transmission. Although user devices consume a little energy (approximately 4.8[J]) by enabling NDRP, this increase is negligible compared to that of NoSup's redundant Interest transmission. Eventually, the users can retrieve contents with small amount of energy consumption with ICOQ compared to routing-less approaches.

Content Retrieval Delay

In this evaluation, we show performance results for content retrieval delay. Average content retrieval delay is shown in Fig. 5.12. In both of forwarding methods, POP retrieves popular contents from the GW and dramatically reduces content retrieval delay compared to TERC.

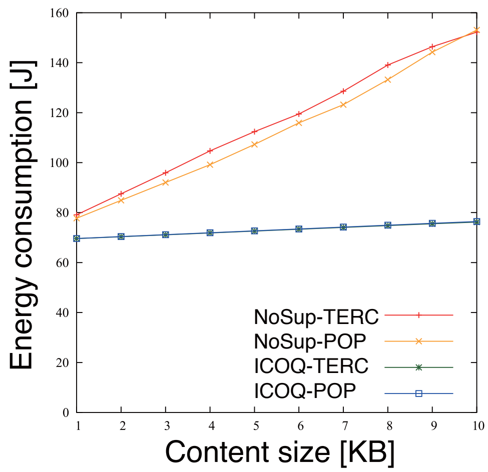


Figure 5.11: Energy consumption vs content size.

Next, we will focus on *content integrity* in the GW. The definition of content integrity is time ratio that all chunks of a content are together in CS on the GW throughout the simulation. Figure 5.13 shows content integrity of the most popular content. POP improves content integrity for popular contents in both of forwarding methods because the GW preferably caches popular contents. POP makes the GW stably retain popular contents in bulk.

In ICOQ, while there is little effect of content size variation on content integrity, content size variation is influential factor for NoSup. This is because, in NoSup, Interest and ContentObject packets are transmitted without any consideration for connectivity of transmission channels and packets are frequently lost when they are transmitted in disconnected phase. If a packet is lost, the packet is retransmitted by hop-by-hop Interest retransmission in the CCN forwarding layer. In such a case, ContentObject packets are not transferred in sequence order. Increase of the number of chunks promotes chunk fragmentation and content integrity decreases. Thus, content retrieval delay drastically increases as in Fig. 5.12. In contrast, ICOQ takes account of connectivity of transmission channels with the routing

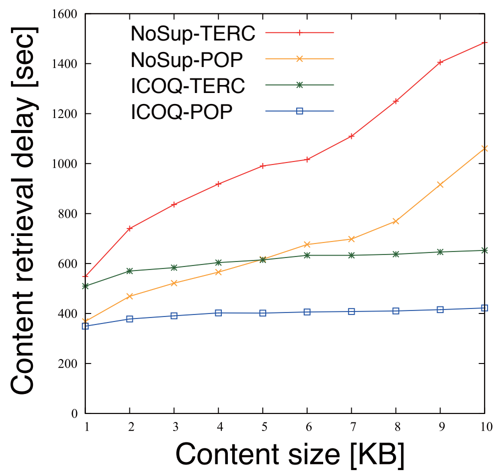


Figure 5.12: Content retrieval delay vs content size.

protocol and senders transmit packets when they are located sufficiently close to receivers. Because bit error rate is small and the packets are rarely lost, they are likely transferred in sequence order. Thus, ICOQ retains more contents in bulk at the GW's CS than NoSup.

In fragmented networks, it is preferable to cache popular contents into a router on a path from user nodes to an intermittent edge. In addition, each content should be cached in bulk at the router's CS. If chunks of the content are not completed in the CS, the cache capacity is eventually wasted. POP enables routers to cache popular contents and the routers possess the complete set of each content. These findings show that the combination of ICOQ and POP achieves the best performance in terms of cache hit ratio, traffic load, energy consumption, and content retrieval delay.

5.4 Related Works

There are several CCN/NDN studies for DTN [3, 25, 36, 4, 55]. Since CCN inherently has DTN-like features, e.g., network cache and flexible routing, CCN can be applied to

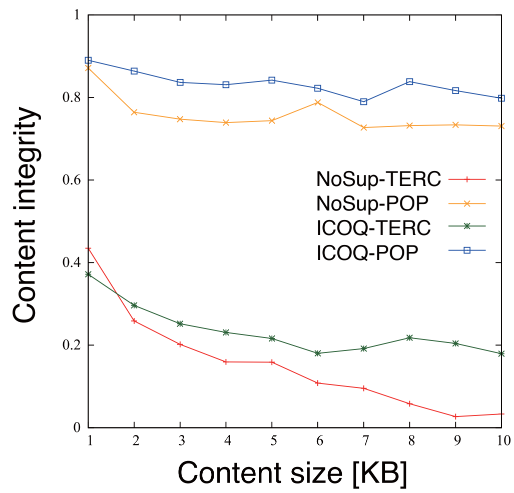


Figure 5.13: Content integrity of the most popular content.

disruptive environments.

CCVN (Content-Centric Vehicular Networking) [3] and V-NDN (Vehicular Named Data Networking) [25] assume highly-dynamic environment compared to our targeted environment. Because such dynamics prevent vehicles from constructing accurate routing tables, they utilize wireless broadcast or soft-state Interest forwarding instead of FIB-based routing. Furthermore, they do not take energy consumption into account because every vehicle has enough battery for communication in VANET (Vehicular Ad-hoc Networks). Since we assume the network in post-disaster situation, the network topology would become less-dynamic yet disruptive and batteries of user devices are very limited. In addition, these studies do not care about efficient bandwidth utilization and packet retransmission control.

Y. Lu, et al., proposed DTN routing based on social-tie among DTN nodes in [36]. Each node records social-tie value for other nodes. When a node encounters another node, this social-tie value of the corresponding node (encountered node) is incremented. So, when a node has high social-tie value for node A, this node frequently encounters node A. When a node has a data to be forwarded to node X and encounters another node, it forwards a data to this node only if encountered node has higher social-tie value of node X. This chapter only focuses on routing and does not take account of retransmission control nor energy consumption.

C. Anastasiades, et al., have proposed agent-based content retrieval method in CCN in [4]. In this method, requesters delegate content retrieval to agent nodes that go around the network. The method is composed of three phases, *agent delegation*, *content retrieval*, and *content notification*. In agent delegation, a requester delegates content retrieval to an agent node. The agent node retrieves contents delegated by the requester in content retrieval phase. In content notification, the agent transfers the retrieved contents to the requester. Although this study assumes less-dynamic scenario, energy/bandwidth-saving aspect is not considered. In content retrieval and notification phase, agent nodes broadcast content request and data because the agent nodes cannot understand link connectivity.

A. Tagami, et al., have proposed Publish/Subscribe based information delivery in disaster situations in [55]. Communication model of this study is push-based and that of our work is pull-based. Also, we assume not Publish/Subscribe communications but request/reply communications. In other words, this study achieves a mutually complementary relationship with our work. With respect to performance evaluation, we use not only delay but also energy aspect although this study only uses delay for performance metrics.

With our approach, the set of NDRP, ICOQ, and POP, users' applications can stably communicate with each other in disaster environment where communication quality is unstable and device energy is highly limited.

5.5 Summary

In this chapter, we propose the new mechanism for fragmented CCNs in the disaster situation. Since original CCN assumes well-connected wired networks, it cannot be simply applied to the fragmented networks whose links are intermittent. We proposed a dynamic routing protocol and a packet forwarding method. The combination of these two techniques realizes energy efficient information retrieval in fragmented networks by recognizing the intermittent links. Also, we proposed a new cache decision policy suitable for the fragmented networks. The policy makes routers stably cache relatively popular contents by measuring incoming Interest packets. Simulation results show that our proposal can improve the performance and efficiency of information retrieval in terms of cache hit ratio, traffic load, energy consumption and content retrieval delay.

In the next chapter, we shift focus of our attention to application of CCN/NDN to IoT data processing, which is the another promising use case of CCN/NDN.

Chapter 6

Information-Centric Function Chaining for Data Processing in Internet of Things

In chapters 2-5, we have studied ICN traffic control as a method only for *information retrieval* thus far. ICN originally innovated for efficient content distribution, is currently discussed to be applied to edge computing in IoT (Internet of Things) environment. In this chapter, we focus on more flexible network processing environment, *in-network processing*, which is realized with ICN architecture. In our assumed environment, multiple functions are executed on different routers widely distributed in a whole network and the end-to-end optimal route for any data processing should be selected to satisfy various IoT applications' requirement. Our proposal, an on-demand routing method efficiently chains data and multiple functions compared to an existing proactive routing method. Also, our method reactively caches routing information in the network and realizes scalable routing for ICN-based in-network processing.

6.1 ICN-Based In-Network Processing

In this section, we explain background technologies for ICN-based in-network processing.

6.1.1 NFN (Named-Function Networking)

NFN has proposed ICN-based in-network processing with the CCN/NDN framework[56]. An NFN router executes an application and provides a function of data processing. Users can download processed data from networks by designating name of desired data and functions.

Figure 6.1 shows an example of in-network processing in NFN. (1) A user sends an Interest packet, which means a data processing request. The Interest packet includes names of required data "$/data$" and a processing function "$/A$". (2) The Interest packet is transferred to an NFN router based on FIB. (3) The NFN router interprets the contents of the Interest packet and divides the Interest packet into two Interest packets. One is for the data "$/data$" and the other is for the function "$/A$". These Interest packets are simultaneously transmitted to the network by the NFN router. (4) Retrieved "$/data$" is processed with downloaded function "$/A$" at the NFN router. (5) Processed data is replied to the user along PIT (Pending Interest Table) trails, i.e. reverse path of the Interest packet. With these procedures, users can download data processed in the network. NFN routers, however, basically download functions to process data, and function placement in the network can be highly dynamic due to undesigned downloads by users.

