Improved Content Finding in Named Data Networking

Atthapol Suwannasa
School of Computing and Communications
Lancaster University

A thesis submitted for the degree of
Doctor of Philosophy

December, 2021

Declaration

I declare that the work presented in this thesis has not been submitted, either in whole or in part, for a degree at this, or any other university, and that the work is entirely my own.

Atthapol Suwannasa

Improved Content Finding in Named Data Networking

Atthapol Suwannasa
School of Computing and Communications, Lancaster University

A thesis submitted for the degree of *Doctor of Philosophy*. December, 2021

Abstract

In today's Internet, the current architecture may not be able to support various challenges (e.g., security, mobility, scalability, and quality of service) in a sufficient level. Information-centric communication model is expected to address the bottleneck of the traditional host-centric model. A number of Information Centric Network (ICN) approaches have been proposed by aiming to replace or augment the current host-to-host routing architecture. ICN focuses on finding and transmitting content to end-users and content routing is location-independent, thereby being able to support multi-sourcing for content consumers.

Named Data Networking (NDN) is one of the promising ICN proposals that allows users (i.e., consumers) to find content objects by their names. In the default forwarding strategy of NDN, an interest packet is forwarded to locate content. A corresponding data packet will be returned back in the reverse path to its requester and will be replicated along this path (called on-path caching). When a consumer requests a content object, it may be found at an intermediate on-path cache. However, several replicas that are often cached off-path especially in nearby nodes of the consumer's vicinity could be the better potential source but they are not effectively utilised, causing a worse than necessary delivery efficiency.

Therefore, this thesis investigates the potential of off-path content finding in NDN. We examine how we can design a flexible and efficient solution to supplement the existing NDN architecture. We then propose a new design called a *Vicinity-based Content Finding* scheme *(VCoF)* to utilise nearby replicas in each vicinity for improving content finding. This includes analysing the efficiency of the proposed scheme in comparison to default NDN. We consider content popularity, which can impact content finding results due to the different number of content replicas (i.e., content availability). We also explore our scheme in supporting mobility, particularly for the issues of missing content because of handover. Through a prototype implementation, we evaluate the delivery efficiency against overhead costs in different scenarios, made possible through effective deployment on real NDN environments.

Publications

A. Suwannasa, M. Broadbent and A. Mauthe. Vicinity-based Replica Finding in Named Data Networking. In *2020 34th International Conference on Information Networking (ICOIN)*, pages 146-151. IEEE, 2020.

A. Suwannasa, M. Broadbent and A. Mauthe. Impact of Content Popularity on Content Finding in NDN: Default NDN vs. Vicinity-based Enhanced NDN. In *2020 10th International Conference on Information Science and Technology (ICIST)*, pages 163-168. IEEE, 2020.

A. Suwannasa, M. Broadbent and A. Mauthe. Effectiveness of Vicinity-based Content Finding in Mobile NDN Environments. In *2021 35th International Conference on Information Networking (ICOIN)*, pages 343-348. IEEE, 2021.

Acknowledgements

First and foremost, I am deeply grateful to my supervisors, Professor Andreas Mauthe and Dr Matthew Broadbent. Throughout my PhD study in the past few years, their insightful comments and plentiful experience have provided me the guidance and encouragement necessary for me to succeed. I also gratefully acknowledge the support of the Royal Thai Government in providing a fully funded scholarship during the study.

I would like to thank my parents, Santisak and Sakorn Suwannasa for supporting me spiritually. Many thanks also go to my good friends and the rest of my family, whom without this would have not been possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Named Data Networking (NDN) is an Information-Centric Networking (ICN) paradigm that is one of the most promising architectural approaches for the Future Internet [1, 2]. In the current Internet, the Domain Name System (DNS) is used to translate numeric IP addresses into domain names to make it easier for people to locate data. Packets are delivered to destinations indicated by IP addresses but content objects are served according to their names. Therefore, the main design of NDN is to eliminate the standard IP addresses by allowing users to directly locate content objects by their names. To find content, requesting packets are forwarded in default forwarding paths indicated by the names of the desired content objects. These objects can be replicated in several caches along the default paths between consumers and content producers (called on-path caching) in the network. Thereby, content can be located from multiple sources.

There are several challenges in the original design of the current Internet architecture. The traditional host-to-host routing infrastructure and its practical implementation have been identified as a cause of several limitations [3]. By using NDN, data can be routed from multiple sources (i.e., caches) due to the caching function. According to this concept, NDN is potentially an appropriate solution to overcome the lacks of the existing point-to-point communication and the current limitations of the TCP/IP Internet [4].

NDN nodes including consumers, routers, and producers are the key elements in content delivery chains. The consumers send interest packets to look for desired content objects that have been provided by their producers. The routers (i.e., nodes or devices) are responsible for two main tasks: caching/seeking content in their caches

and forwarding both incoming and outgoing packets. In each router, a Face represents a network interface, which is mapped with content names. A name is exchanged among NDN nodes to advertise its reachability and its location. An NDN forwarding strategy selects a proper Face for a single-path strategy, or several Faces for a multi-path strategy, to forward an interest packet to find its corresponding content object indicated by its name.

The study of [5] proves that content caching can significantly reduce path lengths for retrieving content since interest packets can be served closer to requesters. Content caching can provide more content availability [6], faster delivery, and lower congestion on long links [7]. Hence, it can be seen that content caching can, indeed, bring content close to end-users, through which several benefits can be gained especially in terms of delivery efficiency.

The default best route strategy [8] of NDN selects a Face mapped with the lowest cost to find a shortest path for locating a content object according to its name. However, the best direction to find the object is, perhaps, not on the default path [9]. The content replicas might be close to the requesters and these replicas could be the better sources for consumers. Notably, the default strategy, by design, could not naturally find nearby replicas located off the default path. Hence, this might not take appropriate advantage of the surrounding replicas, causing sub-optimal delivery efficiency.

Content replicas are replicated within a network regardless of a default forwarding path and they are often cached nearby like in a vicinity (also dependent on their density). Several existing NDN source selection strategies locate a desired content object according to its name without considering off-path or especially nearby replicas, which can be the cause of insufficient delivery efficiency. A number of proposals try to find off-path replicas (discussed in Section 2.5). However, particular shortcomings of these solutions can decrease their effectiveness. For example, by mapping consumers with producers or caching content at rendezvous points or Name Resolution Systems (NRSs), multiple centralised mapping systems incur higher control overhead and a single point of failure. Although an interest flooding strategy might help to find nearby replicas, expensive overhead costs can be incurred. In addition, several data chunks might be generated in return and thereby increasing cache eviction rates.

Content replicas could be available in a particular vicinity of a requester and they can be reachable through different paths. When other requesters in the same vicinity try to look for the same replicas, they can gain benefits (especially in terms of content

delivery) looking for the nearby replicas instead of the original content that might be located further away. Therefore, the work presented in this thesis proposes a content finding strategy (called *Vicinity-based Content Finding* scheme, or *VCoF*) to utilise nearby replicas, that can be the better sources for a consumer. The scheme expands the consumer's view into a vicinity to gain more advantage in nearby content finding. By centering a consumer node, a vicinity contains the set of NDN nodes that are connected to the consumer in different distances. A distance is the number of hops that defines the scope of the vicinity, called vicinity size. This is also crucial to limit and mitigate the overheads of content finding.

The main contribution of this work is the improvement of content finding in NDN using the proposed VCoF scheme. To first understand advantages and disadvantages of the vicinity-based strategy, the designed scheme is developed and investigated. In the first instance, a realistic-like NDN system with static connections is considered to evaluate VCoF to understand delivery efficiency against additional overheads. The investigation of the content popularity is then explored since different popularity levels of content objects/replicas can impact different content finding results. Finally, content finding in the context of mobility could also be challenging and this is also investigated. For example, when a mobile node is moving, some particular contents/replicas might be failed to deliver due to handover. The previous location and the current location of the mobile node are often in the same vicinity. Though if there is a significant jump between the previous and the current location, other nodes in the vicinity and other nearby vicinities can still be alternate sources. Hence, the VCoF scheme can also be beneficial under this mobile environment.

1.1 Research Hypothesis

Due to the rapid growth of the number of devices and the increment of content demand, several challenges in the current Internet architecture (e.g., delivery performance, content distribution problems, and dynamic communication) need to be addressed. Delivering content or enhancing content distribution in the current point-to-point communication protocol can be error-prone and complex.

Named Data Networking (NDN) is a promising alternative that could increase content delivery efficiency. NDN changes the paradigm of the current architecture by delivering a packet to its destination identified by a given name. Although NDN can mitigate or eliminate some issues of the existing architecture, a number of challenging

benefits need to be explored. One of these is the location of content independent from original producers. Each NDN node can replicate content from the original producers and allows requesters to retrieve their desired content from multi-points.

However, several existing NDN routing strategies try to locate a content object according to its name but the delivery efficiency could be sub-optimal, since these do not consider nearby replicas. A number of proposals try to locate off-path or nearby replicas. Nevertheless, their drawbacks can outweigh their benefits (e.g., single point of failure by using resolution-based routing, cache eviction by using flooding-based name routing, and overhead problems). Hence, a suitable content finding (i.e., source selection) strategy would be the possible solution to increase the delivery efficiency and mitigate the existing drawbacks. Getting some information (e.g., content availability) within a neighborhood or a node in the vicinity could help to indicate a consumer to fetch a content object from a better source.

Nevertheless, there are a number of other factors that need to be considered, for instance content density can also affect content finding results. For example, if a consumer try to look for a desired content object and its density (i.e., availability or copies of the content) in the consumer's network is high, the opportunity that a replica of this desired content could be cached nearby would also be high. Thereby, this increases the opportunity that the desired object can be located faster. Further to this, additional overhead in finding content must also be considered to trade off with the efficiency of a content finding solution.

An appropriate source selection strategy can also be beneficial in the context of mobility. For instance, when a source of a content object is going off-line, the proper strategy could still help to find the replica of the object that might be already cached nearby. In several mobile environments, a desired content object is often forwarded to a previous location of a mobile node. The requested object can be missing due to the disconnection during handover. A suitable content finding solution can also help to locate the missing content that is frequently cached close to the mobile node (e.g., in the previous location or other nodes in the vicinity).

Therefore, in our research hypothesis, it is possible to develop a new content finding or source selection strategy to be integrated with the existing NDN architecture to improve delivery efficiency. Nearby content replicas are expected to be the better potential sources for consumers. However, a number of factors need to be investigated in the consideration of nearby content finding. For example, different content densities can impact different content finding results. Overhead costs especially in the context

of locating content generated by the extra solution should be considered to trade off with delivery efficiency. In addition, finding nearby content can also be beneficial in a number of mobile communication models.

1.2 Research Questions

In order to show the validity of the research hypothesis, we need to address the following 4 main research questions while considering different research methods (e.g., analytical research, analysis of requirements, design of appropriate mechanisms as well as practical implementation).

1. Can the content finding of default NDN be improved by considering the vicinity of the consumer?
2. Can the delivery performance of default NDN be increased through the involvement of nearby nodes, and if yes by how much?
3. Does the improvement of content finding create additional overheads, and if so how high are these?
4. What are the effects of content popularity (or replica density) and mobility on the performance and overhead costs of content finding?

1.3 Methodology

This PhD research aims to improve content finding in NDN by taking advantages of replicas that are cached nearby. The highlight of the research methodology is presented as itemised below.

1. **Identify the problem space of off-path content finding and options to improve content finding in NDN.** In Chapter 2, we will describe a set of problems of content finding in NDN. This includes the off-path content finding issues in the existing strategies as well as in a number of related solutions. Further to this, in Chapter 3, the problem identification can help to design a better solution by avoiding the existing problems and shortcomings of several existing forwarding strategies.

2. **Design and implement a new scheme (entitled *"VCoF"*) to improve content finding in NDN.**

   In the current NDN architecture (i.e., default NDN), an interest packet is forwarded to locate a content object indicated by its name prefix. It is difficult to locate off-path replicas that are already cached nearby since there is no knowledge about these nearby replicas. Hence, in this thesis, we will research a new scheme to proactively advertise the availability of content objects and their replicas to enable a consumer to locate the nearby objects according to the advertised information. This technique can help an end node to know the locations of the nearby replicas. The received information should indicate proper paths to find the nearby objects, which means they can be fetched closer instead of going to further producers.

   When a content object is located nearby, there should be another mechanisms to forward the content object back to its requester. Hence, in this thesis, we will also research mechanisms of data (i.e., content) routing. When the interest packet has located the nearby replica in a vicinity, the corresponding data packet should be routed in a shorter path compared to the path to the original producer. According to NDN's stateful forwarding, an intelligent forwarding strategy can be deployed and we can keep track interest/data packets to ensure the success of content delivery regarding to delivery efficiency. Due to the concept of *Vicinity-based Content Finding*, entitled *VCoF*, this will aim to encompass the research of this work. Chapter 3 of this thesis presents the design of VCoF and the implementation is detailed in Chapter 4.

3. **Analyse and evaluate the designed strategy by using appropriate tools to determine how big the improvement is in terms of delivery efficiency.** To indicate how well VCoF can improve content finding in NDN, the evaluations of the design in various aspects are crucial. Although the VCoF scheme might help to increase delivery efficiency, excessive overhead costs can be problematic. Hence, we will examine the performance of content delivery against additional overhead costs. The overhead costs should be acceptable while introducing a better performance of content delivery. The trade-off between the delivery efficiency and the overhead costs must be explored in order to find a balance point. This is presented in Chapter 5.

4. **Investigate how VCoF performs considering different conditions such as content popularity or replica density.** Content popularity/density is another important factor that can affect content finding results. For example, when desired content objects are less populated, it means the density of the desired objects is low. So, the opportunity to find the nearby content objects is also low, resulting in higher delivery delays. When the network is highly populated with the desired content objects, the opportunity to find the nearby replicas is high. So, the delivery performance should be increased. Hence, to understand the effects of content popularity/density, in Chapter 5, this is examined by comparing the VCoF scheme to the default best route strategy of standard NDN.

   Due to the increasing number of objects/replicas, cache replacement rates are another important factor that can affect the content finding results. For example, when a cache is quite large, the replacement rate is low since replicas can be remained in the cache for longer. So, the opportunities to find the replicas are higher compared to a smaller cache size. In this thesis, we assume that the replicas of desired content are available to be located but cache replacements might decrease the replica availability. So, this issue will also be investigated in the evaluation chapter.

5. **Investigate the effects and performance of VCoF in the case of mobility.** According to the concept of VCoF, the area and also directions of content finding is expanded. So, this can increase the opportunity that a desired content object that is already cached nearby can be found faster, resulting in higher delivery efficiency. Due to this concept, VCoF can also have benefits in the context of mobility. For example, when a node has moved to a new location, a requested content object is frequently forwarded to the node's previous location or cached in the node's vicinity. The scope of content finding of VCoF can help to find the object that is cached nearby such as in the previous location, another node in the vicinity, or even in another nearby vicinity instead of requesting the content object again through a new path from the new location since the requested interest is unsatisfied. Re-expressing unsatisfied interest packets might incur excessive overhead, and higher latency. This particular case of mobility will also be investigated in Chapter 5.

1.4 Research Outline

The research outline of this work is threefold by aiming to examine the efficiency of content finding as well as the effectiveness of content delivery. The outline can be ordered in the following sequence.

First, the VCoF scheme is proposed and investigated in a static NDN environment (i.e., structure of the network is fixed) to indicate the performance of the scheme against its overhead costs. This also aims to understand the actual operations of VCoF in realistic NDN networks. A set of metrics is used to measure delivery efficiency and additional overheads in different situations compared to the default NDN strategy. RTT values, as the indicator, are examined to understand how much VCoF can improve the delivery delays over the baseline of default NDN. Message overhead and data volume are investigated to understand the effects of additional costs generated by the VCoF scheme to the entire network traffic.

Second, the impact of content popularity is then evaluated in a larger static topology. Some popular content could be requested several times. Hence, considering different content popularity levels can help to understand the effects of the number of content replicas to content finding results. The effectiveness as well as the efficiency of VCoF (especially the improvement of content delivery) might be significantly leveraged due to the increasing number of replicas.

Finally, although the results in the static networks can indicate how well VCoF improves content finding, finding nearby replicas in a dynamic environment would also be challenging. For example, when a node is moving, a piece of content might be forwarded to a previous location in the same vicinity or other nearby nodes. So, there could be different paths to access the content through a new location. VCoF should help to find a proper path to fetch the content (e.g., in the previous location, other nodes in the vicinity, or other nearby vicinities). Hence, extending VCoF in this particularly dynamic environment (the mobility context) is finally explored.

1.5 Thesis Structure

This thesis is structured into six chapters. Following this introductory chapter, the next chapter, Chapter 2, gives the background and related work. The chapter describes the limitations of the current Internet architecture and its modern role in delivering content leading to understand the reasons of the emergence of ICN

architectures. We describe the fundamentals of ICN in this chapter. The chapter also discusses a number of some representative ICN approaches, alongside their content finding/delivery concepts, and discusses their shortcomings, which indicate the main reasons of choosing NDN as the base architecture in this work. We also present the advantage of content caching and discuss the well-known CDN caching in comparison to NDN caching. The particular issues of content finding in NDN are then presented. Finally, this chapter examines current content finding solutions and their drawbacks. These also include content finding in the mobility context.

Chapter 3 presents the design of VCoF that tries to utilise nearby replicas to leverage content delivery in NDN. The chapter also includes detailed motivations and aims that consider the impact of nearby replicas that can be the better potential sources for consumers, and additional overhead costs that can be incurred by using the design.

Chapter 4 provides the details of the implementation of VCoF. This includes realising a fully running system by implementing the modification of existing NDN code-bases. It also details the existing software (essential) components of deploying an NDN system. The detailed implementation of the designed modules of VCoF to be applied in each NDN node is also presented in this chapter. In addition, the chapter describes the implementation of a number of tools to be used in the next chapter to evaluate the design in comparison to default NDN as the baseline.

In Chapter 5, detailed evaluations of VCoF are presented. These are compared with standard NDN in a number of different scenarios. Three main evaluations are examined to answer the aforementioned research questions. In the first instance, we investigate VCoF in a fully running system to better understand our scheme under the realistic operations of an NDN network. We then evaluate the impact of content popularity distributions in a larger scale network. Finally, we extend the VCoF scheme in the context of mobility and demonstrate how the scheme can leverage content finding in mobile NDN environments that their communication model would fit the advantages of the VCoF scheme.

Finally, Chapter 6 summarises this thesis as well as the future work based on this research. Summary of possible research limitations and concluding remarks are also presented and discussed.

Chapter 2

Background and Related Work

In this chapter, we examine the current Internet architecture and how it has developed to serve the current needs of users. This highlights the limitations of the today's Internet architecture and the main reasons for the emergence of ICN, which are described in Section 2.1.

ICN is expected to overcome the today's Internet limitations by changing host-centric to information-centric, which is the current trend of transferring content. In Section 2.2, we describe how ICN works and the core principle behind its operation. This includes content finding, which is different from the current TCP/IP principle.

A number of ICN approaches have been proposed. Most of these approaches focus on content objects that are independent from their original locations due to enabling in-network caching and replication. Nevertheless, these approaches have their own concepts. In this thesis, we focus on content finding in NDN which is an ICN approach but the number of ICN approaches are also interesting. Hence, in Section 2.3, we examine some representative ICN approaches to explore their concepts and identify their shortcomings especially in the context of content finding and delivery.

One of the main techniques to optimise content access is content caching. This is the key component of most ICN architectures. It is also very crucial in NDN because it is the main mechanism to enhance content delivery. Outlined in Section 2.4, we discuss the importance of content caching in the current Internet architecture with an understanding of the caching technique of the well-known performance optimisation solution for content access called Content Delivery Networks (CDNs). We also discuss CDNs compared to NDN and describe the better potential benefits in NDN that a content finding/delivery solution can take.

By distributing content to several caches, the content can be located closer to consumers. However, several content finding solutions (particularly the default best route strategy of NDN) try to find content objects at their original sources without utilising nearby (cached) replicas, resulting in sub-optimal delivery efficiency. To enhance this, a number of previous solutions try to utilise off-path replicas as an alternate source. Hence, in Section 2.5, we identify particular issues of off-path content finding in NDN and examine how previous items of work have influenced on enhancing content finding in NDN, as well as their shortcomings are considerably presented. These also include a number of solutions in the mobility context.

2.1 The Current Internet Architecture

The principles and essential architecture of the today's Internet are rooted in the system that was created over 40 years ago. The architecture was designed to provide a communication between hosts. As the main concept of the current Internet architecture, a user has to establish a communication to a specific host to exchange information. The primary goals of the Internet were to share resources, connect new networks, and communicate between individual devices in a trustworthy environment.

Various challenges including security, mobility, scalability, and quality of service are the important issues [10, 11] of the current Internet architecture. Due to the shortcomings in the original design, the current Internet architecture cannot effectively handle those challenges [12]. The current design is based on packet switching and used in the telephone network. So, the architecture still resembles to the telephone network. It is difficult to change the original design to support future characteristics of the Internet.

Recently, most people from all over the world use the Internet for searching information, accessing desired content, exchanging information, enjoying and exploring multimedia content, trading, conferencing, taking software services, and etc. The Future Internet should effectively handle these interactions which are performing by billions of people. There is a dramatic increment in information distribution over the Internet. Billions of people increase the information demand [13]. The current Internet architecture becomes insufficiently flexible to the changes of the content demand. Billions of devices will also be connected to the Internet [14]. The increasing number of nodes (e.g., computers and mobile devices) can be problematic to the current Internet architecture, which it may not be able to support to a sufficient level.

So, reliability, availability, robustness, and survivability are imposed to the Future Internet architecture. Connecting mainframes and minicomputers and providing efficient remote access to them were the key goal of the original Internet. However, this end-to-end communication approach [15, 16] and its practical implementation have been identified as a cause of several limitations [3] of the current Internet architecture.

Various add-ons, such as Network Address Translations (NATs), Mobile IP, Content Delivery Networks (CDNs), etc., have been developed to serve the current needs of users. Nevertheless, these were not part of the original requirements [17]. The current Internet architecture was designed in an environment, which is different from today's reality. In particular, according to the growing number of mobile devices, mobile networks should be flexible to enable dynamic communications. Due to the design of the IP address, a mobile host needs to be assigned an IP address to be an identifier for TCP connection. This can be problematic, when the mobile host moves to a new location. It requires to decouple the identifier from a previous location. Establishing a new TCP connection and identification is then required at the new location. Thereby, this limitation of mobility support of the current TCP design can not be able to effectively handle mobile communication.

Several applications such as the MobileIP [18, 19] have been used to serve the needs of the current Internet requirements. However, they also add some complexity and seem to be temporary mechanisms to address the current issues since they are built on top of the original host-to-host communication model [20, 21]. Although some content delivery solutions (e.g., CDNs and peer-to-peer networks) have been deployed to transfer content from producers/publishers to consumers/subscribers, these solutions still suffers the same problems. The designs of these solutions still face the content centric problem due to the limitations of the host-to-host communication model. So, a simple "host-to-host" packet delivery paradigm needs to be transformed into a more flexible paradigm focusing on handling information, content, and users rather than the machines [3]. The needs to support content distribution in a scalable manner, new emerging applications, security, mobility and many more others are required [22, 23]. Due to the multitude of challenges in the original Internet architecture, this is likely that it is time to shift from the current Internet towards a new Internet paradigm.

Recently, distributing and manipulating information has become the main goal of today's Internet [24]. Several Information Centric Network (ICN) approaches have been proposed to handle many challenges that cannot be sufficiently handled in the current Internet architecture. Most ICN architectures are clean-state designs since

several issues of the current Internet can not be effectively addressed on the original design of the Internet. As mentioned previously, exchanging information is one of the major functions of the today's Internet. So, ICN architectures focus on finding and transmitting information to end-users [25] rather than transferring content from host to host. An information object can be located by its name instead of its address. This can overcome the current problems of using IP addresses with location-oriented communication environments, and offer better benefits, which will be detailed in the next section.

2.2 Information Centric Networking (ICN)

According to the original design of today's Internet, the architecture is encountering many challenges as described in the previous section. By using the current Internet architecture to locate a specific content object, the object is mapped to a host using a DNS server. This server is used to translate the host name to its location (i.e., IP address). However, the lack of available IP addresses, increasing network traffic, exchanging massive data, increasing delays can be problematic in the new era of the Internet. Today, users are interested in increasingly large content objects (such as ultra high-definition movies), mobility, security, privacy and other features that were never considered in the original design of the current Internet architecture.

To address the aforementioned issues of the current architecture, several researchers have suggested that there should be a new network paradigm that is a clean state redesign of the Internet. Due to the growing content demand, these researchers have presented their ideas in focusing on content rather than machines or hosts. This prompted the research into changing the current architecture from host-centric to information (content)-centric. So, Information-Centric Network (ICN) has been introduced with the main aim to potentially replace or augment the existing Internet architecture. It is expected to be a key component of the Future Internet [26, 27].

In the current host-centric architecture, when a user needs a piece of content, the user must know the content's location, which is the URL of the content. Today, the content can be located in several locations. According to the concept of Content Delivery Networks (CDNs), the content object should be mapped with the nearest server. However, the content retrieval is still handled by application-layer solutions rather than the network layer itself. Hence, the access overhead can be incurred and

the network-layer optimisations are not optimised.

In an ICN system, a content object is given a name instead of an IP address. The content can be directly exchanged by its name. It means a centralised system is not required to map the IP address to its host name like in the current architecture. Hence, the new network paradigm is more flexible to deploy in various types of networks, especially in the context of mobility. According to the features of in-network caching and replication, the content object becomes independent from its original location. Thereby, better scalability, improved efficiency, better robustness in challenging communication scenarios are also the expected benefits of ICN [28].

2.2.1 ICN Naming

Most ICN approaches focus on content rather than hosts. Hence, some content identification schemes are required to identify every content object in the network. A number of researchers have widely suggested that each content object should be identified by name. Hence, determining a suitable naming scheme becomes one of the most challenging issues in the ICN systems. Instead of providing names to devices or hosts, a proper ICN architecture must be effective and efficient to identify different content objects (such as videos, images, documents, web-pages, etc.). So, the number of available names in the network can be enormous. The ICN systems must effectively handle naming schemes to be more efficient. There are two main types of the naming schemes in ICN [29].

First, hierarchical naming consists of several components of content identification and the structure of the name is very similar to the URL. For example, lancaster.ac.uk/obj1 can be named to uk/ac/lancaster/obj1. The name can be a human-readable name that would be a URL-like name. Several objects in the same node can be located by its prefix. So, the hierarchical naming can be aggregated into a name prefix to indicate the location of several content objects in a particular source. The hierarchical naming can be readable and easy to remember. However, it creates security vulnerabilities because the name is visible.

Second, flat naming provides uniqueness because the hash of the content or the hash of the key is part of the name. One benefits of the flat name is to provide more security compared to the hierarchical naming because the name is not human-readable. The name can naturally authenticate its source. However, flat naming does not support routing aggregation, meaning it does not scale well.

According to the aforementioned issues of the both types of naming, there should be any additional schemes to solve the issues [20]. For example, by using hierarchical naming, some additional security mechanisms should be implemented. In a flat naming system, a number of approaches should be applied to mitigate the naming scalability issue.

2.2.2 Content Finding in ICN

In an ICN forwarding strategy, there is a process that forwards a packet to find a content object at its source. When the object is located, it will be routed back to its requester. There are two main approaches to forward the packet [30]. First, by using a name resolution approach, the process maps a packet to its source location and forwards the packet to the mapped location. To map the location of the source of the content, Name Resolution System (NRS) is responsible for providing the translation of the name to its location. This approach can potentially guarantee that the packet will be forwarded to its source to fetch the desired content object because the name is already mapped with its source at the centralised system. However, this approach can introduce a single point of failure. This is because the NRS requires large amounts of storage to store the mapping information. If it fails to operate, the packet can not be forwarded to the source location and several content objects registered on the NRS would be inaccessible. Resolution delays could be another problem.

Second, by using a name-based approach, the packet is forwarded hop-by-hop to its source by content routers. A content router is responsible for determining a next hop to forward the packet based on its name. There is a Forwarding Information Base (FIB) to map the packet with the best possible next hop to forward the packet to. According to the hop-by-hop forwarding process, this name-based approach does not likely guarantee that the content object of the packet can be located (compared to the previous approach). However, a requested packet can be re-issued and re-transmitted, if its requester has missed the corresponding (desired) data.

Data routing is the process to specify a path to return a data object back to a requester. There are two main approaches for the data routing. One strategy is a coupled approach that the data object is transferred back to the requester in the same path as the initial request (i.e., the reverse path). In contrast, a decoupled approach selects a route independently and the strategy allows different routing mechanisms to be used to route the data object back to the requester.

2.2.3 ICN Caching

To improve the performance of content access, Most ICN approaches provide caching services. By caching content, several replicas of a content object can be distributed across the networks. These replicas can be stored locally on an ICN node or can be shared on any network caches. Content can be cached in several points of the networks (e.g., content routers, and edge nodes or end devices in the networks), thereby being able to support multi-sourcing for content consumers. Content caching can be considered in three levels including object level (a complete content object), chunk level (part of a content object), and packet level (bytes of a content object) [31]. Each ICN architecture can also deploy or customise different caching policies, and schemes depending on its design.

2.3 ICN Approaches

In the previous section, we highlighted the main keys and benefits of ICN to make it the expected solution to replace or augment the current Internet architecture. However, there are recently several ICN architectures that have been proposed. So, this section presents the overview of a number of some representative ICN architectures, which are the widely discussed projects being developed, alongside their content finding/delivery principles. This also includes a comparison of the features and shortcomings of each approach, which is found in Section 2.3.6. By considering the shortcomings, we also describe the main reasons of choosing NDN as the base architecture of this work.

2.3.1 Data-Oriented Network Architecture (DONA)

DONA was developed by UC Berkeley as detailed in [32]. It is considered among the first full ICN architectures. DONA identifies names of data with flat names and these names are not location-oriented. It means the replicas of named objects can be stored in several caches. By the design, the objects can be moved independently. This can also achieve one of the goals of ICN in supporting mobility. A cryptographic technique is used to authenticate and to verify the data integrity. A principal is a trusted entity and is associated with a public-private key pair. A cryptographic hash (called P) of the principal's public key and a label (L), chosen by the principal are

combined together to generate a name. The name is made from P:L to ensure that it is unique.

There is a triplet of the form <data, principal's public key, principal's signature> and other metadata to be sent to a consumer. According to the form, the consumer can immediately check the integrity of the data using the public key and the signature. This can help to ensure the authenticity of the data. However, due to the flat names, it is difficult to consumers to remember the names because these are quite long and are not human-readable making them hard to remember. To eliminate this issue, a name resolution service is designed similar to URLs in a DNS system. A user-friendly name associated with a content object is mapped in the name resolution process. The difference between DONA and DNS is that a DNS system tries to map a URL with its location but in DONA, the names are not attached to a specific location.

In DONA, there is at least one Resolution Handler (RH) (sometimes referred as a Dissemination Handler (DH)) in each autonomous system (AS). These RHs are responsible for the resolution process (DNS-like service) in the DONA system. DONA organises RHs in a hierarchical fashion. A higher RH above another on the hierarchy is called the *parent* of the below RH. When a producer needs to provide a content object or a service, it creates a message packet, namely REGISTER (P:L). The name of this content object is made by (P:L) as described in the aforementioned description. REGISTER (P:L) stores a pointer to the producer and further propagates to the peers and parent RHs. A registration table in each RH is maintained to provide information about the next hop (or next RH) and distance to the available replica(s) of the content object. When a consumer needs this content object, it issues a FIND(P:L) packet to find the object named (P:L). A REGISTER message can be expired, which is indicated by a TTL (time to live). An UNREGISTER message can be issued to indicate that a particular content object is no longer available.

Longest Prefix Matching (LPM) [33] is deployed in each RH. In a longest prefix matching case, the RH tries to match the P:L of a packet in its registration table. If it can be matched, the corresponding data will then be served with the name P:L or forwarded to a next possible hop depending on the existing routing information. If the name P:* or P:L can not be matched in a registration table, it means that there is no corresponding data to be matched in the table. The FIND packet will then be forwarded to its RH parent.

As presented in Figure 2.1., the FIND packet is routed by its name towards a proper source. The packet is forwarded using the LPM technique until it reaches the

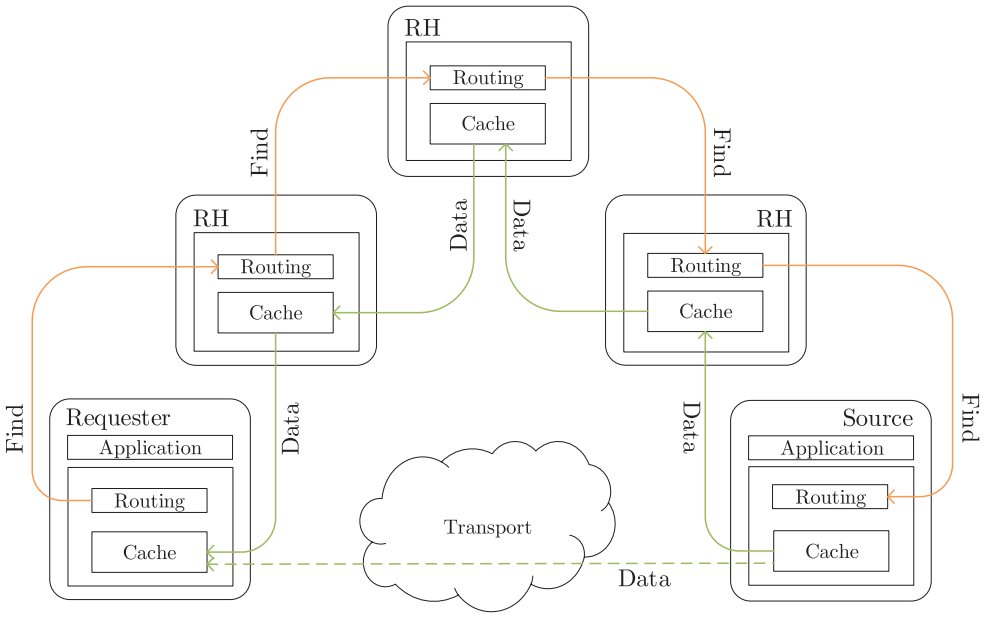


Figure 2.1: DONA Overview

source of its corresponding data. The data packet is then routed back (also cached along this path) to the requester or over a direct route. A FIND message forwards up to the RH hierarchy, unless locates a match. If the FIND does match a record, the corresponding data is returned by using a standard transport-level response to the requester. If the FIND message reaches a Tier-1 AS and does not match any record, the Tier-1 RH then returns an error message to the source of the FIND message. When a data packet is forwarded back to the requester, each RH is not involved in the transport process. The data packet can be transferred by using the standard IP routing. To make this works, transport protocols should bind to names instead of addresses.

DONA generally supports mobility. To locate a content object, a mobile entity simply issues a FIND message and forwards to RHs. The RHs provide a closest replica to the requester depending on routing information. Mobile producers are also supported. When a mobile producer moves to a new location, it simply unregisters its provided content from the old location and registers the content to the new location.

2.3.2 Network of Information (NetInf)

NetInf [34, 35] allows access Named Data Objects (NDOs) using NDO names, instead of using host-dependent addresses. A NetInf node forwards NDO requests to find their corresponding data objects. A name of an NDO is unique and location-independent and the NDO can be replicated or cached in the network. A NetInf NDO can be a complete object or broken down into several chunks. It can also be forwarded by directly using routing information of the object name (name-based routing/forwarding) or using name resolution services. NetInf represents a hybrid architecture that can support both name-based and name resolution routing. An inter-domain interface is defined in NetInf for the name resolution and routing that can be used different mechanisms in different parts of a network. There are two name resolution mechanisms that have been developed in NetInf as described as follows.

First, Multilevel Distributed Hash Table (MDHT) system [36] is used to interconnect separated local NRSs into a global NRS infrastructure. Hierarchical NRS is topologically embedded in the underlying network. Potentially Distributed Hash Tables (DHTs) enables location-aware and scalable resolution of flat namespaces. Second, NetInf provides an NRS approach (named Late Locator Construction (LLC) [37]) that aims to handle highly dynamic network topologies. To support producer mobility, when a content object moves, the NRS updates the movement results to the new network location. In LLC, a Global Locator (GL) is responsible for routing between the Core Network (CN) and the Edge Networks (ENs). GL is built inside the Locator Construction System (LCS). Each network and host in the edge topology has an associated Attachment Register (AR) in the LCS. The host or the network updates its AR with the name of the attached neighbor. Hence, the LCS can construct a GL for the source/destination routing.

NetInf can handle client mobility dependent on a data transport mechanism and a forwarding strategy. For example, Global Information Network (GIN) [38] natively supports client mobility. In GIN, PUT(ID) request is to register the availability of a stored data object at a storage location in Distributed Network Dictionary. A retrieval request (GET message) for the object ID will be resolved into a location address.

By mapping an NDO name to its locations, NRS is deployed to locate the NDO and can indicate where the NDO is cached. To create a hierarchical resolution system, MDHT is applied in the NRSs including the global NRS and local NRSs. MDHT can take different caching decisions at different levels (i.e., multi-level caching). For

example, the more frequently requested objects should be cached in the access node level. In contrast, the less frequently requested objects should be cached in the AS level. Both global and local NRSs can indicate that the selected locator is close to the requester. The NRS can also track the request and keep request statistics. This statistics can help to determine which NDO should be cached locally.

Cached NDOs can bring content closer to a requester, increasing the opportunities of faster content delivery. By using the name-based routing, NDO information is stored and distributed among NetInf routers by a routing protocol. An NDO request can be fetched directly by the requester. Furthermore, by integrating the NRS with name-based routing, this hybrid design can support a flexible routing scheme. A routing hint provided by the NRS can also guide the request to fetch the NDO at its source.

Similar to DONA, a flat name-space with some structures is deployed for NDO names [39]. NetInf provides a fundamental of name-data integrity validation. Requesters or any other nodes can validate names without theoretically infrastructure support (assuming that the names are trustable). In a NetInf NDO, there is a

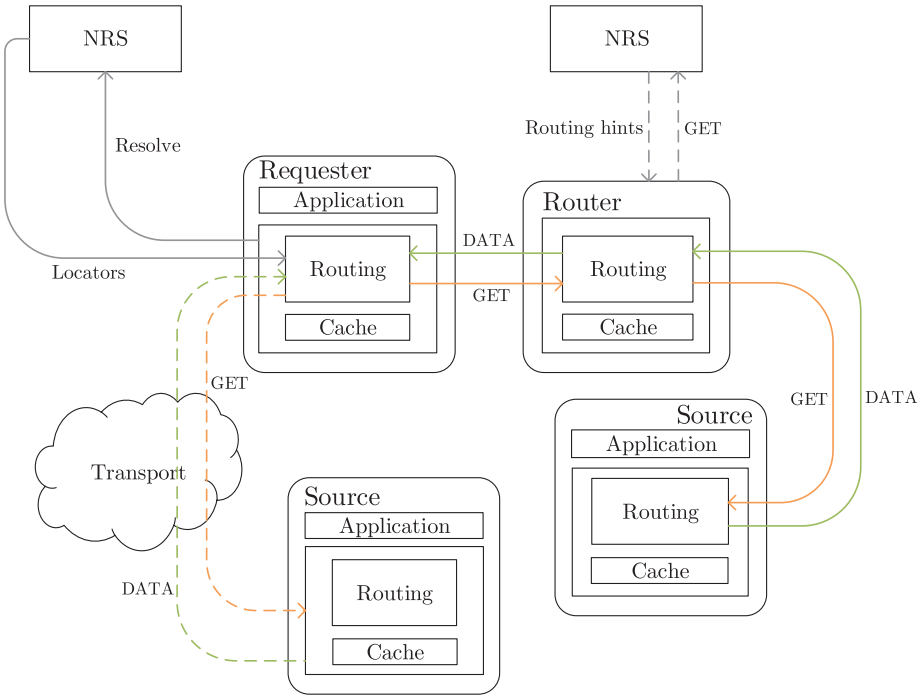


Figure 2.2: NetInf Overview

security metadata that includes the data signature and public key [40] necessary for its verification.

As presented in Figure 2.2, an example of name-based routing, name resolution, and the hybrid approach is illustrated. In the name-based routing, a GET request created by the requester is forwarded hop-by-hop among NetInf nodes unless a cached replica of the NDO is located or the request has reached the original producer of the NDO. If the NetInf routers cannot perform the name-based routing, it means there is not enough information to route the request. The hybrid mode is then activated. Before forwarding the request to the source of the NDO, a name resolution process is performed by sending the GET message to the NRS to get routing hints. The request can then be forwarded indicated by the routing hints. The NDO can be cached along the return path to the requester. Alternatively, the requester can resolve the NDO name into a set of locators and can retrieve a replica of the NDO from the best available source(s) indicated by the locators obtained from an NRS.

2.3.3 MobilityFirst

MobilityFrist [41, 42] aims to design a clean state architecture for a next generation Internet. According to the growing number of mobile devices, the project envisions mobile devices as a key of the next generation communications. One of the current limitations of the traditional IP architecture is the ability to effectively handle seamlessly transition between mobile devices. In the design of MobilityFirst, mobility and trustworthiness are two main keys. "Mobility" focuses on providing seamless communication between wireless devices, thereby mitigating the mentioned limitation of the existing architecture. "Trustworthiness" means that the design should be able to handle security and privacy.

In MobilityFirst, names can be used to identify different entities such as devices, services, consumers, or content. A Globally Unique IDentifier (GUID) contains bit-strings to be used as an identifier and it can be self-certifying. The self-certification of GUIDs means that any entities can be theoretically verified without requiring an external certification authority. Challenge-response procedure is used in the authentication process. In the design of GUIDs, Global Naming Service (GNS) is used to translate human-readable names into corresponding GUIDs.

MobilityFirst uses a Network Addresses (NA) to identify a network. This NA corresponds to an Autonomous System (AS) and it can also be used to identify a

subnet or one or more base stations or an Internet Service Provider (ISP). Global Name Resolution Service (GNRS) is used to translate GUIDs to NAs. When a content producer needs to provide a content object or a service, it asks the GNS for a GUID. The GUID is then registered along with its NA in the GNRS. when the consumer needs a content object or a service, a GET message (Containing the GUID of a desired content object or a service and the GUID of the consumer) is issued and sent to its local Content Router (CR). The CR forwards the GET message to other CRs or network entities. The GNRS provides routing information to route the GET message.

In a data routing process, a hybrid approach between IP routing and name-based routing can be performed. The GNRS is responsible for mapping addresses to GUIDs. Hence, to forward the data packet, traditional IP address routing can also be used. However, the initial request for the data and all end host/node communication must be done by using a name (associated with a GUID) rather than addresses. So, names are exclusively interacted before forwarding data packets.

As presented in Figure 2.3, a host wants to sent a data packet to all of John's devices. The host will lookup and obtain the corresponding GUID from the GNS. A service API is then invoked to use a command such as send (GUID, options, data).

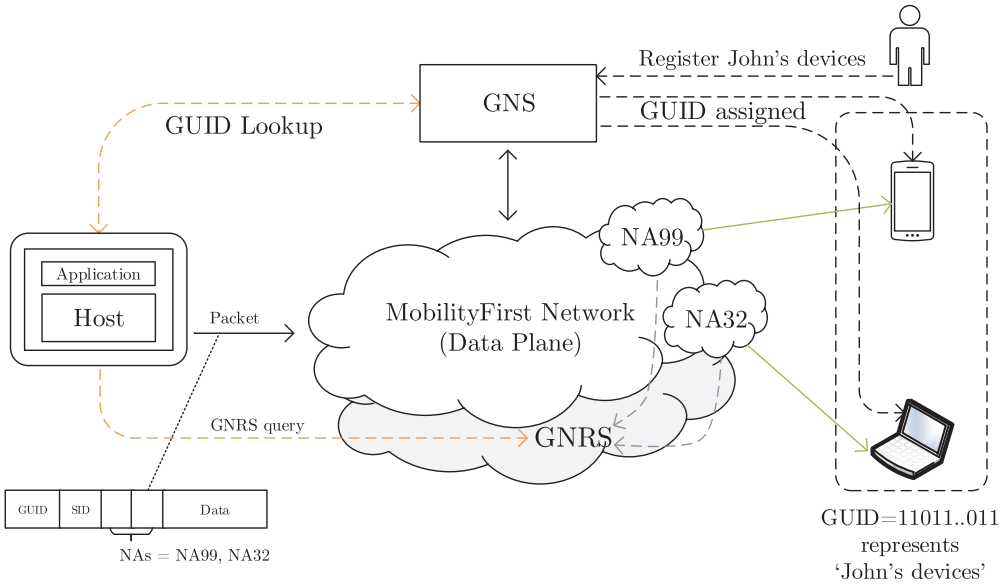


Figure 2.3: MobilityFirst Packet Routing

The options can be service features such as anycast, multicast, and so on. The host then queries the set of network addresses using a GNRS lookup to find the current points of attachments of the destinations. In this case, the network addresses are N99 and N32. The packet is then sent out by the host and it consists of a destination GUID, Service ID (SID), and list of NAs. To make a forwarding decision, each CR uses the current NAs of the destinations. Multicast of the packet is then added to reach both N99 and N32. The data packet is then forwarded to all of John's devices. If the packet fails to delivery according to disconnection or mobility. The packet is still stored in the network and the GNRS is periodically updated to rebind the GUID with the current NAs. The packet is then delivered before a timeout or it will be discarded due to timeout.

According to the data routing process, the path to forward the GET message might not be the same path to forward the data packet. This is due to the fact that the GET and the content packet are routed based on their own destination GUIDs. So, the routing processes are decoupled.

2.3.4 PSIRP/PURSUIT

The Publish-Subscribe paradigm is an alternate way of sending and receiving content. Communication architectures based on this paradigm fundamentally consists of three basic elements including publishers, subscribers, and a network of brokers [43]. Publishers are the content producers who publish content by issuing content publications to be requested by subscribers. Subscribers are content consumers that are interested in the content provided by the publishers. Subscriptions are issued by the subscribers to be sent out to consume the interested content items. A network of brokers located between the publishers and subscribers is responsible for matching the publications and subscriptions. The broker where the publication-subscription matching takes place is known as the Rendezvous Point (RP) [44].

PSIRP (Publish-Subscribe Internet Routing Paradigm) [45, 46] is a clean-state information-oriented architecture aiming to enhance the Future Internet. The core networking functions have been developed based on the publish-subscribe primitives. The architecture focuses on information rather than hosts. The paradigm includes integrated support for anycast and multicast, caching, multihoming and mobility, and security and privacy [44].

In PSIRP, content items are organised within scopes. Scopes can be physical

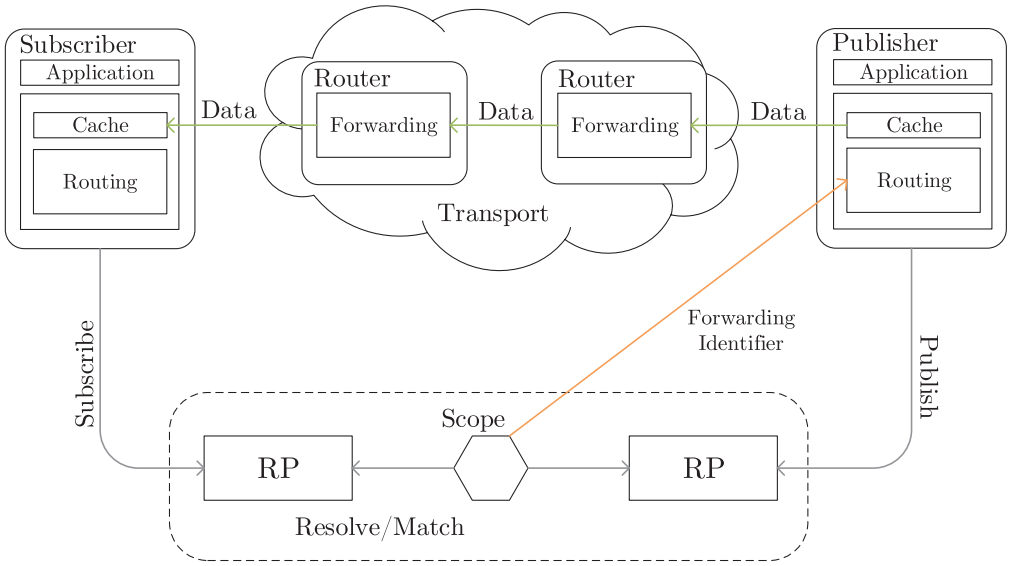


Figure 2.4: PSIRP Overview

structures (e.g., a campus network, a corporate network) or logical structures (e.g., social media friends). A pair of identifiers: a Rendezvous Identifier (RId) and a Scope Identifier (SId) is used to identify every content object with flat-label-based information. The RId must be unique within a scope, while the SId denotes the scope where a content object belongs. As presented in Figure 2.4, to publish a content object, the publisher issues a publication with the SId of the scope. The RId is then created for the publication and it is forwarded to the rendezvous nodes of the SId's rendezvous network. Subscriptions follow the similar process. When the subscriber needs a content object, it learns the identifiers (the RId and the SId of the desired piece of content). The subscriber then issues a subscription packet towards the proper RP using the RId. A data forwarding path is created between the publisher and the subscriber. The Forwarding Identifier (FI) consisting of a Bloom filter [47] of routing information is used by the PSIRP routers to route the content object to the subscriber.

To further explore, PSIRP project has been expanded and called PURSUIT [48] (Publish-Subscribe Internet Technologies). The motivation goals of PURSUIT are to develop a more complete architecture and protocol suite. By taking the advantage of all lessons learned from PSIRP, PURSUIT filters missing components and improves the existing ones. In PURSUIT, the information organisation follows the same principles of PSIRP. However, it also focuses on the lower layers and on mechanisms that can provide better resource allocation and utilisation at the link and the physical

layer by also taking advantage of the information structures at the higher layers [44]. For example, one of the most serious threats on caching networks is a cache pollution attack [49]. However, at the higher layers, ranking information items based on positive votes can achieve better results in terms of polluted content isolation [50].

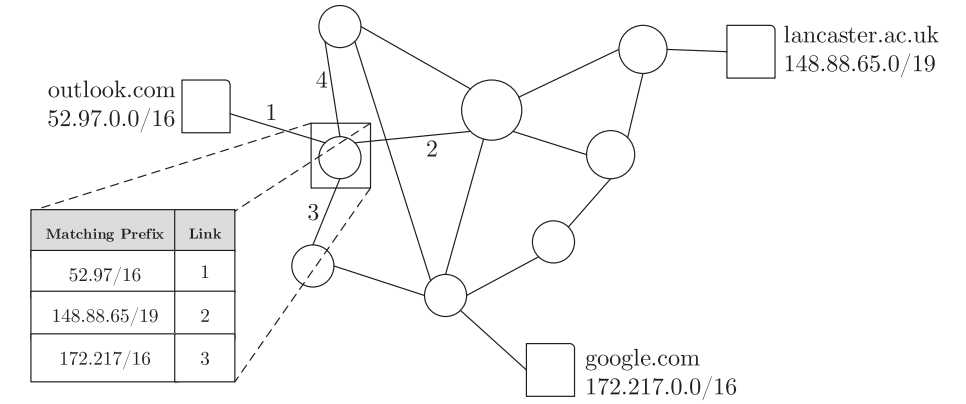
2.3.5 Named Data Networking (NDN)

As mentioned previously, several ICN architectures are poised to address the current limitations of today's Internet, such as scalability, addressing, mobility, security, and privacy. These ICN approaches also aim to be able to handle the requirements of new emerging Internet applications in sufficient efficiency. To realise ICN, Named Data Networking (NDN) [1, 2] is designed and has become one of the most promising ICN architectures for the Future Internet [51]. According to its clean state design and flexible communication model, it is expected to be suitable to handle a number of challenges encountered in the today's Internet requirements.

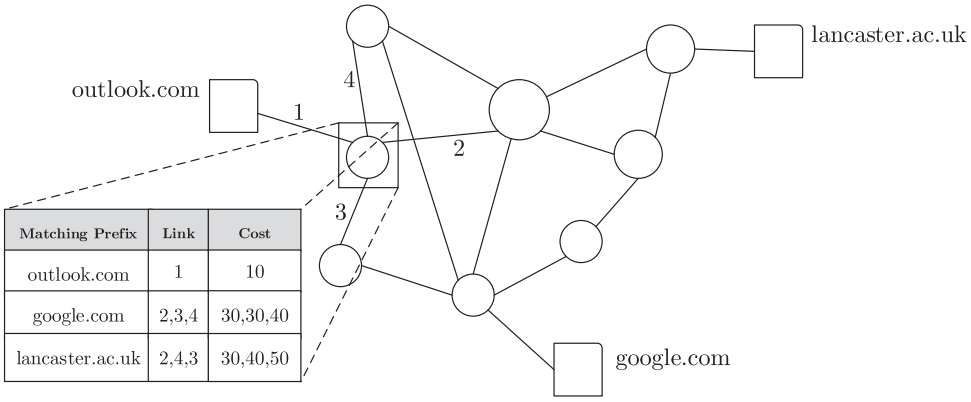
NDN supports information-oriented communication instead of the existing location-oriented model. Every content item is identified by its name. By using a stateful forwarding plane, forwarders keep a state of each request packet and remove the state when its corresponding data returns back. Hence, this can keep track on content delivery. According to the NDN's stateful forwarding, intelligent forwarding strategies can be deployed and loops can be detected. Content caching in NDN allows nodes (e.g., routers, or end devices) to store distributed content objects leading to accommodate scalable content delivery on the Internet. By the NDN's design, the scalability and content distribution issues of the current Internet architecture can be potentially handled. Additionally, data transfer in NDN is secured directly at the network layer. By signing any of named data, packets can be verified and extra security mechanisms such as Public Key Infrastructure (PKI) [52] can also be deployed.

The NDN project originally used Content-Centric Networking (CCNx) as its code-base. However, as of 2013 has forked a version to support research experimentation that aimed to address open questions (e.g., not as much forwarder modularity, API ease-of-use, and trust management tools) [53].

Two packet types including interest and data packets is designed to eliminate the current IP model. An interest packet issued by a requester is to request a desired content object indicated by its name prefix. The corresponding data packet will be



(a) IP Routing



(b) NDN Routing

Figure 2.5: The Main Difference between the Traditional IP and NDN Routing

returned back in the reverse path of the interest packet (i.e., Breadcrumb trail) to be delivered to the requester. The content name is considered by a forwarding strategy to indicate or route the interest packet, which this is also supplied by the routing information in the FIB of each NDN node.

NDN allows multi-path routing that is different from the traditional host-to-host routing. Figure 2.5 describes the overview of the major difference between the IP and NDN routing. In Figure 2.5 (a), if there is a packet that is going to a destination, the prefix of the destination IP address will be matched according to the routing information in each router, which normally provides a single path routing. In NDN as presented in Figure 2.5 (b), a packet can be sent in different paths. In default NDN, each router normally selects a path indicated by every lowest link cost to route the packet. Several paths can also be selected depending on a forwarding strategy.

According to the support of multi-path routing in NDN, we can gain several benefits. For example, in a dynamic topology, a source of content might be going off-line, if a number of desired content objects are cached nearby, NDN could make full use of the surrounding resources. While the traditional host-to-host routing might be failed to get the content.

2.3.5.1 NDN Forwarding Strategy

To route NDN's interest/data packets, there are three core elements in each NDN node as presented in Figure 2.6: a Pending Interest Table (PIT), a FIB, and a Content Store (i.e., called CS, or Cache). The PIT stores interest packets that have been forwarded and are waiting for their content objects. The FIB contains forwarding entries (Faces with name prefixes and costs) to be used as the input in a forwarding strategy. This strategy determines proper Faces (e.g., by the lowest cost) to forward the interest packets to find their corresponding data. The returning content objects (i.e., data) are replicated in each router's CS (i.e., Cache) along the default path from their sources to their requesters. Thereby, this supports multi-sourcing since content can be distributed in any parts of networks. The following strategies are available in standard NDN to be configured for content finding.

- Best Route Strategy: The default forwarding strategy of NDN is configured to find a content object in a shortest default path to its producer. A node selects a Face with the lowest routing cost to forward an interest packet indicated by its name. In this work, **default NDN (i.e., standard NDN)** means the **default best route strategy** of NDN.

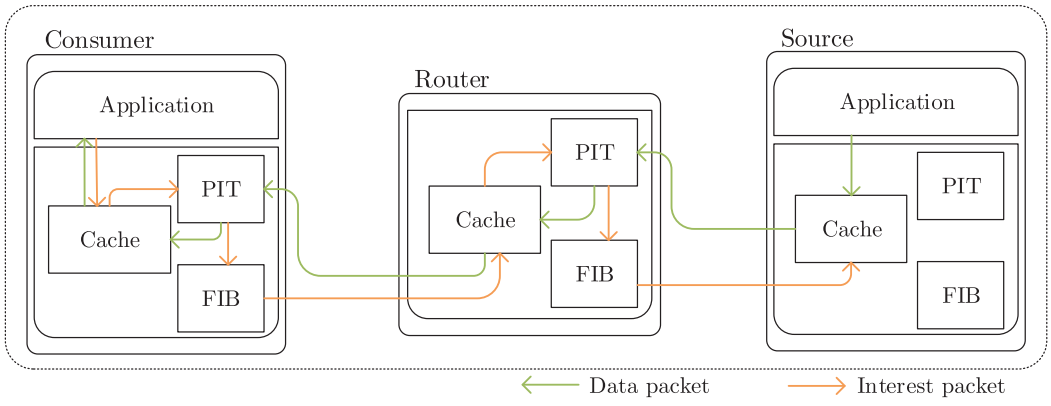


Figure 2.6: NDN Principle

- Multi-cast Strategy: Every interest packet is forwarded to all outgoing Faces to find its content object. These Faces are selected based on the name of the content object in the supplied FIB entry in each NDN node.
- Client Control Strategy: A node specifies an outgoing Face for a particular name of corresponding interest packets.

2.3.5.2 Shortest Path Strategies

In the NDN control plane, a routing protocol is crucial to compute and insert forwarding entries into a FIB. Named-data Link State Routing protocol (NLSR) [4, 54], for instance, is a routing protocol that advertises the reachability of name prefixes to every router's FIB. In the default forwarding plane of NDN, the best route strategy determines a proper Face by considering the lowest cost supplied by the FIB computed by the routing protocol to find the shortest way to fetch a content object.

To reduce NLSR's overhead, a hyperbolic routing protocol [55] uses the coordinates of NDN nodes to calculate a possible shortest path to a producer. Similarly, several forwarding strategies (e.g., [56, 57, 58]) aim to find the best paths by considering other factors (e.g., geolocations) to fetch desired content objects. However, nearby replicas especially in the vicinities of consumers, which are often located off those paths, might be opportune to be efficient looking for them to achieve higher delivery efficiency. These can potentially be the shortest paths for retrieving the desired content.

2.3.5.3 Adaptive Forwarding Plane

By observing interest and data packets, a router can measure delivery performance (e.g., RTT, throughput, and packet loss) to determine alternatively suitable paths to forward an interest packet [59]. The forwarding process in NDN can be adaptive [60]. Thereby, robustness communication is another one of expected benefits of NDN due to the adaptability and variety of content finding or routing paths.

Interface ranking [61, 62], for instance, is applied to find the current best path to fetch a content object. In the design of [8], Faces can be categorised into three colors. Green represents the status of a Face that can bring a desired content object back. Yellow indicates that the Face may bring the content object back but it may exceed an expected time. A Yellow Face turns to Red when there is no data packets back after trying this Face in a certain amount of time or the Face is down. Notably, each Face based on the color scheme can also be ranked by using another factor. For

example, an RTT value can be recorded to rank the Yellow Faces. The lower RTT might help to select a Face that can bring data faster. To find an appropriate path of a content object, [63] creates routes based on an interface ranking. Content names are mapped with a Face as an entry in a Stable BloomFilters(SBF)-based FIB. The more an SBF's entry matches with the content names, the higher associated Face is ranked. An alternate path can then be selected based on the ranked Faces.

Although the ranking design might find a proper path to fetch a content object, the strategies consider only the best way to find the content object indicated by its name. In fact, the replica of the content object could be hidden nearby. It might be difficult to find this replica because there is no indication entry in the FIB by default.

2.3.6 Comparison of ICN Approaches

As detailed previously, the number of representative ICN architectures and their fundamentals are presented. By considering their routing strategies, name-based routing, name resolution routing, and hybrid routing can be categorised. By using name-based routing, an ICN approach like NDN (standard NDN) finds content hop by hop. In this default NDN, the forwarding plane selects every Face indicating a shortest path to an original source of desired content to forward an interest packet to every next hop in this default path according to the FIB constructed by the control plane. The PIT is used to keep track of the content delivery. However, PIT overflow can be problematic, if there are a highly excessive number of unsatisfied PIT entries. In terms of content finding, off-path cached content might not be located by using the default strategy, causing sub-optimal delivery efficiency.

Even though DONA uses a resolution-based technique to ensure that such flat names can be resolved to appropriate locations, FIND packets are routed using the routing information in each RH provided by REGISTER packets that indicate paths to publishers. This is likely also considered as a name-based routing approach. A registration table in each RH maintains location information provided by the REGISTER packets, which can help to point about next hop (or next RH) to available objects/replicas according to their names. Nevertheless, there is a global scalability challenge due to the hierarchical fashion.

By using name resolution routing strategies, a rendezvous point is responsible for mapping request packets to their producers or corresponding caches. For instance, MobilityFirst use GNRS to map addresses to registered GUIDs. A GET message

stores the GUID of a desired content. To forward the GET message, each CR asks the GNRS for a route of the mapped GUID. If the packet fails to delivery due to disconnection or mobility, GNRS is periodically updated to rebind the GUID with the current NAs. This rebinding process might also slow down the name resolution processes. Changing topology/connection and mobility might still be problematic especially in terms of delivery delays, e.g., the rebinding delays can also increase the overall delays of content delivery.

In PSIRP/PURSUIT, a pair of identifiers: an RId and SId is used to identify each information item. A subscriber sends out a subscription message via its local RN in its scope using the RId. A return path is then generated denoting with an FI to be sent to the source, which source routing is then used to deliver content to the subscriber. However, the particularly designed scope in this architecture might limit communication in a particular area, facing scalability challenges. Multiple rendezvous nodes might create higher control overhead. By using the name resolution routing strategy, additional resolution delays can also decrease delivery efficiency. Single point of failure can make desired content inaccessible. In addition, NDO mapping in the centralised manner also requires large storage.

A hybrid strategy like NetInf can switch the operation between name-based routing and name resolution routing. The name resolution routing is activated by sending a request to an NRS to get routing hints or a set of locators. NetInf allows multi-level caching. For example, more requested content will be cached in the access level (nodes) while less frequently requested content should be cached in the AS level. In this hybrid approach, additional resolution delays and single point of failure could still be problematic, when the name resolution routing is activated. Multi-level caching might be complex in terms of cache management. For example, if an access node decides to not cache a specific content object, other nodes that need this object might spend more time in looking for the object at a higher caching level. In addition, distributing NDO information (for the name-based routing) especially when there are massive amounts of NDOs can increase excessive overhead that might impact the entire network traffic.

To summarise those ICN approaches, their routing strategies, main features, and shortcoming remarks are compared in Table 2.1. Additionally, there are other interesting ICN approaches such as COntent Mediator architecture for content aware nETworks (COMET) [64], CONVERGENCE [65], CONET [66], and CURLING [67]. Nevertheless, the main key features that characterise most ICN architectures can

summarily be in-network caching, traffic management and QoS, mobility support, resolution handling and data transport, forwarding and data routing, and congestion control. We discuss each of these features as follows:

- According to location-independent naming of information objects, content can be replicated in several caches, which can be delivered to consumers irrespective of whether the original producer is inaccessible or not. Caching allows more efficient utilisation of replicas and multiple paths for content delivery.

- Exposing content names to the forwarding layer can support traffic management and QoS. For example, the selection of an appropriate source of a desired content can be performed over the essentially suitable routing paths (e.g., shortest paths, less congested, lowest delays, etc.), which can be directly considered at the forwarding layer.

- Several ICN architectures can natively support consumer mobility. Mobile nodes simply re-issue requests for content objects that they have missed due to the disconnection during handover. Additional overhead can be incurred but it is still potentially flexible than the current TCP/IP communication model that has been encountering the limitations of session re-establishment and location-dependent routing.

- By using resolution routing, a separate resolution handler matches consumers with producers to calculate routing paths for data transport.

- Since naming can directly operate at the forwarding layer, an ICN node can make forwarding decisions on its own. Routing protocol can also be deployed to propagate default routing information for the forwarding decisions. Data routing can be coupled with the request forwarding such as returning a data packet back in the reverse path of its request.

- ICN can monitor traffic information using a separated service or a mechanism inside an ICN node itself. For example, when there are highly congested packets in a particular network interface, the ICN node can use alternate interfaces to forward packets.

Although ICN can provide the number of favorable features, there are also the number of shortcoming remarks as mentioned previously and as compared in Table

2.1. By using name resolution approaches, single point of failure and additional resolution delays could be problematic. Some particular ICN approaches might be beneficial in a particular network structure such as RHs in the hierarchical fashion of DONA. Scopes in PSIRP/PURSUIT might also face scalability challenges. In the centralised manner, most resolution mapping strategies require large storage. Multiple NRSs can incur highly control overhead. In addition, when cached objects may change so quickly, resolution-based strategies could not effectively map these objects.

Hence, we are interested in the ICN architectures based on name-based routing since interest/data packets can be routed according to their names irrespective of a separate name resolution service/system. This could be more flexible in various network environments. Although the name-based routing might perform well in NetInf, the excessive overhead of distributing NDO information must be debatable. In addition, the strategy still relies on the name resolution, which can still face the aforementioned problems of the resolution-based routing. Flat naming in some ICN approaches has a scalability challenge since it does not support routing aggregation. Hence, by considering the aforementioned drawbacks, we focus on NDN due to its clean state design and flexible communication model, which can potentially be deployed in various kinds of networks. It is also one of the most promising ICN architectures [68] and widely discussed as a potential paradigm for the Future Internet.

NDN uses hierarchical naming that can be readable and easy to remember and it can also be aggregated into a name prefix to indicate content finding in single-sourcing or multi-sourcing. However, it imposes security vulnerability because the name is visible. Nevertheless, according to its design, it secures data directly [69], which can achieve data authenticity, and confidentiality regardless of whether the data is in transit or at rest. Additionally, extra security mechanisms like PKI can also be deployed. Each ICN approach has its own content finding issues as discussed earlier. Likewise, NDN has also ineffective content finding strategies. Although an NDN node can send a request to a shortest path due to the routing information constructed in the control plane, several off-path (e.g., nearby) cached content objects cannot be located because there is no routing information for these objects, causing a worse than necessary delivery efficiency. Multi-cast strategies might help to find these objects faster. They can increase excessive overhead and cache eviction rates, however. More details regarding these issues will be specifically discussed further in Section 2.5.1.

Architecture	Strategy	Main Feature	Remark
DONA	Name Resolution	In each AS, one logical RH is responsible for the resolution process. RHs support two primitives: FIND and REGISTER. RHs forward REGISTER issued by a producer to a higher layer RH and peer RHs and route FIND indicated by the location information from REGISTER.	Global scalability challenge. Single point of failure. Routing overhead of REGISTER dissemination due to the uniqueness of flat named objects. Off-path content finding.
NetInf	Hybrid	By using the name-based routing, a request is forwarded unless a content is located. If the request cannot be routed, the name resolution is activated by sending the request to an NRS to get routing hints/locators. Local and global NRS can indicate a selected locator close to a requester.	Additional resolution delays. Single point of failure. Complexity of multi-level caching. Overhead of distributing NDO information.
MobilityFirst	Name Resolution	GNRS is responsible for mapping addresses to registered GUIDs. A consumer issues a GET message containing the GUID of a desired object. Each CR asks the GNRS for a route to forward the GET message.	Additional resolution delays. Delay of rebinding name resolution information. NDO mapping requires large storage. Single point of failure.
PSIRP/ PURSUIT	Name Resolution	A subscriber issues a subscription packet towards a proper rendezvous point. Rendezvous function matches the subscriber's interest with a specific content provided by a publisher. Routing table creates a delivery path.	Scalability challenge due to scopes and flat-label-based. Higher control overhead. Single point of failure. Additional resolution delays. NDO mapping requires large storage. Off-path content finding.
NDN	Name-based Routing	Find objects hop by hop. Reliable and global delivery. An Interest packet is to request an object, which will be returned back in a Breadcrumb trail. Routing loops can be detected and eliminated.	PIT overflow. Congestion, if there are a highly excessive number of PIT entries especially mapped within a particular Face. Off-path content finding.

Table 2.1: Comparison of the ICN Approaches

2.4 Content Caching

According to the increasing content demand, various techniques of performance optimisations for accessing content have been developed. For example, load balancing is the technique of distributing network traffic across multiple servers in a server farm. This is to ensure that there is no single server that encounters too much demand, e.g., connections, which can improve responsiveness. However, content objects could still be located further away from their consumers, causing highly delivery delays.

Another one of the most well-known techniques of content access enhancement is content caching such as Content Delivery Networks (CDNs) in the existing architecture. Also, in most ICN approaches, content caching is one of their key components as described in Section 2.2. By caching content, a desired content item can be served from a closer cache rather than its original server, which is frequently located further away. So, this will decrease the content retrieval time. To indicate the potential advantages of content caching in NDN over the existing CDNs in the current architecture, the following sections describe some background of CDNs and discuss the comparison of NDN to CDNs in the caching manner.

Note, this work aims to improve content finding in NDN that has different characteristics of content caching compared to CDNs. This section mainly aims to present some background related to the content finding/caching in CDNs that are widely used in the existing Internet architecture leading to the discussion regarding the potential advantages of content caching in NDN that a content finding solution can provide some benefits.

2.4.1 Content Delivery Networks (CDNs)

CDN is a geographically distributed group of servers. The goal is to provide fast delivery of Internet content by caching them and this aims to move content closer to users [70]. In the early days, web content objects were usually served from one sever to all clients. By using traditional web hosting, several websites struggled to perform adequately. Single server distribution is not a sufficient solution for the increasing demand of content (e.g., ultra high-resolution videos, musics, graphics, software, etc.) we consume today. Although adding more servers and employing load balancing could help to mitigate the problem, scalability is still the important issue. While a CDN does not host content, it does help to cache content at the network edge, which can

improve website performance.

As several of content creators are driven commercial enterprises, it is important for them to ensure that their users received the best possible quality of experience. Small delays in the loading of content could be very sensitive for user experience [71]. Hence, the content creators often employ CDNs to guarantee that their users are not discouraged from enjoying multi-media content due to poor connectivity or availability. The basic operation of content caching in the current Internet architecture can be described as follows:

1. A consumer accesses a desired content object by clicking a link to a web page or by using an application.

2. Assuming that the desired content is not already stored in the browser cache or the consumer has not recently visited the web page, the consumer's browser gets connected to the possible nearest cache automatically. The request is then sent to the cache. This is done by using a redirection technique (e.g., DNS resolution) to route the request.

3. If the requested object is available in the cache, it is then delivered to the consumer. In this case, the content will be fetched extremely fast because the content object is physically closer to the consumer.

4. If the requested object is not stored in the cache, the object will be fetched from the original server and ultimately delivered to the consumer. It can be slower for an initial request to the original server but this case will be rarer. Hence, the consumer can always expect an impressive retrieval time due to content caching.

By distributing proxy/caching servers in data centers all over the world, content can be delivered efficiently without straining the origin servers (the locations that stores the original content). A CDN ensures that content can be delivered to a user directly from a possible closest location. This decreases latency and load times, contributing the better quality of experience while consuming desired content. A content object that is already cached in a CDN server can be a potential source for the next users that request the same object from the same area, resulting in faster load times. The CDN server between the user and the original storage server also prevents the origin not being overwhelmed by a huge number of direct requests. Content can

be delivered from different network locations since CDN servers are distributed. This allows content to go viral without systems crashing because of the heavy load.

By increasing the number of Points of Presence (PoPs) [72], the possibility that there is a nearby server to a larger percentage of end-users is increased. This can ensure that the users derived a consistent standard irrespective of where they are located. Placing content topologically to a user offers a more reliable connection between the user and the nearby CDN server. The traffic must traverse fewer networks, which means cutting distance is cutting latency, thereby potentially increasing the reliability.

The composition of a CDN can vary from provider to provider [73] and frequently changes over time. Nevertheless, the key components [74] can often be generalised as itemised below.

- Replica Servers: These servers replicate a content object in order to serve a request performed by a user. They are the core infrastructure of a CDN.
- Origin Server: This server stores the original file or object of the content. The server is possibly operated by the content provider or a designated server in the CDN operation infrastructure. Generally, changes in the content on this origin server will be updated across the replica servers.
- Clients: They are the end-users requesting the content. They may be connected and located anywhere on the Internet. The connections can be through different networks and technologies.
- Redirection Infrastructure: This is to ensure that a user receives a desired content object from a CDN server. Notably, the redirection strategy must ensure that the request performed by the user can be served from a nearby replica server. This guarantees that the user received the best possible quality of experience due to the lower latency.
- Distribution Infrastructure: The distribution infrastructure is to deliver the content stored on the origin server to a number of replica servers. The content can be *proactively* cached on the replica servers before it is requested by a user. This ensures that the user can receive the content instantly because the replica server does not have to retrieve the content from the original sever before forwarding to the user. The content can also be *reactively* cached, which means

there is no content stored on the replica server by default. The content will be replicated on the replica server once initially requested by the user. So, the initial request will be typically slower, but next requests will take the advantage from the replica server that has stored the content already. This reactive content caching is useful in cases where the cache has a limited storage.

- Accounting and Monitoring Infrastructure: The accounting infrastructure allows the provider to precisely charge for the usage of the CDN to their users. The monitoring infrastructure enables the provider to monitor the health of the other elements of the CDN. This ensures the availability and the reachability of the content. It can also inform that which replica servers should situate what content, by considering previous request data, which can maximise the cost savings.

Redirection techniques are crucial to redirect or route a request to a proper CDN server[1]. Several methods are used to achieve these. For example, the HTTP 1.0 specification [76] defines a number of codes that can redirect a HTTP request to an alternate location. For instance, the *301* code is defined as *Moved Permanently*, which indicates that there is no requested content available at the current location and all next requests should be redirect to a given URI. However, a limitation of using HTTP codes is that the servers must know where the content is currently located. In a distribution system, the content can be located in several locations. So, modifying the target of a redirect in a per-client basis is not scalable.

By using the DNS resolution, a given URL will be resolved to an IP address using a lookup to a DNS server. The resolution process can also be used to redirect a request to a geographically closer cache [77]. To achieve this, the DNS server will inspect the source of the request with a corresponding geographical region. A response will then be returned to the user, which will indicate the request from the user to a nearby edge cache. Nevertheless, the same piece of content can be stored with different identifiers within a cache, causing cache duplication and unnecessary disk utilisation. In addition, when the user caches the DNS response, it can result in a slower response to failures. This is because the user is not aware of changes in the location of the

[1]Note, since content can be distributed in more than one location, caching techniques are also important [75], e.g., for the improvement of a cache hit ratio. However, this section mainly discusses about redirection since this can provide the basic understanding of how content can be located in CDNs which is widely used in the existing Internet architecture for the improvement of content delivery that is different from NDN.

content. Content provider may use low Time To Live (TTL) values in their DNS entries, but it can result in frequent DNS cache misses, causing additional delays.

Another technique to redirect requests is performed on the transport layer. Generally, a request routing middlebox or appliance that acts as an initial gateway is required. A replica server will then be selected from a connected group indicated by the appliance and deliver the content to the user without traversing the middlebox again. Notably, the user requires to purchase the intermediary middlebox. There are also additional maintenance costs and changing the configuration of a device can be a time-consuming process.

By using an *anycast* technique, a routing protocol (such as the Border Gateway Protocol) is used to announce the same IP address from several different places within the Internet. However, not all replica servers replicate the same set of content. In addition, most redirection techniques require additional redirection services, servers, or devices that can introduce a single point of failure. The delay of a redirection process can also increase the overall delivery delay.

2.4.2 Comparing NDN to CDNs

Although NDN is a clean state paradigm, NDN can still operate as an overlay on the today's TCP/IP architecture by mapping network interfaces to NDN Faces. Similarly, a CDN is also operated as an overlay on the existing architecture. Both designs share the same motivation that is to solve the scalability challenges of content demand and distribution. Additionally, when content objects (e.g. popular content) are requested by a huge number of users, content servers without CDNs or NDN would confront load or performance problems.

According to the design of CDNs, content objects can be pushed to edge caches that are close to consumers. When a consumer needs a content object, a common solution to find the object is that a host DNS serves the IP address of a proper server/cache to the consumer. Several CDN servers that are distributed in different locations could help to handle the scalable challenges and to improve delivery efficiency. CDNs tend to be a good solution to address a number of current limitations of the TCP/IP architecture. But as content distribution traffic grows, current solutions like CDNs might not be sufficient and more efficient solutions are needed. NDN would potentially be a long-term solution to address the challenging problems.

A number of comparisons between NDN and CDNs have been examined and

discussed. For example, the study of [78] compared NDN to CDNs in several aspects especially in terms of content delivery efficiency. The results of the study prove that under the same topological network conditions, the number of packet loss by using NDN is significantly less than CDN and the average latency of NDN is lower. The studies in [78, 79] prove that under the same network load and bandwidth, bandwidth utilisation using NDN is better and CDNs need more nodes to equal NDN. Similarly, under the same network condition, in [80], CDNs consume more bandwidth compared to NDN because NDN can support different interest requests in every frequently requested path but a CDN supports only limited interests for content at particular caches. It means that by using the CDN, there are more number of interest packets that must be forwarded to original producers compared to NDN. Together, these studies have proved that NDN can perform better especially in terms of scalability and delivery efficiency.

Furthermore, today's Internet is more dynamic and the multi-point communication of NDN is suitable for dynamic caching [80]. NDN does not require a resolution host (e.g., DNS) like CDNs or edge cache servers. Hence, it can also reduce the deployment costs and make networks more robust.

2.4.3 Benefits of Content Caching in NDN

The primary benefit of content caching is the retrieval time. As many content creators are driven commercially, the content retrieval time is the key factor to ensure the best possible quality of experience to users. Assuming that a content producer is located in the UK, a consumer is accessing the content from somewhere in Asia. Even though the server and network platform perform very well, the retrieval time could be hundreds of milliseconds or more. However, content caching can bring the content closer to the consumer and this ensures faster retrieval time. This can also indicate that the consumer would be satisfied in the delivery efficiency of the desired content. Consumers are mostly sensitive to loading delays, which they can leave using unsatisfied services.

In an NDN network, nodes can consume, publish, and forward content objects according to their unique names. Content caching allows NDN nodes (e.g., routers, and end devices) to replicate content in their local caches. When a node or router receives an interest packet, it checks its local cache to seek a corresponding content object by the interest's name. This can reduce the workloads of end producers

and can bring content closer to consumers. The study of [5] demonstrates that caching content in router-level can lead to significant reductions in path distance. More content/replica availability can also result in lower congestion on long paths [7]. Hence, closer replicas provide advantages to both end-users and content producers. With lossy access links, [81] shows that NDN can gain more advantage in finding cached content. Caching also helps to reduce redundant traffic in the Internet [82].

Nevertheless, although the replicas are often cached nearby (depending on the replica density), the standard strategy of NDN is not designed to find these nearby replicas. So, the strategy does not utilise the existing nearby replicas, which can be the better sources for consumers. This means by using the default mechanism of NDN, the appropriate advantages of cached replicas (especially nearby objects), which are distributed to various locations in the network, can not be taken. For example, finding these nearby objects could improve delivery efficiency and cache utilisation.

2.5 Content Finding in NDN

In the previous section, we presented the importance of content caching that is the important key component of NDN. We also pre-announced the particular content finding issues in the former section that can handicap the benefits of content caching. Hence, this section details these particular issues of off-path content finding in NDN, which brings to the main research motivation of this work.

Several content finding proposals have been proposed to solve their motivation problems. These proposals mostly modify both or either the control plane (to initially define default paths from consumers to origin producers) and the forwarding plane (to forward packets by following the defined paths) to support their schemes. Further to the first sub section (Section 2.5.1), the after sub sections discuss those content finding solutions, which are categorised broadly into two main groups depending on their strategies. We also highlight their shortcomings that should be avoided or mitigated in our design.

2.5.1 Off-path Content Finding Issues in NDN

According to the caching function of NDN, a content object can be replicated in several locations (i.e., router nodes, end devices). A content finding strategy finds the object or its replicas dependent on a particular technique. Based on the default

best route strategy, an NDN router searches content opportunistically **on-path** (i.e., along a default path to a producer) indicated by a corresponding FIB's entry. The content can be found at the origin producer or an intermediary cache in the default path. However, the content can also be found **off-path** (i.e., off the default path), which might be closer than the on-path caches.

The strategies that are available in default NDN are illustrated in Figure 2.7 and described as follows. First, the default best route strategy of NDN finds a content object on its default path. Assuming that the node *h1* is a consumer and needs a content object named *"/h3-site/h3/content"*, this node issues an interest packet to be forwarded on-path. To simply describe the calculation of the cost for finding the content, this cost is generally calculated by multiplying the number of hops with a defined latency. In this case, each link is assumed the homogeneous latency of 10ms.

Assuming that the content origin is the node *h3*, the shortest path takes two hops from *h1* to *h3*. Therefore, 2*10 equals 20, which is the lowest cost to *h3*. Longest prefix matching is used and in this case, *h1* tries to match the entire name *"/h3-site/h3/content"* in its FIB. However, this cannot be exactly matched since the default routing like NLSR propagates reachability to name prefixes. So, the shorter name prefix *(/h3-site/h3/)* is instead matched. There are two Faces in *h1* but it selects the Face 1 due to the lowest cost of 20 to send the interest packet. This process happens again at *h2* and the Face 2 is selected because it takes one hop to reach *h3*. The lowest cost coupled with this Face is 10. The interest packet will then be reached at *h3* and the content will be forwarded to the reverse path to *h1* using "the Breadcrumb" left by the interest packet from *h1*.

Nevertheless, assuming that the content *"/h3-site/h3/content"* is already requested and cached in *h5*'s CS, *h1* cannot find the closer (off-path) replica at *h5* because there is no indication to confirm the availability of the content in *h5*. It always follows the default strategy. In addition to the first strategy, second, by using the multi-cast strategy of NDN, the interest packet is forwarded to all outgoing Faces (e.g., both Face 1 and 2 of *h1*). This can increase the opportunity to locate the nearby replica. Third, by using the client control strategy, *h1* specifies an outgoing Face for the particular content name, which can be Face 1 or 2 depending on the node's administrators.

As mentioned in Section 2.4.3, faster retrieval time is the primary benefit of content caching. Considering nearby (cached) replicas can offer advantages, especially in terms of content access. According to the above described default strategy, the desired

content object is found further whilst it is already cached closer. So, this decreases considerable gains especially in terms of delivery efficiency.

As discussed previously, many content creators need to ensure that their consumers received the best quality of experience. Longer delays can be problematic. Even though content caching can bring replicas closer to consumers, content objects are mostly found along the default paths due to the default strategy. Nearby (cached) replicas are often not utilised. This means that cache utilisation is relatively poor whilst caching feature is one of the promising techniques of NDN to address the current content distribution of the today's Internet. Furthermore, the workloads of end producers or other surrounding caches performed by the default strategy can be higher than a strategy that can find nearby replicas. This is because these producers are the terminals of all corresponding requests. The producers themselves or their surrounding caches have to handle several requests from different locations. However, if nearby content can be located, the number of packets in long paths can also be reduced, thereby mitigating the network load including the workloads of end producers and their nearby nodes.

Although the multi-cast strategy of NDN can help to find nearby replicas, excessive overhead can be incurred because of without defining proper scopes to restrict the area of interest flooding. Notably, several data chunks in return can be created because

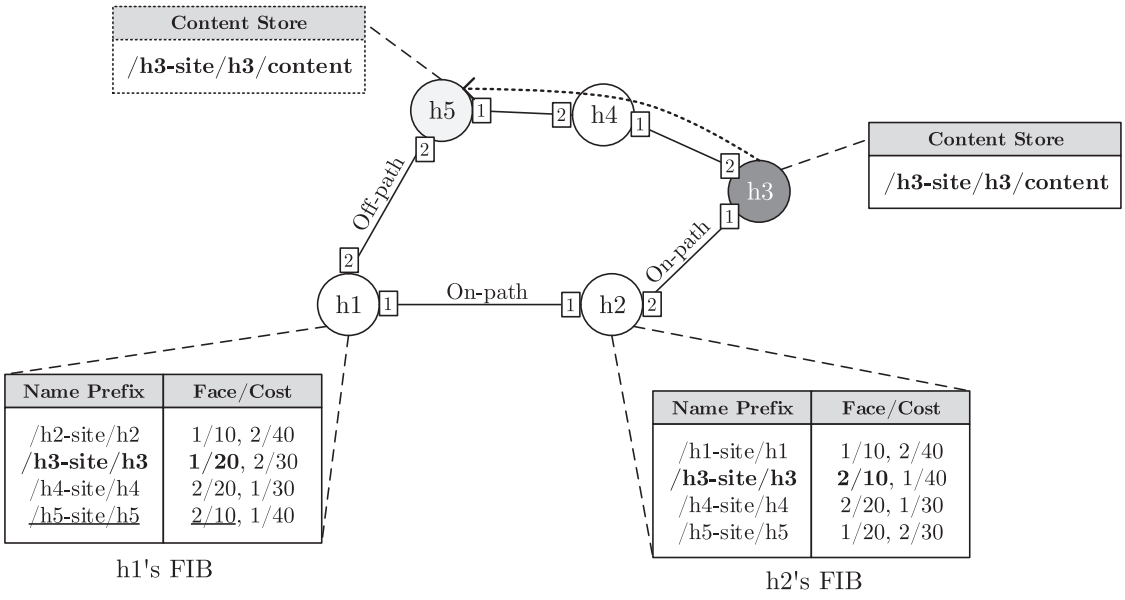


Figure 2.7: Content Finding in NDN

interest packets traverse to many locations. Desired content can be hit in several places, causing higher cache eviction rates in return. By using the client control strategy, it requires good knowledge of the best directions to forward interest packets. Nevertheless, network conditions can change anytime. It is also difficult to configure a node with no permissions granted.

2.5.2 Content Finding Solutions

In NDN, content objects/replicas in each CS allow consumers to retrieve their desired content objects from different sources. Multi-sourcing supports content replica fetching, thereby providing a number of advantages (e.g., a lower delay by fetching a nearby content, and higher cache hit-rate). Several proposals (e.g., [6, 83, 84, 85]) try to pro-actively push a number of replicas into a specific area to increase content availability (i.e., density). The higher number of the replicas of desired content can increase the opportunities that the requested content can be located faster. However, although several replicas are often already cached close to consumers, a few proposals are directly designed to find and gain benefits from these nearby replicas, especially along with the consideration of their density.

To find and deliver desired objects, both or either the control plane and the data plane must be required to create a delivery chain. In the control plane, routers/nodes normally exchange routing updates and calculates the best default paths to proactively construct the FIBs. The information in the FIBs can indicate a default path from a consumer to an origin producer for an interest packet. The actual control plane is stateful, which can adapt to the network changes (e.g., link down, node crashes, new links, or alternate routing paths). According to the constructed routing information from the control plane, in the forwarding plane also known as the data plane, an interest packet is normally forwarded to every next hop that is executed upon the FIB's entries.