6.1.2 ICN-Based Function Chaining

In ICN-based in-network processing, a concept of *function chaining* is originally introduced in [34, 35]. In ICN-based function chaining, functions are located (assigned) at routers

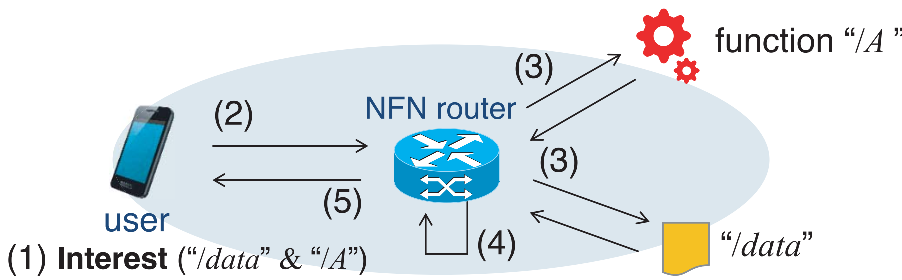


Figure 6.1: Example of in-network processing in NFN.

in a network beforehand and the routers are executing several functions[1]. An Interest packet transmitted by a user chains multiple functions and reserves data processing along its transmission path. During ContentObject packets transmission, the reserved functions are applied to the passing ContentObject packets and required data processing is executed along the ContentObject transmission path.

Figure 6.2 describes an example of data processing in ICN-based function chaining. A user sends an Interest packet to retrieve data "$/video$" which is processed by functions "$/combine \rightarrow /compress$". An arrow means a processing order constraint. The name of Interest packet is expressed as "$/video \rightarrow /combine \rightarrow /compress$". When the Interest packet arrived at the function "$/compress$", the name of function "$/compress$" is removed from the Interest name. After all functions have been correctly chained, the Interest packet finds "$/video$" and ContentObject packets are replied to the user. During the ContentObject packets transmission, functions "$/combine \rightarrow /compress$" are applied along its transmission path. With these procedures, the user can download processed data from the network.

In order to realize this communication, proactive FIB is required to route Interest packets

[1]Computation capacity of routers is generally low compared to computation severs in clouds, and the routers need to conduct another tasks, e.g. frequent table lookup and line-speed packet forwarding. Moreover, distribution of functions might be regulated by licenses of function providers. Hence, we assume that the number of functions provided by one router is highly limited.

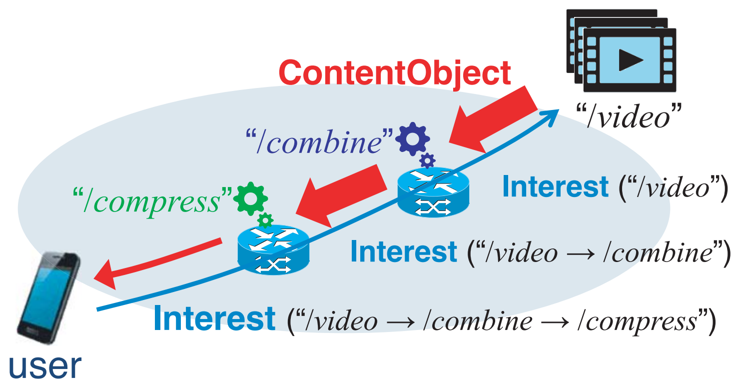


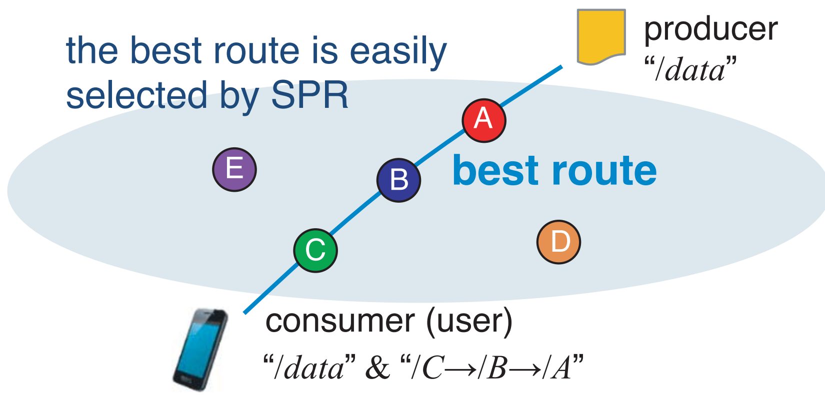
Figure 6.2: Example of ICN-based function chaining.

to the appropriate functions and data in a correct order. This method specifically assumes transcoding "*/compress*" and combination of video clips "*/combine*" as functions. In this case, route information for each destination ("*/video*", "*/combine*", and "*/compress*") only need to be prepared in FIB in advance. However, in IoT data processing, there can be a number of functions and data distributed in a whole network. Moreover, the functions can be redundantly prepared for effective data processing.

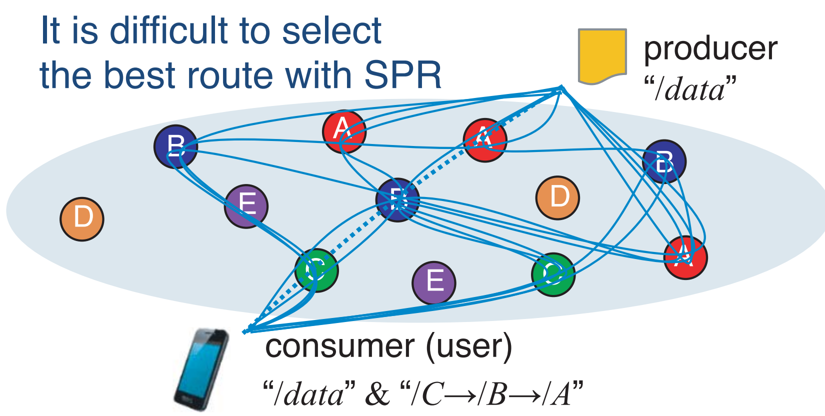
6.1.3 Difficulty of ICN-Based Function Chaining

In function chaining (data processing) in NFV (Network Functions Virtualization), each function is replicated according to demand of services in order to accommodate various users' requirements of the network functions. A function which has high service demand, i.e., a popular function, is proactively replicated and placed to servers as VNFs (Virtual Network Functions) in a cloud. In a similar fashion, in the ICN-based function chaining, popular functions are positively replicated and placed to routers in networks[2]. However, our assumed environment for function chaining is not in a narrow cloud but in a wide network

[2]In this chapter, we only focus on function chaining of named functions, i.e., routing of the functions. Placement of the functions is out of scope of this chapter (linear programming design is one possible approach).



(a) Chaining from a single function.



(b) Chaining from multi-replicated functions.

Figure 6.3: Difficulty of the generalized ICN-based function chaining.

close to edges. There are many kinds of functions for data processing, and a huge variety of IoT data require the data processing. Therefore, we focus on a decentralized approach from viewpoint of routing scalability.

Figure 6.3 shows a difficulty of generalized ICN-based function chaining. As shown in Fig. 6.3(a), in the case that a certain function is served by only one router in the network, i.e. we have only one route selection option for a certain function, proactive SPR (Shortest Path Routing) can route request packets along the best route. In other words, a routing policy with local search, NFF (Nearest Function First) where the nearest function is selected as a destination at each router is reasonable. For example, as in Fig. 6.3(a), function *C* is selected as the first destination, function *B* is the second one, and so on. In this case, NFF can select the global optimal solution. In the general case, however, each function is replicated, and multi-replicated functions that execute the same functionality are served by multiple routers in different locations in the network as shown in Fig. 6.3(b). In this case, NFF occasionally falls into local optimal solutions because the number of possible routes significantly increases according to the number of replicated functions. In addition to this, there are no means to specify each replicated function at a router because the router cannot have a global view for each function with ICN's name-based routing. Even though the routing method that gives the global view to the router exists, the amount of routing information recorded on the router's FIB is innumerable. Combinations of these functions and diverse IoT data cause scalability issues.

6.2 Proposals: OR^3&RR

6.2.1 OR^3 (On-demand Routing for Responsive Route)

Our goal is to provide the best route for function chaining to routers without a global view in a distributed manner (not in a centralized FIB calculation with proactive routing). To this end, we propose OR^3 (On-demand Routing for Responsive Route), a flooding-based routing method that discovers all possible routes and selects the best route for function chaining.

The flooding-based approach brings the following benefits.

1. The Best Route. Flooding means full search of possible routes for function chaining, thus, we can select the "best" one from the discovered routes. In this chapter, the best means shortest path in terms of hop count. This benefit is our main goal of this chapter.

2. Load Balancing. During flooding search, a discovered route might include a congested link, or a function executed on the route can be overloaded for another tasks of data processing. Accordingly, we can control path selection by adding delay or other metrics to flooding search mechanism.

3. Robustness. Since communication links are infrequently disconnected, proactive routing needs some recovery methods. On-demand flooding search can avoid such links adaptively and flexibly select stable route for data processing. In wireless environment, link disconnection frequently occurs and nodes that execute functions have mobility. Even in such environment, flooding search can flexibly discover appropriate routes.