[61] assumes that in NDN, the forwarding plane is the actual control plane because the forwarding strategy module makes forwarding decisions on its own. For example, when a failure is detected, a node can send interest packets to other Faces to discover alternate paths. [82] suggests that cached content objects should left behind the control plane due to two reasons. First, the cached objects may change so quickly that it is difficult to keep track of them in the routing table. The data plane might select inappropriate Faces due to the obsoleted routing information. Second, they are

distributed randomly and cannot be aggregated to achieve scalability, which is likely in the context of routing management. Nevertheless, both control and forwarding plane is still applied in several content finding solutions to address their motivation problems. This also depends on their definitions and aspects.

Content finding can be broadly separated into two main solutions: resolution-based and routing-based solutions [86]. In resolution-based solutions, rendezvous points are responsible for mapping content producers with requesters. In a centralised manner, an NRS [87] is required to register/update communication. For example, NDNS [88] mimics the structure of the DNS system in the today's Internet to provide support for cryptographic key distribution and routing management. Each router has "/net/NDNS" prefix in its FIB to further forward an interest packet to perform lookup at an NDNS server. To find content objects in proper paths, using resolution-based solutions (e.g., [36, 89]), requesters are mapped with producers at rendezvous points. Additionally, several resolution-based solutions (e.g., [87, 90, 91, 92]) are designed to locate off-path content objects cached at nearby (or not) nodes. These solutions also map requests to content objects usually at predetermined rendezvous points.

In the traditional CDN, a request-routing algorithm (e.g., DNS-based request routing [93]) resolves a domain name into a numerical IP address of the optimal server. By embedding NDN into the existing CDN framework, in the design of nCDN [94], content objects or their replicas can be routed from the nearest server to consumers straightforward. According to the static server location with resolution-based content finding, consumers know where to find the content objects. However, a server is still required to translate a HTTP request to the NDN interest format, thereby possibly complicating the communication.

To discover off-path cached replicas, an extra routing table at each NDN node is usually required to map or to create paths from requesters towards the corresponding rendezvous points or the off-path caches. For example, a new component table is used to map requests to collaborative routers that cache desired content objects [95]. The extra routing information can maintain either in a distributed manner by using signaling protocols like [54, 96]. Upon a route request, a controller designed in [97] locates a content producer by computing the sequence of router identifiers in the path from its consumer. A specific interest packet installs the new FIB entry on each router along the path and a corresponding interest packet is forwarded in this path. In another approach [98], every subscription request asks the Mediation System (a rendezvous point) to identify the resolution path to request content. [9] has proposed

a design, named Fetching the Nearest Replica (FNR). A tracker server (similar to the DNS system) calculates a path to forward an interest packet to fetch a popular replica. Similarly, in the approach presented in [99], a topology is divided into different domains and a rendezvous controller in each domain is responsible for forwarding decisions.

However, centralised designs may not take the appropriate advantages of the decentralised concept of NDN [100, 101] especially in dynamic topologies (e.g., mobile environments [102]). Their performance can be degraded when there are dynamic and large content demand [86]. Furthermore, by using resolution-based solutions, resolution delay could also be problematic. This is because the increment of the resolution delay can increase the overall delivery delay. A single point of failure could be another important problem [103]. In addition, it is very difficult to map nearby replicas that may change so quickly.

Several routing-based solutions make use of opportunistic information on the availability of content. [104] considers an ideal Nearest Routing Replica (iNRR) scheme that allows to reach the nearest, possibly off-path, cached replica. The iterative algorithm makes use of an oracle providing information on the availability of content in all caches. The scheme selects a Face with the shortest distance to a desired content item. However, according to the authors, the implementation of the perfect oracle is not feasible in a real environment [105].

In the approach presented in [106], each Content Router (CR) has a neighbor table. Each CR broadcasts content identifiers to its neighboring CRs and adds the identifiers to their neighbor tables. The tables allow opportunistic content routing towards CRs that may have the desired content. There are some drawbacks including false misses and false hits. A false miss is when a CR has a new content, but the content identifiers have not updated. A false hit is when the CR does not store the content anymore, but the content identifiers still reflects it.

A number of reinforcement learning approaches have been proposed. INFORM [107] realises distributed reinforcement learning at each network node to discover routes towards temporary replicas. The strategy adheres the family of approaches [8, 108, 109] that uses local information available at each node to quickly react to dynamic item availability. The strategy is to complement NDN with a dynamic INterest FORwarding Mechanism (INFORM). There are two phases including exploration and exploitation when making forwarding decisions. In the exploration phase, there is no learning incurred and a received interest packet is sent using the

best Face indicated by the FIB. A random Face is then selected to send a copy of the interest. This process is repeated again except the best Face will be the one learnt by the strategy and it will be used in the subsequent exploitation phase. Nevertheless, there is no method to handle interest NACKs, which can increase the possibilities of interest loops [110].

Similarly, Multi-Armed Bandits Strategy (MABS) based on reinforcement learning is proposed in [111], which an NDN node floods a request to its all of Faces to explore nearby replicas, when there is no available forwarding information. By using reinforcement learning, the trade-offs between exploring the environment and exploiting which the best Face has already learned should be discussed. For example, in a highly dynamic environment, the costs for the exploring phrase can be higher.

Scoped-flooding [112, 113] is usually applied to find an off-path content object with a certain probability. A hop limit is usually used to restrict a scope. [113] adds a new component called Downstream Forwarding Information Base (D-FIB) to an original NDN content router to track the directions in which the data packets were sent in the past. The D-FIB entries are based on the "Breadcrumbs" [114]. To increase cache hit rates, the strategy opportunistically multicasts an interest both towards its origin producer and the direction indicated by the D-FIB entries. The similar concept of this strategy is extended in [115]. A strategy proposed in [116] send scanning requests to locate close and possibly multiple copies of a requested content. Content Routing Tables (CRTs) are exchanged among content routers to advertise routing information. In an NDN node, the scanning requests are sent to the next hops indicated by its CRT.

Nevertheless, flooding/multicasting interest packets might create several data chunks in return, causing higher cache eviction rates [105]. Flooding requests or meta-requests to discover content locations might also introduce amounts of overheads. However, the proposal [117] limits the scope of the flood to the neighborhood. The results demonstrate that although there are some additional overhead costs, the approach is far from unacceptable and can indeed scale and achieve considerable gains.

In other multipath strategies [118, 119, 120, 121], the forwarding probability of each Face in each NDN node is dynamically assigned by a forwarding weight, which helps to determine the proportion of traffic (e.g., a number of interest/data packets) sent on it. Achieving load balancing and managing congestion control are the main aims of these strategies. In other approaches like [122, 123], several attributes related to real time network conditions are considered to select a Face with the high

probability to forward an interest packet. Nevertheless, by considering many metrics, a high calculation time can be problematic in making a forwarding decision [124].

A multi-path interest forwarding strategy might increase the opportunity to locate nearby replicas. For instance, each NDN router forwards every interest packet to all upstreams according to the supplied FIB entries. However, the increasing overhead can impact the entire network. The broadcast strategy can also cause the interest flooding attacks [62, 125] and DDoS Attacks [126].

A strategy [82] is designed to advertise cached content objects in a limited scope. Nevertheless, it still lacks the quantitative analysis and the concrete implementation for the routing strategy. For example, how exactly does the strategy define a scope and the impact of the scope's size and its overhead costs. Additionally, a number of implementation techniques are not likely stipulated. In considering a strategy presented in [127], nodes need to be partitioned into different reigns to define a cluster. This is to limit a scope of internal cached content announcement. However, the best source for a content object might be located in a different cluster. It is also difficult to define clusters in complex topologies, e.g., highly dynamic environments. Introducing a new component namely Availability Info Base (AIB), is deployed in [95] to keep track of content availability information from other collaborative routers. Notably, the false hit problem should be discussed.

The approach in [86] advertises content from origin servers using a Bloom filter-based routing. This has claimed the advantages over the shortest path approach in terms of overhead costs and delivery efficiency. However, replicas that are often cached nearby could be another potential sources for consumers rather than the origin servers.

In addition, most of content finding strategies do not consider or provide a concrete analysis of the availability of replicas in regard to their densities in the network. We argue that the popularity or content (replica) densities can be the important factor on impacting content finding/delivery.

2.5.3 Content Finding in Mobile NDN Environments

In the current Internet architecture, by using an IP address to identify a mobile host, when the mobile host moves to a new location, the TCP connection identifier causes TCP connection continuity problem [10]. The mobile node requires decoupling between the location of the mobile host from the identifier for the TCP connection

identification. Establishing a new TCP connection is required to continue the connectivity. To solve this issue, Mobile IP [18] is designed to allow mobile devices to move from one network to another with the same IP address. It ensures the connection continuity without dropping connection's sessions. However, it is not an efficient solution because it is still built on top of the original IP design, which is still encountering the limitation of mobility support. Furthermore, in the triangular routing of the design, the content source has to send packets to a home agent before forwarding to a mobile device. Similarly, the mobile device may have to transfer generated content to a home agent first before sending to a requester. This is very inefficient especially in terms of delivery efficiency.

In NDN, a mobile node needs to send an interest packet to request a desired content object and when it is moving to a new location or NDN Access Router (NAR), a desired content is often forwarded to the NAR that the mobile node has performed the request. The desired content object is returned in the reverse path of the request (interest) packet back to the mobile node. When the mobile node has moved to the new NAR before receiving the content object, it re-expresses the unsatisfied interest packet to create/update a new path to its current NAR. So, consumer mobility is natively supported [128]. Thus, NDN has been extensively used as underlying communication paradigm for mobile environments (such as Vehicular Ad-hoc Networks [129]).

Different types of wireless networks have their own characteristics. For example, in Mobile Ad hoc Networks (MANETs), nodes are infrastructure less. In Wireless Sensor Networks (WSNs), nodes can be fixed or mobile. A content finding solution in NDN as underlying communication architecture for MANET and WSN should consider energy efficiency the most [130, 131] because the energy and power consumption of mobile and sensor nodes is the important issue in these types of networks. In Wireless Mesh Networks (WMNs), mobility of node is less frequent and topology is likely static. The networking infrastructure is simplified and decentralised. Hence, decentralised designs of content finding solutions would be feasible to deploy in WMNs. In Vehicular Ad hoc Networks, fast moving vehicles exchange information with other vehicles and Road Side Units (RSUs) [132]. Since VANET nodes are located at the roadside or are connected to the power supply of each vehicle, they are usually not energy constrained [133]. However, content finding and delivery in these kinds of wireless networks frequently encounter the particular issues as described in the following paragraph. These issues are also particularly examined in this work.

Several objects are often forwarded to a previous NAR of a mobile node due to its movement. The mobile node re-issues an interest packet to re-request a content object, when it has missed this desired object according to the disconnection during handover. This is the cause of higher latency and re-transmission rates. These issues are particularly important, if seamless experience is preferred. The previous studies [134, 135] have proved that sending the lost content from a previous NAR can help to improve the re-transmission rates and handover latency. However, NARs might be located in different Location Areas (LAs) [136, 137]. There could be a significant jump from a previous NAR to a new NAR. Finding content only at a previous NAR that might be located in a different LA of the mobile node could result in higher delays.

Nevertheless, several replicas are often cached in the vicinity of the mobile node and other nearby vicinities, especially when the replica density increases. Hence, locating content in the previous NAR, other nodes in the vicinity, and nearby vicinities can increase the opportunities of fetching the desired content from the most possible optimal sources, which can improve delivery efficiency. This thesis focuses on considering the content finding issues in this particular mobile case, since the VCoF design can potentially provide its benefits in this type of mobile networks.

In considering additional related items of work in the area of improving content finding to particularly enhance consumer mobility support in the mentioned communication model, we focus on the main aim of minimising the loss of data during a hand-off scenario and delivery latency. The proposal presented in [138] introduces a centralised architecture to set up a new path for a re-issued interest. A proxy-based mobility management approach is proposed in [139] to handle consumer mobility with a proxy node. This node holds a desired content packet at its repository after receiving a 'Hold request message' from a mobile consumer. The content will be forwarded to the new location of the consumer after handover. A rendezvous point (called RP) based technique is also introduced in [140]. Content can be fetched from the RP of a mobile consumer after hand-off. This technique requires the name of the mobile node's access router as part of the content name. So, the content name is not unique, which can be problematic when it is relocated.

In the centralised manner, many issues have been highlighted in Section 2.5.2. The remarkable issues could be higher control overhead in deploying multiple rendezvous services and a single point of failure. These are also considered in the mobile case. In addition, most of strategies focus on transferring a lost content object, which is

mainly provided by its original producer, to a new NAR of a mobile node. Nearby replicas could not be beneficially utilised by using these centralised designs, however.

As mentioned previously, sending the lost content from a previous NAR can improve latency and network overheads [134, 135]. Likewise, a technique in [140] is proposed, which is also to retrieve data from the previous NAR of a mobile node. The new NAR of the mobile node signals the previous NAR to retrieve corresponding data of the pending interest packets on behalf of the mobile node. The design in [141] introduces Routing Tag, which is the on-path resolver at the previous NAR of a mobile node. The mobile node re-issues an interest packet towards the previous on-path routers indicated by the Routing Tag. The desired content will then be fetched from a previous on-path router. In another approach [142], a new NAR proactively requests and caches desired content from a previous NAR. By taking these into consideration, although previous NARs can be good sources for mobile consumers, the aforementioned problem of significant jumps between NARs can decrease the advantage of these strategies. By considering nearby replicas in the vicinity of every mobile node or nearby vicinities, these can also be the potential sources for leveraging content finding in the mentioned scenario of mobile communication.

2.6 Summary

This chapter has presented background and related work to this thesis. First of all, we started describing the current limitations of the current Internet architecture and discussed most well-known solutions that have been triggered by the current needs while lacking of the coherent Internet architecture. We identified a particular trend of networking, which focuses on finding and transmitting information to end-users rather than exchanging information from host to host. We then pointed out that why the current architecture must be changed from host-centric to information-centric. This has indicated the emergence of ICN, which is expected to be a key component of the Future Internet. When this ICN is deployed to replacing the core of the today's Internet architecture, better scalability, improved efficiency, better robustness are the expected benefits. We also presented the principle of ICN alongside an understanding of how it generally works. The further topic is a number of representative ICN architectures alongside their concepts and shortcoming remarks. Considering these information has led to the reasons of choosing NDN as the base architecture in this work.

To optimise the performance of accessing content, the important technique is content caching, which is an important key of the existing content distribution solutions like CDNs. Also, content caching is one of the key components of NDN. We described content caching and its benefits and discussed a number of work indicated that content caching in NDN can perform better (e.g., bandwidth utilisation, and scalability) than the well-known content caching technique of CDNs.

Although NDN seems to be a potential architecture to overcome the current limitations of the today's Internet architecture, it is still encountering particular issues of content finding because of without utilising nearby (cached) replicas, which have also been described in this chapter. Notably, the primary goal of this work is to address these issues to effectively leverage the efficiency of content finding in NDN. By considering such issues of content finding, we surveyed the solutions straddling several research projects. We described two phases of content finding operations including control plane and data plane. We also identified the main techniques of content finding of the surveyed proposals, which can be grouped into two main solutions: resolution-based and routing-based solutions. This enables an NDN node to obtain routing information (from the control plane) and to make forwarding decisions (in the data plane). The core commonality all of these related proposals of content finding is the ability to find desired content objects at their original producers or off-path caches to ultimately deliver these objects to their consumers.

As the information from the related items of work, we have identified some shortcomings in the context of content finding that should also be avoided or mitigated in our design. These are follows:

- By using the resolution-based solutions, centralised designs can introduce a single point of failure. Multiple resolution controllers incur higher control overhead. Notably, resolution delays can increase overall delivery delays. In addition, it is very difficult to map nearby replicas that may change so quickly.

- In considering the routing-based solutions, although interest flooding can help to find content objects faster because these are often cached nearby, several data chunks in return can be created, causing higher cache eviction rates. Furthermore, trade-offs between additional overhead and performance gains should be considered. By considering many metrics (e.g., the interest satisfaction rate of a Face, congestion, or previous delivery latency), it requires a high calculation time to make a forwarding decision. In addition, there is also

a lack of suitable implementation of cache advertisement solutions especially in regard to content density.

- In the mobility context of the mentioned communication model, although some routing-based solutions prove that requesting content from a previous NAR can improve the amount of re-transmitted data and handover latency, NARs might be located in different LAs, which means there is a significant jump between two NARs. Finding a content object only at a previous NAR that might be located in a different LA can result in a higher delay while nearby replicas in a vicinity or nearby vicinities can be the better sources. By using a centralised service, a new path of a reissued interest packet can be re-calculated, but the issues of the centralised designs can still be problematic. In the next chapter, we also attempt to avoid theses shortcomings in our design.

Chapter 3

Design

In this chapter, we discuss the motivations and aims behind the need to develop a content finding strategy for improving delivery efficiency in NDN. We describe a brief strategy of default NDN and its shortcomings. We then focus on how we can design a more suitable scheme compared to the strategy of standard NDN. We also discuss additional overheads in the context of content finding that could be generated. We then present the designed VCoF scheme and its modules. In Section 3.1, we discuss the motivations behind the design and define the aims. In Section 3.2, we present the development of the VCoF scheme regarding the aims. We then describe how the scheme can be extended in the context of mobility in Section 3.3.

3.1 Design Motivations and Aims

As described in Chapter 2, NDN can bring content closer to end-users. NDN nodes along a default path from an original producer to a consumer cache every newly incoming content object in their CSs. If there are a higher number of requesters in the vicinity of the consumer that request the same content object, it means the replicas of the content object can be significantly cached in the specific area or in the neighborhood of the consumer. This increases the replica density in the network. When another consumer needs the same content object, several caches nearby can be the potential sources for this consumer. However, the existing concept of the default best route strategy of NDN cannot naturally find the desired content from these nearby sources, causing sub-optimal delivery efficiency.

Therefore, in this chapter, we aim to design the new scheme called *VCoF* that

allow an NDN node to be able to find nearby replicas that could be the better sources. If VCoF can help to find the nearby replicas, it must help to reduce delivery delays because desired content objects can be found closer, resulting in better delivery efficiency. This chapter also describes the detailed operations of VCoF and how it can improve content finding compared to default NDN.

3.1.1 Addressing the Challenges of Content Finding in NDN

In the forwarding plane of NDN, when an NDN node (a consumer) needs a content object, an interest packet is created to be forwarded to other NDN nodes to find the content object. Normally, the name prefix of the interest packet is used to identify the default routing path between the consumer and the original producer of the content. In the caching process, every NDN node along the default path stores the content object in its CS. When another consumer needs the same content object, it can be unintentionally found at any node that caches the content object (e.g., the intermediate node in the default path of the request performed by the previous consumer).

To address the challenges of content finding in NDN, it is important to understand the fundamental (i.e., strategy) of default NDN. This is previously described in Section 2.5, which can be summarised as follows: (1) There is a FIB table in each NDN node that stores the name prefix of each other NDN node in the network. A name prefix is mapped with costs (e.g., number of hops, or link delays) that can indicate the distances to its location. Each name prefix is exchanged among NDN nodes using a routing protocol such as Named Data Link State Routing Protocol (NLSR). (2) To forward an interest packet, the name prefix of the packet is mapped with the name prefix in the FIB to determine the costs to the source of the content object. Every Face ID associated with each lowest cost is then selected to forward the interest packet to every next node along the default path to the original producer or unless the interest reaches an intermediate node that can serve the content object. (4) The content object is then returned to the consumer in the reverse path using the information in the PIT of each node. While the content object is returning back to the consumer, each node along the path stores the received object in its CS.

According to the aforementioned strategy, we can see that the mechanism aims to find the content object in the default path, which is mostly at the original producer. The NLSR can exchange only the name prefix of each producer to define each default

shortest path for every consumer. So, it is difficult to locate any replicas that are located off every default path. It means we can not gain sufficient benefits of the objects that might be cached nearby, such as effective cache utilisation and better delivery efficiency. By caching content objects, several nodes can be the better sources for consumers. However, a content name can only map with an exchanged name prefix provided by corresponding FIBs to forward an interest packet indicated by only the name prefix. Hence, we aim to design the VCoF scheme that can help every consumer to locate nearby replicas by adding extra mechanisms into the current NDN architecture.

3.1.2 Discussion of Designing the VCoF Scheme

In our design, the main goal of VCoF is to find content objects that are already cached nearby. This aims to increase delivery efficiency. Hence, a comparison between fetching content using default NDN and our strategy should be made to understand how the design can improve content delivery. It is challenging to design VCoF to find the nearby content/replicas since in the NDN forwarding plane, there is no indicator that can help to indicate any interest packets to find these nearby replicas.

Content distribution in the network can be the important factor that can impact content delivery according to the caching fashion. As mentioned earlier, the different number of content or replicas can impact different content finding results. So, we need to observe the content finding situations by comparing VCoF against default NDN with different content (replica) densities. This is to understand how VCoF will perform from the case of low replica densities to the case of higher replica densities. For example, when the number of the replicas of desired content is low, the opportunities to find the content should also be low. In this case, an original producer might be the appropriate source for requesters. When the replica density increases, the opportunities to find the desired content should be higher. It might be opportune to locate nearby replicas to leverage delivery efficiency.

Additionally, another factor that can indicate the efficiency of a content finding scheme is content finding overhead. For example, the increased number of requests mean the higher number of packets for finding content. So, the overhead costs of content finding can also be increased. The design of VCoF should also consider the effects of these additional overhead costs.

Another aim of any caches is to provide network efficiency. A high number of

interest packets can impact the entire network traffic. If there is a high number of requests, it can increase the overall number of packets in the network that might overwhelm each NDN node. Similarly, by using the multi-cast content finding scheme, an interest packet is generated and flooded. Although this technique might help to find off-path replicas and desired content can be fetched faster, several interest packets might be generated and forwarded into the large area of the network, causing highly excessive overheads. Furthermore, the multi-cast strategy can also increase cache eviction rates, if there are several hits for desired content objects. These are fetched from several nodes and they might replace several existing content objects in the CS of each cache. So, many replicas might be removed and it might be significantly difficult to find them, when a consumer needs these replicas. Hence, the design must also realise these issues.

3.1.3 The Effects of Vicinity

If there are several consumers in a particular area that requests the same content object from an original producer, many replicas are often cached in several nodes in the area due to the caching mechanism of NDN. It is difficult to locate these nearby replicas because there are no any routing information that directly indicate content finding paths for them. From our observation, replicas are usually replicated in the area dependent on their densities. We can gain benefits of the existing nearby replicas, which they can be the best sources for end-users. Hence, we prioritise nearby replicas first and then their original producers.

Finding nearby replicas may cover a different or larger scope of content finding compared to default NDN, which can introduce some additional overheads. To limit the area of content finding for mitigating excessive/unworthy overheads, we should design a proper content finding scope. Based on our research hypothesis, we define this scope called a "vicinity". The vicinity can expand the larger or alternate view of content finding, which increases the opportunities to find the replicas that are located nearby. This is the reason that we call this content finding technique the "VCoF" (Vicinity-based Content Finding) scheme.

As mentioned in Section 3.1.2, interest flooding can help to find off-path replicas, but highly excessive overheads can be incurred. So, in our design, we try to limit the scope of content finding using the vicinity to mitigate the overheads. In our definition, a vicinity means a set of nodes that are connected to a requester node

in a different number of hops (i.e., distance). By looking into a vicinity, we might find desired content objects that are located closer compared to distances to original producers. We need to design some indications that can help an NDN node to know the availability of nearby content. This availability information should help to indicate forwarding paths to the desired content locations.

The size of a vicinity is also important to define the scope of content finding. It should not be too wide because several spending packets to find a single piece of a content object might outweigh the benefits. The increasing number of requests can also increase overhead costs at each NDN node. A high number of requests mean a high number of processes that needs to be executed. So, it can increase the workload of each NDN node. In addition, the data volume in each node might also be increased due to the higher amount of packets (i.e., requests). Hence, a comparison between fetching content objects at original producers and finding the content objects in different vicinity sizes should be made while considering the different number of requests.

In different vicinity sizes, content popularity can also impact on content finding results. For example, finding content objects in a high content popularity case even considering a small vicinity might still help to find the content faster because the desired content objects are already distributed nearby. As mentioned previously, vicinity sizes can be the important factor to indicate the performance of content finding. Hence, the vicinity in our design should be adjustable. We can then examine the favorable vicinity size(s) by increasing small size(s) to larger sizes in the evaluation.

3.1.4 Content Finding Overhead

By using the NDN's best route strategy, an interest packet is created at a consumer and is then forwarded to an original producer indicated by the name prefix of a desired content object. The interest packet is forwarded hop by hop to the original producer. The overhead of this mechanism is the interest packet that is created in each hop along the path from the consumer to the producer. A higher number of hops that the interest packet have traveled mean the higher number of packets. Thereby, these packets may increase the network traffic in long paths, and the number of processes, causing the higher workload in a number of nodes or the entire network. The VCoF design can add some overheads compared to default NDN due to the designed scope of content finding. The design of VCoF must realise on the overhead costs that might

create some effects to the NDN nodes and the entire network. However, these overhead costs might be worth to spend, if the trade-off between the overheads and delivery efficiency is acceptable.

For example, by looking for a desired content object in a vicinity, the higher number of packets might be created to learn the content availability in each node in the vicinity. Thereby, the message overhead can be developed. In some situations, this message overhead might outweigh the benefits. For instance, in a too-large vicinity, when a distance to an original producer is not very different compared to a distance to a node in the edge of the vicinity. It might not be worth to spend so many packets to find a content object instead of fetching it directly at the original producer. In contrast, when the producer is located further away, it might be worth to spend a higher number of packets in a larger vicinity to locate nearby content objects. Hence, to understand the effects of additional message overhead, comparisons between default NDN and VCoF should be examined.

As mentioned in the previous paragraph, a high volume of packets can impact network efficiency. When there are several packets at an NDN node, the data volume should also be increased that can cause the incrementally required storage in the node. Hence, to understand the effects of the number of packets to the data volume, comparisons of data volume generated by the related packets of content finding between default NDN and VCoF should be evaluated. As mentioned previously, the benefits of the design are not for free. It is difficult to provide a better solution without additional overheads. So, we must consider the tread-offs between performance gains and the content finding overheads.

3.1.5 Flexible Deployment

One of the keys of the VCoF's design is that it should be flexible to be deployed in different network scenarios. For example, in a dynamic scenario that a requester node might move from a location to another location while requesting a content object. The requested content object might be forwarded to the previous location of the requester. If the node moves in the same vicinity, VCoF can benefit this because it can help to find the content object in the previous location, which is mostly located in the vicinity. In another example, when a source node might be going offline but it already forwards the content object to a requester. By using default NDN, it may not be able to locate the content object in the off-lined producer. However, the replicas

of the content object might be cached nearby already. So, VCoF could still help to find a replica at a node in the vicinity that replicates the content object in its CS.

In our design, the VCoF scheme is a decentralised solution, which can eliminate the issues of centralised designs that can make the scheme inflexible. The centralised designs may not take full advantage of the decentralised concept of NDN [100, 101]. For example, a central sever that keeps the locations of replicas could be a single point of failure. Where there are a high number of requests to the single server and the requests cannot be served, the system might fail to service. In addition, in a dynamic topology, several nodes might change their locations frequently. The central server might not be able to update the locations of these nodes instantly. Hence, the decentralised-based system is more flexible to be deployed in different situations. Thereby, this can potentially support various kinds of networks.

3.1.6 Feature Summary

In this section, we have identified a number of features to be used in the design and the development of VCoF. In table 3.1, we list a summary of each functionality and evaluation of the design that we aim to achieve.

Content Availability	Provide content availability to advertise locations of content/replicas to nodes in a vicinity
	Develop a *Content List* to store names in a CS and to advertise the content availability
	Modify an interest packet to push a *Content List*
	Select neighborhood in a vicinity
	Evaluate the effects of vicinity sizes
Content Finding	Improve delivery efficiency by considering extra routing information
	Develop mechanisms to forward an interest packet to locate a corresponding content or replica in a vicinity
	Select the better source between a default path to a producer and a nearby node in a vicinity
	Evaluate the effects of content density/popularity
	Evaluate delivery efficiency alongside the consideration of content finding overheads
	Extend VCoF in a dynamic environment

Table 3.1: Feature Summary

3.2 The VCoF Scheme

The main objective of this work is to find nearby replicas based on the vicinity concept. This is expected to improve content finding in NDN. The code-bases of NDN are modified to support the VCoF scheme. In considering the design, the scheme is a routing-based solution, which can avoid the shortcomings of the resolution-based solutions as discussed in Chapter 2 (Section 2.5). We also try to eliminate or mitigate the number of shortcomings of the other routing-based approaches as also reviewed in the previous chapter.

The core idea is conceptually described in Figure 3.1. The processes can be detailed as follows: The consumer C needs the content object "/P/content" from the producer P. This original producer is located further away (on-path) from the consumer C. We assume that the object "/P/content" is already requested by the consumer R. It means the object is already replicated in the consumer R's CS. In considering the location of the consumer R, it is located in the vicinity of the consumer C (off-path). By comparing fetching the replica at the consumer R with locating the origin content at the further producer P using default NDN, the delay to the producer should be higher.

To find the nearby replica, the consumer C must know the availability of the "/P/content" in the consumer R's CS. So, we design a *Content List*, which is used to store and advertise the name of each content object in an NDN node's CS. In this case, the consumer R pushes its *Content List* to every node in its vicinity. After receiving the list from the consumer R, the consumer C knows where to find the content object by sending an interest packet towards the reverse path to the owner

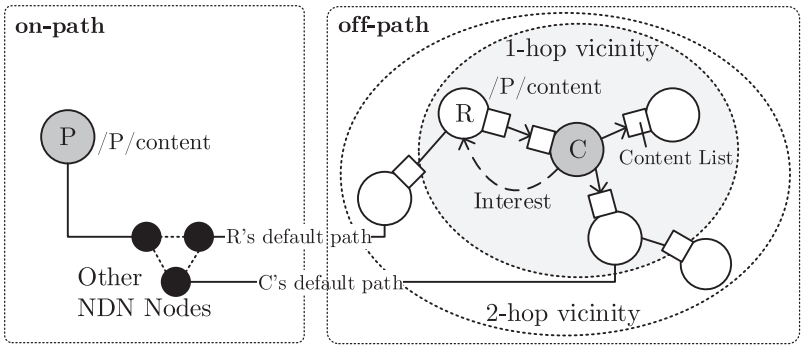


Figure 3.1: Core Scheme

of the list (consumer R). So, the content object can then be fetched closer at the consumer R rather than the further producer.

In our VCoF design, we employ two main modules to process incoming interest packets and data packets as illustrated in Figure 3.2. The first module named *"Content Finding"* module is responsible for finding content using the proactive routing information generated by the second module called *"Content Availability Advertisement"* module. The first module contains three main processes including the *Content List Checking* process, *Face Selection* process, and *Forwarding* process. An incoming interest packet is checked to find a corresponding entry in a *Content List* pushed by another node in the vicinity. This is the process of the *Content List Checking*. If the name is matched in the list, by comparing the lowest cost of a Face to the owner of the list to the lowest cost of a Face mapped with the default name prefix (i.e., two candidate Faces), the better Face to forward the interest packet is selected, which is performed in the *Face Selection* process. This ensures the proper Face to be used in the *Forwarding* process.

The second module is responsible for content availability announcement in each vicinity. It consists of three main processes including the *Caching* process, *Content*

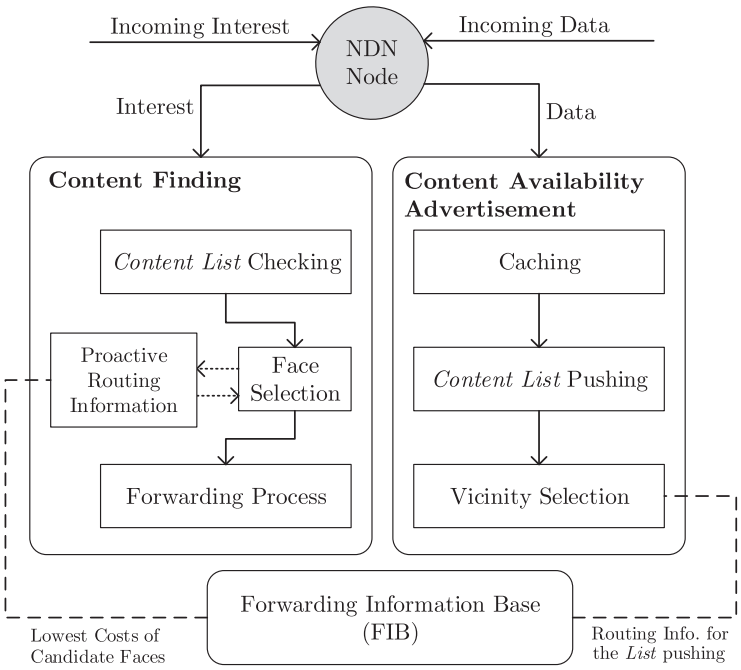


Figure 3.2: The Overview of the VCoF Modules

List Pushing process, and *Vicinity Selection* process. In the *Caching* process, an incoming data packet is cached in an NDN node's CS, which is the default mechanism of NDN. This means the CS is updated. So, the process of the *Content List Pushing* is then activated to advertise the update of content availability of the node to other nodes in its vicinity. Notably, there are several ways to push a *Content List*, which will be discussed in Section 3.2.2.2. To push a *Content List*, the scope of the list pushing should also be considered. A node has to select every node in its vicinity using the *Vicinity Selection* process. FIB of every node manages all of routing information regarding the operations of both designed modules as the overview in Figure 3.2.

3.2.1 Content Finding Module

In this module, as illustrated in Figure 3.3, when a node receives an interest packet that is generated by a consumer. The name of the interest packet is firstly checked to find the corresponding content in the node's CS (cache). The content object will be returned back to the consumer, if it is found in the CS. Otherwise, the name is then checked to seek a corresponding entry in any *Content Listi* containing the name of each entry in the CS of every owner of the list. Each name in a list is mapped with a Face that has received this list with the cost to the owner of the list. This means we can look up the corresponding name to select the Face ID supplied by the FIB with the lowest cost to forward the interest packet to locate the content object at the owner of the list.

The node then compares which Face to the owner of the list or to a default corresponding FIB's entry (e.g., Face to the original producer) that has the lowest cost. The module then selects that Face to be used to forward the interest packet to the next node. In this next node, the same process is repeated again until the interest packet arrives at the node that can return the desired content object back to the consumer. If the name is not in the list, it potentially means the desired content can not be found inside the vicinity. In this case, the default strategy of NDN is activated.

3.2.1.1 Content List Checking

In the default NDN best route strategy, an NDN node selects a Face with the lowest cost to fetch a content object indicated by its name. It does not consider nearby replicas. Hence, to find these replicas, consumers must know what and where content

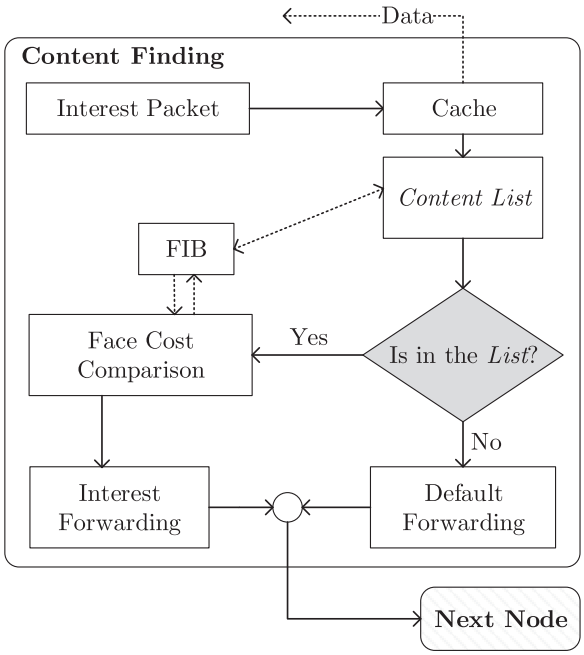


Figure 3.3: Content Finding Module

objects or replicas are in their vicinity. This is done by using the designed *Content List* to advertise the content availability. In this section, we describe the details of the *Content List* and the importance of the *Content List Checking* process.

If an NDN node is a producer, it stores its own content objects in its CS, but if it acts as a forwarder, the CS also stores replicas of other producers. To provide the reachability of the content objects/replicas, we design a *Content List* that contains the names in the CS. The list is used to advertise the availability of the content objects and replicas in the local cache of a consumer to other consumers in its vicinity. To reduce the list's size, we ignore a number of default names (e.g., NLSR messages and local names).

To advertise the content availability of a node, its *Content List* containing each name in its CS will be pushed to neighborhood in the vicinity of the node. When another node in the vicinity receives the list, a Face ID and the lowest cost to the owner of the list is proactively mapped with each name in the pushed list to be used as the routing information to forward corresponding interest packets. We will describe the in-depth details of the *Content List* pushing process in Section 3.2.2.

The *Content List Checking* process enables an NDN node to seek a corresponding *Content List* for a particular interest packet. If the name of the interest packet

matches an entry in a pushed list, the replica of the desired content can be found nearby at the owner of the list. It is important to note that each name in the pushed list couples with a Face with the lowest cost to the owner of the list. We expect that the distance to the owner of the list is potentially shorter than the distance to the original producer of the content. Nevertheless, the producer can be located closer. So, the comparison of the costs to the owner of the list and to the default node (e.g., the original producer) should be made to select the better source.

3.2.1.2 Face Selection

In the *Face Selection* process, we develop the Face comparison mechanism to find a proper Face to forward an interest packet generated by a consumer. The name of the interest packet is mapped to seek a corresponding entry in a *Content List* and a default entry in the FIB to determine a better forwarding Face. By using the default NDN strategy, there is only one Face with a default name prefix in the FIB that can indicate the path to fetch the content object. This is because the default routing protocol like NLSR propagates reachability to name prefixes. The replicas of the content object that are located nearby (i.e., off-path) cannot be found since there is no name coupled with a Face and a cost (i.e., routing information) to indicate the interest packet. By using the designed *Content List*, we can know the content availability of a node in a vicinity. Each name in the list mapped with the lowest cost of a Face to the owner of the list is compared with the lowest cost of a Face provided by a default name prefix of a producer in the FIB. The Face with a lower cost of these two candidate Faces is then used to forward every corresponding interest packet to locate its desired content or replica.

By mapping the name in the *Content List*, the lowest cost of the Face ID can then be found and selected. It will be compared to the lowest cost of the Face ID indicated by a default name prefix (mostly to the original producer) in the FIB to determine that which path is shorter. If the path to the original producer is shorter, the interest packet is then forwarded to the Face with the lowest cost to the producer. Otherwise, if the path to the owner of the list is shorter, the interest packet is then forwarded to the Face with the lowest cost to the owner of the list. This ensures that a better source is selected to forward the interest packet to find the desired content object, which is performed in the *Forwarding* process.

3.2.1.3 Forwarding Process

When the consumer starts to request the content object, it then checks the existing of the content's name at each pushed *Content List*. At the current stage, this name can be found in a pushed list and the consumer knows where to find the replica. According to the routing information obtained from the *Face Selection* process, the consumer or each intermediate node selects the Face with the lowest cost to forward the interest packet after comparing the distance to the owner of the list with the distance to the original producer.

If the name cannot be mapped with any name in each *Content List*, it means a replica that has the same name cannot be located nearby. Hence, the default strategy of NDN is then activated. The interest packet will be forwarded in the default path to the original producer using the default forwarding strategy by determining the default name prefix in each node's FIB.

There could be other Faces mapped with the similar name but we focus on the Face to the nearest replica. The interest packet is originally created at the consumer will be reversely forwarded to each Face of each node unless it reaches the node that is the owner of the list or any intermediate node that can serve the replica. The replica will then be fetched using the mechanism of "Breadcrumbs" without sending any requests to the further producer. In the worst case, if the interest packet arrives at the owner of the list but the desired object has already been removed or evicted, it will be forwarded to the default path from the current node of the interest to its original producer. Therefore, content delivery is highly guaranteed.

In considering the issue of routing loop, when an interest packet has arrived at a node, it will not be forwarded to the Face that has received this packet (i.e., to downstream node(s)). The interest packet is always forwarded to upstream node(s). Even if routing information provided by any content lists might indicate the interest packet to its previous node (i.e., downstream), the Face to this previous node will not be the choice for making a routing decision, thereby mitigating the routing loop. However, in some cases (e.g., in a ring topology), a content finding loop can be happened, if there is no any corresponding data cached in any nodes along the finding path. This data might be removed or replaced by other content and every node always forwards the interest to every Face that might route the interest back to its original node. Nevertheless, this issue can be rarer, if the availability of the desired content is higher and in the networks (e.g., core networks) that offer several alternate routing

paths (i.e., connections). However, addressing the issue is still preferred and this can also be handled by some further mechanisms described in the following paragraph.

To mitigate the issue that a content object advertised by a pushed *Content List* can not be found at the owner of the list due to a cache eviction, some additional mechanisms to the current VCoF scheme need to be considered in the Future. The brief of this is that the entries in the *Content List* do not synchronise with the availability of the content cached in the list owner. This is the current limitation of this work but it can be handled by further mechanisms. For example, we can define a timeout for a *Content List* and all of the content in this list (e.g., $x\%$ in the cache) are not allowed to be replaced by other content unless reached the timeout. Every node that has received the list also knows this timeout and does not forward any interest to the list owner if timeout exceeded. The timeout can be renewed with a newer *Content List*. In addition, we can also push a list by considering only a number of recent content in a cache. This can reduce the opportunity that the recently cached content objects might be replaced if we consider a particular caching policy, e.g., FIFO. In summary, a number of solutions can help to handle this issue and need to be investigated. However, in the early stages of this work, we focus on proving that nearby content replicas in every vicinity can indeed be the better source for content finding.

3.2.2 Content Availability Advertisement Module

In an NDN node, to find a content object, an interest packet is generated and this module of the node checks the content name with the latest *Content List* of this node. This is because the content that the node is trying to find might be already cached in the node itself and the availability information of the content might be already pushed. Hence, there is no need to push the current list again. Otherwise, if this name is not in the list, it means the node is looking for a new content object and the updated *Content List* will be pushed. This is done after the operation of the *Content Finding* module is complete and the desired object is found and has arrived at the node. It should be noted that this is the current mechanism for pushing a *Content List* used in this work. There are some limitations and different list pushing strategies as discussed in Section 3.2.2.2.

When the data packet of the desired content arrives at an NDN node, it is then cached in the node's CS. Assuming it is a new object in the CS, the content name is

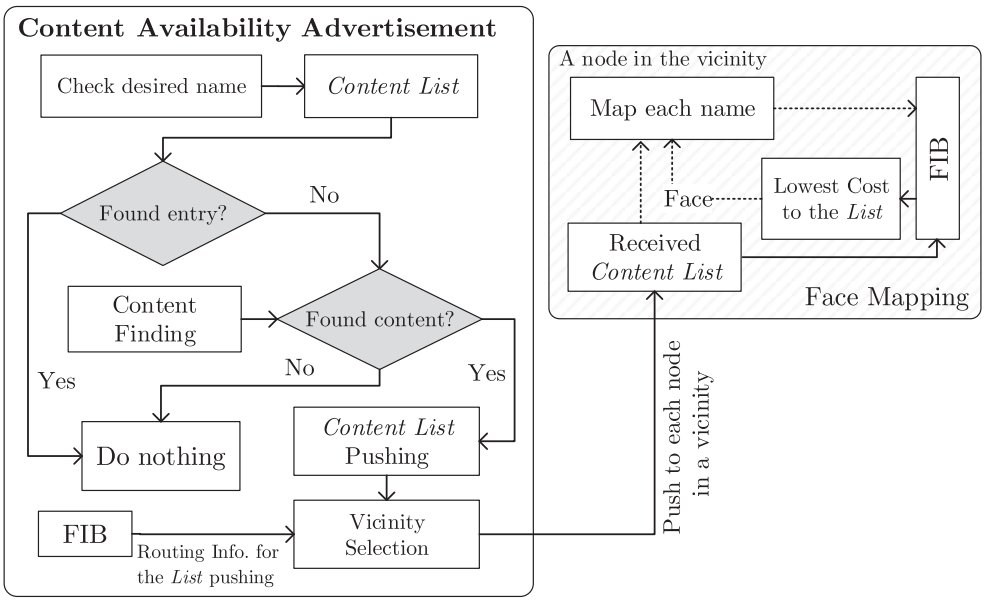


Figure 3.4: Content Availability Advertisement Module

then added into the the node's *Content List*. The list is then pushed to neighborhood in the node's vicinity[1]. As presented in Figure 3.4, once a neighbor node receives the list, each entry in the list is proactively mapped with a Face that has received the list with the lowest cost to the node who has pushed the list (i.e., the owner of the list). Every entry is then stored in the neighbor node's FIB to be used as the extra routing information to select a proper Face for any corresponding interest packet that is looking for the same content object as described in the *Content Finding* module. The lowest cost to the owner of the list is calculated by the routing information in the FIB. If there are several nodes that provide the same lowest cost, we consider the latest node who has pushed its list to potentially ensure that the content object is fresh and available to be located.

In summary, we design a *Content List* for two main reasons. First, we use the *Content List* to be pushed to advertise the content availability of the node that is the owner of the list. When the list arrives at another node, each entry in the list is then mapped with the Face (with the cost) that has received the list. This is the lowest cost of the Face to the owner of the list, which will be used as the extra routing information. Second, we use the *Content List* to check a corresponding name of an interest packet to get a Face with the lowest cost indicated by the list in comparison to

[1]Note, the details of the vicinity-based concept are described in Section 3.2.2.3

a Face with the lowest cost indicated by a default entry in the FIB before forwarding the the interest packet out of a node.

3.2.2.1 Caching

Before searching for a desired content item, if the content's name is already listed in the latest requester's *Content List*, it means the content object is already cached in the node who performs the request. So, we might not need to announce the list again to reduce the redundant of content advertising. Otherwise, the name is newly added into the node's *Content List* after the arrival of the data packet. It will be cached in the node's CS and the *Content List* is updated. To advertise the list, the updated *Content List* will be pushed to neighborhood in the node's vicinity. The process of *Caching* notifies the *Content List Pushing* process. Notably, we might not need to check the list, if we use another list pushing strategy (e.g., a periodically pushing strategy).

3.2.2.2 Content List Pushing

List Pushing Concept

The core idea is that each NDN node has its *Content List* to be pushed to advertise the availability of content objects and replicas in its CS to other nodes in its vicinity. A node who needs a content object can try to find its nearby replica by looking at the names from each pushed list first.

Various techniques can be used to push *Content Lists* and these can be adjusted in different situations. For example, a list is pushed after updating an entry in a node's CS. When a data packet arrives at an NDN node, it is cached in the node's CS, which is the default operation of NDN. After adding the content into the CS, the process of *Content List Pushing* is activated. Before pushing the list, the name of the content object is firstly checked. It will be mapped with each entry in the latest list. If it can be mapped, it means there is the same name as an entry in the list. We do not need to push this list again because of the duplicated name. Otherwise, if the name cannot be found, it means it is a new entry and the list is updated. So, the list will be pushed to the neighborhood in the vicinity. This work uses this technique to push *Content Lists* because the content availability can be instantly updated. However, the overheads of pushing the lists should be considered.

The *Content Lists* can also be pushed periodically such as every n seconds. In some

cases, we do not need to push the lists to advertise the availability of all content in a cache. For instance, an NDN node can also push its *Content List* by considering the changes of some particular content (e.g., popular content). This means the overheads of content list pushing can be mitigated or optimised compared to the current strategy in this work. However, this should be further investigated.

Limitations of the List Pushing

In this work, we push a *Content List* instantly, after a node has received a new content object. This might generate a high number of overheads in some situations. For example, if this pushing mechanism is deployed in core routers that usually see many updates, the increment of the overheads of list pushing could be problematic. As mentioned previously, the mechanisms for the list pushing optimisation can be further investigated and developed. For example, we do not need to push the list instantly. The list can be pushed every n second, considering the changes of some particular content (e.g., popular), or occasionally pushing for a number of recently cached objects by defining a *Content List* lifetime. However, there could be another problem, if the list does not synchronise with its content cached at its owner since an object indicated by the list might be removed or evicted. Nevertheless, this could be further handled by some techniques (e.g., as previously described in Section 3.2.1.3).

In considering consumers or edge networks, nodes might see a lower number of content updates compared to core networks. So, pushing the list instantly could be possible but if the consumers or the edge networks detect significantly excessive costs of the list pushing, the aforementioned mechanisms can also be deployed to mitigate the costs. Hence, different ways of *Content List* pushing can be used in different situations or different parts of the networks. Nevertheless, this work uses the mentioned list pushing strategy since we assume that desired content can be located according to the routing information advertised by every *Content List*. This firstly aims to prove that nearby content in every vicinity can indeed be the better source to improve delivery efficiency. Additional factors with more complex situations could be challenging in the Future work.

3.2.2.3 Vicinity Selection

By centering a consumer node, a vicinity contains a set of NDN nodes whose are connected to the consumer in different distances. A distance is the number of hops to the central node and is less than or equal to a threshold. We define the scope

of the vicinity called "vicinity size", which is a threshold number. For example, as illustrated in Figure 3.5, when the vicinity size is set to 2, it means that the coverage area includes every node that connects to the central node C in 1-hop and 2-hop distances. The number of nodes in the vicinity depends on the topology and the number of links around the consumer.

This work expands the consumer's view into a vicinity. Unlike the narrow view of the current NDN paradigm, the consumer can gain higher advantage of the designed scope especially in terms of content finding. The vicinity is a limited scope that can help the consumer to find a desired content object in a proper area of neighbors. However, the scope can be too broad or too narrow. Thus, the experiments will also investigate the impact of different vicinity sizes.

In the *Vicinity Selection* process, to select a vicinity (i.e., the area of a *Content List* pushing), a node that is the central node should define the vicinity size[2]. The vicinity might be defined in a configuration file by system administrators and in this work, we define vicinity information in each node's configuration based on the evaluation topologies, which will be presented in Chapter 5. To select paths to push a *Content List*, we modify the current interest packet structure to be able to carry the list. The modified interest packet is pushed to its destinations indicated by each name prefix

[2]Note, in this thesis as detailed in Chapter 5, we evaluate VCoF by considering different vicinity sizes. So, we assume that every experiment defines its vicinity size depending on the experiment setup. However, to be more practical, vicinity sizes can be varied in different parts of the network and this can also be further investigated in the Future work.

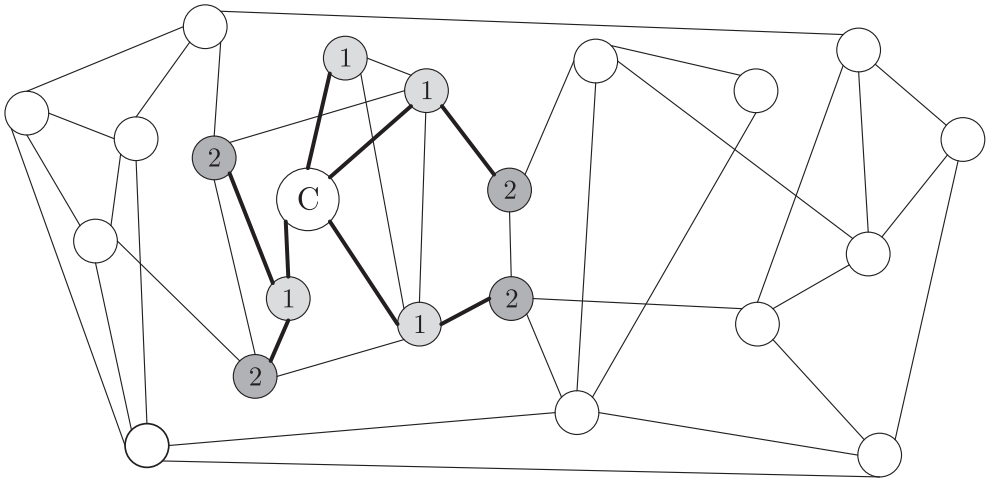


Figure 3.5: An Example of a 2-hop Vicinity

of each node in the vicinity. To mitigate the redundant of advertising a particular content, we ignore pushing a *Content List* containing the name of the object that has been recently forwarded to its requester back to the owner of the list. This is because this node already stores the content in its CS and the content is potentially available at the list owner.

The possible routing cases using the VCoF scheme are presented in Figure 3.6. In considering these cases, VCoF not only helps to find nearby replicas inside a vicinity, but also increases the opportunity to locate replicas in nearby vicinities. For example, if a replica cannot be found inside a consumer's vicinity, the interest packet is forwarded to the default path. Based on the operations of VCoF, when a node in the default path receives the interest packet, it then checks the corresponding *Content List* to compare the distance to find the content in the default path to the distance to the owner of a list. In this case, we assume that this node is fortunately a member of a nearby vicinity, which another node in this vicinity is storing the desired replica. The replica can then be found in this nearby vicinity instead of the further producer.

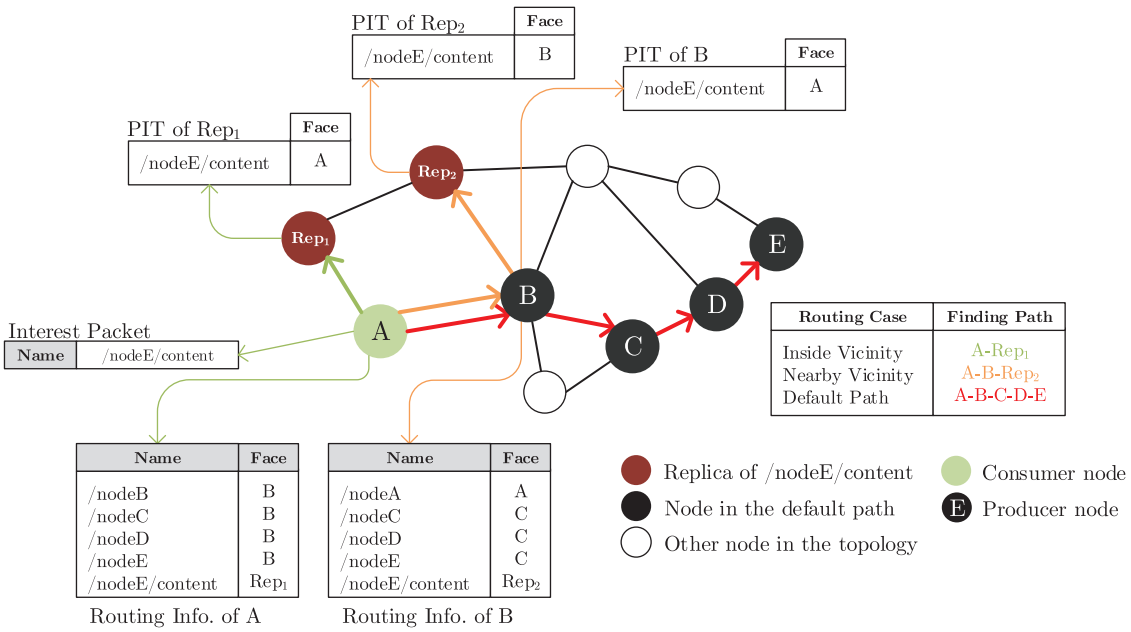


Figure 3.6: Routing Cases

3.3 Mobility Support

Different types of wireless networks have their own unique communication models. We are interested in a particular communication model as discussed in Section 2.5.3. The brief summary of the model and its issues are described as follows: When a mobile node moves to a new location, a requested content object is usually cached in the previous location of the mobile node. The mobile node might have missed the content object due to disconnection during handover. By using default NDN, the mobile node re-transmits the interest packet to re-request the object again at the new location. A new path of content finding is re-defined, causing poor delivery efficiency and higher re-transmission rates.

Notably, the previous location and the new location are often in the same vicinity. A number of related items of work have proved that retrieving the content object from the previous location can reduce the delivery delay. However, the previous location and the new location might be located in different Location Areas (LAs), which means there is a significant jump between them. Finding the content only at the previous location can be insufficient to improve the delay whilst other nodes in the vicinity and nearby vicinities can be the better potential sources.

To be specific about the mobile communication model that is focused on this work, one example that can also potentially indicate the benefits of VCoF is Vehicle-to-Infrastructure (V2I) communication in VANETs. For instance, assuming that a mobile node (e.g., a car, or a train) is moving in a particular direction, there are a number of access stations (i.e., NDN Access Routers) that provide the connectivity to the mobile node. Several content objects are usually forwarded to the nearby area of the mobile node due to the generated interest packets performed by the mobile node, which is moving in the area. VCoF tends to fit this mobile communication model since it increases the opportunity to find the replicas of desired content in those sources (i.e., the nearby area). So, we develop the VCoF scheme in a mobile NDN environment, which represents the aforementioned communication model.

3.3.1 Applying the Design in a Mobile NDN Environment

The core idea of VCoF in a mobile environment is presented in Figure 3.7[3]. We assume that the mobile consumer is moving from time T_1 to T_2 and it needs a content object from the producer P. It travels from the previous NAR to the new NAR. All NARs are connected to the NDN infrastructure network. Before handover, the mobile node starts requesting the content object by sending an interest packet through the current NAR (the first node on the left hand-side), which will be the previous NAR when it has moved to the new NAR. The content object is then fetched from the original producer P and it is replicated along the default path from the producer towards the consumer.

When the previous NAR receives the content object, it will cache the object (replica R) in its CS. The VCoF scheme is then activated by pushing the latest *Content List* to each node in the vicinity except the node who has forwarded the list since the object is already cached in its CS. In this case, there is only one node (the new NAR) in 1-hop vicinity that needs to update the pushed list. Assuming that there is another node in the vicinity of the new NAR storing the replica (replica R in

[3]Note, this is the example that illustrates how VCoF can be applied in the mobile environment. There could be a higher number of nodes in a real network.

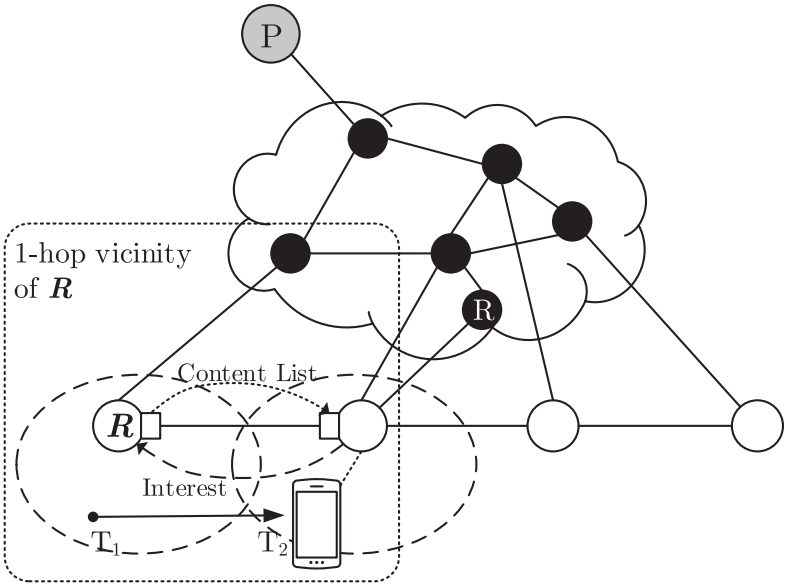


Figure 3.7: The Core Scheme of VCoF in the Context of Mobility

the infrastructure network), this node also pushes its *Content List* to the new NAR. The new NAR then maps each name from every list to each Face that has received every list to the FIB. If there is the same name from several sources that provide the same lowest distance, we select the latest source who has pushed its list.

After the mobile node has moved to the new location, we assume that the content object has been forwarded to the previous NAR. So, in this case, the content object fails to deliver to the mobile consumer due to handover. The consumer then sends a new request again. The new NAR then checks its FIB and it knows where to find the content object because of the updated *Content List*. The new NAR then selects the Face (the mapped Face with the content name) to send the interest packet in the reverse path to the previous NAR. The content object can then be fetched at the previous NAR that is located closer to the mobile consumer rather than the producer P that is located further away. The replica R might be found at the another node in the vicinity (the black node in the infrastructure network) but in this case, we find the replica *R* at the previous NAR due to its latest update of the *Content List*. The opportunities that nearby replicas of desired content can be located are increased since using VCoF covers both the previous NAR in the vicinity and other nodes in the vicinity. Additionally, the scheme can also increase the opportunities to find replicas even in other nearby vicinities as described in Section 3.2.2.3.

3.3.2 Content Availability Announcement

To locate an off-path content object (cached in a vicinity), each node should know the availability of the content object and which Face should be used to forward an interest packet. Hence, in the mobility context, each NAR and NDN node must push its *Content List* to announce the content availability in its CS to every node in its vicinity using the *Content Availability Advertisement* module. It means a node or an NAR that is the owner of the list is ready to serve all objects in its CS. After receiving the pushed list, a node in the vicinity maps each name in the list with a Face that has received the list. For example, an NAR maps each name of each content object in a received list to a Face that indicates the path to a previous NAR. When a mobile node needs a content object, it just sends an interest packet through its NAR. This NAR just checks the desired name with the mapped Face. The NAR then selects the Face to send the interest packet in the reverse path to the owner of the list (i.e., in this case, the previous NAR) and then the off-path content object will be found.

3.3.3 Fetching a Content Object

In the mobility context that is particular focused in this work, while a mobile node is moving to a new NAR, several replicas are often cached in a previous NAR because most requests are performed in the location of the previous NAR. The previous NAR is often located in the same vicinity of a new NAR. Hence, it is our expectation that VCoF can help to find the nearby replicas. The distances between a requester to the previous NAR or some nodes in the vicinity are mostly closer compared to the distance to an original producer. So, by looking at the previous NAR or nodes in the same vicinity, the delivery efficiency can be improved, which is the key concept of VCoF. To fetch a content object, when an interest packet arrives at the owner of a *Content List*, the object will be forwarded back in the reverse path towards the requester using "Breadcrumbs" left by the interest packet. The owner of the list can be a previous NAR, another node in the vicinity, or another node in a nearby vicinity, which are usually located closer than the producer.

3.4 Discussion

The VCoF scheme detailed in this chapter is designed to meet the aforementioned motivations and aims. The design of VCoF realises on delivery performance that can be improved with the consideration of some additional overheads. The design is a significant modification of the existing NDN architecture. So, it is expected to be directly deployed on real NDN systems. The concept of packet forwarding of VCoF is based on the shortest path strategy of NDN. However, instead of forwarding interest packets to the default shortest paths to their producers, VCoF offers better opportunities to forward the packets to the alternate shortest paths to nearby replicas.

The VCoF scheme requires the modification of the core design of the current NDN architecture. The official code-bases of NDN need to be modified by integrating the proposed design to support VCoF. Additionally, the design also realises on supporting the flexibility of deployment in various scenarios. The VCoF scheme has been designed with the aim to be flexible on deploying in both static and dynamic environments.

The *Content Availability Advertisement* module has been designed to advertise the content availability of an NDN node. The *Content List* is pushed to every node in the NDN node's vicinity. Each node in the vicinity then knows what are the current content objects storing in the owner of the pushed list. A Face that is mapped with the

list indicates the node that has received the list to find a desired content object mapped in the list at its owner. This is done by using the *Content Finding* module. According to the VCoF design, the packets used to push the *Content Lists* can increase network traffic overhead because of the higher number of packets regarding content finding in comparison to default NDN. So, we must evaluate the effects of the additional overhead while considering the trade-offs of improvement performed by the designed scheme.

By realising on the discussion in Section 2.6, our VCoF scheme has also been designed by avoiding the mentioned shortcomings. These are summarily described as follows:

- The VCoF scheme is a decentralised design, that can eliminate the mentioned issues of the centralised designs, which are mostly found in resolution-based solutions.
- By using VCoF, only a desired content object/replica in a proper potential source is fetched. A single interest packet is used to fetch a corresponding content object. This means that VCoF does not introduce the cache eviction problem caused by multi-sourcing in return due to multiple interest packets of content finding in several interest-flooding based techniques. Based on the design, VCoF introduces some additional overheads. Nevertheless, the trade-offs between the overheads and the performance gains will be discussed further in Chapter 5. Additionally, without considering many metrics, VCoF does not require a high calculation time to make a forwarding decision. Furthermore, VCoF is a cache advertisement solution, which introduces the concrete implementation that can potentially deploy on real NDN networks.
- In the mobility context of the mentioned communication model, unlike several previous proposals on gaining benefits of replicas only located in previous NARs, VCoF also offers alternate opportunities to find nearby replicas inside a vicinity or nearby vicinities. This can result in better delivery efficiency.

We have detailed the design of VCoF in this chapter. As mentioned previously, the design will be integrated with the existing NDN architecture. Hence, in the following chapter, we present how we implement the VCoF design by integrating with the current NDN primitives.

Chapter 4

Implementation

In the previous chapter, we described the core modules that make up the VCoF scheme. In the modules, we also presented the detailed processes of content finding and content availability advertisement. These include the core forwarder that is responsible for processing interest/data packets. Once these are taken into consideration, a number of elements must be implemented specifically for use in the VCoF scheme.

In this chapter, the existing essential components that are necessary for making up a fully running system of NDN are described in Section 4.1. The NDN library, tools for providing default routing information, and a core forwarding daemon are required to be installed in every NDN node in the network. The explanations of these elements lead to the detailed implementation of VCoF described in Section 4.2. For example, by integrating VCoF into the existing elements of NDN, the designed modules implemented specifically to proactively provide routing information for finding nearby content are presented in this section. The process of implementing VCoF integrated with the current NDN forwarding plane (consisting of the three major tables) is also detailed. Finally in Section 4.3, the implementation of a number of tools to be used in the next chapter to evaluate VCoF is presented.

4.1 The Existing Essential Components of NDN

The items described in this section are the essential components (i.e., software) to be installed in every NDN node for implementing an NDN network. These components have been released by the NDN project [143], which operate together to provide an

actual instance (a fully running system) of NDN. In this section, we describe the implementation details of the main components. As mentioned in Section 3.2, the official code-bases of NDN are modified to support VCoF. So, the descriptions of the existing core components can lead to understand the implementation of the extra modules that we have added into the current NDN architecture.

In order to run an actual instance of an NDN system, the essential components (presented in the following sub-sections) need to be installed in each NDN node. All of these components are built using the C++ programming language. This is to support the clean state architecture of NDN, which mainly aims to replace the today's TCP/IP protocol architecture. Nevertheless, it is possible to have IP with NDN at the same time. NDN can run as the overlay on an IP network by converting an Ethernet/IP network interface to an NDN Face in every host. Also, those core components can support NDN as the overlay.

When we have built a core NDN network, every NDN node in the network can store and forward packets using the basic NDN primitives. In addition, there is a common Application Programming Interface (API) that allows different applications developed by different languages such as C++, Java, JavaScript, Python, and .NET (C#) using the NDN Common Client Libraries (NDN-CCL) [144] to connect to the core NDN network. In this work, however, we focus on the core architecture of NDN to be modified to support VCoF. Hence, all of our extra modules and tools are implemented in C++ because these are easier to be compatible with the existing core components.

4.1.1 ndn-cxx

ndn-cxx [145], NDN C++ library with eXperimental eXtensions, implements all of NDN protocol primitives such as "Name", "Interest", "Data" to facilitate the development of NDN and to provide a foundation for NDN application development and experimentation in NDN. It also drives the development of NDN Forwarding Daemon (NFD), another core component of NDN to maintain packet forwarding, which will be detailed further in Section 4.1.3. Since the initial release in 2014, it has been developed to support the definitions of NDN protocol incorporating with the development of NFD. In addition, it simplifies the development of NDN applications (e.g., [146, 147]). It is an open source project licensed under the GNU Lesser General Public License, version 3 or later. Developers can redistribute it and modify it under

the terms of the license.

An example of using the ndn-cxx library can be describe as follows: assuming that a consumer currently uses an NDN client application in order to look for a particular content object, the application makes use of the library to generate an interest packet based on a defined format. At a producer side, a data packet is defined in another format. A Type-Length-Value (TLV) format is used to encode each NDN packet. It is noted that unlike the TCP/IP packet headers, an NDN packet format does not have a fixed packet header. This means it is more flexible to add or remove some types in the format depending on how the protocol evolves.

The version of interest (at the time of writing this chapter) TLV-based format contains the following elements. "Name" is the only required element. "Nonce" is required when the interest packet is forwarded over the network. It is important to carry a randomly-generated 4-octet long byte-string. This "Nonce" should uniquely identify the interest packet by combining with its "Name". By this, looping interests can be detected using a duplicated nonce of an interest packet. Other optional elements (e.g., "MustBeFresh", and "InterestLifetime") can be set to guide interest forwarding or matching ("ForwardingHint"). Signature/key information (contained in "Selectors" element) can also be included for security reasons. This process of formatting the interest is done using the ndn-cxx library. The library then expresses the interest to the network after prompted by the client application.

A server-side application at the producer proactively publishes the data (i.e., content) object and waits for any corresponding interest request. Similarly, to generate the data packet for the published content object, the application makes use of the library by relying on the TLV format of data packets. When the data object is found at the producer matched with the received interest, it will be returned back along a default path to the consumer.

The following elements are composed in the version of the response data in a TLV-based format. "Name" of the content is required to identify the data packet, which contains some arbitrary binary data (hold in the optional "Content" element). "Signature" represents a digital signature for data verification. Additional bits of information labeled as "MetaInfo" element specify additional information of the content. "ContentType" is a sub-element, which defines a specific type of the data packet. In this case, it is "BLOB" (binary large object), which represents the default type of payload. "KEY" and "NACK" are another types representing public key and application-level negative acknowledgment (NACK), respectively. The expiration of

an optional "FreshnessPeriod" indicates that the producer may publish newer content. The optional "FinalBlockId" provides the end of the final block in a sequence of fragments. This is done as part in the library. The server-side application then prompts the library to put the content to be ready to serve once the corresponding interest has been received.

4.1.2 Routing Information

To forward interest packets, a forwarding strategy requires knowledge of where to forward. This can be done by considering routing information in the FIB of the NFD in each NDN node. By default, each node has an empty FIB. Inputting routing information into the FIB can either be manually configured, or can be assigned using a routing protocol. The protocol generally disseminates producer prefixes and updates their availability to simplify FIB management especially in large topologies. NLSR is one of the most well-known routing protocols in NDN, which is used in this work and described in Section 4.1.2.1. In NFD, the Routing Information Base (RIB) stores static or dynamic routing information registered by applications, the routing protocol, the operator, and NFD itself. FIB entries are calculated from the RIB to be used directly by NFD to determine next hops to forward the interest packets. This also depends on each forwarding strategy.

4.1.2.1 Named-data Link-State Routing (NLSR) Protocol

NLSR [54], Named-data Link-State Routing protocol, is a link state routing protocol like OSPFN [96], developed to overcome the limitations of OSPFN in NDN such as the inadequate multi-path support. NLSR populates NDN's RIB, which exchanges the link state information using interest/data packets. This builds the FIB table in each NDN node by discovering adjacencies and disseminating name prefixes regarding the network topology. In NLSR, each NDN node has a hierarchical name structure, which identifies the location it belongs to, such as "/ndn/uk/ac/lancaster/scc". The NLSR process on a node has the node's name as its prefix, e.g., "/ndn/uk/ac/lancaster/scc/nlsr" to periodically send *hello* messages at a default interval of 60 seconds between adjacent NLSR nodes to detect the changes of neighboring connections.

When an NDN node's NLSR detects a failure or a new connection of neighbor, it disseminates a new Link State Advertisement (LSA) message to the entire network.

Two types of LSA are carried out by NLSR nodes. A Name LSA stores all name prefixes registered locally with NLSR, which are injected by connected end-nodes. An Adjacency LSA stores all of the active links of a node, which each entry represents a neighboring node's name and a link cost. A Link State Database (LSDB) in each node stores the latest version of the LSAs. A simple extension of Dijkstra's shortest path first (SPF) algorithm is executed to calculate multiple next hops to reach each node. Each node recalculates the latest routes and updates its FIB, upon receiving any new LSAs. NLSR creates Faces for the neighbors using the information obtained from the NFD. To enable this protocol, each node must install NLSR using its source code available in [148].

4.1.2.2 ChronoSync

NLSR propagates LSAs using sync protocols such as the ChronoSync protocol [149]. This is to synchronise changes in the nodes' LSDBs. Each node exchanges a crypto digest form (i.e., hash) of a name set, which contains all the latest LSAs in its LSDB. A node compares a received hash of a neighboring node's LSDB to its local hash. If the comparison is different, the node will request the different LSA from the neighbor to update its LSDB. To deploy NLSR, a sync protocol, which is ChronoSync (available in [150]) used in this work, is required before installing NLSR due to its core function to disseminate routing information for building and managing the FIB, which is performed by NLSR.

The processes of disseminating an LSA message in the network can be described as follows: To synchronise a hash of LSAs in an LSDB, ChronoSync on a node broadcasts a *Sync Interest* to all of its neighbors. When an updated LSA is added to a neighbor's LSDB, the hash of the neighbor's LSA name set will be different compared to the node's hash. This causes the ChronoSync protocol of the neighbor replying to the *Sync Interest* with a *Sync data* packet containing the new LSA name. The node's ChronoSync then receives the *Sync data* packet, and notifies NLSR of the new LSA name to update its LSA name set. The node's NLSR sends an *LSA Interest* to retrieve the updated LSA from the neighbor, since the NLSR process on the node has been notified by the new LSA name. The neighbor responds to the *LSA Interest* with its updated LSA data. When the node's NLSR receives the LSA data, it updates the LSA in its LSDB. LSDBs of the two nodes are now synchronised. A new hash of the new LSA name set is computed and exchanged between the two nodes using new

Sync Interests to ensure the similarity.

Since LSAs are used to calculate routes, it means by default, no existing routing information to be used to send *LSA Interests*. Hence, a multi-cast forwarding strategy configures the specific name prefix of LSAs (/localhop/<network>/nlsr/lsa). This name allows an NDN node to forward the *Interest* for an LSA to all of its neighboring nodes. If any of the neighboring nodes have a copy of the corresponding LSA data in its NLSR process or its cache, the neighbor node will return the LSA data. Otherwise, it will discard the *Interest* because of the restriction of the *localhop* scope.

4.1.3 NDN Forwarding Daemon (NFD)

NDN Forwarding Daemon (NFD) [151] is a network forwarder responsible for processing/forwarding interest/data packets. All incoming interest packets are processed by NFD to make forwarding decisions depending on a forwarding strategy and to seek their corresponding data. NFD returns data packets towards their consumers using "Breadcrumb trails" left by the interest packets. NFD implements the core NDN structure. So, it must be installed in each NDN node with ndn-cxx. These operate together to process incoming/outgoing packets.

NFD implements three major tables including the CS, the PIT, and the FIB evolved together with the NDN protocol. The forwarding module interacts with Faces, Tables, and Strategies to implement basic packet processing pathways. NFD determines whether, and where to send packets using the provided information in the three tables. In the data routing module, when an incoming interest packet hits a corresponding content object in the CS, the object will be reversely forwarded back to the requester using the "Breadcrumb trail" left in every PIT along the path. NFD selects Face(s) to forward the interest packet using the information supplied by the FIB entries (filled by NLSR or system administrators) depending on a forwarding strategy.

Both modules of NFD are modified to support the VCoF scheme. A sub-module, "Face Counters", is used to monitor some evaluation metrics (e.g., message overhead, and data volume). In this work, we mainly focus on two tables, which are the CS and the FIB. According to our design, the CS of a node creates its *Content List* as a new element after the node startup. This is to prepare the list for the *Content Availability Advertisement* module as described in Section 3.2.2. NFD gets the latest *Content List* from the CS to advertise the content availability. The CS also inserts received

content objects as a normal operation of NDN.

By considering the NFD's forwarding module, the FIB indicates routing information for interest packets, as usual. It also inserts the *Content List* coupled with the cost to the list owner to be used in the *Content Finding* module as described in Section 3.2.1. The PIT is responsible for keeping unsatisfied interest packets and forwarding data packets in the reverse paths of the interests. After sending the data packets, the PIT then makes the pending interests of the corresponding data packets satisfied.

4.2 The VCoF Scheme

To supplement the existing NDN functionalities of content finding, we develop the VCoF scheme based on the detailed design described in the previous chapter. As mentioned in Section 4.1, the existing core structure of NDN needs to be modified to support the VCoF modules. First of all, we take advantage of NLSR to propagate name prefixes and to calculate each default shortest path to reach each node in the network. We then calculate shortest paths to the owners of *Content Lists* using the information of the name prefixes of the list owners provided by NLSR. Hence, there is no modification in NLSR. NDN nodes still require the default routing information to forward interest packets to their default paths, especially when nearby (off-path) replicas can not be found by using the VCoF scheme. This is a function of our design that can potentially ensure content delivery.

ndn-cxx is modified to attach the *Content List* into the default interest packet format. We also modify NFD to support our content finding strategy. This is because NFD implements packet processing pathways. The core operations of NFD directly involve in the content finding strategy. The following detailed implementation of the core modules of the VCoF scheme describes how we develop the scheme.

4.2.1 Content Finding Module

In NFD, a core forwarder is responsible for processing incoming/outgoing interest and data packets using the three major tables (PIT, FIB, and CS). Hence, we modify the forwarder to support the VCoF packet processing. Figure 4.1 shows how an interest packet is processed by the modified forwarder, realising the three main processes of the *Content Finding* module as described in Section 3.2.1. In the diagram, we assume

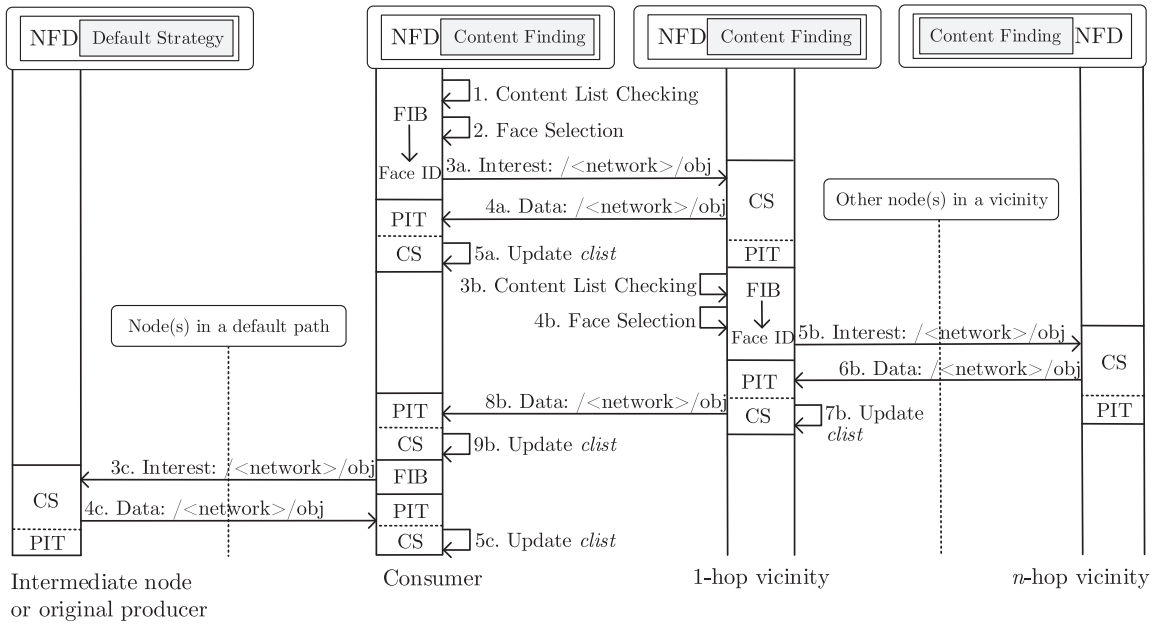


Figure 4.1: The Flow of Content Finding

that the consumer is looking for a particular content object. It should be noted that there are three major forwarding decisions in the flow depending on the current information in the three tables of each node. The description of the flow is detailed as follows.

Assuming that the consumer uses an application requiring the particular content object from the producer, the interest packet named /<network>/obj is generated. The <network> component in the name of the interest represents the network of the object residing in. It can contain additional details in the hierarchical format such as /<core-network>/<sub-network>/<node>/obj. In the *Content List Checking* process (step 1), the consumer checks the FIB to find a corresponding entry in a *Content List* pushed by another consumer in its vicinity as illustrated in Figure 4.2. This is done by matching the exact name of the content.

The exact name of the content (or the interest) is the first priority in the matching process. If the name can be found in the FIB, it will response the routing information of this particular name. This name is in the list pushed by its owner, which couples with the Face ID and the lowest cost to the owner (representing the routing information to find a nearby replica). The default routing information of the name prefix will also be provided to determine the better direction to forward the interest

packet in comparison to the information provided by the matching of the exact interest name. Based on the given routing information, candidate Faces represented by their Face IDs are considered in the next process.

In the *Face Selection* process (step 2), NFD compares the given candidate Faces to find the proper path to forward the interest packet. This is done by comparing the costs of the Faces to their destinations. In this case, the cost to the owner of the list is compared with the cost to the original producer. It is noted that the default prefix indicates the shortest path to the producer. It means the exact content name to the list owner is compared with the name prefix to the producer. After completing the cost comparison, the better Face is selected to be used in the *Forwarding* process.

After selecting the proper Face, in the *Forwarding* process, the Face ID provided by the FIB is used to forward the interest packet to find the nearby replica at the neighboring node in its 1-hop vicinity (step 3a). Notably, this is the **first** forwarding decision, representing the case that the exact name can be matched an entry in a list. This means the cost to the owner of the list (nearby replica) is lower than the cost to the producer. When the neighboring node receives the interest, the node's NFD look ups the corresponding object in its CS. If it can be found, the data packet named /<network>/obj is then returned back in the reverse path to the consumer (step 4a) and the interest will be satisfied in the neighbor's PIT. So, the desired content can be found nearby instead of going to the further producer. The consumer's NFD then marks the interest satisfied in its PIT, inserts the content into its CS, and updates the CS's *Content List* (i.e., *clist*) (step 5a). The updated list will be used in the *Content Availability Advertisement* module.

In the case of the **second** forwarding decision, the neighboring node, which is an intermediate node in the consumer's 1-hop vicinity, adds the interest as a pending entry in the PIT, which couples with the Face ID that has received the interest. This is to indicate a corresponding incoming data packet to the reverse path to the consumer, i.e., creating a Breadcrumb trail. If another interest packet looking for the same content arrives at this node, the node look ups the PIT to add the incoming Face ID of this interest as another pending downstream Face (as illustrated in Figure 4.4). This neighboring node repeats the *Content List Checking* process again (step 3b). We assume that this is because the desired content cannot be found in this neighbor's CS. It means this node is not the actual owner of the list. So, this neighboring node checks a corresponding entry of each pushed list in its FIB to find a Face of the exact interest name to be used in the *Face Selection* process (step 4b).

After the completion of the cost comparison, assuming that the cost to the actual list owner in the n-hop vicinity is still lower than the cost to the original producer, the interest packet is then forwarded to the list owner performed by the *Forwarding* process using the Face ID provided by the FIB (step 5b). Notably, if there are some intermediate other node(s) in the vicinity before reaching the list owner, the three major processes will also be repeated in every intermediate node. This is the case when the cost to the list owner in each intermediate node is still lower than the cost to the producer.

When the actual owner of the list receives the interest packet, its NFD look ups the corresponding object in the CS, as usual. Since it is the owner of the list, the CS stores the replica of the desired content object and the interest packet hits this replica. The data packet containing the replica is then generated and returned back to the intermediate node who has forwarded the interest (step 6b). The interest packet will be marked satisfied in the list owner's PIT. In this case, the intermediate node is the neighboring node (in the 1-hop vicinity) of the consumer.

Once the data packet arrives at this intermediate node, the PIT entry that stores the incoming Face ID of the received interest packet (as mentioned previously) is used to forward the data packet in the reverse path to the consumer. It is then satisfied in the intermediate node's PIT. Notably, there could be more than one Face of the pending interest. The data packet will be forwarded downstream to all Faces of this interest. This technique is called "Breadcrumbs" and will be repeated in each intermediate node towards the consumer. It should be noted that if the data packet cannot be matched any entry in the PIT, it is dropped since it is an unsolicited data.

The intermediate node stores the content obtained from the data packet before forwarding downstream to the consumer. This content replica can be further located by other consumers. The CS's *Content List* of this intermediate node is then updated (step 7b). Notably, there is an option to (or not to) allow caching the content (named *"allowCache"*) in each intermediate node consisting of *"setAllowCache(boolean)"* and *"getAllowCache()"*. This is to control content popularity/density for some evaluation purposes, which will be described further in Chapter 5. This also depends on different customisation of VCoF to be further deployed in each particular network, which may have different characteristics of content caching. If this option is set to false, step 7b is not required to updating the list since the content object is not allowed to be cached at any intermediate nodes.

The consumer's NFD then seeks a corresponding PIT entry of the received data

packet (step 8b). The pending entry of the interest packet is then satisfied and the data packet is then added into the consumer's CS, thereby activating the update of the *Content List* (step 9b). The received data packet notifies the application by triggering a sub process (called "afterReceiveData") in the Face module of ndn-cxx and it can be further used in the application level.

The **third** forwarding decision considers the case of no entry of the exact name found in the FIB or the lower cost to the original producer compared to the cost to the list owner. Hence, in the *Face Selection* process (back to step 2), the Face ID provided by the FIB indicating the default path to the original producer is selected to forward the interest. It may travel through several nodes in the default path before reaching a particular node that can serve the content. The desired content can be opportunistically found at an intermediate node, or be ultimately located at the producer (step 3c).

Once the interest packet hits its corresponding content in an on-path node's CS, the node's NFD returns the data packet back to the reverse path towards the consumer (step 4c). The node's PIT marks the interest satisfied. Each intermediate node processes the incoming data packet using the Breadcrumb trail in its PIT left by the interest. When the data packet arrives at the consumer, the same process of the PIT satisfaction as described previously is activated. The content object is then inserted into the consumer's CS and can be used in the application level. The *Content List* is also updated to be prepared for the *Content Availability Advertisement* module.

It should be noted that an intermediate node can be a member of another nearby vicinity. When an interest packet arrives at the intermediate node, NFD always performs the three major processes of the *Content Finding* module. This ensures that a candidate Face with the lowest cost is selected to forward the interest packet to find its corresponding content at the nearby source or at its original producer. The nearby source can be in either the node vicinity itself or another nearby vicinity as described previously in Section 3.2.2.3.

4.2.2 Content Availability Advertisement Module

To advertise the content availability of an NDN node, the core forwarder of NFD is also responsible for pushing the node's *Content List* to each node in its vicinity. To achieve this, we modify the format of interest packets in ndn-cxx to support the *Content List* pushing operation. The main change is illustrated in Figure 4.2. This

spacial interest packet is pushed to the vicinity by following the flow demonstrated in Figure 4.3. The node (i.e., consumer) must perform the three major processes of the *Content Availability Advertisement* module as described in Section 3.2.2. It is noted that the flow presents the *Content List* pushing operations in the smallest vicinity size of 1 and the larger vicinity size of n. The node's NFD must push its *Content List* dependent on a defined vicinity size. The details of the packet modification and the implementation of the list pushing are described as follows.