In the rest of this chapter, we address the benefit 1 (The Best Route), and the remaining parts are left to future works.

Mechanism of OR^3

Figure 6.4 shows an overview of OR^3. OR^3 utilizes flooding of Interest packets (R-Interest) to discover the best route among all combinations of data and functions. R-Interest is a route discovery packet used for finding a transmission path of data processing. When an R-Interest packet, whose name is "$/data$ & $/C \rightarrow /B \rightarrow /A$", has discovered the first required function C, the name of the function "$/C$" is removed from the name of the R-Interest packet. And a new renamed R-Interest packet "$/data$ & $/B \rightarrow /A$" is created and flooded. This flooding is continued until all required functions are successfully chained. The producer of data responds only to the earliest R-Interest packet and replies R-ContentObject, i.e., the response packet for the R-Interest packet. As a result, the best route, which is expected to be the shortest path of all required functions and data, is selected from all combinations of required functions and data. The other unutilized routes will be invalidated with NAK (Negative Acknowledgement) packets.

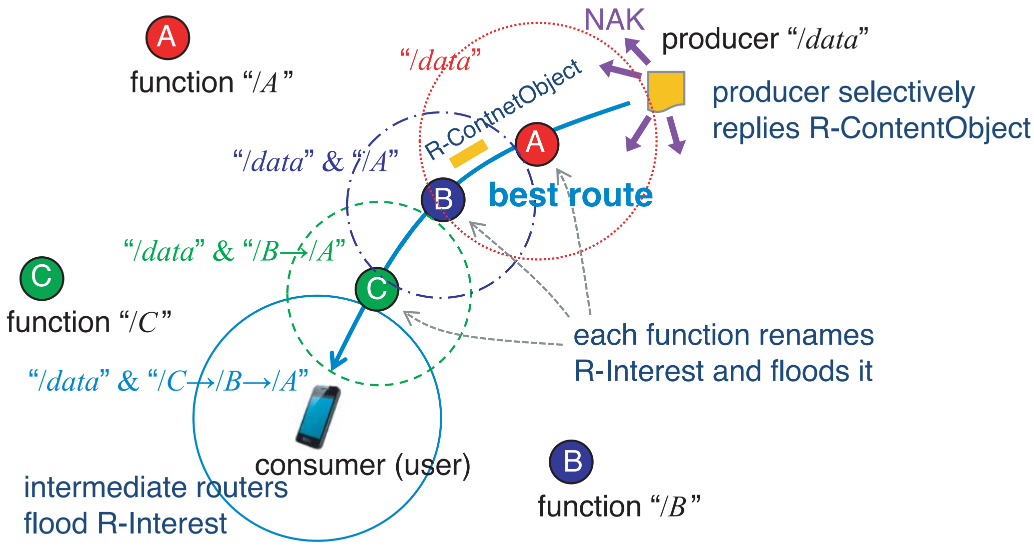


Figure 6.4: Overview of OR^3.

Algorithm 3 shows procedures of R-Interest packet processing at a router and a producer in OR^3. A route discovery request (R-Interest) r is expressed as $\{k_r, \mathcal{F}_r, C_r\}$. k_r denotes the required data and $\mathcal{F}_r$ denotes the set of required functions that R-Interest r has not chained yet. C_r is the ordered set of functions and denotes an order constraint among functions. In this chapter, we assume two kinds of conditions that there is no order constraint between functions, i.e., $C_r = \phi$, and there is an order constraint, e.g. $C_r = \{f_c, f_b, f_a\}$. Due to limitations of space, we only explain the ordered case. $\mathcal{R}_{k_r}$ is the set of nonce values of received R-Interest packets in producer of data k_r. $\mathcal{R}_{k_r}$ is used for identifying the earliest R-Interest packet and selectively replying ContentObject packets.

For a better understanding, we explain the procedures of OR^3 in the network and the application layer, respectively.

- Network layer

  When a router has received an R-Interest packet, the router looks up its PIT. If this router has been already discovered by R-Interest flooding, there should be an exact-matched PIT entry. In this case, the router discontinues the flooding and invalidates the discovered route by sending a NAK packet. Otherwise, the arrived R-Interest packet successfully discovers this router and flooding is continued to explore remaining parts of routes. At the same time, if function f_n executed on the application layer in this router is one of the functions to visit, i.e., $f_n \in \mathcal{F}_r$, the router replicates the R-Interest packet toward the function f_n to pick up a possible route. In such a way, OR^3 explores all possible routes by flooding search.

- Application layer

  When function f_n receives an R-Interest packet from the network layer, the function checks order constraints C_r of the R-Interest packet. If function f_n satisfies order constraints C_r, the function removes its name from the R-Interest and transmits the renamed R-Interest packet into the network layer. The renamed R-Interest packet will be flooded later by the network layer.

These procedures are continued until all required functions are successfully discovered. Finally an R-Interest packet $\{k_r, \phi, \phi\}$ that has chained all required functions arrives at a producer of data k_r. Since the earliest R-Interest packet by which all required functions are fully-chained means the best route of the function chaining, the producer replies an R-ContentObject packet only to this R-Interest. The other late arrivals are invalidated by NAK packets. During the R-ContentObject packet transmission, each intermediate router records route information of the R-ContentObject packet in RR. Detail of packet processing in a router is to be mentioned in section 6.2.2.

Algorithm 3 OR^3 algorithm.

Given: $k_r, \mathcal{F}_r, \mathcal{C}_r$
Initialize: $\mathcal{R}_{k_r} \leftarrow \phi$
- Procedures in a router ———————————
- - Network layer -
When a router receives an R-Interest packet $\{k_r, \mathcal{F}_r, \mathcal{C}_r\}$
if (there is an exact-match PIT entry for $\{k_r, \mathcal{F}_r, \mathcal{C}_r\}$) **then**
  detect Interest loop, and reply NAK
else
  if (there is an application face of function f_n) $\wedge$ ($f_n \in \mathcal{F}_r$) **then**
    replicate the R-Interest and forward it to function f_n
  else
    do not replicate the R-Interest
  end if
  flood the R-Interest to all network-device faces
end if
- - Application layer -
When a function f_n receives an R-Interest packet $\{k_r, \mathcal{F}_r, \mathcal{C}_r\}$
if (*SatisfyConstraints* $(f_n, \mathcal{C}_r)$ = true) **then**
  $\mathcal{F}_r \leftarrow \mathcal{F}_r \setminus f_n$
  $\mathcal{C}_r \leftarrow \mathcal{C}_r \setminus f_n$
  the renamed R-Interest $\{k_r, \mathcal{F}_r, \mathcal{C}_r\}$ is replied to the network layer
else
  discard the received R-Interest and reply NAK to the network layer
end if

SatisfyConstraints $(f_n, \mathcal{C}_r)$
if ($\mathcal{C}_r = \phi$) $\vee$ (f_n is the first function of processing order $\mathcal{C}_r$) **then**
  return true
else
  return false
end if

- Procedures in a producer ———————————
When a producer of data k_r receives an R-Interest packet r
if ($\mathcal{F}_r = \phi$) $\wedge$ ($r \notin \mathcal{R}$) **then**
  reply a ContentObject packet k_r
  add nonce value of R-Interest r to $\mathcal{R}$
else
  reply a NAK packet
end if

OR^3 in an example

Figures 6.5(a) and (b) show an example of route discovery in OR^3. Figure 6.5(a) is topology and Fig. 6.5(b) shows the sequence diagram of R-Interest and R-ContentObject packets. Function f_a and f_b are executed in node v_5 and v_2, respectively. The producer of data k_r is node v_6. The consumer is node v_1 and requests data k_r processed by function f_a and f_b in order of $f_a \rightarrow f_b$. R-Interest packets need to chain these functions in order of $f_b \rightarrow f_a$ since ContentObject packets is transferred on the reverse path of R-Interest packets along a PIT trail. Thus, the order constraint is $C_r = \{f_b, f_a\}$. An R-Interest packet includes all of these information, i.e., $\{k_r, \mathcal{F}_r, C_r\}$.

As shown in Fig. 6.6(a), when node v_2 receives the R-Interest $\{k_r, \mathcal{F}_r, C_r\}$, node v_2 replicates the R-Interest and forwards it to the function f_b on the application layer in order to find a possible route for required function chaining. The R-Interest is also flooded to the node v_3, v_4 and v_5 to explore other possible routes. In the application layer at node v_2, function f_b checks an order constraint of the R-Interest $\{k_r, \mathcal{F}_r, C_r\}$. Because f_b is the first element of C_r and meets the order constraint, function f_b removes its name from the R-Interest and the name is converted to $\{k_r, \mathcal{F}_r \setminus f_b, C_r \setminus f_b\}$. This renaming of Interest packets is for avoiding PIT aggregation with the original name packet of $\{k_r, \mathcal{F}_r, C_r\}$ in the network layer because Interest packets which have the same name are aggregated to one PIT entry in CCN/NDN router. The renamed R-Interest $\{k_r, \mathcal{F}_r \setminus f_b, C_r \setminus f_b\}$ is replied to the network layer and flooded into the network again.

Next, node v_5 receives the R-Interest $\{k_r, \mathcal{F}_r, C_r\}$ which has not visited function f_b from node v_2 as in 6.6(b). In this case, function f_a invalidates a part of the route discovered by this R-Interest packet because this R-Interest does not satisfy the order constraint. A little behind this time, node v_5 receives the R-Interest $\{k_r, \mathcal{F}_r, C_r\}$ from node v_3 in this example. This R-Interest is invalidated at the network layer because the corresponding PIT entry has been created by the previous Interest packet in node v_5. This means node v_5 has been already explored by the R-Interest $\{k_r, \mathcal{F}_r, C_r\}$. Therefore, node v_5 stops discovering the route for $\{k_r, \mathcal{F}_r, C_r\}$ and avoids Interest loop.