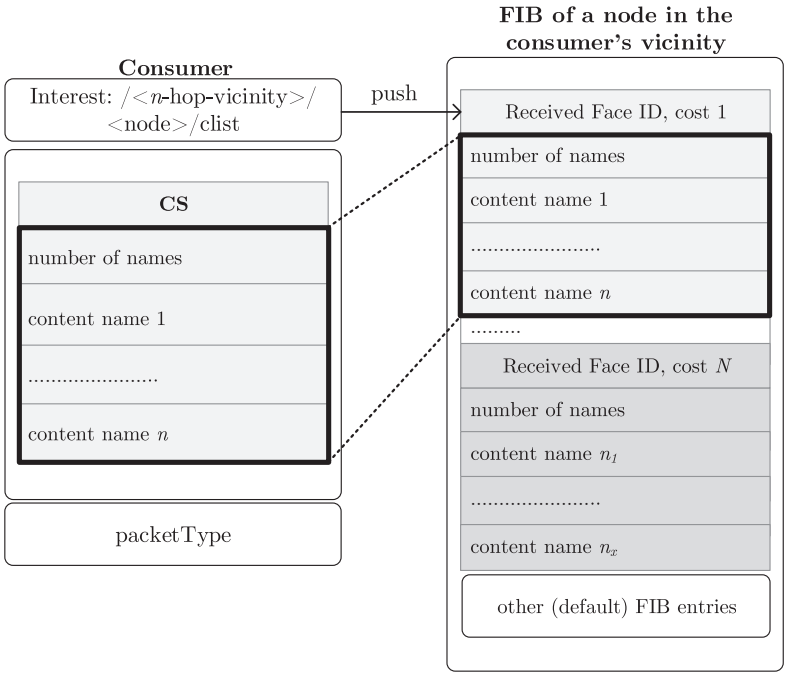


Figure 4.2: Modified Interest and FIB Interaction

As shown in Figure 4.2, the consumer pushes its *Content List* to a node in its vicinity. The interest packet is modified to carry the extra table of the *Content List* in the consumer's CS. The table stores each of exact content name in the consumer's CS. This is done by altering the interest TLV-based format as described in Section 4.1.1. A new element called *"contentList"* is added into the current format. In ndn-cxx, each interest packet is encoded into TLV block. So, we have to prepend the TLV block of the extra table to expand the total length of the interest packet for encoding. A new function called *"getContentList()"* is added into the interest processing. This is to enable NFD to get the extra table in the new *"contentList"* element of the interest. A function *"setContentList(list)"* is likewise attached to enable NFD to update the

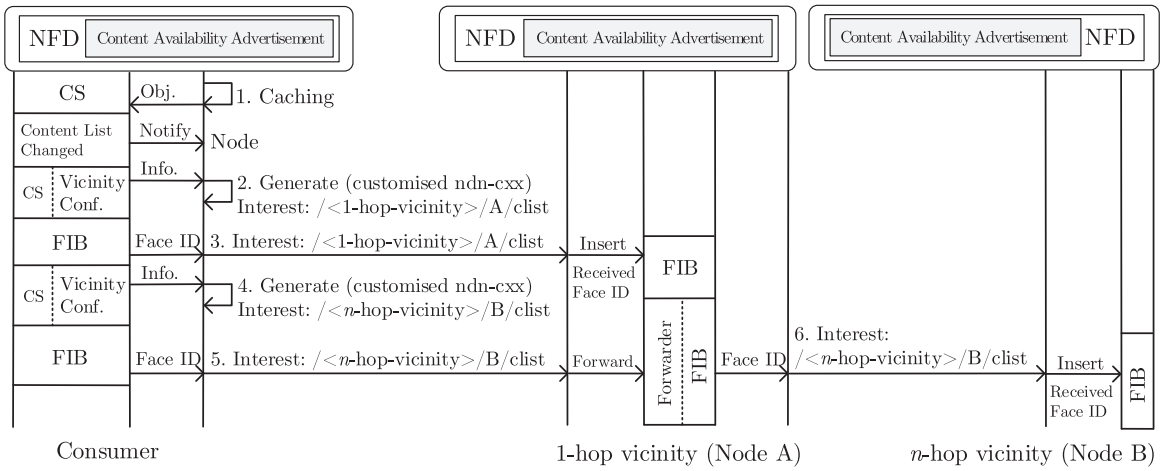


Figure 4.3: The Flow of Content Availability Advertisement

latest extra table.

The interest packet is named according to the network of a node in the consumer's (n)-hop vicinity that it resides in, i.e., /<n-hop-vicinity>/<node>/clist. The <n-hop-vicinity> and the <node> components contain the name of the network and the name of the node, respectively. The name prefix routes the interest to the node that is one of the members in the consumer's vicinity. We assume that this prefix is obtained from the consumer's configuration file. This file contains each name prefix of each node in the consumer's vicinity, which is used in the *Vicinity Selection* process of the *Content Availability Advertisement* module. The "clist" component identifies the type of the interest packet (i.e., specifically for pushing the *Content List*).

Although the "clist" component can identify the packet type, a name of a content object can be duplicated. To avoid this issue, a new element called *"packetType"* is added into the interest TLV format. Functions *"getPacketType()"* and *"setPacketType(type)"* are also attached for packet type checking and setting, respectively. The consumer then pushes the interest packet according to its name. When the destination node of the interest receives the pushed list, each content name in the list is then inserted into the destination node's FIB to be used in the *Content Finding* module. The list is coupled with the cost to the owner of the list (i.e., the consumer) and the Face ID that has received the list. The destination node, which is another consumer, uses an entry in the list to find a corresponding content object at the owner of the list. Since the consumer pushes the list in a shortest path to the destination node, the Face ID that has received the list indicates the reverse shortest

path (the same path) to the list owner.

It should be noted that there might be several lists from different sources inserted in a node's FIB. A duplicated name can be aggregated into an entry. The lowest cost of a particular Face ID of this name will be selected to be used in the *Face Selection* process as described in Section 4.2.1. In addition, the lowest cost of this name can be coupled with several Face IDs, which means there are more than one direction to find a nearby replica with the same distance. In this case, the node selects the latest Face ID that has received a list containing this name.

In Figure 4.3, when a newly requested content object arrives at its consumer, it is inserted into the CS of the consumer's NFD (step 1). The CS's *Content List* is then changed. This notifies the node to provide its updated content availability to each node in its vicinity by using a developed *Content List* pushing application. This is the *Caching* process of the module. If the object is already in the latest list that has been pushed, we ignore pushing the list. In the *Content List Pushing* process, the list is pushed using the mentioned strategy in Section 3.2.2.2. Each prefix of each member in the vicinity listed in the consumer's configuration is used to generate its special interest packet to be pushed (step 2). This is done by the *Vicinity Selection* process. The interest named /<1-hop-vicinity>/A/clist, for instance, is created using the customised ndn-cxx, which the <1-hop-vicinity> component represents the network of a member (i.e., Node A) located in the consumer's 1-hop vicinity. The interest packet is then sent to Node A according to the name prefix (/<1-hop-vicinity>/A/) through the Face ID provided by the FIB (step 3).

When Node A receives the interest, each name in the pushed list is inserted into the FIB of Node A, which couples with the Face ID that has received the list and the lowest cost to the list owner. The process of *Content List* pushing to a member (another consumer) in the consumer's 1-hop vicinity is then finished. Other names in the 1-hop vicinity repeat the same process. It should be pointed out that if there are a number of entries of n-hop vicinity, the consumer must push the list to all entries listed in the configuration. For example, the interest named /<n-hop-vicinity>/B/clist is also pushed to Node B residing in the network of the consumer's n-hop vicinity (step 5). An intermediate node's forwarder (e.g., Node A) forwards this interest towards Node B indicated by its name prefix using the mapped Face ID in the FIB. Each name in the pushed list is then added into Node B's FIB to be used in the *Content Finding* module.

The interest/data packet processing flow is illustrated in Figure 4.4. The flowchart

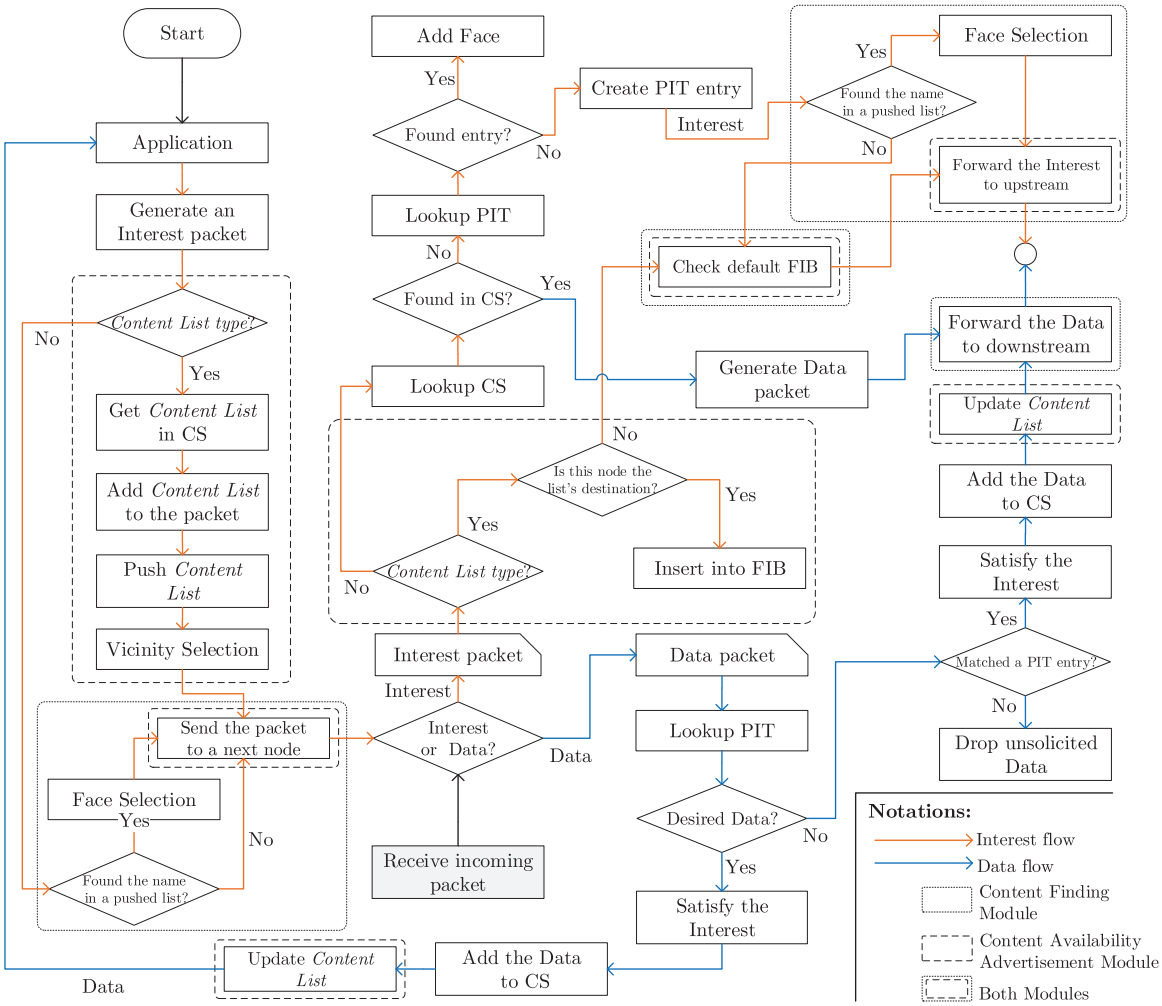


Figure 4.4: Upstream Interest Forwarding Flow and Downstream Data Forwarding Flow Integrated with the Designed Modules

describes the integration of the major modules of VCoF added into the current NDN. It presents where the modules are applied in the NDN architecture. It starts from generating the interest packet triggered by the application. The interest packet is then classified to identify its type. If the packet type is to push the *Content List*, the updated list is carried in the interest to provide the content availability. Otherwise, the interest packet is processed to find its content. After the arrival of an interest/data packet in a node, the node processes the incoming packet differently. The interest packet processing is to forward the interest to upstream node(s) supplied by each FIB, create the Breadcrumb trail in each PIT, look up a corresponding content object

in each CS, and push the *Content List*. The data packet processing is to return the corresponding content obtained from the CS to downstream node(s) (towards its consumer) in the created Breadcrumb trail, drop unsolicited data, and update the *Content List*.

To attach a *Content List* into an interest packet, the packet must be altered in NFD. This is because we cannot currently get the list provided by its CS in the application level. The list is likewise updated using NFD before sending to the application level. When the interest packet (carrying the list) arrives at a node, it must be checked that the node is the destination of the list or not. This is to ensure that the pushed list will be advertised in the target node of the defined vicinity.

4.3 Evaluation Tools

There are various ways to evaluate an NDN scenario including using a testbed, virtual machines, an emulator, and a simulator. In considering the testbed, it is slower to boot/restart compared to other tools. It also requires several dedicated devices to run a large-scale network. Hence, the testbed might scale poorly. Virtual machines can connect different NDN nodes using a smaller number of machines compared to the testbed. However, NDN installation and configuration must still be manually performed in each VM. Changing and managing event(s) of an experiment might also be difficult due to no central experiment controller by default.

Since this work aims to leverage content finding in NDN, it requires a higher amount of nodes in a network, while the content finding events performed by several consumers can be controlled by a central controller. For example, content finding requests should be executed by different consumers in several locations to realise the realistic evaluation. Hence, we start focusing on an emulator (i.e., Mini-NDN [152]). This is because a fully running system consisting of the mentioned components in Section 4.1 can be easily emulated. Any changes in these components can directly apply to the NDN official code-bases. Each NDN node can run real applications (e.g., *Content List* pushing or finding). We can understand the flow of a packet better because we can interact directly in real time. For instance, current name prefixes in a FIB of a node can be observed using an actual command of NFD (<*node name*> *nfdc fib list*) to investigate routing information.

After understanding the actual operations of the VCoF scheme (in comparison to default NDN) in a fully running system by using the emulator, the evaluation requires

higher-scale networks to understand content finding results in larger topologies. This is done by using a simulator (i.e., ndnSIM [153]) since it can simulate a larger-scale network compared to the emulator. It should be noted that ndnSIM uses discrete time steps. Hence, an application might run differently from real world.

As mentioned in Chapter 1 (Section 1.4), the research outline is threefold. Based on the outline, Mini-NDN is used to evaluate VCoF in a static (emulated) NDN network to first understand the deployment of VCoF in the NDN network that consists of the actual software components (ndn-cxx, NLSR, NFD, and ndn-tools), i.e., for the fully running system. ndnSIM is then used to evaluate VCoF in a larger topology compared to the emulated network by considering the impact of content popularity with an actual data-set of content requests, which provides a higher content count. Finally, ndnSIM with a mobile simulation package evaluates VCoF in handling mobility. The evaluation tools and their implementation details are described as follows[1].

4.3.1 Mini-NDN: A Mininet based NDN Emulator

Mini-NDN [152] is an emulation tool based on Mininet [154] to realise NDN implementation. The prior Mininet based emulator, Mini-CCNx [155], is a prototyping tool for implementing basic CCNx protocols, which operates on CCNx architecture. Currently, the CCNx is no longer available, however. Unlike Mini-CCNx, Mini-NDN uses the current NDN code-bases to support the needs that CCNx was not amenable as mentioned in Section 2.3.5. Since it uses real world time steps, an emulation network can be close to a real world environment. The network topology can be easily generated by a GUI tool named MiniNDNEdit as illustrated in Figure 4.5. The NDN software components described in Section 4.1 can be emulated to examine a fully running system of NDN in a single machine using Mini-NDN. It runs the components on top of visualised nodes and uses each of different socket file for each node. Therefore, the mentioned essential components need to be installed in the machine before setting up Mini-NDN.

Since Mini-NDN is built on top of Mininet, we must install Mininet prior to Mini-NDN. Python is also required since it is used to run experiments. Process-based visualisation and network namespaces are the features in recent Linux kernels, which allow Mininet to create virtual networks. In a network namespace, Mininet runs hosts

[1]Note, the detailed evaluation will be described in the next chapter.

as "bash" processes. Any commands or applications that can normally execute on a Linux system should also be able to execute within a Mininet host. So, commands such as *nfdc <option>* (NFD command) can be virtually executed in the host. The host will only see its own processes and will have its own private network interface(s). Mini-NDN maps each network interface to each NDN Face. In Mini-NDN, hosts can be connected by virtual links (processed by the Linux kernel) in a point to point fashion instead of using switching elements like in Mininet.

As shown in Figure 4.4, an interest packet is generated by an application and its corresponding data will be used in the application level. Hence, to transmit a single interest/data packet for any content to be used in the application, an NDN node needs some essential tools. A collection of basic tools for NDN called ndn-tools [156] is recommended to be installed in all NDN nodes. So, ndn-tools is installed after the installations of the mentioned NDN software components. We focus on a major tool involved the evaluation of this work, which is "ping". The "ping" tool is generally used to test reachability between a consumer and a producer. It can also transmit the single packet between the two nodes because it is based on a simple interest/data transmission.

To generate a special interest packet for carrying a *Content List* provided by

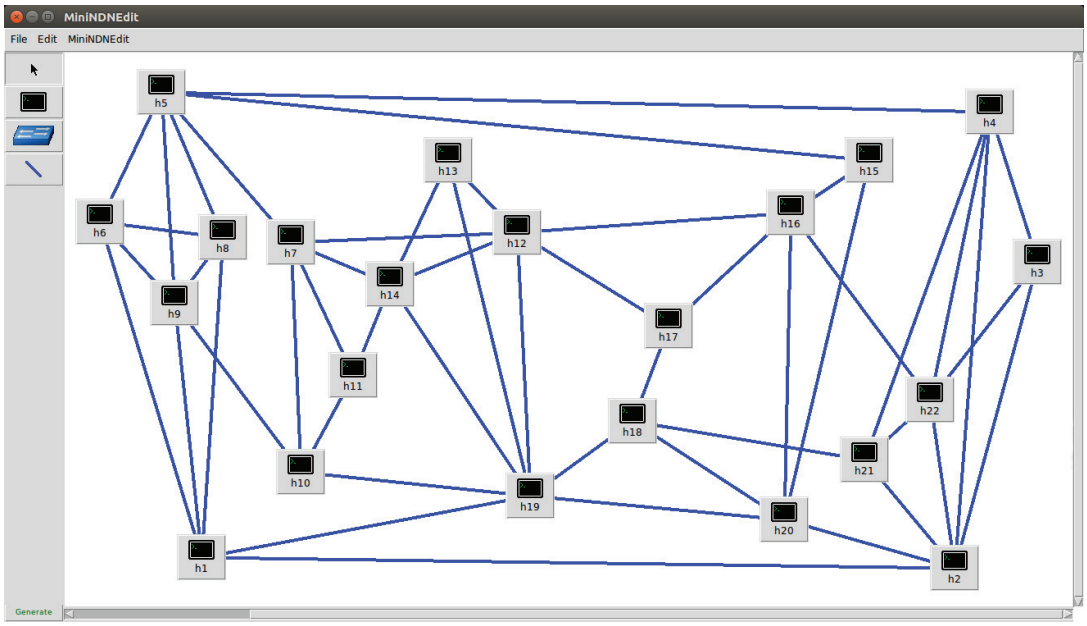


Figure 4.5: Generating an Evaluation Topology using MiniNDNEdit

a node's CS, we modify the "ping" tool by setting an interest option to identify the packet type representing the list pushing. This calls the *"setPacketType(type)"* function that we have added into the interest packet format in ndn-cxx. The tool then sends the interest to NFD. In NFD, we develop a function to allow getting the current *Content List* of the node's CS. The interest packet is then altered by attaching the obtained list before forwarding out of the node. When the destination node receives the list, each name in the list is inserted into its FIB as described previously. No data packet is returned since the interest packet is the one-way packet for pushing the list. Hence, a tool named "pingserver", a pair application of the "ping" tool for serving the corresponding data in response to the interest, is not required. We name the tool *"pushlist"* without removing the origin "ping" tool from ndn-tools.

An interest packet for finding content is generated in a consumer node using the another modified "ping" tool renamed to *"findcontent"*. The tool prepares the packet type to be set in the *"setPacketType(type)"* function to identify the default packet type of content finding. The packet is then sent to NFD. The *"getPacketType()"* function in NFD is used to check the packet type as illustrated in Figure 4.4. Since it is the default packet, it will be processed by the described *Content Finding* module. When a node that can serve the corresponding data in response to the interest, receives the interest, the data packet generated by the modified "pingserver" tool named *"servecontent"* is returned back to the consumer.

4.3.2 ndnSIM: An NS-3 based NDN Simulator

ndnSIM [153, 157] is an open-source simulator written in C++ to realise NDN implementation. It is based on NS-3 [158], a discrete event network simulation platform. The latest release of ndnSIM integrates NFD and ndn-cxx code-bases, increasing its value in understanding the NDN implementation such as finding content, developing applications, applying NDN to different network environments (e.g., mobile environments), designing congestion control and caching. Since the NDN team started developing the new implementation of NDN forwarder and supporting library to realise the latest NDN protocols and to enable several new features [159] (e.g., security abstractions with trust schemas, formalised forwarding pipelines, forwarding hints, etc.), ndnSIM 2.x [160, 161] integrates those code-bases inside itself instead of re-implementing the same features for NS-3 based on the prior version (ndnSIM 1.x). ndnSIM is built on top of NS-3. Hence, we must install NS-3 prior to ndnSIM.

Unlike Mini-NDN, the NFD and ndn-cxx components are already included. So, we do not need to install them separately. As mentioned previously, NLSR provides default routing information into FIBs. However, in ndnSIM, NLSR is not originally included. In ndnSIM, a global routing controller can also simplify FIB management by calculating and installing FIBs on every node using "GlobalRoutingHelper". The helper calculates each shortest path for every destination node and installs routes for all of default name prefixes. This is done by doing Dijkstra for each destination node. Hence, by using ndnSIM, we do not need to use NLSR since "GlobalRoutingHelper" can also provide default routing information, which is sufficient for the current version of VCoF especially in terms of content finding evaluation in this work. It is noted that "GlobalRoutingHelper" does not require a central controller. It is indeed a specific helper in ndnSIM that calculates default routing information to simplify FIB management after the startup of an experiment.

ndnSIM allows simulating two distinct types of NDN applications including ndnSIM-specific and real-world applications. ndnSIM-specific applications simulate basic interest/data packet flows. The version of the ndn-cxx library (used in this work) bundled in ndnSIM alongside the operations of NFD enables simulating real-world NDN applications. We port our modified real world applications based on ndn-tools as described in Section 4.3.1 into ndnSIM. Unlike ndn-tools in Mini-NDN that can directly set their attributes via Command Line, ndnSIM uses "AppHelper" to set the underlying application attributes and to install the applications in each node. The flows of interest/data packets are processed based on the modification described in Section 4.2. This is done by applying the customised core forwarder of NFD and ndn-cxx library into the existing NFD and ndn-cxx code-bases inside ndnSIM.

Before running an experiment, we need to define a network topology. In ndnSIM, a topology is specified using a specialised helper or in a text file. It is easier to use a text file because we can define a topology in a single file. This is very convenient when we need to change a topology by simply changing a current file name to the new file name of the new topology in a simulation script. Hence, each text file that is simple user-readable format, is used to define every evaluation topology. The "AnnotatedTopologyReader" class reads annotated topology from a file and applies parameter settings to corresponding nodes and links. In the file, we can define topology parameters such as node names, links between desired nodes, link bandwidth, link delays. An (x,y) coordinate pair represents the position of a node in the network, which can also be defined.

4.3.2.1 ndnSIM Mobile Simulation Package

ndnSIM Mobile Simulation Package [162] is an NDN hackathon project, which contains a simple WiFi mobility simulation. In the simulation, assuming that a WiFi AP acts as an NAR connected with other NARs and nodes in the infrastructure, it is responsible for associating mobile nodes in its coverage area. The scenario simulates a three topology provided by a text file using the "AnnotatedTopologyReader" class. A mobile node moves from a previous AP to its current AP in a constant velocity. After association, the mobile node sends a uni-cast interest packet to the AP. The interest packet is then forwarded to a producer and the corresponding data packet is returned back to the mobile node through the AP. The AP broadcasts the data packet to be forwarded to its consumer.

"GlobalRoutingHelper" is also used to propagate default name prefixes of the nodes in the topology to provide default routing information. By considering the mobility simulation in the package, we modify this simulation to support the evaluation topology in Section 5.4 and evaluate simulation events to examine how VCoF handles the mobile case, which is previously discussed. The NFD and ndn-cxx code-bases are also replaced by following the modification described in Section 4.2 to deploying VCoF.

4.3.2.2 A Live Simulation of the VCoF Scheme

In NS-3, a live simulation visualiser can be useful for debugging. i.e., to understand if a packet flow is what we expect. Since ndnSIM is built on top of NS-3, it can also support the visualiser. To debug a flow performed by VCoF, for instance, we can monitor a simple interest/data processing flow by simply defining a simple topology with static locations of a consumer and a producer. We can then check the flow by using the visualiser as soon as the consumer sends a request. However, the visualiser is not compatible with real applications by default. Hence, the real applications mentioned previously are modified to be executed in the visualiser mode.

It should be noted that the evaluation topologies in Chapter 5 contain several nodes and the locations of requesting content and producers can be varied. There are also a high number of interest/data packets. This might be impossible to check the evaluation flows using the visualiser. So, we only use the visualiser for checking packet flows with simple defined NDN networks (i.e., for debugging). An example flow for debugging is described as follows.

In the example, the topology represents a grid network consisting of 4x4 nodes as shown in Figure 4.6. Assuming consumer A is the first requester of a content object provided by the producer, the consumer A's location is at the (0,0) coordinate. The producer is located further away at at the (3,2) coordinate. The shortest path to reach the producer is 5 hops. By requesting the content, an interest packet is generated by the consumer A's application and forwarded along the default path indicated by each FIB towards the producer. After receiving the interest, the producer then returns the corresponding content in the reverse path towards consumer A by following each PIT entry left by the interest. This is the default best route strategy of NDN.

By following the operation of the *Content Availability Advertisement* module, when consumer A receives the content, it then pushes its *Content List* to the other nodes in its vicinity. Figure 4.7 presents the flows of the list pushing. In this case, we set the vicinity size to two. Every node in the vicinity then inserts the received list into its FIB coupled with the Face ID and the lowest cost to the list owner (i.e., consumer A). This is the proactive routing information for each node in the vicinity that might look for the same content.

Assuming consumer B is another requester looking for the same content, the *Content Finding* module tries to find the content by considering the routing information in its FIB. This case assumes that consumer B is located at the (2,0) coordinate. After receiving the list pushed by consumer A, consumer B knows where to find the same content object nearby by considering the added extra routing

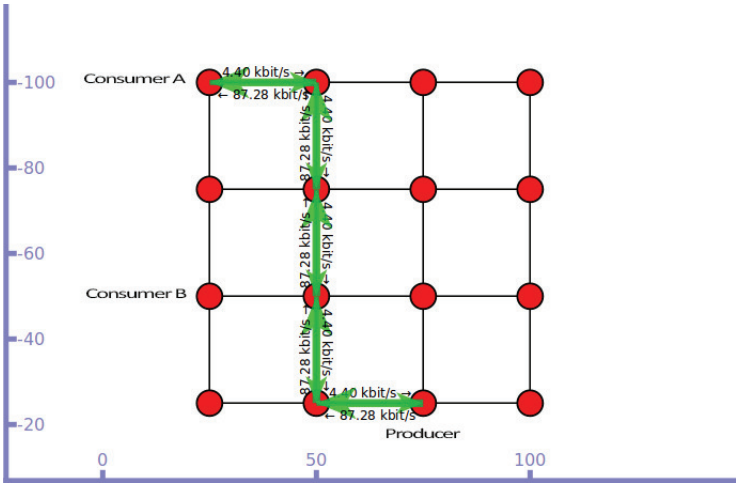


Figure 4.6: First Request Performed by Consumer A

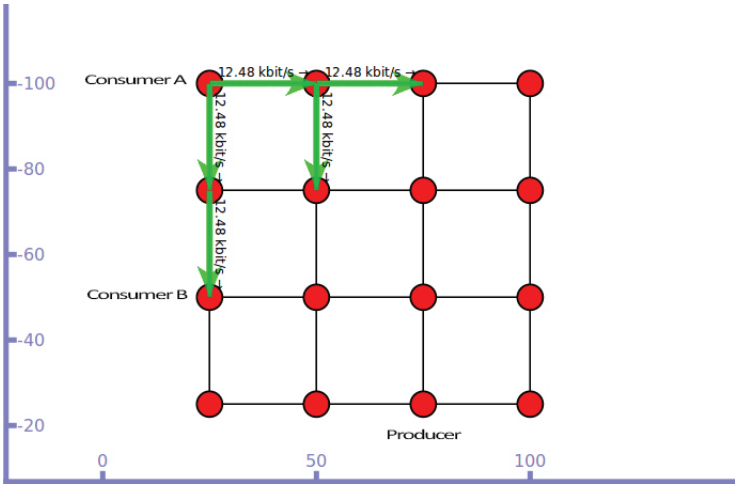


Figure 4.7: List Pushing Performed by Consumer A

information in its FIB. This consumer then sends an interest packet to fetch the content from consumer A as shown in Figure 4.8.

The visualiser ensures that the flow of finding the nearby content is correct. Instead of taking 3 hops, which is the shortest path to the producer, consumer B can find the content in the shorter distance of 2 hops (assuming no consideration of other caches), which is located in a different path to the producer. Notably, in this case, we allow only the requester (i.e., consumer A) to push its *Content List*. It is also noted that the content might be opportunely found at an intermediate node in the default path.

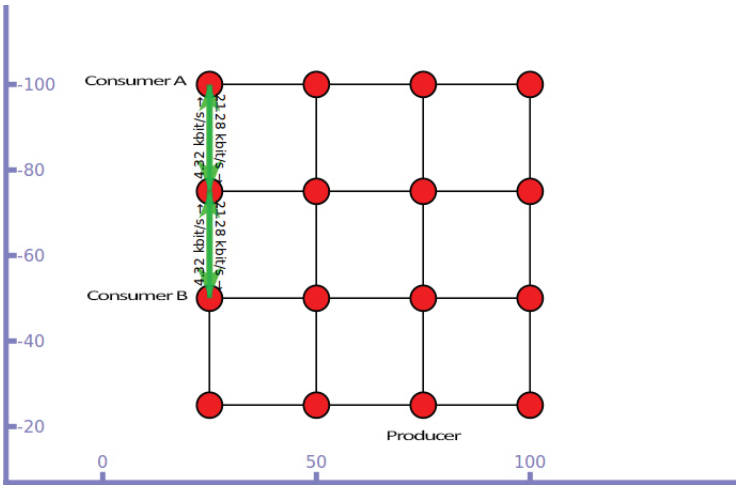


Figure 4.8: Second Request Performed by Consumer B

Nevertheless, the content can also be found at an intermediate node to the list owner or in a nearby vicinity.

4.4 Summary

In this chapter, we presented the detailed implementation of the VCoF scheme. We started introducing the core software components necessary for setting up a fully running system of NDN. These led to understand how the implemented modules are applied in the existing NDN architecture. We then described the implementation of VCoF consisting of the designed modules. In particular, we presented the detailed packet processing flows integrated with the modified core forwarder of NDN. We also described a number of tools to be used to evaluate the VCoF scheme, including the emulator to firstly understand our scheme in a fully running system, and the simulator to simulate larger-scale networks compared to using the emulator. The simulator is also used for the mobile case. In the next chapter, we take the implementation and evaluate it in a number of scenarios using the aforementioned evaluation tools.

In table 4.1, we summarise the implemented modification of the three software components of NDN. We list the added element(s) in each interest/data packet structured in ndn-cxx, which enable the packet to carry the extra information for use in the VCoF scheme as detailed in Table 4.1 (a). The modification of the core forwarder and the major tables in NFD is then summarised in Table 4.1 (b). Finally, the set of tools to be used in the application level is presented in Table 4.1 (c).

Each tool has its attributes to generate interest/data packets according to its requirements. The *"name"* attribute is to set the name of an interest, data, or list pushing packet. The *"packetType"* attribute indicates the type of a packet. The interest timeout is defined by the *"interestLifetime"* attribute, which the default value is 4000 ms. The *"mustBeFresh"* attribute is to define allowing a stale data request and the default value is False. The freshness of a content object is declared by the *"freshnessPeriod"* attribute, which the default value is 1000 ms. The *"allowCache"* attribute defines the allowance of caching a content object at each intermediate node. It is optional (with the default value of True) to control replica densities for the evaluation purposes (will be described later in Chapter 5).

Packet	Original	Element Added	Function
Interest	name, selectors, nonce, interestLifetime, forwardingHint	packetType	setPacketType(type), getPacketType()
		contentList	setContentList(list), getContentList()
Data	name, metainfo, content, signature	(optional) allowCache	setAllowCache(boolean), getAllowCache()

(a) ndn-cxx

	Original	Feature Added	
Core forwarder	Process in/out packets	Check packetType (getPacketType()) and process packets	
	Interest	*Content List*	Normal Interest
	Look up PIT. Detect interest loops. Add an interest as a pending entry. Find a corresponding Data in CS. Forward the interest to upstream using default routing information in FIB.	Get current *Content List* from a CS to be inserted in a list pushing packet.	Found in a pushed list? (Look up FIB for a corresponding entry to find a list owner)
		Add the list into the packet (setContentList(list)) to carry each name in the list for advertising its availability.	Face selection (Compare the lowest cost of a Face ID to a producer and a list owner)
	Data	Push the list according to the name of each node in a consumer's vicinity.	Send the packet according to the selected Face or to a default Face in case of no entry found in each pushed list.
	Return the data back in the reverse path left by the interest. Satisfy the interest in PIT.	Is the list's destination?	
		Yes	No
	Nack	Add the list into FIB (call getContentList()).	Forward the list to upstream.
	Forward Nack to the requester. Drop Nack, if there is no matching entry in PIT.	Send a corresponding data from the list owner to its consumer.	
		Optional	
		Call getAllowCache().	Check and set allowCache to each data packet.
Table	Original	Feature Added	
CS	Store and serve content objects.	Provide and update its *Content List*	
FIB	Provide each default name prefix of each producer (default routing information)	Provide the exact name of each content in every pushed list.	
		Each name couples with the lowest cost of a Face ID to a list owner.	
		Insert *Content List*.	
PIT	Store pending interest entries.	Ignore creating a reverse path for a list pushing packet.	
		Create a routing path for a data packet from a list owner.	
Content List	None	Store each name in a node's CS into TLV block to be encoded in the interest packet of a list pushing.	

(b) NFD

Added tool	Attributes	Features
pushlist	("name" : [<name>]), ("packetType" : [<type>])	Push a node's *Content List* into its vicinity. Call setPacketType(type) to define the type of *Content List*.
findcontent	("name" : [<name>]), ("packetType" : [<type>]), ("interestLifetime" : [<ms>]), ("mustBeFresh" : [<boolean>])	Generate a normal interest to find a content object. Call setPacketType(type) to define the type of normal interests.
servecontent	("name" : [<name>]), ("allowCache" : [<boolean>]), ("freshnessPeriod" : [<ms>])	Provide a content object. Call setAllowCache(boolean) to (or not to) allow caching a data packet in each intermediate node.

(c) ndn-tools

Table 4.1: Summary of the Implemented Modification of the NDN Code-bases

Chapter 5

Evaluation

In this chapter, we comprehensively evaluate VCoF by considering various factors alongside the trade-offs between delivery efficiency and additional overhead costs in comparison to default NDN. In Section 5.1, we describe the overview of the evaluations. We then perform the first evaluation to understand how VCoF enhances content finding in a fully running system of NDN using the emulator as detailed in Section 5.2. Further in the larger topology, we examine the impact of content popularity with the consideration of an actual data-set of content access. This is done by using the simulator as presented in Section 5.3. Finally, we investigate VCoF in handling mobility by considering the well-known mobile communication model that can also take the benefits of VCoF. This is detailed in Section 5.4.

5.1 Evaluation Overview

To answer the research questions in Section 1.2, we evaluate this work in three different evaluations that give insights in different conditions under which NDN operates. This is to understand VCoF in different evaluation scenarios. Although the three major evaluations look at different aspects, together they focus on measuring the effectiveness as well as the efficiency of content finding performed by VCoF in comparison to default NDN as the baseline. These are briefly discussed as follows.

The first evaluation aims to firstly understand how VCoF enhances content finding in realistic NDN networks. Hence, the actual code-bases of the essential components of NDN consisting of NLSR, ndn-cxx, NFD, and ndn-tools are installed in every node of a snapshot of an actual NDN test-bed. We assume that this is a fully running

system of NDN since all of the essential components are installed. The modified code of VCoF is also installed to deploying the scheme, which is to compare with default NDN. This is done by using the emulator (Mini-NDN). In this evaluation, we can track the flows of packets by looking at the validity of each FIB of every node via the actual command lines. Nevertheless, this is also confirmed by the results.

Based on the previous evaluation, we know that the number of the replicas of desired content (i.e., replica density) is the important factor that can affect different content finding results. So, in the second evaluation, we analyse how the content popularity impacts on content finding by considering the actual behaviors of content access obtained from a data-set. We use the simulator (ndnSIM) which is scale better than the emulator to support the higher content count.

In the final evaluation, we extend the VCoF scheme in the consideration of the mobility context. The important objective of this evaluation is to demonstrate that the scheme can also help to address the issues of failed content delivery and delays due to handover as discussed previously in Section 2.5.3.

In every evaluation, a number of metrics are used to measure VCoF compared to the baseline. A metric may have the similar name in each evaluation but it is slightly different. The overview of the metrics is described here:[1].

- Round Trip Time (RTT) is crucial to indicate the delivery efficiency. The basic definition of RTT in this work is the duration in milliseconds (ms) it takes for a content request to go from a requester to a node that can serve the desired content and back again to the requester. It is used in the same pattern for all evaluations, that is the average RTT of a number of requests is calculated after the end of each experiment. According to the results, the number of requests defined in each of the evaluation in this work are sufficient to indicate the delivery efficiency. This is also confirmed by error bars that represent 95% confidence interval. Nevertheless, it is noted that network conditions and evaluation topologies which could be different from the experiment setups in this work might require a different number of requests to sufficiently calculate the average RTT results.

- Message overhead is to measure a number of interest packets in the context of content finding. In the first evaluation, we focus only on the interest packets generated within a number of request(s) to see the number of additional packets

[1]Note, more details are discussed in each sub-evaluation section.

produced by the VCoF scheme. In the second evaluation, we consider the average message overhead per node, which can help to understand how the entire network traffic in the context of locating content can be affected. In the final evaluation, we calculate the total message overhead after finishing each experiment. This is because the particular requesting behavior performed by the mobile node from specific areas, which means the average message overhead per node might not properly indicate the overhead results.

- Data volume is to understand how much the extra packets produced by the VCoF scheme consume data volume, that can indicate another additional overhead in addition to the message overhead. By considering the first evaluation, the average data volume per node is investigated from the beginning to the ending of each experiment. This is to observe how the results develop from the small number of requests to the higher number of requests. In the second evaluation, the overall data volume results (after completing each experiment) are examined. Based on the results from the first (Section 5.2) and second evaluation (Section 5.3), the data volume does not significantly play a major role. So, we consider only message overhead in the third evaluation to understand the extra costs.

- Re-transmission percentage is specifically used in the final evaluation since the mobile node re-transmits a request if it has missed a desired content object. We expect that the VCoF scheme can find the content faster, i.e., the re-transmission percentage should be reduced because the content can be delivered before handover. Replica hit rate is also considered in the final evaluation. This is because of the particular node mobility and a desired content object that is frequently forwarded to the area of the current NAR of the mobile node. Replicas can be hit frequently due to the concept of VCoF itself. Hence, the hit rate can also help to better understand how nearby replicas can be the potential sources in the context of mobility.

5.2 Enhancing Content Finding

This first evaluation aims to firstly understand the effectiveness as well as the efficiency of VCoF in an instance of actual NDN environments. We need to see how VCoF improves content finding results in NDN while considering against additional

overheads. Mini-NDN, the emulator, is used in this evaluation. We inspect a router-level network with a limited scale to better monitor and understand the flows of packets. The evaluation topology is composed of 22 nodes and 50 links as shown in Figure 5.1. It is a snapshot of an actual NDN testbed [55][2]. A cost to find a content object can be varied dependent on each link cost and the number of hops. To consider only the number of hops, we assume that each link has a homogeneous latency of 10ms. This is because path reductions can be indicated by hop count. The lower number of hops can cause the reductions of delivery delays. The NDN nodes in the topology can be either consumers, routers, and producers, which are the key elements in the content delivery chain of NDN. They are designed to emulate content finding situations.

5.2.1 Experimental Model

Two main scenarios are employed in the evaluation. In the first scenario, we analyse the delivery efficiency of VCoF by investigating how much it can reduce the average round trip time (RTT) compared to the default NDN mechanism. As discussed previously, replica density is important in content finding. A higher number of replicas of desired content can offer better possibilities to find the desired content. So, the caches are filled up with a number of replicas dependent on a specific replica density of a desired content. Replica density is defined by the percentage of the number of

[2]Note, this topology is used to evaluate the main aims of this work and there are some evaluation limitations as described in Section 5.2.4.4

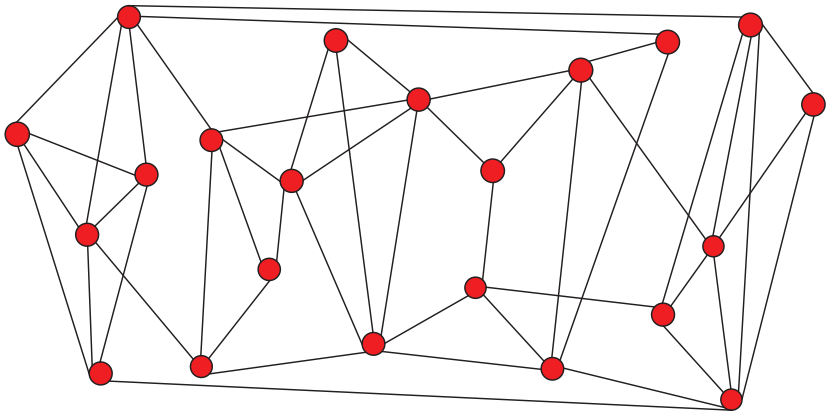


Figure 5.1: Emulation Topology

distinct replicas in the network compared to the potential number of distinct locations that they could be cached. A higher percentage means a higher number of replicas in the topology. In every experiment, each cache is fixed to prevent that successive requests affect the further outcome of the test, i.e., an increasing of the replica density is prevented. In each density, we execute 30 requests from randomised consumers and the average RTT of these requests is then calculated.

In the second scenario, we examine the overhead costs of VCoF in comparison to default NDN. This is to consider the trade-offs between delivery efficiency and additional overheads. To clarify how the message overhead can be generated, we consider the additional messages that only involve in our VCoF scheme. These messages include the interest packets for requesting the desired content, and for pushing the *Content List(s)*[3]. We execute 30 runs in every experiment to calculate the average message overhead.

In considering the message overhead, we focus on two factors that can affect the overhead results. First, a higher number of Faces (i.e., connected links) of an NDN node can increase the number of packets for pushing its *Content List*. This might result in higher message overhead costs. Hence, to evaluate this, we set only one randomised requester and producer in the topology within each of different number of links, including 30, 35, 40, 45, and 50 links, respectively. This is to understand how high the message overhead is generated in each request in consideration with each of defined number of links, which can represent low to high connections in the topology. Second, we also investigate the impact of the number of requesters. This is because a higher number of requesters can generate a higher number of requesting packets and list pushing packets. In each experiment, the interest packets generated by the requests from the requesters are counted. The number of randomised requesters vary from 5 to 35.

In addition, VCoF might consume additional storage or traffic in each node to process the packets especially for *Content List* pushing. Hence, we also analyse the average data volume in each node. We test from the small number of requests to the higher number of requests (the beginning to the ending of every experiment (time t_1 to t_{30})) to see the development of the data volume. It is noted that each time step has one requester that requests a desired content object, thereby starting with the low number of requests to the higher number of requests.

[3]Note, due to no list pushing by using default NDN, in this evaluation, the message overhead costs of default NDN are only the interest packets for requesting the content.

5.2.2 Vicinity and Content Placement

As seen in [5], the number of hops can be used to indicate path reductions and delivery performance. Hence, based on this knowledge, we solely investigate on the number of hop(s). It should be noted that there are a number of other factors in real networks that can affect delivery delays such as different link latency characteristics, jitter, throughput, packet loss, etc. However, this work aims to first prove that nearby content can be the better source for content finding. So, the number of hops can properly indicate the improvement in the context of locating content, which means cutting distances is cutting delays. This is fair to compare VCoF to standard NDN under the consideration of the same network conditions. Nevertheless, to be more practical, those factors need to be further investigated and additional optimisation mechanisms might be required to be able to properly deploying in more complex network conditions.