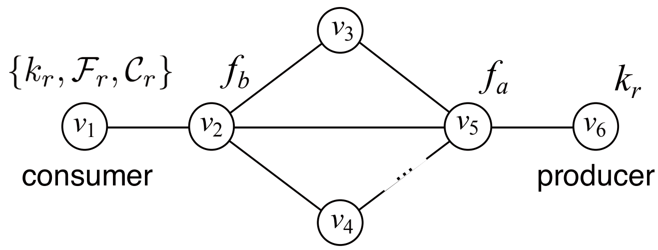
As shown in Fig. 6.6(c), an R-Interest $\{k_r, \mathcal{F}_r \setminus f_b, C_r \setminus f_b\}$ arrive at node v_5 from node v_2. The R-Interest is forwarded to function f_a and completes its function chaining because all required functions are fully chained successfully. The fully-chained R-Interest $\{k_r, \phi, \phi\}$ is transmitted from function f_a to the network layer and flooded again to explore the final destination, data k_r.

Finally, we explain procedures in node v_6, the producer of data k_r. As shown in Fig. 6.6(d), the producer may receive many kinds of R-Interest packets that have not chained all required functions until receiving the appropriate R-Interest $\{k_r, \phi, \phi\}$. The producer never responds to this kind of R-Interest packets other than R-Interest $\{k_r, \phi, \phi\}$. Only for the earliest R-Interest packet that has chained all required functions, the producer replies the ContentObject packet. Due to this, we can reduce redundant ContentObject packet transmission and only utilize the best route for data processing.

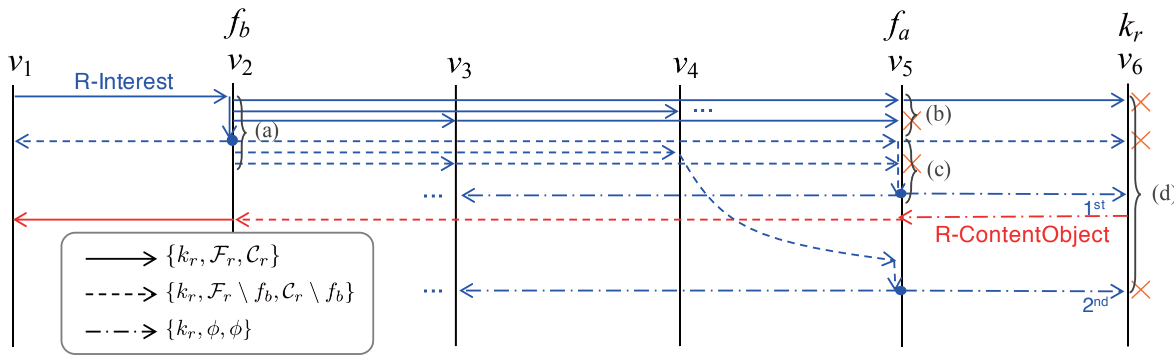
6.2.2 RR (Route Records)

RR is a forwarding table to reactively manage forwarding information of Interest packets at each router. This table is used for temporally caching route information of some destinations that have not been registered in FIB. Route information for function chaining discovered by OR^3 is usually stored in RR and reused especially for decreasing flooding overhead. This is because this kind of route information has high temporal locality. Stably-popular route information, which are utilized by many users, are stored in FIB, and not stably-popular but highly-temporal route information are temporally stored in RR. In such a way, we can manage where to save route information according to popularity of routes and suppress memory usage in a router.

Figure 6.7 shows packet forwarding procedure of Interest/ContentObject at a router.



(a) Topology.

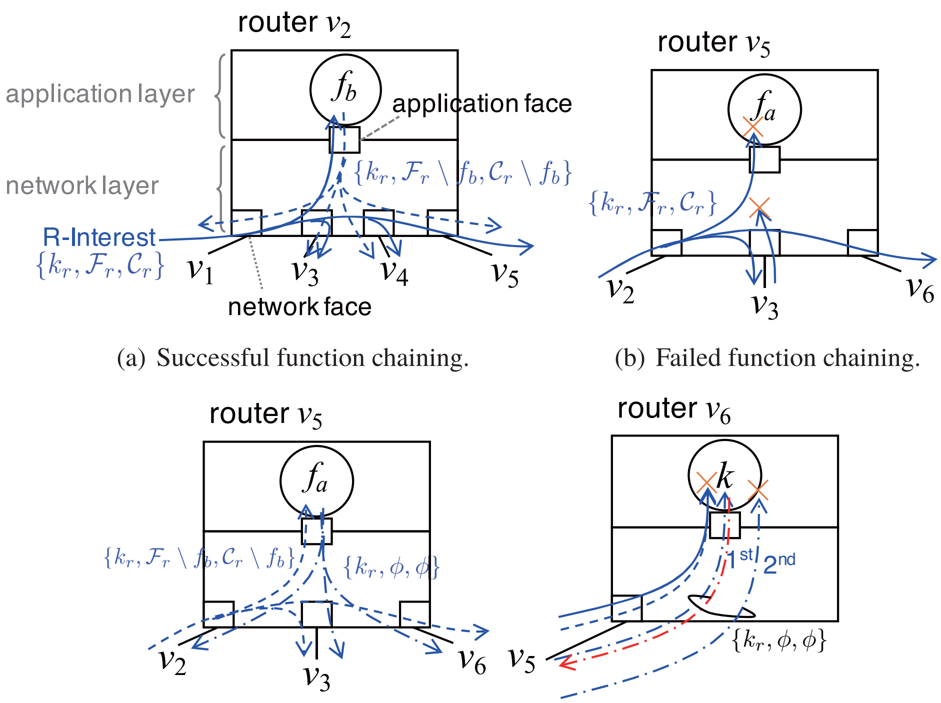


(b) R-Interest/R-ContentObject packet sequence.

Figure 6.5: Example of route discovery in OR³.

Dashed lines mean components of the original CCN/NDN, i.e., CS (Content Store), PIT, FIB. When the router receives an Interest packet and its name of Interest does not match any FIB entry, the Interest packet is translated to R-Interest and discover routers for data processing. When the router received R-ContentObject, the router records route information of the R-ContentObject packet into RR. Specifically, an entry of RR is created for the name of the R-ContentObject packet, and the input face of the R-ContentObject packet is registered as the output face of future R-Interest packets.

With this manner, we would like to leverage characteristics of “caching” to promote reuse of route information. In cache networks, caching natively makes popular contents stably stayed in the network because the popular contents are frequently downloaded by many users. Popular route information for function chaining is also to be spread in a similar fashion and stably retained in RR around the network. In contrast, unpopular route



(a) Successful function chaining. (b) Failed function chaining.

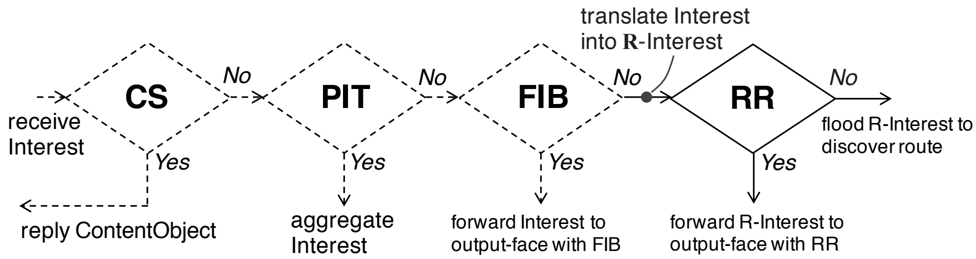
(c) All functions were chained. (d) Selective R-ContentObject reply.

Figure 6.6: Packet processing patterns inside routers.

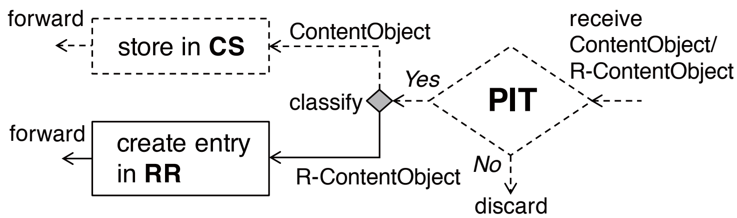
information is not distributed to the network. This effect can lead to efficient memory usage in routers and effective Interest forwarding.

6.3 Performance Evaluation

In this section, we evaluate performance of our proposal, OR^3 and RR. Our evaluation is twofold. Firstly, we compare our proposed reactive routing method, OR^3, with a proactive routing method in terms of delay and traffic load performance. Secondly, we investigate performance of RR and evaluate the effect of route information caching.



(a) Interest packet.



(b) ContentObject packet.

Figure 6.7: Packet forwarding procedures in a router.