A trade-off that has to be essentially examined is what the optimal size of the vicinity is to find nearby replicas by still keeping the costs low or acceptable. In our hypothesis, the vicinity size should not be too large. We expect that, by centering a consumer, a replica should be located nearby. If it cannot be found nearby, the default path is still a suitable way to locate the content. This is to control the scope of content finding especially for mitigating the potentially expensive message overhead of the list pushing packets. Hence, to evaluate a narrow vicinity, the size is set to 1 hop. We expand the vicinity size to 2 and 3 hops to evaluate the larger views of content/replica finding. It means that the scope of the *Content List* pushing (i.e., the vicinity size) varies form 1 to 3 hops.

In the first scenario mentioned in Section 5.2.1, a producer is randomised and then random consumers (i.e., requesters) request a desired content object from the producer to create replicas in the topology depending on the percentage of their density. For example, 50% replica density means there could be around 11 nodes in the evaluation topology that cache the replicas of the desired content. This is to assume that the replicas of the desired content are distributed in the network based on their pre-defined percentage that each of this includes the cases of low-to-high replica densities. We expect that different replica densities in the network can impact different content finding results. To control the replica density, we allow only the requesters to cache the replicas. This is because if we let any nodes along default paths from the producer to the requesters cache the replicas, it is impossible to control the percentage of the

Number of nodes			22 nodes
Scenario	1	Replica density (percentage)	[5,20,50,70,90]
		Number of requests after each cache is fixed (to get avg. RTT)	30 requests
	2	Number of links	[30,35,40,45,50 (default)]
		Number of requesters	[5,10,15,20,25,30,35]
Vicinity size			[1,2,3]
Content chunk size (a complete object)			1024 bytes
Link latency			10 ms

Table 5.1: Summary of Emulation Parameters

replica density that we would like to evaluate. So, we assume that the replicas of a desired content item are already distributed by following a defined percentage that we need to examine. This is done by using the added *"allowCache"* element described in Section 4.2.1.

A producer provides a content object that is a piece of data. It is carried in a data packet to be forwarded to a consumer when a corresponding interest packet has arrived at a node that can serve the content. In the evaluation, each object is created in one chunk and has the same size of 1024 bytes.

In the second scenario mentioned in Section 5.2.1, we perform the three major experiments to study the overhead costs. The first experiment investigates the overhead costs in consideration with the number of links. We analyse the average number of interest packets (message overhead) that each request produces by using the VCoF scheme compared to default NDN. The second experiment examines the average message overhead costs, varying the number of requesters (5 to 35). In the third experiment, we aim to understand the data volume that might affect the storage consumption or traffic in each NDN node. Emulation parameters are summarised in Table 5.1.

5.2.3 Metrics

In this evaluation, we focus on three main aspects including delivery efficiency, message overhead, and data volume in the consideration of the aforementioned scenarios. We analyse these aspects by comparing VCoF within different vicinity sizes to default NDN. The evaluation metrics are itemised below.

- RTT: the average RTT per request by each consumer to retrieve its desired

content object. The goal of this metric is to measure the delivery efficiency. The comparison between VCoF and default NDN is investigated. A lower RTT indicates that each consumer can fetch its desired content object faster.

- Message Overhead: We consider the overhead packets performed by the requester(s) including the interest packets for requesting the desired content, and for pushing the *Content List(s)*. We examine the impact of the number of links and requesters. We focus only on messages that involve in content finding and therefore ignore some system messages such as NLSR messages or local messages. A higher average number of messages mean a higher number of packets in each node that might increase the network traffic. However, we need to consider that the overhead is acceptable or not to perform a better performance in terms of content delivery.

- Data Volume: To understand how much the packets in the context of content finding (especially the *Content List* pushing packets) can affect the data volume of each node, the average data volume in each node is hence measured. The data volume results include every packet that passes through each single node. Higher data volume results mean higher traffic and disk utilisation. This might impact the entire network and higher storage consumption in the nodes.

5.2.4 Results

In this section, the results of the experiments alongside the consideration of the aforementioned metrics within a number of emulation scenarios are presented. In the following results, we also discuss main findings. It is noted that these results are the numbers obtained from the experiments performed in this work and they could be different in real world with more complex network conditions. Nevertheless, the results can still strongly support the proof of the improved content finding by using the VCoF concept compared to the baseline in an NDN network, made possible through deployment on real systems.

5.2.4.1 Round Trip Time (RTT)

This analysis is to quantify how well default NDN deliveries content compared to VCoF to see how much the proposed scheme can improve. The results of the average

RTT per request (with error bars that represent 95% confidence interval) are shown in Figure 5.2.

The results indicate that VCoF benefits in almost every case of replica densities. We start considering the case of the low replica density (e.g., 5%). In this case, there are a low number of replicas in the network. Hence, for both default NDN and VCoF, the average RTT results are comparatively high while considering against the cases of higher replica densities. For example, it is 128.26ms for default NDN and 115.2ms for the vicinity size of one in this case of 5% density. Interestingly, VCoF is better since it offers alternate opportunities to locate nearby replicas in each vicinity or nearby vicinities.

When the replica density is medium (e.g., 50%), VCoF can gain the most advantage compared to default NDN. This is because the higher number of replicas and the ability to find these replicas performed by VCoF increase the opportunities that the desired content can be located nearby and thereby faster. In this medium replica density case within the 1-hop vicinity, the average RTT is approximately 67.86ms while default NDN is around 101.98ms. Nevertheless, in the case of the highest replica density (90%), the replicas are cached at almost every node. Hence, they can be easily located, not just in the off-paths but also in the default paths. This is why in this specific case, the differences between VCoF and default NDN are marginal.

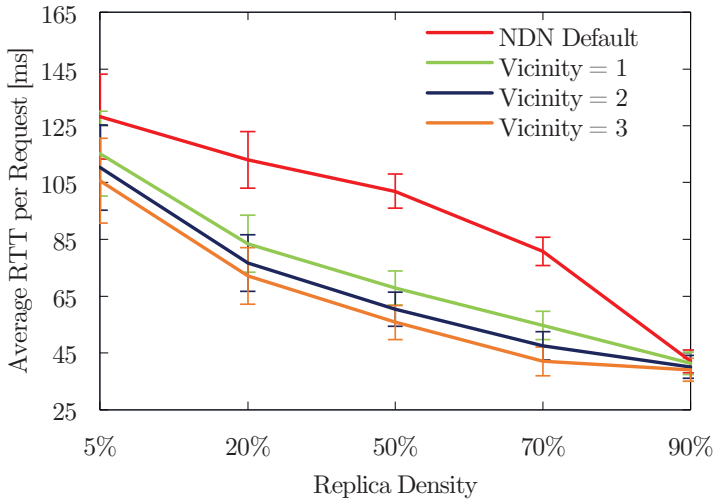


Figure 5.2: Average RTT per Request Varying Vicinity Sizes and Replica Densities

5.2.4.2 Message Overhead

In this analysis, two factors that can affect the message overhead results are considered. First, we examine the impact of the number of links. As presented in Figure 5.3, by using default NDN, the results of the overhead are quite stable. The results performed by VCoF within the 1-hop vicinity are slightly higher than default NDN since VCoF has to push *Content Lists* in each vicinity within a narrow scope. The overhead results increase alongside the increment of the number of links. When the vicinity size is set to two, there are around 20 additional packets from default NDN in the default 50 link case. By considering the trade-off between the gained performance and the generated overhead, the results are likely too high when the vicinity size is set to three.

Second, further to the investigation of the number of links, the involvement of the number of requesters on the additional message overhead generated by VCoF is examined and the results are presented in Figure 5.4. The results are slightly increased by using default NDN alongside the increment of the number of requesters. Similarly, there are slight increments, when the vicinity size is set to one. There are only 22 additional packets within the 1-hop vicinity in the case of 5 requesters. When the vicinity size is set to two or three, the results are higher. In particular, the overhead packets increase to approximate 196 packets in the case of the 3-hop vicinity with 5 requesters. It can be seen that a larger vicinity means that VCoF must push the

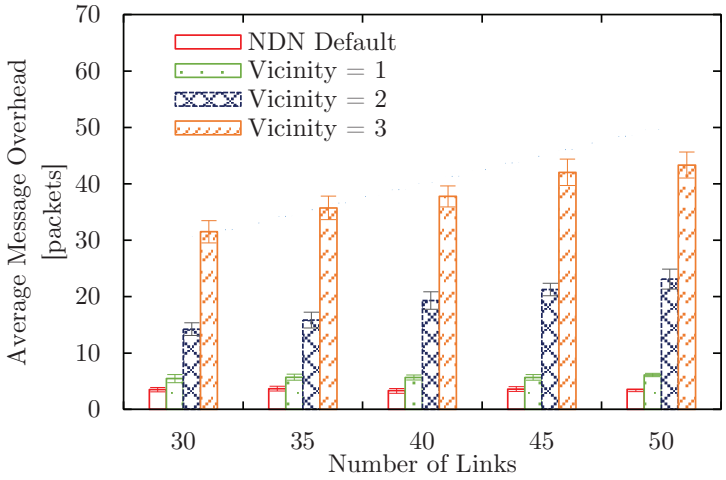


Figure 5.3: Average Message Overhead Varying the Number of Links

Content List to a higher number of nodes and the increased number of requesters creates a greater number of packets to be pushed. According to the results, the message overhead growth is almost linear. If we increase the number of requesters (more than 35 requesters), the graph would grow in the same pattern.

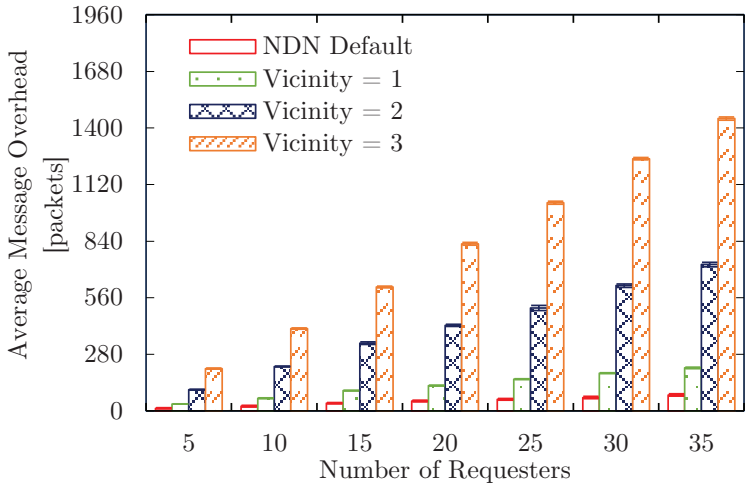


Figure 5.4: Average Message Overhead Varying the Number of Requesters

5.2.4.3 Data Volume

According to Figure 5.3 and 5.4, the message overhead results seem to be significantly increased in the 3-hop vicinity cases. This overhead might consume higher data volume in each node. Hence, the goal of this analysis is to understand how much the data volume overhead can be incurred in every node in addition to the increment of the requests. We start considering default NDN and VCoF within the 1-hop vicinity. As illustrated in Figure 5.5, the difference is very marginal and the results are approximately 0.57±0.01MB. These are quite stable from time t_1 to t_{30}. This is because each node sends a small number of packets even including the list pushing packets. Although the results include all messages (e.g., default routing messages, and messages created by VCoF), the data volume results are still low (the maximum value is less than 0.6MB) due to the small size of the pushed packets.

When the vicinity size is set to two, the results are slightly increased (e.g., VCoF adds only 0.048MB to the baseline after the ending of the experiment) since there are the higher number of packets to be processed. Similarly, the results are higher, when

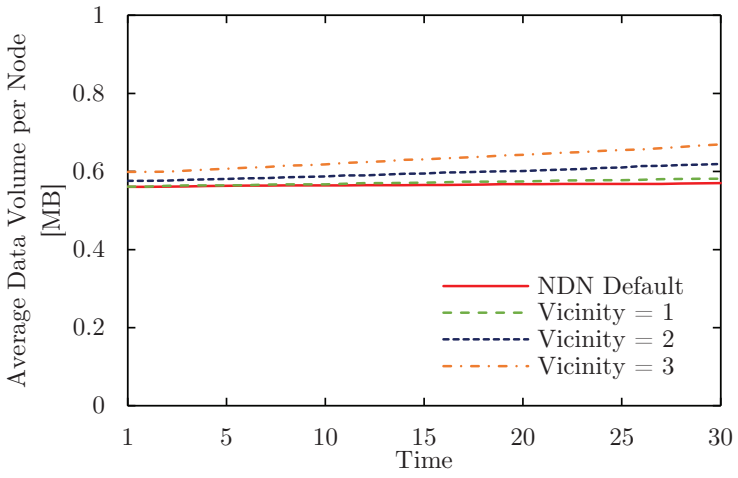


Figure 5.5: Average Data Volume per Node from Time t_1 to t_{30}

the vicinity size is three. Interestingly, in considering the results in Section 5.2.4.2, although the message overhead costs seem to be higher alongside the increment of the vicinity sizes, the differences of the data volume results are negligible. In the worst case (the 3-hop vicinity at time t_{30}), it is only ≈15% high compared to the baseline of default NDN. It can be seen that the list pushing packets are quite small and do not introduce a huge impact to the network traffic and storage consumption.

However, it is important to note that the more packets created, the more processing has to take place, which may overload the network. So, we suggest that the vicinity size should not be too large to keep the overhead as low as possible, while introducing considerable gains. For example, in the medium replica density case (50%) within the 1-hop vicinity, the performance of content delivery is improved to 33.46% while the additional average message overhead per request can be only 3 packets. However, although the improvement is higher in the larger vicinity size of three (e.g., in this same case, 11.8% improved over the vicinity size of one), the overhead is significantly increased (13 times over the smallest vicinity size). This can outweigh the benefits.

5.2.4.4 Summary and Discussion

The default best route strategy of NDN focuses on finding a content object according to its name prefix, which this object is normally located on-path (i.e., default path). We argue that nearby off-path replicas can be the better source. VCoF is designed to proactively provide routing information of these nearby replicas. This can be of

		Vicinity Size		
		1	2	3
Improvement over Baseline (%)	Low Replica Density (5%)	10.18	13.97	17.61
	Medium Replica Density (50%)	33.46	40.74	45.32
	High Replica Density (90%)	1.73	4.64	7.07
Increased Average Message Overhead over Baseline (packets)		3	20	40
Increased Average Data Volume over Baseline (MB)		0.011	0.048	0.099

Table 5.2: Results Summary of Improvement with Additional Overhead

advantage by providing a higher efficiency in locating content. This first evaluation has demonstrated how VCoF enhances content finding in NDN. The results have indicated that VCoF can significantly improve delivery efficiency with some additional overhead costs especially in the cases of the narrow vicinity size of one. Although, a larger vicinity size such as the vicinity size of three can provide further improvement, the trade-off between the gained performance and the overhead might not be worth it to spend too much costs.

According to the results, considering the trade-off between the achieved benefits and the overhead costs, the experimental results have demonstrated that the 1-hop vicinity can be considered as a good trade-off since VCoF provides considerable benefits while introducing slight increments of the overhead costs compared to the default NDN cases. As shown in Table 5.2, the percentage of improvement compared to the baseline can reach up to 33.46% in the 1-hop vicinity case. In considering the overhead in this case of the 1-hop vicinity, a small increment of the average of 3 additional message overhead packets and also the slightly increased average data volume of 0.011MB have incurred.

In some particular cases, due to the further decreased RTT (e.g., 40.74% of improvement in the 50% replica density), the 2-hop vicinity might still be worthwhile. For example, where a producer is quite far from a consumer and the neighborhood within which there is interest in the content is larger than one hop. The overhead results of the 3-hop vicinity are likely to be too high in consideration of the performance gained. 40 packets of the increased average message overhead per request have demonstrated the highly excessive overhead while getting only a slight improvement over the smallest vicinity. It should be noted that the overhead packets do not consume high bandwidth since VCoF introduces only 0.099MB of the increment of the data volume in the worst case of the 3-hop vicinity.

The experiments have indicated the effects of content finding using VCoF in a static network environment. Though, there are other scenarios that VCoF can also provide benefits. In the mobility context, for instance, VCoF can help to find content in the vicinity or nearby vicinities of a mobile node. In particular, when it has missed desired content due to handover. Hence, this leads to extend VCoF in supporting mobility, which will be described further in Section 5.4. Furthermore, locating nearby replicas with different content popularity distributions can help to understand the impact of content popularity to content finding. The different number of replicas of desired content that are distributed by the requests of consumers can impact content/replica finding results and this is explored in the next evaluation section.

Evaluation Limitations: In considering the topology, it may not be considered as a practical topology that represents a current network, which is based on the existing Internet architecture. We need to consider the topology that can potentially represent an NDN topology in the Future. This is because routers and end devices are allowed to cache/serve content. We believe that hardware customisation specifically for NDN must be required. This can change the way of content delivery. Hence, the snapshot of an actual NDN test-bed is likely an appropriate topology that can help to understand content finding performed by VCoF in comparison to standard NDN since every node in the topology can perform the full of NDN primitives.

In contrast, if the network topologies in the Future are very similar to the current topologies even integrated with NDN, we can still deploy VCoF specifically in different parts of the networks. For example, an ISP backbone network can be possible. VCoF can also deploy only in core routers that are connected with their network segments. At edge networks (e.g., a home network), a topology could have various communication technologies [163]. It is common to have Bluetooth, Zigbee, WiFi, Ethernet, etc. This means there are a different number of routing parts or devices to find content. Hence, VCoF can likely help to handle these networks.

Nevertheless, the strategies of *Content List* pushing should be adjusted or optimised depending on the different characteristics of networks, e.g., the number of content updates and parts in the networks as described in Section 3.2.2.2. Content finding results in different network conditions might be different from the evaluation results in this work especially the overheads. Nevertheless, we still believe that if desired content can be found nearby, the delivery efficiency must be improved since due to the results, distances to get desired content performed by VCoF are usually

closer than performed by standard NDN. In the Future, some further mechanisms to optimise and handle the issues such as the overheads, network congestion, content updates regarding different network conditions can be developed afterwards.

5.3 Impact of Content Popularity

The goal of this evaluation is to examine the impact of content popularity to content finding. In the previous evaluation, we have found that the different number of replicas of desired content can significantly affect the content finding results. Hence, this evaluation aims to profoundly examine the popularity distribution of content using a real dataset [164] of actual content access. One limitation incurred using the emulator in the previous evaluation is the ability to emulate a higher-scale network. For example, default name prefixes propagated by the routing protocol (NLSR) take a considerable time for routing convergence, especially in the case of a high number of nodes. Using the emulator, each name prefix of every node should be disseminated to other nodes in the topology to calculate default routing information. This is the important process of routing convergence[4]. The experiment fails, if there is a node's name prefix that is not fully disseminated to every node in the topology because the process of routing convergence is not complete.

Therefore, we port the implementation code to the simulator (ndnSIM), which can simulate a higher-scale network compared to the previous one. This is to support the higher content count in the dataset. The simulation topology which is another snapshot of the actual NDN testbed introduced in [165] consists of 46 nodes with 138 links as presented in Figure 5.6. This is to cover the number of consumers/producers selected from the dataset. We still assume that each link has the same cost (a homogeneous latency of 10ms). It is noted that this environment is to understand VCoF compared to the baseline under the same network conditions defined specifically for this second evaluation and a number of limitations should be considered as discussed in Section 5.3.4.4.

[4]Note, routing convergence is the process that the routing algorithm computes routes and builds default routing information in a set of NDN nodes in a topology. This is done by exchanging each name prefix of each node across the network.

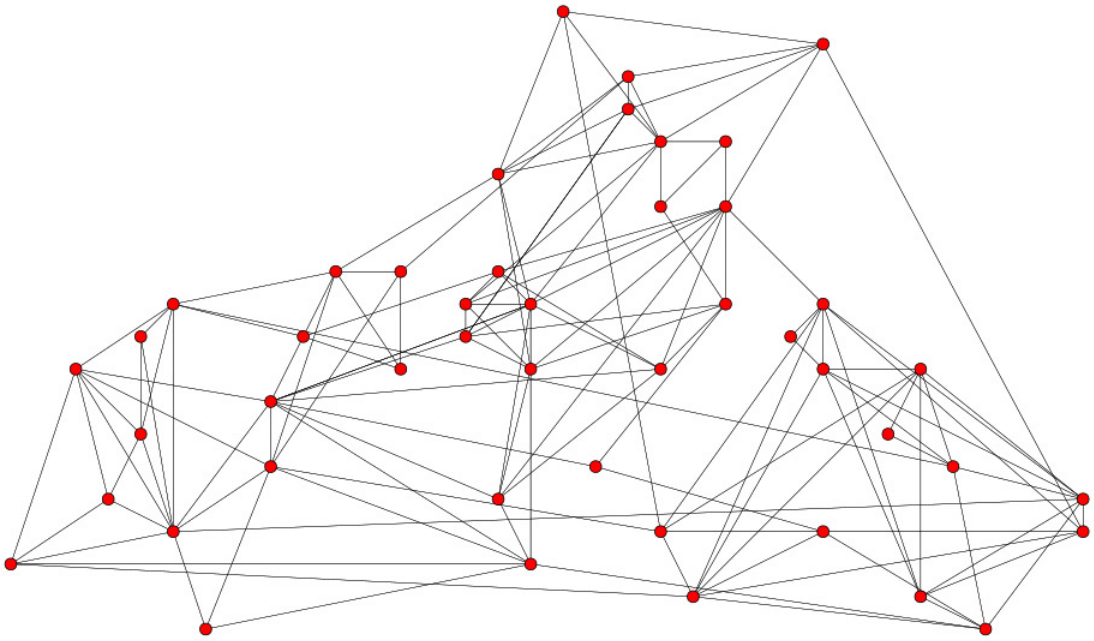


Figure 5.6: Simulation Topology

5.3.1 Experimental Model

We consider the dataset [164] that is collected from an IPTV-service that provides content for consumers. The dataset shows content popularity since there are several consumers requested the same content. We examine different TV programs in the dataset since they can represent the popularity levels in every case including low, medium, and high. A content popularity distribution is defined by the percentage of the number of requests (created replicas) of a program compared to the overall requests[5].

Before running an experiment, the caches are filled up based on the dataset by considering two selections of different numbers of content objects. First, we investigate a less populated network by filling up the caches with a 100 selected objects. Second, 500 objects are filled to evaluate a more populated network. A less populated network represents an NDN network that at the time where it has been exposed to a low number of overall requests, which means the caches or the consumers have seen low traffic passing through. In contrast, a more highly populated network is an NDN network that has been exposed to a higher (frequent) number of requests, which

[5]Note, a program might be requested more than one time by a single consumer.

means a higher number of content objects are in the caches or at the consumers (i.e., highly densely populated).

When a requester node receives a desired content object[6], the *Content Availability Advertisement* module is activated to push the latest node's *Content List* into its vicinity. This is to proactively advertise content availability of the node to other nodes in its vicinity. It might create additional overhead but the scope (i.e., vicinity size) of the list pushing can mitigate the overhead. Hence, we investigate the list pushing in different vicinity sizes to understand how the overhead costs develop in the real behaviors of content access. We push *Content Lists* for only desired content since we need to see how overhead costs develop in the context of locating the particular content that represents a case of content popularity that we would like to investigate. We focus on the spending overhead to find only each desired program (i.e., content). Unlike the previous evaluation where we examine how the overhead costs develop with a number of request(s), this evaluation is to understand the average message overhead generated by VCoF for each particular content program based on its popularity.

Three packet types of interest messages necessary include: packets for filling up the caches depending on the number of content objects (i.e., 100 or 500), for requesting content (30 requests) after the caches are fixed, and for pushing the *Content Lists* for each desired content by the requesters. It is noted that message overhead costs in the context of locating desired content are considered by including the baseline of the costs (i.e., packets) of other content provided by the dataset to fill up the caches.

Content popularity might be affected by cache sizes. For example, a small cache size can increase the cache replacement rates and decrease the opportunities to locate replaced replicas. This means the replacement can affect the content availability. It is noted that the impact of cache performance might also affect content finding results since desired content objects can be removed depending on a caching policy. However, in this work, we do not want to determine which caching policy performs the best while applying VCoF. In our hypothesis, the actual impact would be cache replacement because it can reduce the number of desired content items that decrease the opportunities to locate the content/replicas. A smaller cache size can also force a higher replacement rate. Therefore, to understand the impact of the cache

[6]Note, a desired content object means a complete content object that a consumer is trying to find. In this second evaluation, a desired content is a TV program in the dataset that represents a particular case of content popularity (could be low, medium, or high depending on the percentage of the popularity) that is examined.

replacement, the small cache size of 10 and the large cache size of 1000[7] are evaluated within different content popularity levels.

5.3.2 Vicinity and Content Placement

A particular level of content popularity is considered based on the dataset. The replicas of a desired content representing a level of content popularity are placed (i.e., filled) in the caches by requesting the content generated by its producer based on the popularity percentage. One issue of each experiment is that requesting a desired content object several times can increase its popularity. However, we have to control content popularity depending on each predefined percentage calculated from every selected TV program in the dataset, representing each of content popularity level. Similar to the first evaluation, we assume that replicas are distributed only at their requesters based on their percentages. We then control every level of content popularity by freezing (i.e., fixing) each cache before collecting the average RTT of 30 requests. The average value of the requests performed in each experiment is then calculated. Therefore, all of the average RTT values are compared to indicate the performance of VCoF alongside the consideration of default NDN.

Through the previous evaluation, it is to be expected that replicas should be located closer especially alongside the increment of content density or popularity. Based on the knowledge of the previous evaluation, 1-hop vicinity can be the good

[7]Note, this is the default cache size. We do not expect any cache replacement since the number of content in the evaluation are not sufficiently high compared to this default cache size of 1000. This is only for the evaluation purpose as the large cache size can remain content for longer compared to the small cache size (which is also evaluated) that can have some cache replacement cases.

Topology	46 nodes, 138 links	
Link latency	10 ms	
		Content popularity (%)
Number of content objects	100	[1,7,11,37]
	500	[0.6,2.6,7.4,12.2,20]
Number of requests after each cache is fixed (to get avg. RTT)	30 requests	
Cache sizes	[10,1000(default)]	
Vicinity sizes	[1,2,3]	
Content chunk size (a complete object)	1024 bytes	

Table 5.3: Summary of Simulation Parameters

trade-off, while 3-hop vicinity has demonstrated the unworthy trade-off between the performance gained and overhead obtained. Hence, we again investigate the narrow vicinity size of one and expand the vicinity size to 2 and 3 hops, respectively. The simulation parameters of this second evaluation are summarised in Table 5.3.

5.3.3 Metrics

In this second evaluation, we employ three key metrics consisting of the average RTT per request, the average message overhead per node, and the overall data volume. The details of each metric are itemised below.

- RTT: the average RTT per request performed by the random consumers to find a content object within each of content popularity level represented by its percentage. A lower average RTT indicates that the consumers can locate the desired content object faster.
- Message Overhead: After finishing an experiment, the interest messages are counted from every node. The average message overhead result is then calculated. We filter only messages that are involved in content finding. A higher result indicates the increment of traffic. However, the trade-off between the overhead and the achieved performance in terms of delivery efficiency should be ultimately considered.
- Data Volume: the overall data volume obtained from each node after an experiment is ended. In the evaluation, all incoming packets are counted as megabytes. A higher data volume means a higher traffic or overall storage consumption that could impact the workload of each node or the entire network.

5.3.4 Results

The experimental results are presented and examined in this section alongside the consideration of the aforementioned metrics. In each metric, we also discuss main findings considered from the results. Similar to the previous evaluation, it is again noted that these results obtained from the experiments performed in this work could be different in other environments or real world with more complex network conditions.

5.3.4.1 Round Trip Time (RTT)

This analysis is to indicate how well VCoF deliveries content compared to default NDN. The results with error bars that represent 95% confidence interval are shown in Figure 5.7 and 5.8. It should be noted that Figure 5.7 represents the results for both cache sizes (10 and 1000 objects) since there is no significant difference by considering the impact of cache sizes in the less populated network[8] (described further in the following paragraph).

In the case of the less populated network, as presented in Figure 5.7, by comparing the cases of the cache size of 10 and 1000, cache size does not have any significant effects since the number of content objects are not high enough to increase cache replacement rates even in the case of the small cache size of 10. When the content popularity levels increase, VCoF can provide higher benefits compared to default NDN due to the higher opportunities of finding nearby replicas. In the case of the highest popularity level (37% in this case), the overall distribution of the replicas of desired content is approximately average compared to the total number of nodes. Nevertheless, the increment of these replicas still shows the visible gaps between VCoF and default NDN. For example, in the 1-hop vicinity, 55.3ms is from default

[8]Note, the definition of the less populated network and more populated network is already defined in Section 5.3.1, which is also used to describe the results of this second evaluation.

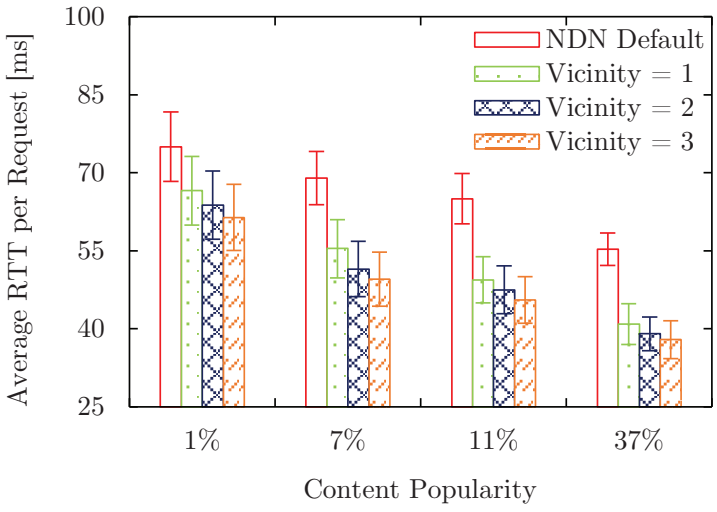
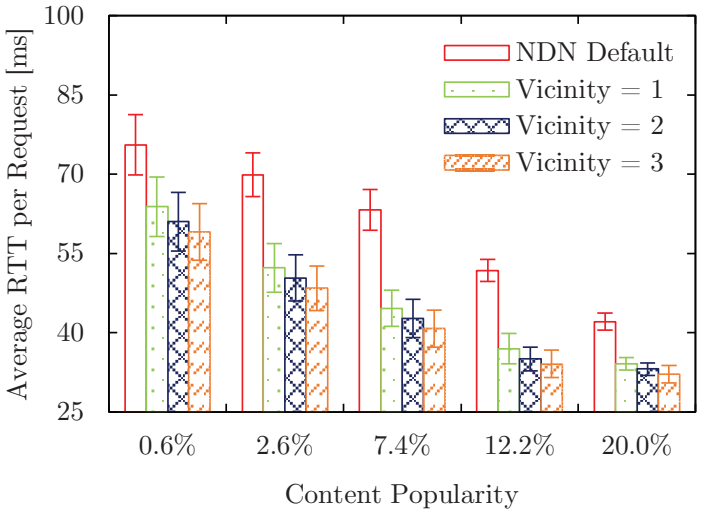
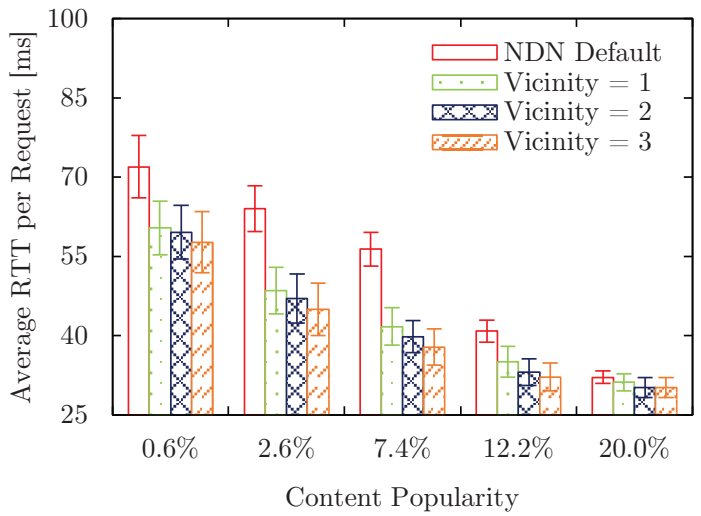


Figure 5.7: Average RTT per Request in the Less Populated Network



(a) Cache Size = 10 objects



(b) Cache Size = 1000 objects

Figure 5.8: Average RTT per Request in the More Populated Network

NDN while 40.91ms is performed by VCoF. This has demonstrated the significant improvement provided by VCoF even in the less populated network.

As presented in Figure 5.8 (representing the more populated network), VCoF still shows higher advantage over the baseline. This is because the designed scope of content finding can increase the opportunities to find nearby replicas in each vicinity

or even closer vicinities. However, when the popularity level is high (e.g., 20%), in the case of the 1000 object cache size, the differences between VCoF and default NDN are negligible because the replicas can be remained in each cache for longer. It is effortless to locate the replicas since they are already cached in almost every node even using default NDN. However, in the case of the cache size of 10, the replicas might have been replaced with other content more, reducing their availability. So, it is more difficult to find the replaced replicas but VCoF can still significantly reduce the RTT with the noticeable gaps to default NDN. For instance, in the 1-hop vicinity case, the result of default NDN is 42.10ms while the result of VCoF is 34.13ms.

By comparing the same percentage of content popularity between the two cache sizes, the results have demonstrated that the large cache size (Figure 5.8 (b)) slightly decreases the RTT compared to the small cache size (Figure 5.8 (a)). For example, in the case of 0.6% with the small cache size, the average RTT result of default NDN is 75.5ms, while the result of the large cache size is 71.9ms. Similarly, the differences of VCoF itself develop in the same direction. This is because in the large cache size, the replicas can be remained for longer and it is easier to find these available replicas, resulting in the lower RTT.

5.3.4.2 Message Overhead

In this analysis, we examine the message overhead costs generated by VCoF in comparison to default NDN. Figure 5.9 and 5.10 present the average overhead (with error bars of 95% confidence interval). It is again noted that Figure 5.9 contains the results from both cache sizes due to no noticeable difference, which is described later.

We consider the overhead results along with the improvement of delivery efficiency presented in the previous section. As shown in Figure 5.9, when we increase the content popularity levels, the message overhead costs also increase. This is because the replicas of desired content are distributed to the higher number of nodes, increasing the *Content List* pushing packets. It should be noted that the desired content can be found in the local cache of a single node if this node performs a number of requests more than one time. Hence, there is no *Content List* pushing packet for any request looking for this content after the first request.

In the case of the 3-hop vicinity, the costs are likely high after considering the achieved performance compared to the smaller vicinity sizes. The 2-hop vicinity can be worth it while the RTT can be significantly reduced with considerable overhead.

In the case of the 1-hop vicinity, the costs are slightly increased compared to default NDN, while the RTT results are improved significantly. In this case, the improvement can reach 26.02% while the increased average message overhead is only 5.5 packets. In this less populated network, cache size does not again affect the results since the number of filled content in the caches are not sufficiently high to create some cache replacements.

In the more populated network as presented in Figure 5.10, the message overhead results slightly increase considering the cases of the 1-hop vicinity. For example, the additional average overhead is only 5 packets in the case of the cache size of 10 and medium content popularity. This is due to the fact that the packets are pushed in the small areas caused by the narrow scope of the vicinity. In considering the delivery performance, the results have shown favorable outcomes performed by VCoF. For instance, the improvement can reach up to 29.39% within this narrowest vicinity size. In the 2-hop vicinity cases, there are still performance gained but with the higher overhead costs. In particular, when a producer is located further away, expanding the vicinity size from one to two can be beneficial. In cases of the 3-hop vicinity, the increased overhead results could considerably outweigh the benefits. Interestingly, in the case of the large cache size, the results are slightly reduced compared to the small cache size. This is because the number of packets for finding desired objects travel shorter distances (i.e., hops). This is due to the higher number of replicas remaining

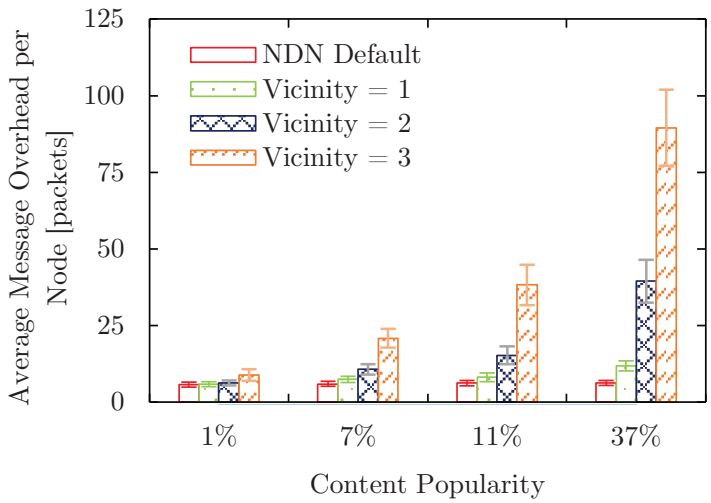
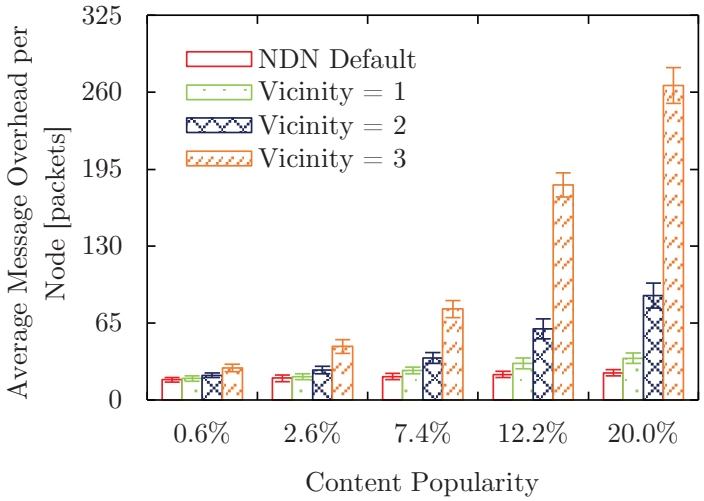
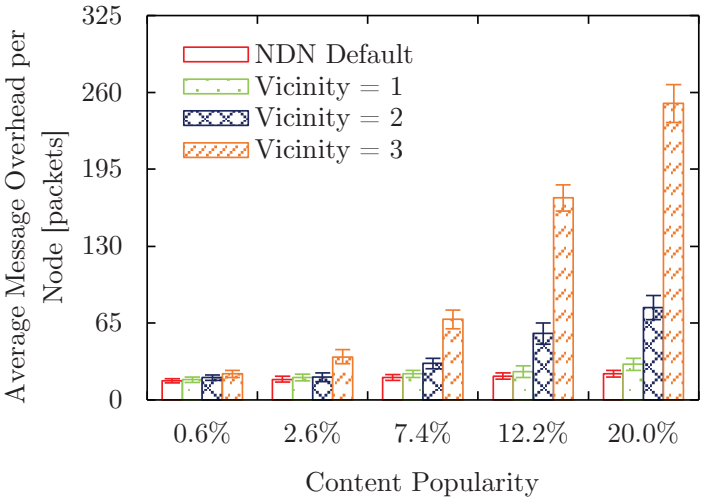


Figure 5.9: Average Message Overhead per Node in the Less Populated Network



(a) Cache Size = 10 objects



(b) Cache Size = 1000 objects

Figure 5.10: Average Message Overhead per Node in the More Populated Network

longer in the caches. Another reason is that in the case of the small cache size, the availability of some replaced replicas is lower causing the higher number of packets to locate them, for instance if they are re-requested again to fill up the caches.

5.3.4.3 Data Volume

The goal of this analysis is to quantify how much the data volume overhead is incurred. As presented in Figure 5.11 (covered the results from both cache sizes), cache size does still not play a major role in the less populated network since the number of replicas in each cache remain similar. In considering the low content popularity, the differences are slight since there are a low number of packets to push the *Content Lists*. This is because the replicas of the desired content are distributed in a few nodes due to the low levels of the content popularity of these replicas. This means the number of list pushing packets for this particular content marginally increase the number of packets in the topology compared to the default strategy. So, the overall number of increased packets in the topology are not high resulting in the slight increment of the data volume results. When we increase the vicinity sizes, the results are higher due to more number of list pushing packets in the larger scopes. Nevertheless, although in the case of the highest content popularity (37%) within the 3-hop vicinity (i.e., the worst case), the results are still slightly different (the increment is only 1.01MB compared to the baseline).

In the more populated network, as shown in Figure 5.12, due to the higher number of packets related to content finding, the overall results increase compared to the less populated network. These packets include the packets for filling up the caches, finding content, and pushing the lists. However, since the packets for pushing the lists are

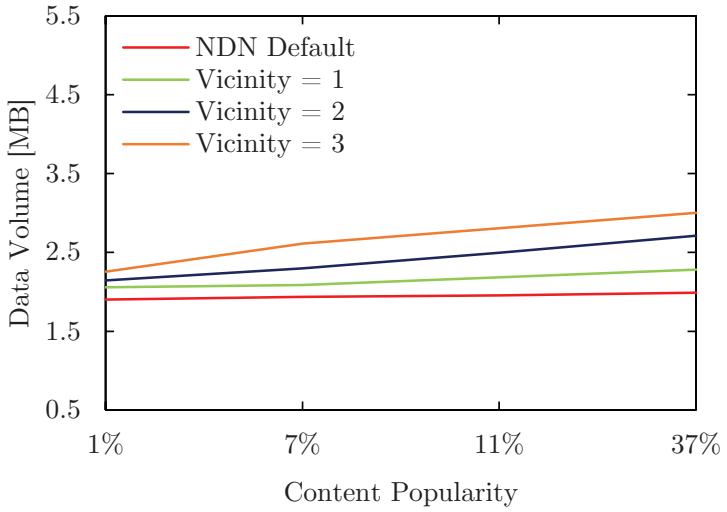
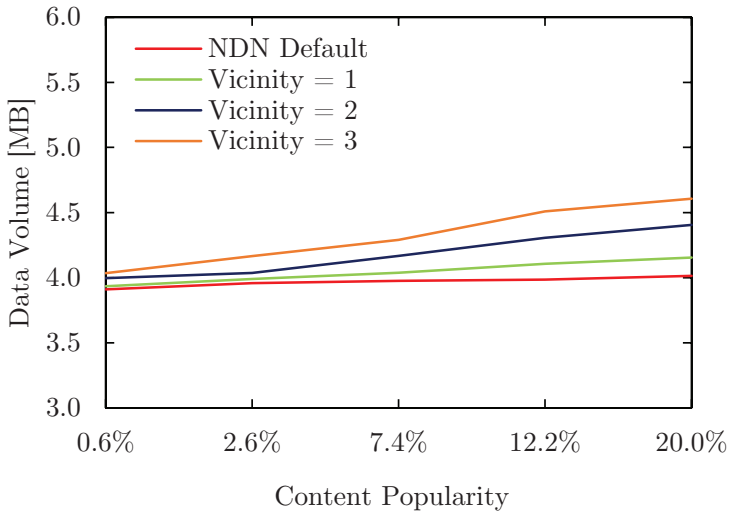
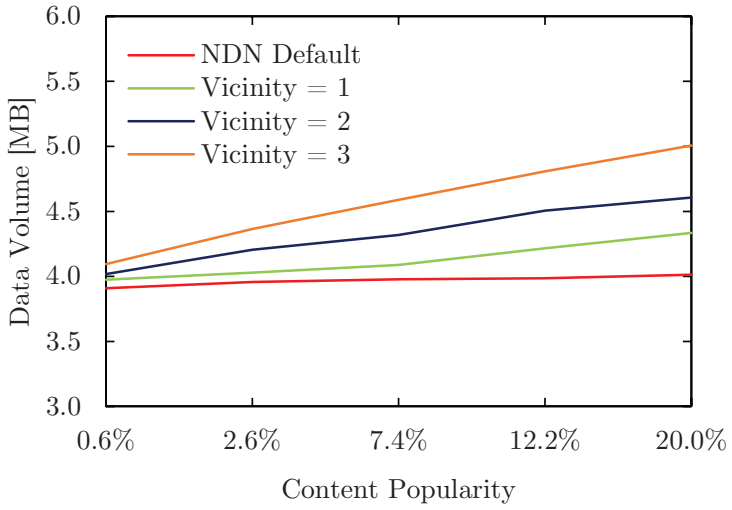


Figure 5.11: Data Volume in the Less Populated Network



(a) Cache Size = 10 Objects



(b) Cache Size = 1000 Objects

Figure 5.12: Data Volume in the More Populated Network

quite small, the differences are still negligible. In the large cache size (Figure 5.12 (b)), the results slightly increase compared to the smaller cache size (Figure 5.12 (a)) because the larger cache size means the lager sizes of the lists to be pushed. Interestingly, the overall increments are still slight (less than 1MB in the worst case of the 3-hop vicinity compared to default NDN). This has indicated that the small

packet sizes of the lists do not introduce a huge impact to the entire network traffic and space consumption in each node.