6.3.1 Proactive vs. Reactive

First, we explain the evaluation model. We use ndnSIM 2.3 [67] as a simulation tool. Evaluation topology is BA model [2] with 100 nodes. The number of data is 1. A consumer requests the data provided by the one producer. The number of functions is 3, and 4 replicated functions are deployed in the network for each function. The producer, the consumer and all functions are placed to randomly selected nodes, and we conduct simulations in 1000 different patterns of the placement. The consumer requests data that is processed by three functions. This chapter mainly focuses on the evaluation in the case that there are no processing order constraints among the functions, i.e., Interest packets can chain the functions executed on routers in arbitrary order. Although it is easier to find a route that meets requirement of function chaining, the number of possible routes becomes larger compared with the cases that there are some processing order constraints. Thus, it is difficult to select the best route from all possible routes. Performance evaluation in

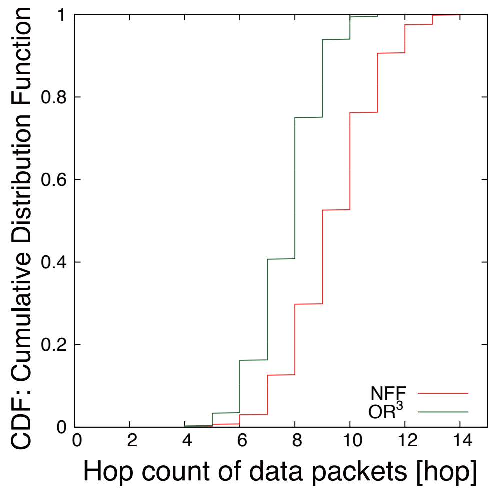


Figure 6.8: CDF of data hop count.

the case with processing order constraints is left to our further study. The comparison method is a proactive routing method, NFF as mentioned in section 6.1.3. In NFF, a router forwards Interest packets towards the nearest function for all required functions based on SPR without a global view. In other words, NFF is a local search approach that frequently falls into local optimal solutions.

The first evaluation metric is delay performance in terms of hop count. As shown in Fig. 6.8, total performance of OR^3 is improved compared to NFF. With NFF, function chaining occasionally falls into local optimal solutions in terms of total route length because the global view is not given to each router. OR^3 can discover the best route from all combinations of data and functions by flooding search, and our proposal always achieves the best performance throughout all patterns of function placement. Figure 6.9 shows *improvement* by OR^3. Definition of improvement is $\frac{h_i^{\mathrm{NFF}} - h_i^{\mathrm{OR}^3}}{h_i^{\mathrm{NFF}}}$, where $h_i^{\mathrm{NFF}}(h_i^{\mathrm{OR}^3})$ is data hop count of NFF(OR^3) in random placement i. As shown in Fig. 6.9, OR^3 totally shows better performance than NFF and decreases hop count by half at a maximum.

Then, we investigate traffic load characteristics. Since OR^3 floods control packets to

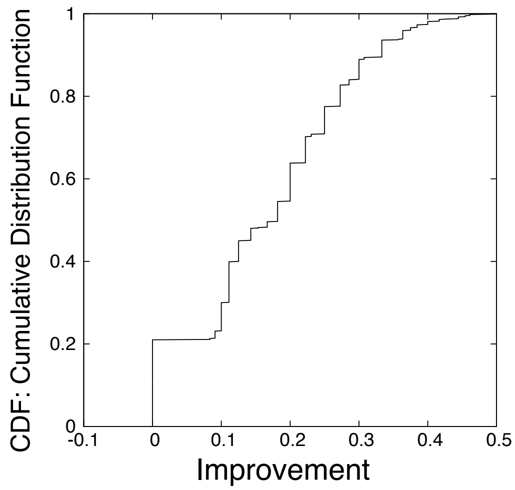


Figure 6.9: Improvement by OR^3.

explore the best route, our method may have large control overhead. Figure 6.10 shows traffic load characteristics for data size variation. Definition of traffic load is the sum of the products of a transmitted packet (Interest, NAK, or ContentObject) size and hop count of the packet. When data size is small, influence of flooding overhead is relatively large, and traffic load of OR^3 is higher than that of NFF. As data size increases, however, the influence of overhead decreases and becomes negligible especially around data size of 200 [KB]. This is because the average data hop count of NFF is larger than that of OR^3 (NFF = 9.37, OR^3 = 7.71). When data hop count is large, data packets detour and generate much traffic in the network. With OR^3, hop count decreases and data traffic is reduced. As a result, traffic load is relatively low for large data size.

6.3.2 Effect of Caching Route Information

In this section, we investigate effects of reactive caching of route information, RR, on network performance. Evaluated methods are $OR^3 without(w/o)$ RR and $OR^3 with(w/)$ RR. In OR^3 w/o RR, consumers always flood Interest packets to discover the best routes for data processing. With OR^3 $w/$RR, discovered route information is reactively cached in routers' RR and these information can be reused by other consumers' requests. We alter the number of consumers that request the same data processing from 1 to 64. Each consumer is placed to a randomly selected node. Other simulation parameters are the same as the previous evaluation.

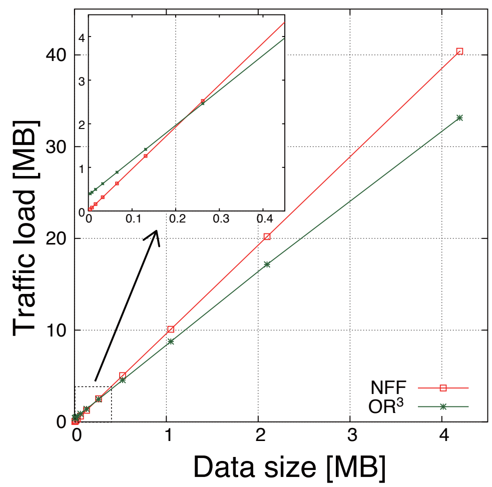


Figure 6.10: Traffic load vs. Data size.

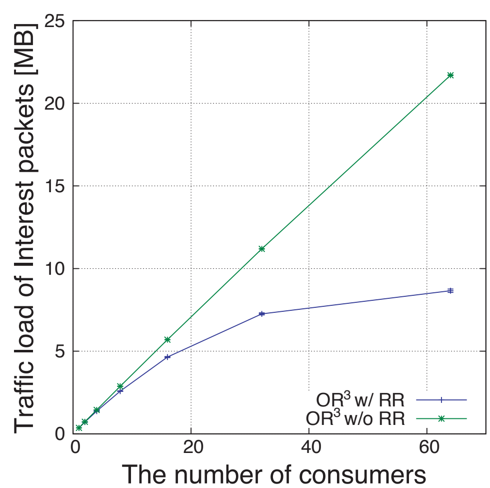


Figure 6.11: Traffic load of Interest flooding with/without RR.

Figure 6.11 shows traffic load characteristics for variation of the number of consumers. Traffic load for Interest flooding linearly increases with increase of the number of consumers in OR^3 w/o RR. In contrast, traffic load is saturated with OR^3 $w/$ RR because reuse ratio of route information increases according to the number of consumers. When a user node has sent an R-Interest packet and this node fortunately stores route information corresponding to the R-Interest packet in RR, flooding of R-Interest never starts from this node. Traffic load for flooding overhead is significantly reduced. This event repeatedly affects route discovery of other subsequent users. Therefore, with RR, popular route information would be distributed to the entire network and traffic load for flooding is significantly reduced. From these results, our proposal, the combination of OR^3 and RR is especially effective for popular route information and this combination enables scalable routing for ICN-based function chaining.

6.4 Summary

In IoT data processing, routing method that efficiently chains multiple functions is an important technical issue for ICN-based in-network processing. Our proposal, an on-demand routing method, discovers the end-to-end optimal route for any combination of required data and functions with low control overhead. Also, we introduced the concept that routers temporally cache route information and reactively manage them. The combination of these proposals realizes scalable ICN-based function chaining in IoT data processing.

In the next chapter, we conclude this thesis and describe the remaining technical issues for future works.

Chapter 7

Conclusions

Toward efficient information retrieval, CCN/NDN has been developed, and a lot of effort has been devoted by many ICN researchers thus far. However, there are a number of technical issues to be resolved for spread of CCN/NDN on implementation, security, etc.. The most important benefit of CCN/NDN is "in-network cache", which is the inherent feature of CCN/NDN. In order to utilize this effective resource as much as possible, traffic control scheme that steers data traffic where to transport is a key research topic. In this thesis, we made studies on traffic control schemes in ICN especially for basic technologies: route control, cache control and congestion control, and applied technologies: disaster communications and IoT data processing.

In Chapter 2, we evaluated several combinations of routing and caching and revealed the combination effects of routing and caching. For routing, we took up off-path cache routing, Breadcrumbs, as one of the content request routing proposals. When shortest path routing is adopted as the forwarding policy, cache decision policies, Fix(p), LCD, and ProbCache that distribute high-popular contents in a whole network shows high performance because content search for cached contents is limited to the on-path. However, when we use Breadcrumbs, the range of cache discovery is expanded to off-path caches. In TERC, not only high popular contents but also middle and low popular contents are cached. TERC makes effective bc-trail because all routers on this trail store the corresponding content. Therefore, we clearly reveal that combination of BC and TERC brings effective usage of off-path cached contents, which has not been deeply investigated thus far. Our detailed evaluation results in two network environments, emerging CCN/NDN and conventional IP, show these insights hold in both of these two network environments.

In Chapter 3, we proposed a cache decision policy based on betweenness centrality and content popularity, which is suitable for Breadcrumbs. Our proposed cache decision policy makes popular contents located edge area of a network, which stabilizes popular cached contents. Moderate popular contents tend to be stored in core area, which induces in-network guided requests to encounter cached contents more frequently. Unpopular contents are not likely to be cached in the network. As a result, frequency of cache replacement decreases and cache hit performance is improved due to the off-path cache guidance by BC forwarding. Our performance evaluation results revealed that the combination of Breadcrumbs and our proposed cache decision policy improves cache hit performance compared to existing cache decision policies.

These insights obtained from Chapters 2 and 3 imply that the combination of BC and our proposed caching policy enables to utilize ubiquitous caches in the network with multiple paths. This information-centric multi-source download brings various benefits. From the perspective of users, download throughput and robustness for link failures are improved. From the networking aspect, traffic load is reduced, and load balancing is achieved.

In Chapter 4, we assumed the information-centric environment where contents can be obtained from multiple sources with multiple paths, and then we changed our focus to congestion control to adequately steer transferred data traffic on multiple paths for congestion mitigation. MSC^{4}N, the previously proposed multipath congestion control in CCN/NDN, can utilize these multiple paths by adequately controlling congestion on each branched path. In multipath download, many users share highly-limited common network resources, thus fairness among the users sharing the network resources is required for transmission control. In Chapter 4, we discussed congestion control for CCN/NDN from the viewpoint of the resource pooling, which is a key concept for multipath resource sharing. MPTCP which is the most famous congestion control addressing the resource pooling in IP networks has an analogy with MSC^{4}N. Firstly, we discussed the analogy of window mechanism between our proposal and MPTCP. Also, in the performance evaluation, we have shown that MSC^{4}N satisfies the resource pooling concept and achieves macroscopic fairness among users at a certain level as well as MPTCP.

In Chapters 2 through 4, we studied basic technologies of traffic control in CCN/NDN to support efficient content distribution. Owing to these technologies, concrete traffic control schemes have been developed for information-centric communications where contents are downloaded from anywhere in the network. Hence, from Chapter 5, we shifted the focus of attention away from basic technologies to applied technologies to clearly show many benefits of applications of CCN/NDN.

Firstly, in Chapter 5, we made a study on an application to disaster communications, which is one of the most prospective applications. We proposed the new route control mechanisms for fragmented CCN in the disaster situation. Since original CCN assumes well-connected wired networks, it cannot be simply applied to the fragmented networks whose links are intermittent. We proposed a dynamic routing protocol and a packet forwarding method. The combination of these two techniques realizes energy efficient information retrieval in fragmented networks by recognizing the intermittent links. Also, we proposed a new cache decision policy suitable for the fragmented networks. The policy makes routers stably cache relatively popular contents by measuring incoming Interest packets. Simulation results show that our proposal can improve the performance and efficiency of information retrieval in terms of cache hit ratio, traffic load, energy consumption and content retrieval delay.

Next, in Chapter 6, we studied not only the case of emergency but also the case of data processing in IoT environment, which is a beneficial and proactive application of CCN/NDN. From the viewpoint of traffic control, it is important to control data processing path because large amount of collected data generate much traffic on the processing path in a network. Hence, in ICN-based in-network processing, routing method that efficiently chains multiple functions is an important technical issue. We proposed an on-demand routing method which discovers the end-to-end optimal route for any combination of required data and functions with low control overhead. Also, we newly introduced the concept that routers temporally cache route information and reactively manage them. The combination of these proposals realizes scalable ICN-based function chaining in IoT data processing.

In addition to the basic technologies described in Chapter 2-4, here in Chapters 5 and 6, we studied two applied technologies: application to disaster communications and application to IoT data processing. Robust communications are realized by the former applied technology because we can adaptively switch traffic control schemes according to condition of networks. In other words, we use the basic technologies as described in Chapters 2-4 in normal times and use the applied technology of Chapter 5 in an emergency. Also in Chapter 6, we extended our study on traffic control in IoT data processing. This extension brings high-speed and scalable data processing in IoT environment. The studies on these two applications would be powerful incentive for spreading CCN/NDN.

Future Works

In this section, we explain the future works on this thesis.

In Chapter 2, for cache control, we assumed that the size of all contents is identical for simplicity. We need to assume more realistic situations where the size of each content is different. Also, for route control, we only studied combination effects of route control and cache control. In order to operate CCN/NDN as a large-scale network system, we need to combine route control not only cache control but also congestion control. These three control schemes are comprehensively to be take into account.

In Chapter 3, we assumed that distribution of content popularity follows Zipf distribution. However, in the realistic situation, routers need to locally estimate content popularity by a measurement method, e.g. counting the number of received request packets. Also, we will conduct more evaluations in various network models because our approach is based on betweenness centrality, which is topological feature.

In Chapter 4, we only made a study on congestion control from the resource pooling viewpoint. As previously mentioned, we need to consider both route control and congestion control in order to find out combination effects among traffic control schemes. In addition, we plan to conduct simulations under the network environments which have heterogeneous communication quality.

In Chapter 5, we proposed a dynamic routing protocol that recognizes intermittent links in disaster situations. However, our study of this routing protocol is not sufficient because our proposed routing protocol has only assumed a scenario of single-source communications. Thus, we need to extend our proposal to multisource communications in disaster cases.

Chapter 6 addressed the best route selection in ICN-based function chaining for IoT data processing. Although our proposed routing method can select the end-to-end optimal route for data processing, this path might be unavailable in dynamic situations such as high-load processing and wireless environment. In the future works, we plan to modify our proposed routing method for these kinds of situations.

Bibliography

[1] B. Ahlgren, C. Dannewetz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Commun. Mag.*, vol.50, no.7, pp.26-36, Jul. 2012.

[2] R. Albert and A. Barabási, "Statistical mechanics of complex networks," *Rev. Mod. Phys.*, vol. 74, no. 1, pp. 47-97, 2002.

[3] M. Amadeo, C. Campolo and A. Molinaro, "Enhancing content-centric networking for vehicular environments," *Computer Networks*, vol. 57, no. 16, November 2013, pp. 3222-3234.

[4] C. Anastasiades, T. Schmid, J. Weber, and T. Braun, "Information-centric content retrieval for delay-tolerant networks," *Computer Networks*, In Press, Corrected Proof, Available online, Mar. 2016.

[5] S. Arianfar, P. Nikander, J. Ott, "On content-centric router design and implications," in *Proc. ACM ReArch'10*, Nov. 2010.

[6] M. Badov, A. Seetharam, and J. Kurose, "Congestion-Aware Caching and Search in Information-Centric Networks," in *Proc. ACM ICN'14*, Paris, France, September, 2014, pp. 37-46.

[7] C. Bernardini, T. Silverston, and O. Festor, “MPC: Popularity-based caching strategy for content centric networks,” in *Proc. IEEE ICC’13*, Budapest, June 2013, pp. 3619-3623.

[8] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, “Fog Computing and Its Role in the Internet of Things,” in *Proc. ACM MCC’12*, Helsinki, Finland, Aug. 2012, pp. 13-15.

[9] U. Brandes, “A Faster Algorithm for Betweenness Centrality,” *Journal of Mathematical Sociology*, vol. 25, no. 2, pp. 163-177, 2001.

[10] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, “Web Caching and Zipf-like Distributions: Evidence and Implications,” in *Proc. IEEE INFOCOM’99*, New York, NY, USA, March 1999, pp. 126-134.

[11] G. Carofiglio, M. Gallo, and L. Muscariello, “ICP: Design and Evaluation of an Interest Control Protocol for Content-Centric Networking,” in *Proc. IEEE INFOCOM NOMEN’12*, Orlando, FL, USA, Mar. 2012, pp. 304-309.

[12] G. Carofiglio, M. Gallo, and L. Muscariello, “Joint Hop-by-hop and Receiver-Driven Interest Control for Content-Centric Networks,” in *Proc. ACM ICN’12*, Helsinki, Finland, Aug. 2012, pp. 37-42.

[13] G. Carofiglio, M. Gallo, L. Muscariello, and M. Papalini, “Multipath Congestion Control in Content-Centric Networks,” in *Proc. IEEE INFOCOM WKSHPS’13*, Turin, Italy, Apr. 2013, pp. 363-368.

[14] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, “Delay-tolerant network architecture,” IETF RFC 4838, April 2007. `http://tools.ietf.org/html/rfc4838`

[15] W. Chai, D. He, I. Psaras and G. Pavlou, "Cache "less for more" in information-centric networks," in *Proc. IFIP Networking'12*, Prague, Czech Republic, May 2012, pp. 27-40.

[16] W. Chai, D. He, I. Psaras and G. Pavlou, "Cache "less for more" in information-centric networks." *Computer Commun.*, vol. 36, no. 7, pp. 758-770, Apr. 2013.

[17] D. Chiu, R. Jain, "Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks," *Computer Networks and ISDN Systems*, vol. 17, no. 1, pp. 1-14, June 1989.

[18] J. Choi, J. Han, E. Cho, T. Kwon, and Y. Choi, "A Survey on Content-Oriented Networking for Efficient Content Delivery," *IEEE Communications Magazine*, vol. 49, no. 3, pp. 121-127, March 2011.

[19] R. Chiocchetti, D. Rossi, G. Rossini, G. Carofiglio and D. Perino, "Exploit the Known or Explore the Unknown? Hamlet-Like Doubts in ICN," in *Proc. ACM ICN'12*, Helsinki, Finland, August 2012, pp. 7-12.

[20] R. Chiocchetti, D. Perino, G. Carofiglio, D. Rossi and G. Rossini, "INFORM: a dynamic INterest FORwarding Mechanism for Information Centric Networking," in *Proc. ACM ICN'13*, Hong Kong, China, August 2013, pp. 9-14.

[21] S. Eum, K. Nakauchi, M. Murata, Y. Shoji and N. Nishinaga, "CATT: Potential Based Routing with Content Caching for ICN," in *Proc. ACM ICN'12*, Helsinki, Finland, August 2012, pp. 49-54.