5.3.4.4 Summary and Discussion

This evaluation revealed the effectiveness and the efficiency of VCoF by considering the impact of content popularity based on the actual data-set of content access. We assumed that the replicas of desired content are requested to be stored in the caches based on the percentage of each level of content popularity. The less populated network and the more populated were used to examine the different amounts of traffic passing through the caches. The large cache size of 1000 objects is to ensure that each node's CS caches the desired content objects for longer, causing more content availability. In contrast, we also examined the case of the cache size of 10 objects to understand if there are some cache replacements due to the small cache size. The results have again demonstrated that VCoF can provide effective delivery efficiency over the baseline of standard NDN.

According to the RTT results, the values decrease dependent on the increment of the content popularity levels. It means when the replicas of desired content are distributed in a higher number of nodes, this increases the popularity of the desired content, which increases the opportunities that the desired content object or its replicas can be found especially in nearby nodes. VCoF can help to locate these nearby replicas effectively, improving the delivery delays compared to the baseline, even in the less populated network. For example, as shown in Table 5.4, the scheme presents 26.02% of improvement in the case of the 37% content popularity, even using the narrowest vicinity size of one.

			Content Popularity		
			Low (1%)	Medium[9] (11%)	High (37%)
Improvement over Baseline (%)	Vicinity Size	1	11.25	23.99	26.02
		2	15.02	26.91	29.41
		3	18.08	29.95	31.46
Increased Average Message Overhead per Node over Baseline (packets)		1	0.1	1.8	5.5
		2	0.6	9.0	33.1
		3	3.1	32.0	83.2
Increased Overall Data Volume over Baseline (MB)		1	0.155	0.231	0.291
		2	0.241	0.541	0.723
		3	0.352	0.853	1.013

Table 5.4: Results Summary of the Less Populated Network

			Content Popularity		
			Low (0.6%)	Medium (7.4%)	High (20.0%)
Improvement over Baseline (%)	Vicinity Size	1	15.49	29.39	18.93
		2	19.23	32.46	21.30
		3	21.80	35.53	23.61
Increased Average Message Overhead per Node over Baseline (packets)		1	1.0	5.0	12.3
		2	3.8	15.7	65.1
		3	9.9	57.0	242.7
Increased Overall Data Volume over Baseline (MB)		1	0.025	0.060	0.141
		2	0.087	0.122	0.392
		3	0.125	0.313	0.593

(a) Cache Size = 10 objects

			Content Popularity		
			Low (0.6%)	Medium (7.4%)	High (20.0%)
Improvement over Baseline (%)	Vicinity Size	1	16.09	25.93	3.02
		2	17.23	29.37	6.05
		3	19.85	32.82	6.05
Increased Average Message Overhead per Node over Baseline (packets)		1	1.0	3.0	8.0
		2	2.7	11.7	55.9
		3	5.9	49.0	228.6
Increased Overall Data Volume over Baseline (MB)		1	0.065	0.110	0.321
		2	0.107	0.341	0.592
		3	0.185	0.921	0.993

(b) Cache Size = 1000 objects

Table 5.5: Results Summary of the More Populated Network

Interestingly, cache size does not play a major role in the less populated network since the number of replicas remain similar. In the more populated network, RTT results of the large cache size slightly reduce compared to the small cache size due to the more content availability as mentioned in Section 5.3.4.1. In this case, there are also slight reductions of the overhead results since the requesting packets travel shorter distances (considering the improved RTT). For example, as presented in Table 5.5 (a), the increased average message overhead result of 7.4% content popularity is approximate 5 packets in the case of the 1-hop vicinity while the result of the larger cache size in Table 5.5 (b) is around 3 packets. In considering the data volume, the results of the large cache size are higher than the small cache size according to the larger sizes of the *Content Lists*. Notably, the increments are still negligible and do not introduce a huge impact to the entire network. For instance, the increment of the data volume in the worst case of the 3-hop vicinity is only 0.993MB as indicated in

[9]Note, medium in this context does not mean the middle value but it would mean one of the values between the low and the high.

Table 5.5 (b).

The overhead costs increase dependent on the content popularity levels. In particular, the higher number of requests increase the more list pushing packets. In the 1-hop vicinity, the results slightly increase (due to the narrow scope) with the higher efficient content finding compared to default NDN. For example, in the medium content popularity, VCoF adds only 5 packets to the baseline of the average message overhead per node while efficiently providing better improvement (29.39%) as presented in Table 5.5 (a). The costs might still be worthwhile in the case of the 2-hop vicinity while providing the improved RTT results with acceptable overhead. However, the costs could be likely high within the 3-hop vicinity because of the larger scope of the list pushing (higher number of hops) and the unworthy trade-off of the gained performance. For example, considering every content popularity level, the large vicinity size of three offers less than 7% of the maximum improvement compared to the narrowest vicinity size of one as shown in Tables 5.4 and 5.5 while the highest difference of the message overhead can reach up to ≈230.4 packets.

The VCoF scheme has again demonstrated that nearby replicas in every consumer's vicinity or nearby vicinities can indeed be the potential source for improving content delivery. Most of the results ensure the higher delivery efficiency over the baseline. In the next evaluation, we will explore that how replicas that are frequently forwarded to the vicinity of a mobile node can also be a suitable source for leveraging delivery performance in the mobile case.

Evaluation Limitations: In this evaluation, we again assume that the topology could be one of potential NDN topologies in the Future. This is because every node in the topology can perform all of the NDN primitives. In the existing architecture of the Internet, routers do not consume, serve or publish content. Hence, the topology used in this evaluation is not likely sensible if we consider current topologies based on the existing architecture. However, as discussed previously in Section 5.2.4.4, NDN could change the way of accessing content. Although core routers might not consume content, several devices in their network segments can consume and publish content. Hence, content generated by user's devices can be cached and served at the core routers due to the NDN primitives. This create replicas in the networks that VCoF can still provide the benefits.

Furthermore, considering edge networks, a single node in the evaluation topology can be a device (e.g., a smartphone, a laptop, a smart device, etc.). Nodes in the

topology do not always represent core routers and the evaluation topology could also represent an edge network (e.g., a home network). Due to the design of NDN, it can be flexible to be deployed in various kinds of networks and parts in a network system. VCoF can also provide its advantages in different networks. Nevertheless, a number of Future researches (e.g., caching techniques along with overhead cost optimisation as described in Section 3.2.2.2) can be further examined and developed specifically to be used in different network systems. In this evaluation, we aimed to examine that the nearby content finding performed by VCoF in the consideration of different content popularity levels can indeed improve the delivery efficiency compared to the baseline of standard NDN and this has been confirmed by the results obtained.

In considering the content popularity based on the dataset, we do not expect that the dataset can represent an actual NDN traffic since it is the dataset from a VoD IPTV-service based on P2P communication in the existing Internet architecture. However, the dataset can represent different levels of content popularity by considering different TV programs requested by a number of consumers selected from a specific section of the dataset. This dataset can indicate the actual behavior of content access. For example, we know the potential content distribution by considering different locations of consumers that perform their requests after applying the dataset into the evaluation topology. Hence, by applying NDN, the replicas of a desired TV program can be cached and distributed based on its popularity in the topology dependent on the realistic behavior of requesting content performed by the number of consumers. This is sufficient to first understand VCoF compared to standard NDN in the context of finding content by considering the actual content distribution provided by the dataset.

It is noted that in the Future where might have a real (i.e., native) dataset of NDN traffic, the results could be potentially similar or different depending on information in the dataset, the types of networks, and other related factors. Nevertheless, we still believe that the results proved in this work can indicate that nearby content finding can also provide the advantages in those network conditions. To ensure this, further improvement of the VCoF scheme itself and evaluations in different network conditions can be examined afterwards.

5.4 Handling Mobility

In Section 2.5.3, the issues regrading content delivery in the mobility case are discussed. These are mainly about the missing of desired content objects for a number of requests performed by a mobile node due to handover, causing a higher number of re-transmissions and delivery delays. In this section, we examine how VCoF handles consumer mobility considering these issues. In considering the communication model of this mobility case, the replica of a content item is usually cached in the previous NAR (i.e., the access station) of the mobile node. It is often located in the same vicinity of the current NAR. This is likely that VCoF can also help to find the replica to improve delivery efficiency. However, some of these NARs can also be located in a different Location Area (LA). In this case, the previous NAR could not be a suitable source. Based on the knowledge from the previous evaluations, the other replicas in the vicinity or nearby vicinities can be alternatively potential sources (depending on the content density). Hence, VCoF has the potential of being an appropriate solution to handle the issues linked to the mentioned mobility case.

To understand VCoF in a mobile NDN environment within the mentioned communication model, we need to scale up the evaluation topology since the mobile node needs to travel between NARs a longer distance. It means there should be a sufficiently large number of NARs that the mobile node can perform requests through different NARs. This means the locations of requesting content should be different. If there are a small number of NARs, it means the mobile node can travel only a short distance. Some requests might be performed in the same NAR and a desired content object can be fetched from the same cache. In fact, the mobile node can request the desired content object from various locations of NARs according to its movement.

So, we set the number of NDN nodes that are located in the infrastructure network along with the number of NARs, in which content objects can be cached in various locations. The challenges related to this to the VCoF scheme need to be investigated. Hence, the simulation topology is composed of 94 nodes of normal NDN nodes (or routers), 35 nodes of NARs, 14 LAs, and 265 links. The main infrastructure topology is generated based on considering the number of NARs and their locations. It is again noted that a number of evaluation limitations (including the simulation topology) are discussed in Section 5.4.4.3.

We assume that each LA should be connected to different parts of the infrastructure network depending on its location. This is also based on considering the

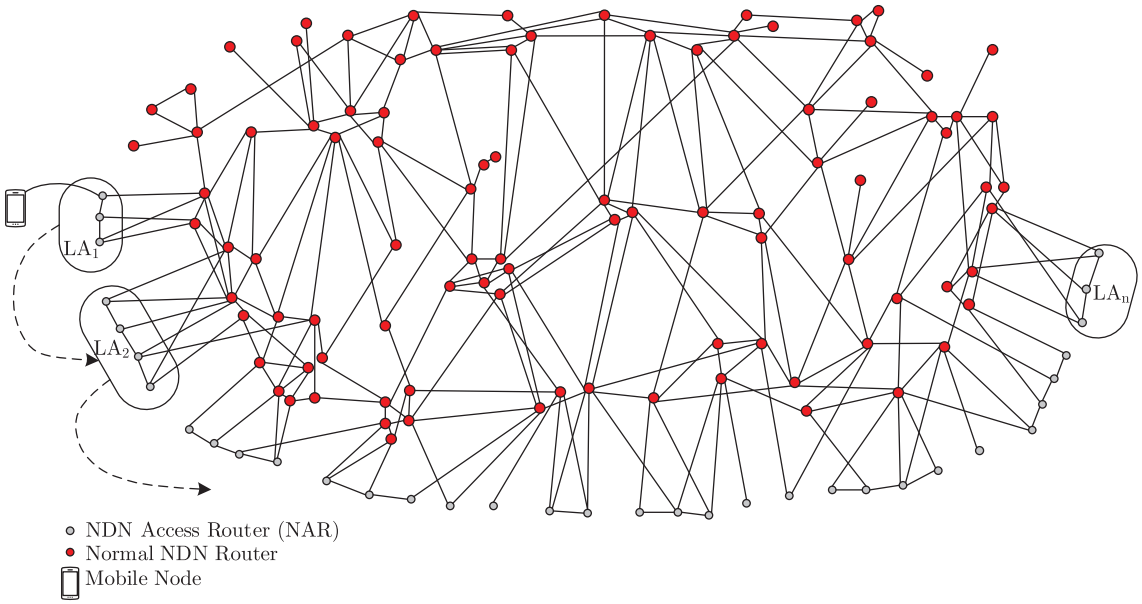


Figure 5.13: Mobile Topology Simulation

aforementioned communication model, which the mobile node moves from its current location to every next location. The opportunity that the mobile node performs every request from different locations would be more realistic in the considered mobile case. So, the NARs and LAs are randomly connected to different areas of the infrastructure network along the direction of moving the mobile node. As presented in Figure 5.13, the mobile node moves from the first NAR to the next NAR in the same LA (LA_1). It then moves to the next NAR of the next LA from the left hand-side to the right hand-side until finishing the communication with the last NAR of the last LA. It moves in a constant velocity of 25m/s. We again assume that each link has a homogeneous latency of 10ms.

5.4.1 Experimental Model

In this mobility evaluation, we expect that replica density is still the important factor since the NARs are statically connected to the infrastructure network. The replicas can be distributed in several locations in the topology. So, we investigate different replica densities at 5%, 25%, 50%, 75%, and 90%, respectively. This can cover low-to-high replica density cases, which can affect different content finding results. To fill up the caches, we request an origin content object to be randomly distributed

in different requesters in the infrastructure network by following the percentage of each replica density. For example, in the case of 50% density, the total number of nodes that can cache the object are 129 nodes (including normal NDN nodes and NAR nodes) and in this case, there should be ≈65 nodes that cache the replica of the desired content. However, if we let all of intermediate on-path nodes store the object, we can not control any replica density based on the percentage that needs to be evaluated. So, we assume that the replica is distributed throughout the topology based on the locations of the requesters[10], i.e., only at the requesters under a defined percentage.

Further to the experiments of the case of fixed caches, we also evaluate the case of non-fixed caches. In the first case, we assume that an interest packet tries to find its corresponding data under the network with different pre-cached replica densities. This is to understand how well every interest packet finds the desired content with the different defined percentages of replica densities. However, in real NDN networks, the content can be cached at any nodes in the Breadcrumb trails. This can cause the increment of a replica density especially in the areas closer to the mobile node due to the directions of interest packets. Hence, under the same setup of the first case, in the second case, we do not fix the caches.

After filling up the caches, the mobile node runs 30 requests from different NARs according to its movement. It is noted that each normal requester in the infrastructure network has to push its *Content List* to its vicinity due to the VCoF concept. This creates some additional overhead. To clarify this, we consider only the interest packets involved in content finding. The packets include the interest packets for requesting a content object to distribute the content depending on each replica density, the interest packets for requesting the content performed by the mobile node, and the interest packets for pushing the *Content Lists*. Only two packet types consisting of the interest packets for filling up the caches (i.e., distributing the content) and for requesting the content (performed by the mobile node) are considered as the overhead of default NDN.

According to the results presented in Section 5.3, we have found that a large cache size can remain content objects for longer and thereby the availability of the desired content makes it easier for finding them compared to a small cache size. This is because each content object can be replaced easier in the smaller cache size due to the limited caching capacity. In this evaluation, we do not consider the impact

[10]Note, the mobile node is not one of these requesters.

of cache size. We focus on finding replicas that are distributed in several caches depending on their density and are assumedly available to be located. There is no replacement from other objects and this is a note of this evaluation that we need to focus only on investigating content finding within the availability of the replicas based on their different percentages of densities. Additionally, the impact of some cache replacements due to the limited cache size has been already examined in the previous evaluation.

5.4.2 Vicinity and Content Placement

We assume that the replicas of a desired (i.e., finding) content object are randomly distributed in the nodes of the main infrastructure network (depending on a defined parameter (i.e., percentage) of the replica density). When a node receives the finding content object, it replicates the object into its CS and it then pushes its *Content List* to other nodes in its vicinity. If there is an interest packet generated by the mobile node that has been forwarded to a node (probably in the same vicinity or a nearby vicinity) to find the content in the node's vicinity, the content will then be fetched at the owner of the pushed list due to the *Content Finding* module. The content object is then forwarded to the mobile node considering the two cases of the fixed caches and non-fixed caches. In the first case, any intermediate nodes in the infrastructure network do not cache the content to prevent the increasing of a replica density. In the second case, the intermediate nodes are allowed to cache the content.

After the content object has been requested to fill up the caches for distributing the content, each node that is the requester and is caching the replica of the object has to push its *Content List* to advertise the content availability in its CS. This process can introduce additional overhead. Due to the previous evaluations, we have found that the vicinity size of the list pushing should be one because it has shown to have the most favorable tradeoff between the delivery efficiency and overhead costs. In specific cases, the vicinity size can be expanded to two but this comes at the costs of higher overhead. A larger vicinity can increase the opportunities of locating replicas. However, the overhead costs are likely to be excessively high in the case of the large vicinity size (like 3-hop vicinity) while considering the slight improvement over the case of the narrowest vicinity size of one. So, in this evaluation, we still control the vicinity sizes from one to three because we can evaluate the impact of the smallest vicinity size to the large vicinity size that is sufficiently large to understand excessively

Topology	94 nodes (in the infrastructure), 35 NARs, 14 LAs, 256 links
Link latency	10 ms
Mobile node's velocity	25 m/s
Handover delay	50 ms
Coverage area of the access station	100 meters
Number of requests to get the avg. RTT (randomly send before every handover)	30 requests
Replica density (%)	[5,25,50,75,90]

Table 5.6: Summary of Mobile Simulation Parameters

high overhead costs.

During a connectivity with an NAR, the mobile node randomly sends a request to find the content prior to every handover (at any time in the period of a hundred milliseconds before handover). This is to understand the higher possibility of re-transmissions if a number of requests cannot be delivered before handover. We assume that the default handover delay is set to 50ms. So, if there is a re-transmission, the RTT will include this hand-off delay with the delay of finding the content which can be different in each request. In each experiment, the mobile node generates 30 requests to indicate the performance of VCoF alongside the consideration of different metrics (described further in the next section).

When an NAR receives the content object, it then replicates the object in its CS. To understand the benefits of VCoF in the mobile topology, we also allow each NAR to cache the content object and to push its *Content List* to its vicinity (i.e., VCoF deployed). This might slightly increase a replica density from its base percentage but we can gain more understandings of the VCoF scheme particularly in this mobile environment from the deployment of the scheme at each NAR. This is because nearby NARs in a vicinity can also be a potential source especially every previous NAR. Based on the evaluation setups, the summary of the simulation parameters of this final evaluation is presented in Table 5.6

5.4.3 Metrics

In this evaluation, we employ four metrics i.e., (i) the average RTT per request performed by the mobile node, (ii) re-transmission percentages, (iii) replica hit rates, and (iv) message overhead. We again compare VCoF to the baseline of the default

NDN strategy. The details of each metric are itemised as follows.

- RTT: the average RTT per request performed by the mobile node to retrieve the desired content. This is to quantify how much VCoF can improve delivery efficiency. A lower average RTT demonstrates that the mobile node can find the desired content faster in comparison to default NDN.

- Re-transmission Percentages: The mobile node re-transmits a request again, if it has missed the desired content due to handover. VCoF proactively provides routing information of nearby replicas. So, we expect that the number of re-transmission percentages can be reduced especially in the case of a higher replica density. This metric is to understand how VCoF can improve the re-transmission percentages compared to default NDN. A lower percentage means the decrement of the number of re-transmission packets. This can relate to the improvement of delivery efficiency since a small percentage means the content can be found faster through the current NAR (i.e., before handover) without re-transmitting a new request through a new NAR.

- Replica Hit Rates: According to all requests performed by the mobile node, each replica is often cached in each previous NAR that is frequently located in the same vicinity of the current NAR of the mobile node. The other replicas can also be cached in area of the NAR of the mobile node depending on the replica density. So, this is to measure how often the replicas can be found. A higher replica hit rate demonstrates that the replicas can be located frequently rather than the original content object at its producer. This can also indicate higher cache utilisation.

- Message Overhead: the number of all interest packets related to content finding as described in Section 5.4.1. The interest messages are counted after the end of each simulation. An increment of message overhead results means a higher overall spending costs in the context of content finding. Nevertheless, although VCoF creates some additional overhead, it is important to consider that the overhead costs are acceptable or not in the consideration of the achieved performance gains of content finding. Most benefits of several solutions are not for free. In these experiments, we have also to trade off the overhead costs to the delivery efficiency.

5.4.4 Results

This evaluation examines the content finding results by still focusing on delivery efficiency alongside additional overhead specifically from the perspective of the consumer mobility under the discussed communication model. As discussed previously, we consider two cases of the fixed caches and non-fixed caches. The following results demonstrate the aforementioned metrics regarding both cases. This can help to understand how well VCoF finds and deliveries the content compared to default NDN in the mobile NDN environment by considering the different cases of content distributions. It is again noted that the results are the numbers obtained from the experiments performed in this work and different environments or network conditions can differentiate the results.

5.4.4.1 Fixed Caches

In the case of fixed caches[11], the following results demonstrate how well every single request finds the content, if the network has been populated with each initialised replica density. Replicas can be the suitable source for the mobile node. However, this might depend on the replica density and the replica's distributed locations.

Round Trip Time (RTT)

In this analysis, we aim to quantify how well VCoF improves delivery efficiency in the mobile network. The results (with error bars that represent 95% confidence interval) are presented in Figure 5.14. By considering the results, even in the cases of the low replica densities (e.g., 5% or 25%), VCoF can help to reduce the delivery delays. For example, in the case of 5% replica density, default NDN can perform around 100.24 ms while VCoF can decrease the average RTT to approximate 78.50 ms (the 1-hop vicinity case). However, the RTT results are still higher than the cases of the higher replica densities because of a lower number of the replicas in each vicinity, and a previous NAR in a different LA.

Furthermore, when the mobile node requests the content before handover, although the object can be found at the previous NAR, it might not be delivered to the mobile node since the node might have moved to a new NAR already. The mobile node needs to still re-issue an interest packet to find the content again, causing

[11]Note, the definition of the case of fixed caches and the case of non-fixed caches is already defined in Section 5.4.1, which is also used to describe the results of this final evaluation.

some delays. Nevertheless, VCoF can still offer the better advantage of content finding due to the designed scope of the vicinity-based concept (considering the reductions of the RTT results).

When we increase the replica densities to medium (50%) or high (75%) or even highest (90%), the RTT results are improved significantly. These are almost on a similar level ($\approx$ 35 ms to 43 ms) according to the small number of hop(s) to find the replicas. Since the increment of the nearby replicas in the vicinity of each of the current NAR of the mobile node, and in other nearby vicinities adds more content availability and more list pushing packets (to proactively provide routing information for the nearby replicas). So, the content can be found faster. Furthermore, the mobile node does not re-transmit a request again because a path for finding a nearby replica is proactively advertised. An NAR can instantly forward an interest packet to fetch the content through the path, resulting in the faster content delivery. Additionally, the gaps between VCoF and default NDN are reduced due to the higher number of replicas. For example, in the 90% replica density case, the desired content can be located easily, even in the default paths due to the higher availability of the replicas.

When we increase the vicinity sizes, there are slight improvements compared to the smallest size of the 1-hop vicinity. A larger vicinity size might increase replica hit rates but it also increases the number of hops to locate the replicas due to the larger scope. Nevertheless, the 1-hop vicinity has still shown the favorable performance

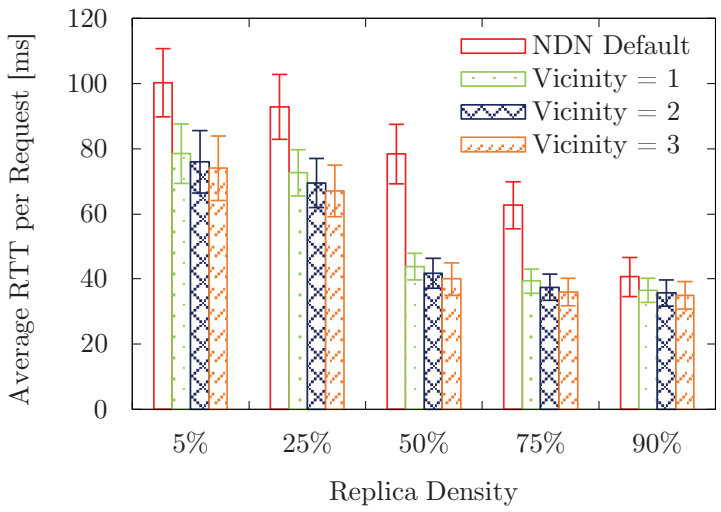


Figure 5.14: Average RTT per Request in the Case of Fixed Caches

gains. Interestingly, as mentioned earlier, several replicas are forwarded to previous NARs and VCoF can help to find these replicas faster. However, when a previous NAR is located in a different LA of a current NAR, finding a replica in the previous NAR can not effectively reduce the RTT. This is because there is a significant jump between the previous NAR and the current one. However, other nodes in a vicinity and other nearby vicinities are still the alternate potential sources for the mobile node, proved by the reductions of the RTT results.

By considering the RTT results presented previously in Section 5.2 and 5.3 with the RTT results in this analysis, the results in the previous evaluations decrease dependent on the higher number of content popularity/density percentages. However, the results in this evaluation have demonstrated that the RTT values are reduced to the almost similar level, when the replica densities are high enough (e.g., medium to high densities). This is because several replicas are forwarded to the area of each NAR and cached closer to the mobile node. So, it is easier to locate these replicas using VCoF. This is unlike the previous experiments that the content objects can be freely distributed to any areas due to the experimental setups and the real dataset of content access in the static networks. This decreases the RTT results gradually along with the increment of the replicas.

Re-transmission Percentages

This analysis examines how VCoF handles re-transmission percentages. According to the results presented in Figure 5.15[12], when a replica density is low (e.g., 5%), the maximum percentage can reach up to 70% by using default NDN. This is because the desired content object or its replica can not be delivered through each of the current NAR of the mobile node before handover. So, the mobile node re-transmits a number requests again, increasing the percentage. Even though VCoF can reduce the re-transmission percentages since the opportunity to locate a replica in a vicinity is higher, the results are still higher than the cases of the higher replica densities according to the small number of the nearby replicas. There are also re-transmission packets for retrieving the replicas from previous NARs and every current vicinity of the mobile node, and also nearby vicinities.

When we increase the replica densities (e.g., more than 50%), VCoF can improve the percentages efficiently. For example, in the case of 50% replica density, 46.6%

[12]Note, no cofidence intervals are given since each experiment contains 30 requests per replica density but no repetition of the experiment for every replica density. This is also considered in the metric of replica hit rates.

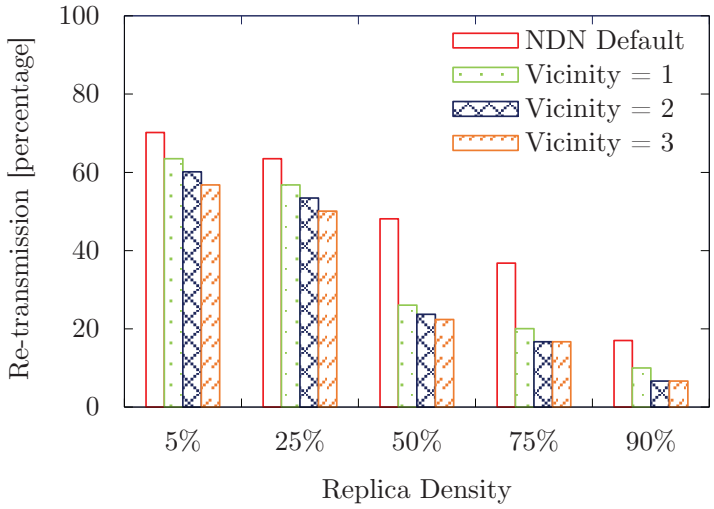


Figure 5.15: Re-transmission Percentage of Requests in the Case of Fixed Caches

is the re-transmission percentage of default NDN while VCoF introduces only 23.3% within the 1-hop vicinity. According to the higher number of the replica densities along with the increment of the list pushing packets, VCoF can proactively indicate a number of faster routes to find nearby replicas in every vicinity or nearby vicinity. This increases the higher possibility that the content can be served through each of the current NAR of the mobile node, reducing the re-transmission percentages. It is noted that although the results in the medium density case are higher than the higher density cases due to the lower number of proactively routing information advertised, the re-transmission packets still travel short distances. Thereby, the RTT results with some additional handover delays are also reduced.

The results quite relate to the RTT results. For example, when a replica can be located nearer, i.e., fewer hops away, it can also decrease the RTT value and re-transmission percentage. In some cases of the high replica densities, when the vicinity sizes are increased, some percentages could be probably the same because the replicas can be found in just a small number of hop(s). A too large vicinity might be unnecessary.

Replica Hit Rates

The goal of this analysis is to understand how much the replica hit rates can be increased. The results are presented in Figure 5.16. We have found that even in the

cases of low replica densities (e.g., 5% or 25%), the replica hit rates are significantly increased compared to default NDN. This is because several replicas can be hit in each previous NAR. However, the results are still low compared to the higher replica densities due to the lower number of the replicas in each vicinity. Furthermore, when there are significant jumps between NARs, which means they are located in different LAs, VCoF might not provide a path to locate a replica at a previous NAR. This cannot increase the hit rates. In a very specific case, when the mobile node instantly sends a request (e.g., a re-transmission packet) after it has moved to a new NAR before this NAR receives a pushed *Content List* due to some delays, VCoF may not indicate the request to find a corresponding replica at a proper source. This can result in lower replica hit rates.

When the replica densities are sufficiently high (e.g., 50% or 75%), we have found that the hit rates are improved significantly. For example, in the case of the 50% replica density within the 1-hop vicinity, 40% has been improved in comparison to default NDN. This is because the designed scope of content finding can increase the replica hit rates especially from each previous NAR and at other nodes in each vicinity. In the case of the highest replica density of 90%, the gaps between default NDN and VCoF within different vicinity sizes are reduced since the replicas can be found easily, even in the default paths. In addition, there are some increments in each replica density, when the vicinity sizes are increased because a larger vicinity can cover a larger

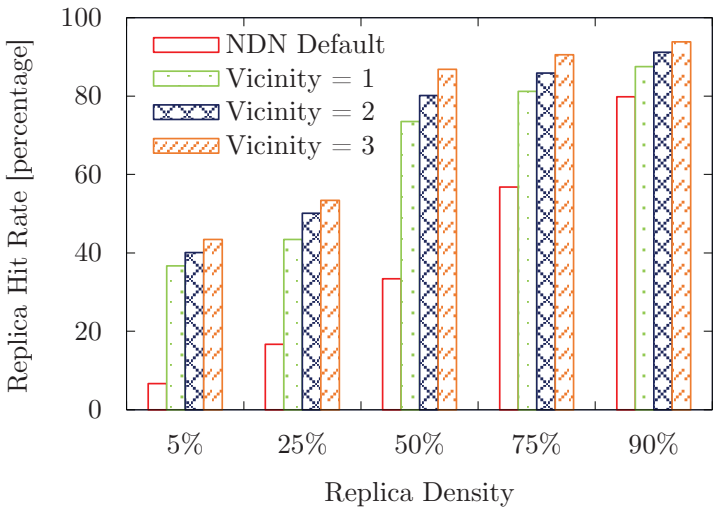


Figure 5.16: Replica Hit Rate of Requests in the Case of Fixed Caches

scope of content finding. So, the replicas hit rates can also be increased. Another finding is that the increment of the replica hit rates can also reduce the RTT results. Notably, by using default NDN, the replicas are mostly found further away in the default paths according to the higher RTT results compared to VCoF.

Interestingly, even though the increment of vicinity sizes can increase replica hit rates, the RTT results might not be significantly improved. This is because when a replica is found in a larger scope of a vicinity, it might take a higher number of hops to reach the node storing the replica. So, the larger vicinity sizes can not improve the RTT results significantly in this type of networks, where the replicas can be found in a small number of hop(s). 1-hop vicinity can be sufficient to improve the RTT results. A larger vicinity size might help to slightly reduce the delivery delays but the higher overhead costs should be considered as the tradeoff. This is discussed in the next metric.

Message Overhead

This metric examines the additional overhead costs performed by VCoF in comparison to default NDN. The results are shown in Figure 5.17. When we increase the replica densities, the overhead results are also increased due to the higher number of requests. Even though VCoF can find the content faster and the interest packets generated by the mobile node travel a shorter number of hop(s), the list pushing packets still provide the dominant effect on the increment of the message overhead related to the higher number of requests (i.e., content distribution) in the topology.

Nevertheless, by comparing default NDN to the 1-hop vicinity cases, the overhead costs performed by VCoF are slightly higher because the pushed packets travel the narrow scope (i.e., 1 hop). The costs can still be acceptable because of the improved results of delivery efficiency. In the 2-hop vicinity, the results are notably higher since the larger scope of content finding covers more hops. The *Content Lists* have to be pushed in larger areas. The costs might not be worth while considering the small improvement of the delivery performance compared to the smaller vicinity (1-hop vicinity). In the 3-hop vicinity, the overhead costs are likely to be dramatically higher, outweighing the benefits.

By considering the overhead costs in this analysis alongside the RTT results obtained as shown in Figure 5.14, in this kind of mobile topologies, we suggest that the good trade-off for a vicinity size should be one since the larger vicinity sizes slightly improve the delivery delays. However, these larger vicinity sizes generate remarkably

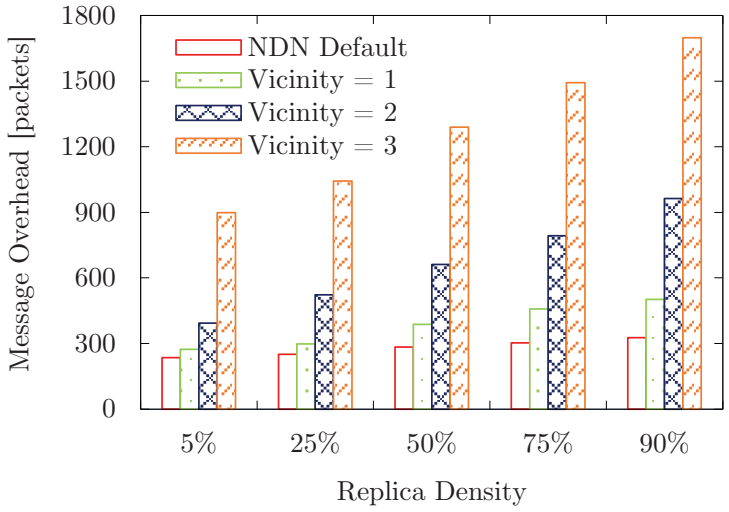


Figure 5.17: Message Overhead of Content Finding in the Case of Fixed Caches

overhead costs, outweighing the benefits. The movement pattern in the topology and the caching behavior as discussed previously allow the desired content to be cached in the closer areas of the mobile node, providing the advantage for VCoF. 1-hop vicinity would be sufficient to find the nearby replicas of the desired content object to improve content finding in this kind of mobile networks.

5.4.4.2 Non-fixed Caches

According to the requests performed by the mobile node, content replicas can be cached closer especially in each of the current vicinity of the mobile node. Both VCoF and default NDN can take the advantage of these replicas, particularly when the caches are not fixed[13]. In this analysis, the main findings are discussed and also compared to the previous case of fixed caches.

Round Trip Time (RTT)

In considering the RTT results presented in Figure 5.18 with error bars that represent 95% confidence interval, the results develop in the same pattern of the case of fixed caches. However, VCoF can provide the higher advantage over the previous case particularly in the low replica densities. The gaps between VCoF and default NDN are reduced when the replica densities are medium to high.

[13]Note, the definition of this case of non-fixed caches is already defined in Section 5.4.1.

In the low replica densities of 5% and 25%, default NDN gets the higher advantage over the case of fixed caches since the replicas are forwarded to each of the area of the NARs, increasing the content density closer to the mobile node. This means the opportunity of a node that has forwarded (and cached) a replica to the mobile node through a previous NAR will be the node that can serve the replica in the next request is higher. This is because in this default path, the replica can be cached in several nodes (especially in the area of a current NAR). This is the reason of the reduction of the RTT values performed by default NDN compared to the previous case (e.g., from 100.24ms to 91.21ms in the 5% replica density).

Nevertheless, VCoF provides the higher advantage over the case of fixed caches because several recently cached replicas can be hit in nearby nodes due to the routing information proactively provided by the *Content List* advertisements. A node that recently caches a replica can be a potential source after a next request is forwarded to another node in the same vicinity. Thereby, this can effectively improve the delivery delays. For example, in the 25% replica density and 1-hop vicinity, the percentage of improvement over the baseline is 35.37% while 21.76% is the result of the case of fixed caches.

When the replica density is medium or higher (e.g., 50% or 75%), the gaps between VCoF and default NDN are reduced compared to the case of fixed caches. This is because the increment of the replica availability can also improve the RTT values

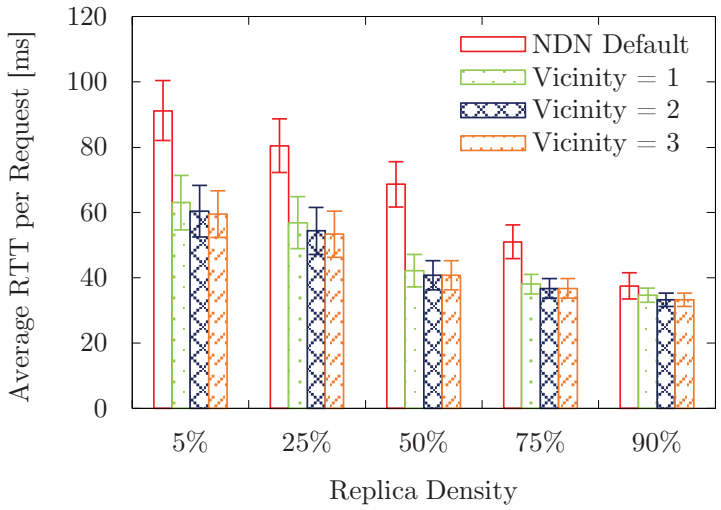


Figure 5.18: Average RTT per Request in the Case of Non-fixed Caches

of default NDN. However, the values performed by VCoF improve to almost the minimum delays (around 40ms to 34ms) of all of the experimental scenarios of this final evaluation in this work. The increment of the replica availability (when the caches are not fixed) can slightly reduce the RTT results but it does not provide a significant difference since the level of the RTT values is already close to the base level of the best case (e.g., considering the 90% replica density).

In considering the high replica density of 90%, the results again confirm that the slight increment of the replica availability can marginally reduce the RTT values (compared to the case of the fixed caches) but this is not a significant difference. This is because the replicas are already distributed to almost every node.

Re-transmission Percentages

The results have again demonstrated that VCoF can still provide the better advantage in reducing re-transmission packets over default NDN. The percentages reduce when the replica densities increase. However, by comparing the results to the case of fixed caches, every re-transmission percentage of each replica density is significantly reduced even using default NDN as shown in Figure 5.19. This is because of the increment of the availability of nearby replicas. The opportunities of locating the desired content before handover are higher.

In the case of low replica density (e.g., 5%), the percentage of default NDN is

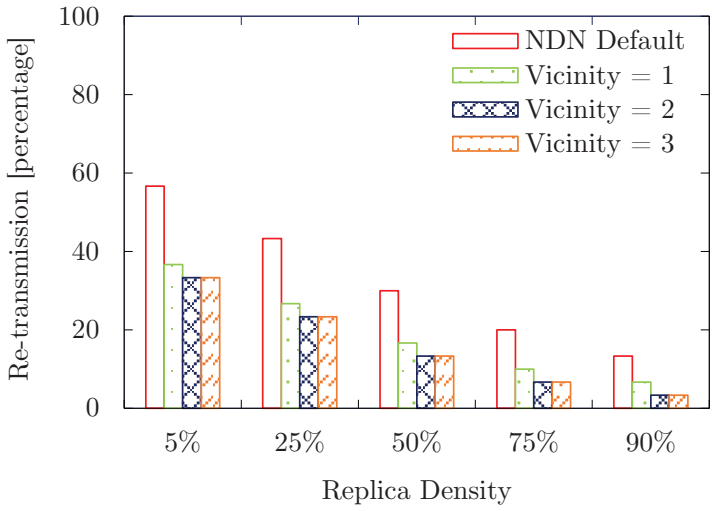


Figure 5.19: Re-transmission Percentage of Requests in the Case of Non-fixed Caches

56.67% while VCoF (1-hop vicinity) is 36.67%. In this case, the gap between VCoF and default NDN is higher (20%) compared to the previous case (6.7%). This is because every previous NAR and the increment of the replicas in nearby nodes provide the higher opportunities to indicate faster paths for retrieving the content. Hence, the possibilities that the content can be delivered before handover are higher, causing the reduction of the re-transmission packets and also the RTT results. This is similar to the case of the low replica density of 25%.

In considering the cases of higher replica densities including 50%, 75%, and 90% respectively, both default NDN and VCoF can also reduce the percentages compared to the previous case of fixed caches. However, the gaps between VCoF and default NDN are likely to be reduced since default NDN can also get the advantage from the increment of nearby replicas. For example, 13.3% is the improvement over the baseline while in the same condition, 23.3% is the improvement of the previous case (considering the condition of 50% replica density with the 1-hop vicinity). Nevertheless, VCoF can still provide the higher re-transmission reductions over the baseline. Interestingly, by considering the RTT results, although default NDN can reduce the re-transmission percentages, the content objects are significantly located further away due to the higher delays compared to VCoF.

Replica Hit Rates

As presented in Figure 5.20, the incremental results are significantly different from the previous case of fixed caches. The overall hit rates are dramatically increased especially in the cases of low replica densities. This is because every single request towards the original producer can create a number of replicas that are cached in the network especially around the producer. The opportunities that the next requests will be hit these replicas are reasonably higher.

In considering the low replica densities, the hit rates are higher than the case of fixed caches even using default NDN. For example, in 5% replica density, the percentage performed by default NDN can reach to 73.48% while the previous case is only 6.68%. This is because the surrounded replicas can be hit before reaching the origin producer. The results performed by VCoF are slightly better than default NDN since the opportunities that VCoF can indicate paths to nearby replicas are higher. The values are significantly higher (almost to 100%) particularly in the highly replica densities. There are some gaps before reaching the maximum percentage because a very slight number of requests might be hit directly at the origin producer.

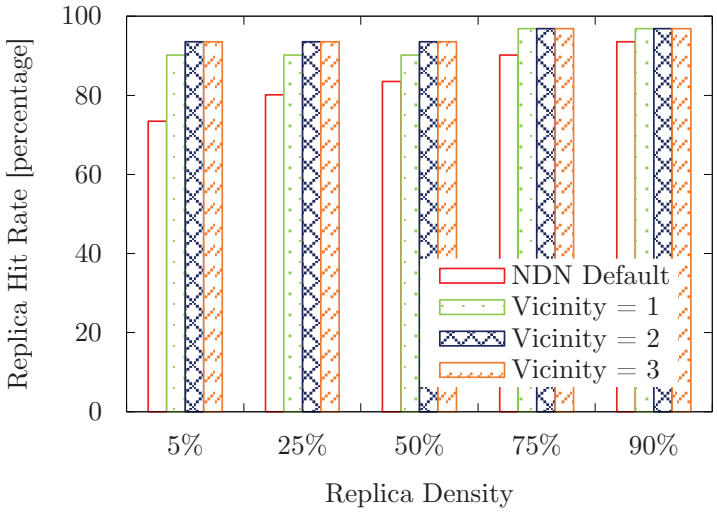


Figure 5.20: Replica Hit Rate of Requests in the Case of Non-fixed Caches

Message Overhead

The results of message overhead again develop in the same pattern to the previous case of fixed caches as presented in Figure 5.21. However, the overall values of both VCoF and default NDN are slightly lower than the previous case. This is because the requesting packets travel shorter distances even using default NDN. Although VCoF has to push a higher number of *Content List* packets when replicas are newly stored at new nodes, many of them are cached in the previous NARs. According to the topology, a small number of pushed packets is generated. Some of these packets are pushed in a vicinity or nearby vicinities, which means in a limited scope and does not introduce a higher number of message overhead over the previous case.