[22] S. K. Fayazbakysh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. M. Maggs, K. C. Ng, V. Sekar, and S. Shenker, "Less Pain, Most of the Gain: Incrementally Deployable ICN," in *Proc. ACM SIGCOMM'13*, Aug. 2013, pp. 147-158.

[23] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397-413, Aug. 1993.

[24] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure "TCP Extensions for Multipath Operation with Multiple Addresses," *RFC-6824*, Jan. 2013. `https://tools.ietf.org/html/rfc6824`

[25] G. Grassi, D. Pesavento, G. Pau, R. Vuyyuru, R. Wakikawa and L. Zhang, "VANET via Named Data Networking," in *Proc. IEEE INFOCOM'14 Workshop on Named Oriented Mobility*, Toronto, Canada, April 2014, pp. 410-415.

[26] M. Gritter and D. Chariton, "TRIAD: A new next-generation Internet architecture," July 2000. `http://gregorio.stanford.edu/triad/`

[27] A. Ioannou and S. Weber, "Towards On-Path Caching Alternatives in Information-Centric Networks," in *Proc. IEEE LCN'14*, Edmonton, Canada, September 2014, pp. 362-365.

[28] V. Jacobson, D. Smetters, J. D. Thornton, M. Plass, N. Briggs, and R. Braynard, "Networking Named Content," in *Proc. ACM CoNEXT'09*, Rome, Italy, December 2009, pp. 1-12.

[29] Y. Jin, W. Qu and K. Li, "A survey of cache/proxy for transparent data replication," in *Proc. IEEE SKG'06*, Nov. 2006, pp. 35.

[30] T. Koponen, M. Chawla, B. Chun, A. Ermolinskiy, K. Kim, S. Shenker and I. Stoica, "A Data-Oriented (and Beyond) Network Architecture," in *Proc. ACM SIGCOMM'07*, Kyoto, Japan, August 2007, pp. 181-192.

[31] M. Król and I. Psaras, "NFaaS: Named Function as a Service," in *Proc. ACM ICN'17*, Berlin, Germany, Sept. 2017, pp.134-144.

[32] N. Laoutaris, S. Syntila and I. Stavrakakis, "Meta Algorithms for Hierarchical Web Caches," in *Proc. IEEE ICPCC'04*, Bangalore, India, August 2004, pp. 445-452.

[33] N. Laoutaris, H. Che, and I. Stavrakakis, "The LCD interconnection of LRU caches and its analysis," *Performance Evaluation*, vol. 63, no. 7, pp. 609-634, July 2006.

[34] L. Liu, L. Xie, M. Bahrami, Y. Peng, A. Ito, S. Mnatsakanyan, G. Qu, Z. Ye, and H. Guo, "Demonstration of a Functional Chaining System Enabled by Named-Data Networking," in *Proc. ACM ICN'16*, Kyoto, Japan, Sept. 2016, pp. 227-228.

[35] L. Liu, L. Xie, M. Bahrami, Y. Peng, A. Ito, S. Mnatsakanyan, G. Qu, Z. Ye, and H. Guo, "ICN-FC: An Information-Centric Networking Based Framework for Efficient Functional Chaining," in *Proc. IEEE ICC'17*, Paris, France, May 2017.

[36] Y. Lu, X. Li, Y. Yu, and M. Gerla, "Information-Centric Delay-Tolerant Mobile Ad-Hoc Networks," in *Proc. IEEE INFOCOM Workshop on Name-Oriented Mobility 2014*, Toronto, Canada, Apr. 2014, pp. 428-433.

[37] V. Martina, M. Garetto, and E. Leonardi, "A unified approach to the performance analysis of caching systems," in *Proc. IEEE INFOCOM'14*, Tronto, Canada, May 2014, pp. 2040-2048.

[38] J. Miyoshi, S. Kawauchi, M. Bandai, and M. Yamamoto "Multi-Source Congestion Control for Content Centric Networks," in *Proc. ACM ICN'16*, Kyoto, Japan, Sept. 2016, pp. 205-206.

[39] T. Mori, H. Nakajima and M. Yamamoto, "A Study on In-Network Interest Guidance Method in Content-Centric Networking," in *Proc. IEICE Communications Society Conference'15, B-6-42*, Sept. 2015.

[40] H. Nakajima, T. Tsutsui, H. Urabayashi, M. Yamamoto, E. Rosensweig, and J. F. Kurose, "Performance Evaluation of Partial Deployment of an In-Network Cache Location Guide Scheme, Breadcrumbs," *IEICE Trans. Commun.*, vol. E99-B, no. 1, pp. 157-166, Jan. 2016.

[41] S. Oueslati, J. Roberts, and N. Sbihi, "Flow-aware traffic control for a content-centric network," in *Proc. IEEE INFOCOM'12*, Orlando, FL, USA, Mar. 2012, pp. 2417-2425.

[42] C. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," in *Proc. ACM SIGCOMM'94*, London, England, UK, 1994, pp. 234-244.

[43] S. Podlipnig and L. B. Osz, "A Survey of Web Cache Replacement Strategies," *ACM Computing Surveys*, vol. 35, no. 4, pp. 374-398, Dec. 2003.

[44] I. Psaras, W. K. Chai and G. Pavlou, "Probabilistic in-network caching for information-centric networks," in *Proc. ACM ICN'12*, Helsinki, Finland, August 2012, pp. 55-60.

[45] I. Psaras, L. Saino, and G. Pavlou, "Revisiting Resource Pooling: The Case for In-Network Resource Sharing," in *Proc. Hotnets'14*, Los Angeles, CA, USA, Oct. 2014, pp. 1-7.

[46] C. Raiciu, D. Wischik, and M. Handley, "Practical Congestion Control for Multipath Transport Protocols," *UCL Technical Report*, pp. 1-13, Jan. 2009.

[47] E. Rosensweig and J. Kurose, "Breadcrumbs: Effcient, Best-Effort Content Location in Cache Networks," in *Proc. IEEE INFOCOM'09*, Rio de Janeiro, Brazil, April 2009, pp. 2631-2635.

[48] G. Rossini and D. Rossi, "Coupling Caching and Forwarding Benefits Analysis and Implementation," in *Proc. ACM ICN'14*, Paris, France, September 2014, pp. 127-136.

[49] N. Rozhnova and S. Fdida, "An effective hop-by-hop Interest shaping mechanism for CCN communications," in *Proc. IEEE INFOCOM WKSHPS'12*, Orlando, FL, USA, Mar. 2012, pp. 322-327.

[50] L. Saino, C. Cocora, and G. Pavlou, "CCTCP: A Scalable Receiver driven Congestion Control Protocol for Content Centric Networking," in *Proc. IEEE ICC'13*, Budapest, Hungary, Jun. 2013, pp. 3775-3780

[51] M. Satyanarayanan, P. Bahl, R. Cáceres, and N. Davies, "The Case for VM-Based Cloudlets in Mobile Computing," *IEEE Pervasive Computing'09*, vol. 8, no. 4, pp. 2-11, Oct.-Dec. 2009.

[52] K. Scott and S. Burleigh, "Bundle protocol specification," IETF RFC 5050, November 2007. `http://tools.ietf.org/html/rfc5050`

[53] M. Sifalakis, B. Kohler, C. Scherb, and C. Tschudin, "An Information Centric Network for Computing the Distribution of Computations," in *Proc. ACM ICN'14*, Paris, France, Sept. 2014, pp. 137-146.

[54] I. Stoica, D. Adkins, S. Zhuang, S. Shenker and S. Surana, "Internet indirectioin infrastructure," in *Proc. ACM SIGCOMM'02*, Pittsburgh, PA, USA, August 2002, pp. 73-86.

[55] A. Tagami, T. Yagyu, K. Sugiyama, M. Arumaithurai, K. Nakamura, T. Hasegawa, T. Asami, and K. Ramakrishnan, "Name-based Push/Pull Message Dissemination for Disaster Message Board," in *Proc. IEEE LANMAN'16*, Rome, Italy, June 2016.

[56] C. Tschudin and M. Sifalakis, "Named Functions and Cached Computations," in *Proc. IEEE CCNC'14*, Las Vegas, NV, USA, Jan. 2014, pp. 851-857.

[57] D. Wischik, M. Handley, and M. B. Braun, "The Resource Pooling Principle," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 5, pp. 42-52, Oct. 2008.

[58] M. Yamamoto, "[Invited Paper] A Survey of Caching Networks in Content Oriented Networks," *IEICE Trans. Commun.*, vol. E99-B, no. 5, pp. 961-973, May 2016.

[59] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. Thornton, D. Smetters, B. Zhang, G. Tsudik, K. Claffy, D. Krioukov, D. Massey, C. Papadopoulos, T. Abdelzaher, L. Wang, P. Crowley, and E. Yeh, "Named Data Networking (NDN) Project," *PARC Technical Report* 2010-003, Oct. 2010.

[60] BRITE, `http://www.cs.bu.edu/brite/`

[61] `https://tools.ietf.org/html/draft-irtf-icnrg-ccnxsemantics-03`

[62] Cisco, "Cisco Virtual Networking Index," June 2017. `https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html`

[63] "FG IMT-2020: Report on Standards Gap Analysis," TD 208 (PLEN/13), ITU-T SG13, 2015.

[64] Ministry of Internal Affairs and Communications, "WHITE PAPER 2011 Information and Communication in Japan," `http://www.soumu.go.jp/johotsusintokei/whitepaper/eng/WP2011/2011-index.html`

[65] ns-3, network simulator 3. `http://www.nsnam.org/`

[66] A. Afanasyev, I. Moiseenko and L. Zhang, "ndnSIM: NDN simulator for NS-3," *NDN Technical Report NDN-0005*, 2012. `http://ndnsim.net/1.0/`

[67] S. Mastorakis, A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM 2.0: A new version of the NDN simulator for NS-3," NDN, Technical Report NDN-0028, 2015.

List of Publications

A. Refereed Jornal Papers

1. Y. Hayamizu, A. Shibuya, and M. Yamamoto, “The Combination Effect of Cache Decision and Off-Path Cache Routing in Content Oriented Networks,” *IEICE Trans. Communications*, vol. E102-B, no. 5, May 2019. (to be published)

2. Y. Hayamizu, T. Yagyu, and M. Yamamoto, “Energy Efficient Information Retrieval for Content Centric Networks in Disaster Environment,” *IEICE Trans. Communications Special Section on Information Centric Networking: Paradigms, Technologies, and Applications*, vol. E99-B, no. 12, pp. 2509-2519, December 2016. DOI: 10.1587/transcom.2016CNP0003

B. Refereed Conference Papers

1. Y. Hayamizu, K. Otsuka, M. Bandai, and M. Yamamoto, “Resource Pooling in Multipath Congestion Control for Content Centric Networks,” in *Proc. 2018 IEEE Global Communications Conference (GLOBECOM 2018)*, Abu Dhabi, UAE, December 2018. (to be presented)

2. Y. Hayamizu, A. Nagata, and M. Yamamoto, "On-Demand Routing for Chaining Multiple Functions in ICN-Based In-Network Processing," in *Proc. 2018 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN 2018)*, Washington, DC, USA, June 2018, pp. 7-12. DOI: 10.1109/LANMAN.2018.8475111 **(Best Paper Award)**

3. Y. Hayamizu, K. Hirata, and M. Yamamoto, "CCAR: Caching and Content-Aware Routing for Content Oriented Networking," in *Proc. 2018 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR 2018)*, Austin, TX, USA, May 2018, pp. 1-6. DOI: 10.1109/CQR.2018.8445945

4. Y. Hayamizu, A. Shibuya, and M. Yamamoto, "Effective New Cache Decision Policy for Breadcrumbs in Content-Centric Networking," in *Proc. 2017 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR 2017)*, Naples, FL, USA, May 2017, pp. 1-6. DOI: 10.1109/CQR.2017.8289446 **(Best Paper Award)**

5. A. Shibuya, Y. Hayamizu, and M. Yamamoto, "Cache Decision Policy for Breadcrumbs in CCN," in *Proc. 2016 IEEE Globecom Workshops (GC Wkshps 2016)*, Washington, DC, USA, December 2016, pp. 1-6. DOI: 10.1109/GLOCOMW.2016.7848851

6. Y. Hayamizu, T. Yagyu, and M. Yamamoto, "Energy and Bandwidth Efficient Content Retrieval for Content Centric Networks in DTN Environment," in *Proc. 2015 IEEE Globecom Workshops (GC Wkshps 2015)*, San Diego, CA, USA, December 2015, pp. 1-6. DOI: 10.1109/GLOCOMW.2015.7414126

7. Y. Hayamizu and M. Yamamoto, "Receiver-driven Congestion Control for Content Oriented Applications with Multiple Sources," in *Proc. 2015 IEEE International*

Workshop Technical Committee on Communications Quality and Reliability (CQR 2015), Charleston, SC, USA, May 2015, pp. 1-6. DOI: 10.1109/CQR.2015.7129085

C. Non-Refereed Technical Papers

1. A. Nagata, Y. Hayamizu, and M. Yamamoto, “A Study on ICN-based In-Network Processing,” in *Proc. 2018 IEICE General Conference*, BP-2-2, Tokyo, Japan, March 2018.

2. Y. Hayamizu, A. Nagata, and M. Yamamoto, “Performance Evaluation of ICN-based Multi-Function Routing in Mobile Ad Hoc Network,” in *Proc. 2018 IEICE General Conference*, B-6-47, Tokyo, Japan, March 2018.

3. Y. Hayamizu and M. Yamamoto, “A Study on the Influence of Spatial Locality of Content Popularity in CCN with In-Network Guidance,” *IEICE Technical Report*, vol. 117, no. 459, NS2017-204, pp. 209-214, Miyazaki, Japan, March 2018.

4. K. Otsuka, Y. Hayamizu, M. Bandai, and M. Yamamoto, “Performance Evaluation of CCN Multipath Congestion Control from Resource Pooling Viewpoint,” *IEICE Technical Report*, vol. 117, no. 459, NS2017-203, pp. 203-208, Miyazaki, Japan, March 2018.

5. Y. Hayamizu, A. Nagata, and M. Yamamoto, “Performance Evaluation of ICN-based Multi-Function Routing under Processing Order Constraints,” in *Proc. 2017 IEICE Society Conference*, B-6-21, Tokyo, Japan, September 2017.

6. Y. Hayamizu, A. Nagata, and M. Yamamoto, “Novel Routing Strategy for Multiple Functions in ICN-based In-Network Processing,” *IEICE Technical Report*, vol. 117, no. 204, NS2017-86, pp. 91-96, Miyagi, Japan, September 2017.

7. Y. Hayamizu, K. Hirata, and M. Yamamoto, “Performance Evaluation of Caching and Content-Aware Routing [Poster Session],” *IEICE Technical Report*, CQWS3-03, pp. 26, Osaka, Japan, January 2017.

8. Y. Hayamizu, A. Nagata, and M. Yamamoto, “Multi-Function Routing for In-Network Processing in Information-Centric Networking,” *IEICE Technical Committee on Information-Centric Networking*, Hiroshima, Japan, December 2016.

9. A. Shibuya, Y. Hayamizu, and M. Yamamoto, “A Study on In-Network Guidance and Cache Decision Policy in CCN,” *IEICE Technical Report*, vol. 116, no. 111, NS2016-30, pp. 13-18, Hokkaido, Japan, June 2016.

10. Y. Hayamizu, T. Yagyu, and M. Yamamoto, “Performance Evaluation of Cache Decision Policy for Fragmented Content Centric Networks,” in *Proc. 2016 IEICE General Conference*, B-6-122, Fukuoka, Japan, March 2016.

11. Y. Hayamizu, K. Hirata, and M. Yamamoto, “Content-Oriented Traffic Engineering with Content Caching,” *IEICE Technical Report*, vol. 115, no. 404, NS2015-165, pp. 109-114, Fukuoka, Japan, January 2016.

12. Y. Hayamizu, T. Yagyu, and M. Yamamoto, “Retransmission Control for Content-Centric Networking in DTN Environment,” *IEICE Technical Report*, vol. 114, no. 297, NS2014-145, pp. 99-104, Kochi, Japan, November 2014.

D. Miscellaneous

1. Y. Hayamizu, M. Yamamoto, E. Rosensweig, and James F. Kurose, “Multiple-Breadcrumbs: A New In-Network Guidance for Off-Path Cache in Cache Networks,”

IEICE Trans. Communications, vol. E101-B, no. 12, December 2018. (to be published)

2. Y. Hayamizu, A. Nagata, and M. Yamamoto, "A Study on Reactive ICN-Routing for IoT Data Processing [Poster Session]," in *Proc. the 13th International Symposium in Science and Technology at Cheng Shiu University 2018 (ISST 2018)*, Kaohsiung, Taiwan, August 2018, pp. 96. **(Poster Presentation Award)**

3. Y. Hayamizu, A. Nagata, and M. Yamamoto, "Named-Function Chaining for IoT Data Processing [Poster Session]," in *Proc. the 4th International Symposium on Electrical Engineering and Computer Science (ISEECS 2016)*, Osaka, Japan, September 2017, pp. 117-118. **(Best Poster Presentation Award)**

4. A. Shibuya, Y. Hayamizu, and M. Yamamoto, "Cache Decision Policy for Breadcrumbs in CCN [Poster Session]," in *Proc. the 11th International Symposium in Science and Technology at Kansai University 2016 (ISST 2016)*, Osaka, Japan, July 2016, pp. 211.

5. Y. Otsuji, Y. Hayamizu, and M. Yamamoto, "Random Order Content Request in Content-Oriented Networking [Poster Session]," in *Proc. the 11th International Symposium in Science and Technology at Kansai University 2016 (ISST 2016)*, Osaka, Japan, July 2016, pp. 209.

6. Y. Hayamizu, T. Yagyu, and M. Yamamoto, "Information-Centric Content Delivery in the Aftermath of Disasters [Poster Session]," in *Proc. the 11th International Symposium in Science and Technology at Kansai University 2016 (ISST 2016)*, Osaka, Japan, July 2016, pp. 208. **(Poster Presentation Award)**

E. Awards

1. The 13th International Symposium in Science and Technology at Cheng Shiu University 2018 (ISST 2018) Poster Presentation Award, August 2018.

2. The 24th IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN 2018) Best Paper Award, June 2018. (acceptance rate $\simeq$ 36%)

3. The 4th International Symposium on Electrical Engineering and Computer Science 2017 (ISEECS 2017) Best Poster Presentation Award, September 2017.

4. IEEE Communications Quality and Reliability 2017 (CQR 2017) Best Paper Award, May 2017. (acceptance rate = 40%)

5. The 13th IEEE Kansai Section Student Paper Award, February 2017.

6. The 11th International Symposium in Science and Technology at Kansai University 2016 (ISST 2016) Poster Presentation Award, July 2016.