According to the results, the VCoF scheme has again confirmed that 1-hop vicinity can be the good trade-off of the vicinity size for this kind of communication model. For example, considering the medium replica density of 50%, the scheme has introduced only 110 additional packets while the improvement of the RTT reductions can reach 38.55%. There are slight improvement in the cases of larger vicinities (2 and 3 hops vicinities). However, the costs might not be worth for the slight improvement over the 1-hop vicinity. For instance, in the highest replica density of 90%, the maximum cost can reach to 1,594 packets but only 3.61% is improved over the the 1-hop vicinity that introduces only 464 packets.

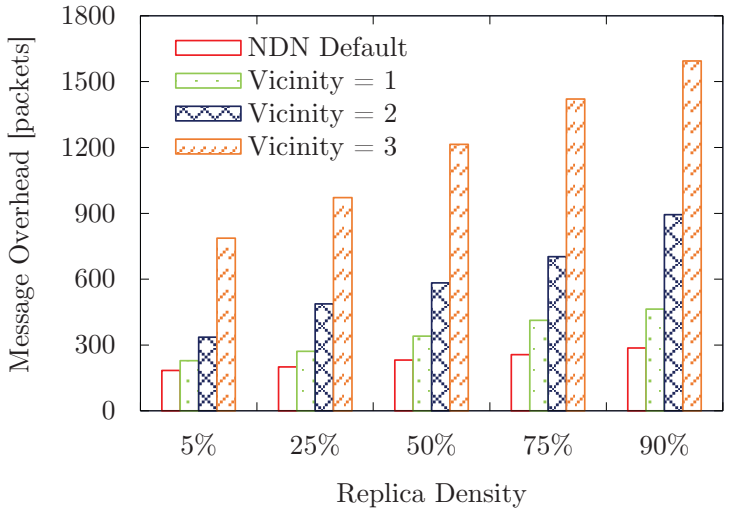


Figure 5.21: Message Overhead of Content Finding in the Case of Non-fixed Caches

5.4.4.3 Summary and Discussion

In this evaluation, we examined VCoF in comparison to default NDN in terms of handling consumer mobility in a particular mobile NDN network. According to the average RTT results, we have found that VCoF can improve delivery efficiency effectively, especially when the replica densities are ranged from medium to high (50% to 90%). This is because the number of nearby replicas are sufficiently high to be located easily. Furthermore, due to the request pattern performed by the mobile node, several replicas are forwarded to each of the previous NAR, which is frequently located in the same vicinity of the mobile node. In addition, in a higher density of the replicas, the opportunities that the replicas are cached in a closer area of each NAR of the mobile node are also higher, causing faster content discovery.

Two main experiments have been done. In the first experiment, we considered the case of fixed caches to understand how well every request can find content if the network is pre-populated with different initialised replica densities. In the second experiment, the caches are not fixed to gain more understanding about the impact of natural content caching of NDN specifically in the mobility context.

We summarised the obtained results in Table 5.7. According to the results, for instance, 44.1% of improvement (the case of fixed caches) over the baseline of default NDN has indicated the better performance gained considering the 1-hop vicinity with the medium replica density of 50%. This has proved that VCoF can also fit well with

this mobile communication model. VCoF has provided the higher performance, even in the low replica density of 5% in the 1-hop vicinity case, which the improvement result is 21.68%. Interestingly, the improvements are higher in the low replica densities when the caches are not fixed (e.g., 30.85% of the 5% density). This is because the opportunities of nearby replicas cached are higher, which is a central benefit for the VCoF scheme. In the case of 1-hop vicinity with the high replica density of 90%, there are slight improvements. In these cases, the content can be found easier, even in the default paths. Nevertheless, VCoF has again demonstrated the higher delivery performance over the baseline due to the higher utilisation of nearby replicas.

VCoF also helps to reduce the re-transmission percentages because when the corresponding replica can be found faster, it can be delivered through the current NAR of the mobile node. The mobile node does not need to re-transmit a request again through a new NAR due to the missing content object after handover. So, the number of re-transmission packets can be decreased. In table 5.7 (a), 23.3% of the re-transmission reduction is incurred by the medium replica density (50%), even in the narrow vicinity size of one. However, in the case of non-fixed caches, the reduction is lower (13.3%) as shown in Table 5.7 (b) since default NDN can also get the benefit of the higher content availability. Nevertheless, considering the RTT results VCoF still outperforms the baseline. Interestingly, the percentages of re-transmission reduction are higher in the case of non-fixed caches and the low replica densities (e.g., at least 20% over 6.7%). This is because VCoF can indicate the better paths for the requesting packets due to the higher content availability in the nearby areas.

In the previous evaluations, replicas are distributed freely to any nodes and the RTT results decrease along with the increment of the replica density/popularity. Unlike those evaluations, the replicas are often cached in the areas of the NARs. Hence, this evaluation also focuses on the replica hit rates to gain more understanding of how much the replica hit rates impact on content finding results. By using VCoF, the desired content can be found easier, especially in a previous NAR, another node in a vicinity, or another nearby vicinity which are the better potential sources.

Particularly, when the replica density is high enough (e.g., more than 50%), the hit rate results are increased significantly. This is because several replicas can be found particularly in the area of each NAR. In considering the case of fixed caches, in the 1-hop vicinity with 50% replica density, 40% of the increased hit rate (presented in Table 5.7 (a)) ensures that replicas are more utilised compared to default NDN. In this density case, larger vicinity sizes offer the higher hit rates (e.g., 46.8% and 53.4%

			Replica Density		
			Low (5%)	Medium (50%)	High (90%)
Improvement over Baseline (%)	Vicinity Size	1	21.68	44.10	10.16
		2	24.17	46.70	12.18
		3	26.18	48.93	13.87
Re-transmission Reduction from Baseline (%)		1	6.7	23.3	10.0
		2	10.0	26.6	13.3
		3	13.3	30.0	13.3
Increased Replica Hit Rate over Baseline (%)		1	30.0	40.0	7.6
		2	33.4	46.8	11.3
		3	36.7	53.4	14.0
Increased Overall Message Overhead over Baseline (packets)		1	38	103	175
		2	158	378	636
		3	662	1006	1371

(a) Fixed Caches

			Replica Density		
			Low (5%)	Medium (50%)	High (90%)
Improvement over Baseline (%)	Vicinity Size	1	30.85	38.55	7.68
		2	33.73	40.52	11.29
		3	34.68	40.52	11.29
Re-transmission Reduction from Baseline (%)		1	20.0	13.3	6.6
		2	23.3	16.6	10.0
		3	23.3	16.6	10.0
Increased Replica Hit Rate over Baseline (%)		1	16.7	6.6	3.3
		2	20.0	10.0	3.3
		3	20.0	10.0	3.3
Increased Overall Message Overhead over Baseline (packets)		1	45	110	177
		2	152	353	607
		3	603	983	1307

(b) Non-fixed Caches

Table 5.7: Results Summary of Handling Mobility

of the 2 and 3 hop vicinity cases, respectively). The increased hit rates reduce along with the increment of replica densities since replicas can be hit easier in the default paths. In the case of non-fixed caches, the overall hit rates increase in both default NDN and VCoF which means the gaps between these two mechanisms also reduce (e.g., less than 10% in the medium replica density). This is because the opportunities that the replicas around the original producer will be hit again for every next request are significantly higher.

In considering the message overhead results, the *Content List* pushing packets

create the most dominant effect on the increment of the overhead. Focusing on the RTT results, the 1-hop vicinity can be sufficient to perform the significant improvement of delivery efficiency. A larger vicinity might help to increase the replica hit rates but an interest packet still needs to travel a higher number of hops due to the larger scope of the vicinity. This might help to slightly reduce the RTT values because some replicas can be located faster compared to finding them at their original producer. However, by expanding too large vicinities, the slight improvement of the RTT results might not be worth to spend higher overhead since a number of the content replicas can be significantly found in the smaller scope of each NAR's vicinity.

For example, in considering the case of fixed caches and comparing the smallest vicinity size of one to the larger vicinity size of three, less than 5% of improvement is noticed while the difference of the message overhead is significantly higher (e.g., 1,196 packets added considering the high replica density of 90%). In the case of non-fixed caches, the message overhead costs develop in the same pattern of the previous case. However, the overall costs marginally reduce compared to the fixed cache case. This is because the requesting packets travel shorter distances (caused by the higher content availability). Additionally, though the *Content List* packets create the most dominant impact on the overhead costs, these are also pushed in the limited scopes and many of them are from every previous NAR (in each of current vicinity) and some of them are from nearby vicinities.

Evaluation Limitations: By considering the mobile simulation topology, as discussed previously in Section 3.3, we focus on a particular type of wireless communications that VCoF can likely provide its benefits. To be more specific, one example of these mobile communication models could be V2I in VANETs. Imagine that when a vehicle (i.e., mobile node) is moving from a previous location (i.e., access station) to every next location such as a car in a highway, a tram or a train. What happens if a number of desired content are forwarded to the previous location but the mobile node has moved to its current location? There could be a number of missing objects caused by the delays of content finding. The content cannot also be delivered before handover causing re-transmissions. According to the results, VCoF can also provide the advantages in this kind of networks.

Nevertheless, it should be noted that the results could be different in other mobile communication models and network conditions. For example, some wireless networks that mobility of node is less frequent and handover rates could be low compared to

the topology used in this work such as Wireless Mesh Networks, a particular type of Wireless Sensor Networks, Wireless LANs, and etc. However, this does not mean that VCoF cannot provide its benefits but further investigation should be made and some optimisation techniques can also further be developed specifically for different kinds of networks as discussed previously.

Further to the limitation of the mobile evaluation topology, content distribution in the main infrastructure network is assumed based on different replica densities defined and this is to mimic the potential locations that the content could be cached in the network. However, the results might be different if the content items are distributed differently within different network conditions. Nevertheless, in the second (previous) evaluation, we already understood the potential content distribution based on the actual content access performed by the requests of consumers. From our observation, each replica density can also potentially reflect every content popularity level investigated in the previous evaluation. Note, when the caches are not fixed, the desired content can be naturally cached at any nodes dependent on the requests that are randomly generated by the mobile node. This can also realistically demonstrate how content caching alongside the operations of VCoF can enhance this mobile communication in the context of content finding.

5.5 Evaluation Summary

In this chapter, we evaluated VCoF considering a number of different aspects. The designed evaluations are to examine the particular context of content finding in NDN performed by VCoF compared to the baseline of standard NDN. This is particularly in respect to the design goals as summarised in Table 3.1. The evaluation results have indicated the efficiency of the VCoF scheme (i.e., how well the scheme finds content alongside the consideration of the efficient trade-offs to additional overheads) and the effectiveness of the scheme (i.e., how nearby replicas located by VCoF effectively improve content delivery). The three major evaluations are summarily discussed as follows.

For the VCoF scheme to be a practicable alternative to current content finding strategies of NDN, it must be applied in an actual running system of NDN. Hence, in the first evaluation (presented in Section 5.2), we examined how VCoF can enhance content finding in NDN by considering the system that drives the major software components of NDN. The implemented modification is added into the core system.

According to the evaluation results, VCoF has significantly improved content finding (in terms of delivery efficiency) compared to default NDN with considerable additional overheads.

The first evaluation was crucial to understand the effectiveness as well as the efficiency of content finding by considering improved delivery performance within the realistic NDN network. In this evaluation, we have observed a limitation of emulating a larger-scale network. This could be problematic, if we have a higher number of nodes. Nevertheless, the introduced topology is sufficient to understand how VCoF enhances content finding in a real system.

To further examine a larger network with an actual data-set of content access, the second evaluation (described in Section 5.3) was used. According to the results of the previous evaluation, we have found that a different number of content/replicas can affect different content finding results. This has led us to investigate the impact of content popularity. To highlight the impact of content popularity, we identify the different popularity distributions in a less populated network and in a more populated network. Through this evaluation, VCoF has shown again that it can indeed leverage delivery efficiency. This evaluation has guaranteed that consumers can gain the advantage of VCoF since the actual behaviors of accessing content were simulated.

This second evaluation was also used to demonstrate the effect of cache sizes with the understanding that existing content can be replaced by other objects, especially in a limited cache capacity, causing lower replica availability. This can reduce the opportunity that the replica can be found nearby. Nevertheless, VCoF has still demonstrated the better content finding performance in comparison to default NDN.

In our final evaluation (outlined in Section 5.4), we examined the benefits of VCoF in handling mobility. In particular, we looked to demonstrate how VCoF enhances content finding when a consumer node has missed its desired content due to handover within a well-known mobile communication model.

The experimentation has demonstrated that VCoF has provided considerable gains (e.g., up to 44.1% of improvement, even in the narrowest scope of the vicinity size of one), which delivery efficiency can be improved significantly. We have also shown the flexibility of deploying VCoF in the mobile environment due to its decentralised concept. This evaluation has also guaranteed that nearby replicas can be the better sources for content requesters. This has also confirmed the same direction of improvement as demonstrated in the previous evaluations in considering the trade-

offs between the performance of content finding and the overhead costs.

Together, these evaluations have demonstrated the favorable outcomes of the VCoF scheme. Not only does the scheme efficiently leverage content finding in NDN, it has also highlighted additional benefits. The most important of these is the flexibility of deployment and the effective cache utilisation, which takes the advantage of content naming and caching that is the important key concept of NDN.

Summary of Evaluation Limitations: In the three main evaluations, a number of limitations should be noted. We highlight some important limitations that could provide different results to this work, as discussed as follows. The first limitation is about the evaluation topologies. These might not represent practical topologies based on the existing architecture of the Internet. This is because according to the concept of NDN, content can be cached or served at any devices (e.g., routers, devices in edge networks), which is explicitly different from the current architecture. Hence, there could be further development possibly including some hardware customisation for NDN. This can potentially change the way of accessing content in the Future. So, we consider the topologies that can be the potential topologies of native NDN networks (no global scale native NDN networks and traffic at the time of this thesis that we can essentially examine the VCoF scheme) since all of the NDN primitives can be deployed in the topologies. In addition, potential content distribution is also explored in these topologies.

However, if actual NDN topologies are very similar to the topologies based on the existing Internet architecture, we still believe that finding nearby content in every vicinity can provide the benefits. To ensure this, once NDN can indeed be deployed to replacing the current architecture, further investigation must be explored. Some optimisation techniques specifically for different kinds of networks can also be examined afterwards.

The next limitation is about other related network conditions. Since the VCoF scheme proposed in this work is in its early stages, we have aimed to develop the scheme to first prove that nearby content in every vicinity can be the better source for content finding. Hence, we need to focus on a number of important factors that can likely impact content finding results the most first. However, some other network conditions could also provide some effects such as network congestion, content sizes, different link latency values, bandwidth, packet loss, etc. These should be further investigated.

Further to the challenge of the network conditions, the investigation of the impact of additional overhead costs of content finding in different networks would also be challenging. Although the results have indicated the favorable outcomes of the delivery efficiency with the good trade-offs against the additional overhead especially in small vicinity sizes, these are based on the essential evaluation setups with the main aim to prove that the concept of VCoF can indeed improve content finding in standard NDN. However, different networks or parts of networks could have different characteristics of content caching. For example, core routers could see a higher number of content updates compared to edge networks. Hence, further mechanisms to understand or even optimise the overhead costs must be examined to adjust the VCoF scheme specifically to be used within different networks such as discussed in Section 3.2.2.2.

Chapter 6

Conclusions

The Internet has become an important and useful technological tool impacting on many people's lives. As the patterns of using the Internet have changed over time, it has seen growing demands of content delivery. Hence, the ways of delivering content have also evolved in order to fulfill the current needs of users. However, the traditional Internet technologies have been encountering several limitations of the existing architecture. In particular, the traditional paradigm of the host-centric model is considered insufficient and hence a change to the information-centric network is proposed.

In this thesis, we particularly look into the latest novel content delivery approaches of Information Centric Networking (ICN). We consider Named Data Networking (NDN), which is one of the most promising ICN architectures for the Future Internet. Its impact on the future content delivery architecture is widely discussed. By taking this into consideration, we have found that some limitations of content finding can handicap the actual benefits of deploying NDN for content delivery enhancement, causing ineffective cache utilisation (one of the major advantages of NDN), which can result in sub-optimal delivery efficiency.

Therefore, developments of a particular solution (called *Vicinity-based Content Finding* scheme or *VCoF*) have been made. The essential components of the NDN code-bases are modified to apply the VCoF scheme in order to ensure that the solution is capable to be deployed in real NDN networks. In considering the scheme, it includes a set of techniques to proactively provide routing information for off-path (nearby) content replicas, which can be the better sources for consumers rather than original producers. Advances in the content finding techniques have also made the

best possible quality of experience for consumers due to the improved delivery delays. This also includes mobile devices that encounter particular content delivery issues due to their movement.

6.1 Thesis Summary

The major purpose of this thesis is to improve content finding in NDN, which is one of the most promising ICN architectures of the Future Internet. Through the envisaged benefits of NDN, several limitations of the current Internet architecture can be eliminated or mitigated. However, NDN has also a number of shortcomings that prevent to exploit its full potential. This is usually inefficient in terms of nearby cache usage, causing sub-optimal delivery efficiency. The proposed solution in this thesis (i.e., the VCoF scheme) has shown the favorable outcomes regarding to enhancing content delivery. This thesis is structured in 6 chapters, summarised here:

Chapter 1 introduces the motivations of this thesis and describes the research hypothesis, aiming to examine NDN and its shortcomings. This leads to develop a suitable solution realising on the defined research questions. The chapter also presents the research methodology and research outline of the thesis.

In Chapter 2, a number of limitations of the current Internet architecture are described along with the emergence of ICN and its fundamentals. A set of representative ICN approaches are then compared leading to the reasons of choosing NDN as the base architecture of this thesis. We also describe the differences between NDN and the well-known solution (the CDNs) of improving content delivery in today's Internet architecture since both of them provide caching as the key component. This includes how NDN can be a better solution. We then discuss the issues of off-path content finding in NDN that handicap the benefits of using NDN. This also considers the existing solutions and their drawbacks that should be avoided or eliminated in our design. In addition, the chapter highlights these particular issues within the well-known mobile communication model.

A set of motivations and aims is defined and described in Chapter 3. In this chapter, we introduce the design of VCoF to address the mentioned issues of content finding in NDN, mainly aiming on the improvement in terms of delivery efficiency. The design is also realised on avoiding the particular issues of several existing solutions (described in the previous chapter).

Chapter 4 presents the overall implementation of VCoF to realise the off-path

content finding issues in NDN and to fulfill the design aims in the previous chapter. The core software components of NDN are firstly introduced leading to understand how VCoF is applied into the current NDN code-bases. A set of tools is described to be used in the next chapter to evaluate VCoF in a number of aspects as discussed in Chapter 3 and also in order to answer the research questions.

Chapter 5 evaluates how well VCoF achieves the overall objectives of the thesis, i.e., of improving content finding in NDN by considering standard NDN as the baseline. It includes a set of different scenarios that is evaluated using the emulation and simulation. Various parameters have been examined such as a number of requests, content density/popularity, high/low densely networks, and various scales of networks. The evaluation metrics of RTT, message overhead, data volume are used to measure the trade-offs between the achieved performance and the additional overhead along with the consideration of re-transmission percentages and replica hit rates specifically in the mobile case.

In this final chapter, Chapter 6, concludes this work by summarising the thesis structure and highlighting the contributions of the thesis. This chapter also revisits the research questions (goals), summarises research limitations, and describes the directions of future work. Concluding remarks are also discussed.

6.2 Thesis Contributions

This thesis considers the content finding issues in NDN, a potential realisation of the ICN paradigms for the Future Internet. We focus on NDN due to its communication model that can be flexible to be deployed in various kinds of networks. It is also widely discussed as a next potential architecture for the Future Internet. This is done after understanding the basics and principles of several ICN approaches (summarised in Chapter 2). According to the study of the well-known content finding strategies in NDN, as well as their shortcomings, a particular set of motivations and aims is developed with the aim of improving the issues by utilising nearby (off-path) content objects. This is achieved alongside avoiding the existing deficiencies of several research proposals.

In considering the issues, we provide a comprehensive design capable of enhancing content delivery in NDN. This is separated into different modules, which aim to proactively provide routing information for nearby content replicas. The area of finding nearby objects is called a "vicinity". It is very crucial to limit the scope

of content finding for mitigating excessive overhead.

Building on this design, we implement a prototype content finding strategy called "VCoF" by integrating into the current NDN code-bases. This ensures that VCoF is able to be deployed in actual NDN systems. The implementation is also built to evaluate VCoF regarding a set of important factors that can affect different content finding results.

In the first instance, the VCoF prototype is used to examine and understand its deployment in an actual emulated NDN network. A set of factors has been investigated to indicate how much the implemented VCoF design can improve delivery efficiency compared to default NDN. According to the concept of NDN, desired content can be cached in several locations dependent on the content availability (i.e., density). Hence, various content (or replica) densities have been explored alongside the consideration of the number of request(s), which provides different overhead costs. This is to find an appropriate trade-off between the achieved performance in terms of delivery efficiency and the additional overhead.

Through this evaluation, we have found that finding nearby content can effectively improve delivery efficiency with acceptable overhead (depending on the scope (i.e., vicinity) of content finding). It is noted that a too large vicinity is potentially unnecessary because of the highly excessive overheads that outweigh the benefits gained. Although this proof of the design has shown that VCoF can be practically deployed in actual NDN networks, the emulator can not replicate a larger scale evaluation topology, which contains a higher number of content and consumers.

Further to this, we scale up the experimental environment using a simulator and use this to measure the impact of content popularity in the consideration of an actual data set of content access from more various consumers. This has been led by the understanding of the effect of a different number of replicas in the previous evaluation. This evaluation however has provided more understanding of the VCoF scheme aided by simulations of the real behaviors of requesting content. This also includes the understanding of the overall overheads generated particularly to find the desired content based on their popularity.

By examining this evaluation, the results has indicated the similar direction of improvement of the previous evaluation. This is shown through the ability to use nearby replicas, which can be the better source. It is confirmed by considering the reduction of delivery delays along with the increment of content popularity. Interestingly, in a more populated network (representing a network that has seen

high traffic passing through), existing content objects can be potentially replaced by a number of other content objects, reducing their availability, especially in a small cache size. This can decrease delivery efficiency compared to a larger cache size that can remain content objects for longer. Nevertheless, VCoF has still shown the advantage over default NDN due to the designed scope of content finding that provides the better opportunities of nearby cache utilisation.

Another contribution of this thesis is the content finding enhancement in mobile NDN scenarios, which due to their communication model encountering the particular frequent issues of content finding and delivery. A short description of the issues is that a mobile node has missed its desired content due to handover. According to our investigation, a content object is often forwarded to the previous location (i.e., NAR) of the mobile node that is frequently located in the same vicinity of the current NAR. However, the object might be cached in a different vicinity, if the NARs are separately located in different Location Areas (LAs). Hence, VCoF also fits this communication model since the area of content finding covers the previous NAR and other nearby replicas in the vicinity, and either other nearby vicinities.

In this final evaluation, we have demonstrated the enhancement and flexibility (by taking the advantage of the decentralised design) of deploying VCoF in the mobile environment. The results have also confirmed the effective improvement of content finding that efficiently handles the aforementioned issues. Notably, unlike the previous evaluations, in this evaluation, content replicas are often forwarded into the area of the mobile node due to its movement, which means they can be found in a small number of hop(s), especially in 1-hop vicinity. A larger vicinity is significantly unnecessary due to the very slight improvement with the higher overhead compared to the small vicinity.

Together, these evaluations examine a number of aspects by exploring a potential form of the Future Internet considering NDN as the base architecture. They highlight the benefits and limitations of deploying NDN and also ICN, which has been emerged as the promising paradigm for future content delivery and also how it can be enhanced. This allows us to better understand the important steps towards the development of the next generation Internet.

6.3 Research Goals Revisited

We described the research goals in the introductory chapter of this thesis by defining a set of research questions that must be answered to achieve the purposes of the study. In this section, a summary of each research question is given below:

1. **Can the content finding of default NDN be improved by considering the vicinity of the consumer?** This thesis aims to address the off-path content finding issues in standard NDN as discussed in Section 2.5.1. According to the designed scheme, VCoF offers higher opportunities of finding nearby replicas, which can be the potential source for the consumer. This has been proven by the reduction of the delivery delays demonstrated in the evaluations. Not only VCoF has improved delivery efficiency but also higher cache utilisation (confirmed by the higher delivery performance and replica hit rates). Comparing this to default NDN, although content caching is one of the key principles of NDN, the default strategy does not utilise nearby cached content by the design itself causing sub-optimal delivery efficiency. Due to the evaluation results, VCoF can indeed help to improve content finding by considering nearby replicas based on the vicinity concept.

2. **Can the delivery performance of default NDN be increased through the involvement of nearby nodes, and if yes by how much?** According to the evaluation results, the VCoF scheme has demonstrated the effective improvement of delivery efficiency compared to standard NDN. VCoF can indeed gain the advantage of the replicas in nearby nodes to be the better source for content delivery. In considering the overall evaluations, the percentage of the improved performance can reach up to 44% with acceptable overhead[1]. More than 48% can be achieved but the excessive overhead may outweigh the benefits.

3. **Does the improvement of content finding create additional overheads, and if so how high are these?** VCoF introduces higher message overhead (i.e., packets) for locating a desired content object compared to default NDN due to the dominant effect of the *Content List* pushing mechanism. Nevertheless, the

[1]Note, all of the results are based on the evaluation setups specifically performed in this work which can be different within other network conditions. These evaluations mainly aim to prove that nearby replicas based on the vicinity concept can indeed improve content finding in standard NDN.

trade-offs between the overhead and the achieved delivery performance have also been discussed. For example, VCoF can effectively offer a 29% of improvement by efficiently adding only ≈5 packets to the average message overhead per node. However, in some cases, such as when the vicinity size is large (e.g., 3 hops), less than 4% of improvement (compared to the smallest vicinity size of one) is noticed while the overhead cost is significantly increased, potentially outweighing the benefits. It should be keep in mind that the list pushing packets are quite small and the data volume results have demonstrated that they do not consume high storage or traffic in each single node that may impact the entire network.

4. **What are the effects of content popularity (or replica density) and mobility on the performance and overhead costs of content finding?** We have examined different content popularity levels ranging from low to high. We have found that VCoF can significantly offer the better performance of content finding over default NDN in almost every case but the highest advantage is when the number of content replicas are sufficiently high (e.g., up to 44% with the case of the medium density of 50%, even in the narrowest vicinity size of one). The differences of improvement between VCoF and default NDN will then slightly decrease since the replicas of desired content are distributed to several (or almost) every node. They can be found easier, even in the default paths. Nevertheless, VCoF has still shown the advantage in the high content popularity cases over standard NDN. For instance, the improvement can reach up to 18%, even in the smallest vicinity size of one. VCoF provides the favorable performance with the slight increment of the overhead costs which are acceptable especially in the vicinity size of one. Expanding too large vicinity sizes such as the 3-hop vicinity might outweigh the benefits since the narrow vicinity can also provide the opportunity to find content in nearby vicinities while introducing a lower number of overhead costs.

This thesis also focuses on the frequent issues of content finding and delivery in the well-known mobile communication model mentioned in Section 2.5.3. The evaluation results have again demonstrated that VCoF outperforms default NDN in terms of delivery efficiency. In particular, unlike those evaluations in the static topologies (the increments of RTT reductions depend on the increments of content density), the RTT results reduce to an almost similar level when the content density is high enough (e.g., more than 50% of replica density). This is

because the replicas are often cached in the specific area of each NAR, which desired content can be found in a few number of hops. Also, the costs of content finding can be worthwhile since the scope of content finding is not excessively large. These also fit well with the advantage of the VCoF scheme.

6.4 Summary of Research Limitations

This work aims to prove that nearby content finding based on the VCoF concept can indeed improve content finding in standard NDN. Since the development of VCoF is in its early stages, a number of possible research limitations should be noted. These can be considered in two main categories: 1) design and implementation, and 2) evaluation.

In considering the design and implementation, we assume that a content object (as an entry in a node's *Content List*) which is advertised by the *Content Availability Advertisement* module is available to be located. However, some issues should be addressed if a number of entries in the *Content List* do not synchronise with the available content objects at the owner of the list due to cache replacement or removal. For example, in a ring topology, assuming the worst case that an interest packets always cannot find its corresponding data in every node in the topology due to cache eviction, the interest is always forwarded to every upstream node (default NDN concept) in the path and the circling direction to its requester according to the assumed topology and proactive routing information generated by VCoF. So, an interest loop is then detected and the desired content can not be found. Nevertheless, multi-sourcing (e.g., from multiple caches) offers a number of alternate paths to find the desired content, thereby naturally mitigating these issues.

Although the aforementioned issues especially the loop problem could be the rare case, some mechanisms to handle this are preferred. Extra mechanisms to the current VCoF scheme can be developed. For instance, all of the content in the latest *Content List* are not allowed to be replaced or removed for a specific period (e.g., timeout). By considering the caching policy like FIFO, the *Content List* can also advertise only a number of recent content that have been stored in the cache to mitigate the opportunities that the older content objects are replaced but the list still reflects these objects. The *Content List* can also consider only a number of particular content (e.g., popular content). The main aim of these mechanisms is to guarantee that the proactive routing information can still indicate any corresponding

interest packets to their available content at nearby sources. Every content store might reserve some space for these content (e.g., $x\%$ of the cache) and the *Content List* can only consider the content in the reserved space. Although some experiments in Chapter 5 investigate the impact of cache replacement, more complex conditions and the handling mechanisms as discussed previously would be challenging to the Future of VCoF.

Further to the first category, in the second category of research limitations, a number of challenges should be noted. This is also discussed in the three main evaluations as detailed in Chapter 5. Discussing about the topologies (including the emulation and simulation topologies), these are not likely the practical topologies regarding the consideration of existing topologies based on the current Internet architecture. Since in this architecture, routers/end devices cannot naturally cache and serve content, accessing content by using NDN could be different. Based on the principles of native NDN, every node in a topology can support content caching and multi-sourcing. Hence, we consider the potential topologies (e.g., actual NDN test-beds) that can represent realistic NDN topologies since all of NDN primitives can be deployed. This is also inspired by the NDN research communities such as several participating institutions in [165] or in [55] investigating on NDN forwarding.

Since NDN is in its early stages of development and at the time of this work, no global scale native NDN traffic that we can essentially examine the impact of content popularity/density to content finding. In our hypothesis, this is one of the most important factors impacting on different content finding results. Hence, we consider content distribution that can likely represent realistic content access with the different levels of content popularity performed by a number of consumers in the evaluation topologies (e.g., the real VoD dataset). Furthermore, different parts in a network system generally see the different number of content updates. This could impact the VCoF scheme especially the overhead costs. For example, core routers can see the higher number of content updates compared to edge networks. Thereby, the overhead results could be different from this work.

Nevertheless, the main aim of this work is to first prove that the vicinity-based concept can indeed improve content finding in NDN. In the same network conditions, VCoF is fairly compared to standard NDN and the results have indicated the favorable trade-offs between the delivery efficiency and additional overhead costs especially in the small vicinity sizes. However, more complex network conditions (e.g., different link latency characteristics, jitter, throughput, etc.), different parts of the network,

and more various kinds of topologies would be challenging to the Future of VCoF. Memory and CPU overheads could also be challenging and some matching algorithms can be further applied in VCoF to optimise these overheads especially for the process of *Content List* checking. It is noted that the current design of VCoF especially the *Content List* and its advertisement strategy is not the final design and it is customisable for specific use in different kinds of networks. This will be interestingly to be further investigated and developed. In this thesis, the results have mainly indicated that locating nearby content based on the vicinity concept can indeed improve content finding in NDN, and this has proved the main hypothesis of the research.

6.5 Future Work

In this thesis, we presented the VCoF design with the main aim to improve content finding in standard NDN. We then implemented the prototype of VCoF by integrating with the current NDN code-bases and examined a number of evaluation aspects. Nevertheless, further development and deployment continue on VCoF could help to strengthen its strategy of content finding in NDN. Not only a number of further research challenges described in the previous section (Section 6.4) need to be investigated but a number of future work possibilities could also be explored. These can be described as follows.

The important goal of VCoF is to develop a concrete technique of effective content finding in NDN mainly aimed to increase delivery efficiency. The development is integrated into the current NDN code-bases. Hence, in the future, VCoF can be an alternate content finding strategy in NDN since the scheme is built based on the existing essential components of NDN.

Future work also includes automatic discovery of nodes in a vicinity since the current technique assumes that a node knows each member in its vicinity by its configuration defined by system administrators or evaluation scripts used in this work. The idea of Neighbor Discovery Protocol (NDP) [166, 167] might be interesting. In the protocol, a router solicitation message is sent to discover routers on an attached link. By extending this concept, we can send a message to discover each name prefix of every node in a vicinity depending on the hop limit of the defined vicinity size.

Interestingly, in NDN, when a pushed *Content List* packet has been sent to an NDN node, we can check the number of hops that the pushed packet has passed. So, due to this information, we can limit the scope of a vicinity by considering the hop

limit. When the pushed packet reaches the threshold of the vicinity size at an NDN node, this node just stops forwarding the pushed packet to any next hops because they are out of the vicinity's scope.

In recent years, mobile traffic grows more rapidly [13, 168]. In the context of mobility, NDN can also improve delivery efficiency according to the concepts of naming, caching and multi-point communication. However, in a mobile network, paths could change frequently. Sources of content may be going off-line or there are new local connectivities in a vicinity. These can be challenging to ensure that content objects can be forwarded reversely to their requesters in a dynamic environment [169].

Further to this, according to the decentralised concept of VCoF, consumers can gain benefits of locating nearby replicas under different dynamic situations. In this thesis, we specifically considered the mobile topology consisting of the infrastructure network. The result revealed the favorable content delivery performance. Nevertheless, another future research direction can be the extension of VCoF in other mobile topologies such as Mobile Ad Hoc Networks (MANETs) [170] and Vehicle-to-Vehicle (V2V) communications [171] in Vehicular Networks.

Those topologies represent more complex distributed systems that mobile nodes can freely and dynamically self-organise into temporary "ad-hoc" network topologies without communication infrastructure. The proposed scheme might also help to improve content finding in these kind of topologies since a moving node might carry desired content objects to another node in the same vicinity that is looking for the same content objects. However, to deploy VCoF in a highly dynamic environment, the aforementioned feature of automatic discovery of nodes in each vicinity should be further explored.

One of the most important factors that can affect the performance of content finding is content placement and the popularity of cached replicas. Caching popular content objects can decrease the network load and the access delay [172] and the results in this work have proved that higher content popularity of desired content can effectively increase delivery efficiency. In particular, if desired content objects can be remained longer in caches, the opportunities to locate these objects are also higher. In a small cache size, several content objects might be replaced quickly, causing the difficulty of finding replaced content. This can also depend on different caching policies. Once these were taken into consideration, we have found that caching policies might not play a major role in terms of content finding, if the cache size is large enough

or desired content objects can be remained longer in each cache[2]. However, if a cache size is small, we suggest that caching highly frequent used objects would offer more benefits because most desired content objects can be easily located compared to less frequent used objects. Hence, in the case of limited cache capacity, cache replacement policies that can hold popular content objects might be more beneficial and these can be further investigated.

6.6 Concluding Remarks

In this thesis, contributions to enhancing content delivery of the Future Internet based on the novel (promising) network paradigm (i.e., NDN) are developed. This includes the prototype in which future deployments can be applied. The extensive evaluations have shown the favorable benefits of using the emerged architecture specifically integrated with the proposed VCoF design. There is a clear advantage to properly utilise caches, which is the key principle of NDN, and significantly ensure that the appropriate cache utilisation (e.g., fetching nearby content) integrated into the current NDN architecture will be the key to realise increased delivery efficiency.

Although early signs are encouraging that NDN is one of the most promising architectures for the Future Internet, these are not guarantee that it will replace the existing Internet architecture in the near future. Since it is still in its early stages, there are a number of challenges to be met. For example, at the time of this thesis, most NDN test-beds run as overlay networks on top of traditional IP networks. Global scale native NDN networks must be required to examine the practical implementation of NDN in real world. This might require further specific hardware customisation for NDN, however.

By considering the NDN code-bases themselves, although these are designed to be executed over various kinds of networks, the NDN forwarder (i.e., NFD) is currently supported in a number of platforms. Users (mostly developers) need to install the NDN components manually. This might be too complex for ordinary users. However, it is understandable since NDN is in the current stage of development. Hence, in the future of a deployment stage, NDN need to be built-in in the kernels of Operating Systems to avoid the complexity.

Nevertheless, there is a good sign of providing a common API across several

[2]Note, in our experiments, we used the default Least Recently Used (LRU) cache replacement policy.

programming languages. This can speed-up adoption of NDN by enabling applications developed by various languages connecting to the core forwarder. Hence, a growing number of developing communities can push NDN towards the actual implementation in real world. It will then be an efficient architecture for the Future Internet. Furthermore, the proposed VCoF scheme presented in this thesis will also potentially supplement NDN usage since this has been proved by the evaluation results.

Bibliography

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '09. New York, NY, USA: ACM, 2009, pp. 1–12.

[2] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, k. claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, pp. 66–73, July 2014.

[3] J. Pan, S. Paul, and R. Jain, "A survey of the research on future internet architectures," *IEEE Communications Magazine*, vol. 49, pp. 26–36, July 2011.

[4] L. Wang, V. Lehman, A. K. M. M. Hoque, B. Zhang, Y. Yu, and L. Zhang, "A secure link state routing protocol for ndn," *IEEE Access*, vol. 6, pp. 10 470–10 482, 2018.

[5] G. Tyson, S. Kaune, S. Miles, Y. El-khatib, A. Mauthe, and A. Taweel, "A trace-driven analysis of caching in content-centric networks," in *2012 21st International Conference on Computer Communications and Networks (ICCCN)*, July 2012, pp. 1–7.

[6] M. B. Lehmann, M. P. Barcellos, and A. Mauthe, "Providing producer mobility support in ndn through proactive data replication," in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, April 2016, pp. 383–391.

[7] N. Abani, G. Farhadi, A. Ito, and M. Gerla, "Popularity-based partial caching for information centric networks," in *2016 Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, June 2016, pp. 1–8.

[8] C. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, and L. Zhang, "A case for stateful forwarding plane," *Comput. Commun.*, vol. 36, no. 7, pp. 779–791, April 2013. [Online]. Available: http://dx.doi.org/10.1016/j.comcom.2013.01.005

[9] J. Cao, D. Pei, X. Zhang, B. Zhang, and Y. Zhao, "Fetching popular data from the nearest replica in ndn," in *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, August 2016, pp. 1–9.

[10] T. Zahariadis, D. Papadimitriou, H. Tschofenig, P. Daras, G. Stamoulis, and M. Hauswirth, "Towards a future internet architecture," in *The Future Internet*. Springer Berlin Heidelberg, January 2011, pp. 7–18.

[11] FIArch Group, "Fundamental Limitations of Current Internet and the path to Future Internet," *White Paper*, 2010.

[12] A. Feldmann, "Internet clean-slate design: What and why?" *Computer Communication Review*, vol. 37, pp. 59–64, July 2007.

[13] Cisco, "Cisco Annual Internet Report (2018–2023)," *White Paper*, March 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

[14] P. Ray, "A survey on internet of things architectures," *Journal of King Saud University - Computer and Information Sciences*, vol. 30, no. 3, pp. 291 – 319, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1319157816300799

[15] T. Moors, "A critical review of "end-to-end arguments in system design"," in *2002 IEEE International Conference on Communications. Conference Proceedings. ICC 2002 (Cat. No.02CH37333)*, vol. 2, 2002, pp. 1214–1219 vol.2.

[16] A. Afanasyev, N. Tilley, P. Reiher, and L. Kleinrock, "Host-to-host congestion control for tcp," *Communications Surveys and Tutorials, IEEE*, vol. 12, pp. 304 – 342, September 2010.

[17] N. Fotiou, K. Katsaros, G. Polyzos, M. Särelä, D. Trossen, and G. Xylomenos, "Handling mobility in future publish-subscribe information-centric networks," *Telecommunication Systems*, vol. 53, July 2013.

[18] C. E. Perkins, S. R. Alpert, and B. Woolf, *Mobile IP; Design Principles and Practices*, 1st ed. USA: Addison-Wesley Longman Publishing Co., Inc., 1997.

[19] Y.-a. Chen, "A survey paper on mobile ip," January 2000.

[20] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Communications Magazine*, vol. 50, no. 7, pp. 26–36, July 2012.

[21] M. Handley, "Why the internet only just works," *BT Technology Journal*, vol. 24, no. 3, p. 119–129, 2006. [Online]. Available: https://doi.org/10.1007/s10550-006-0084-z

[22] A. Barakabitze, T. Xiaoheng, and G. Tan, "A survey on naming, name resolution and data routing in information centric networking (icn)," *IJARCCE*, vol. 3, pp. 2278–1021, October 2014.

[23] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos, "A survey of information-centric networking research," *IEEE Communications Surveys Tutorials*, vol. 16, no. 2, pp. 1024–1049, 2014.

[24] W. Stallings, *Foundations of Modern Networking: SDN, NFV, QoE, IoT, and Cloud*. Addison-Wesley Professional, 2016.

[25] A. Kostopoulos, I. Papafili, C. Kalogiros, T. Levä, N. Zhang, and D. Trossen, "A tussle analysis for information-centric networking architectures," in *The Future Internet*, F. Álvarez, F. Cleary, P. Daras, J. Domingue, A. Galis, A. Garcia, A. Gavras, S. Karnourskos, S. Krco, M.-S. Li, V. Lotz, H. Müller, E. Salvadori, A.-M. Sassen, H. Schaffers, B. Stiller, G. Tselentis, P. Turkama, and T. Zahariadis, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 6–17.

[26] D. Kutscher, H. Flinck, and H. Karl, "Information-centric networking: a position paper," in *Proc. of 6th GI/ITG KuVS Workshop on Future Internet*, November 2010.

[27] L. Galluccio, G. Morabito, and S. Palazzo, "Caching in information-centric satellite networks," in *2012 IEEE International Conference on Communications (ICC)*, 2012, pp. 3306–3310.

[28] Information-centric networking research group. [Online]. Available: https://irtf.org/icnrg

[29] 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.

[30] M. F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and B. Mathieu, "A survey of naming and routing in information-centric networks," *IEEE Communications Magazine*, vol. 50, no. 12, pp. 44–53, December 2012.

[31] G. Pavlou, "Information-centric networking and in-network cache management: Overview, trends and challenges," *IFIP/IEEE CNSM*, 2013.

[32] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, p. 181–192, 2007. [Online]. Available: https://doi.org/10.1145/1282427.1282402

[33] R. Govindaraj, I. Sengupta, and S. Chattopadhyay, "An efficient technique for longest prefix matching in network routers," in *Progress in VLSI Design and Test*, H. Rahaman, S. Chattopadhyay, and S. Chattopadhyay, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 317–326.

[34] C. Dannewitz, "Netinf: An information-centric design for the future internet," January 2009.

[35] B. Ahlgren, M. D'Ambrosio, I. Marsh, C. Dannewitz, B. Ohlman, K. Pentikousis, O. Strandberg, R. Rembarz, and V. Vercellone, "Design considerations for a network of information," January 2008, p. 66.

[36] M. D'Ambrosio, C. Dannewitz, H. Karl, and V. Vercellone, "Mdht: A hierarchical name resolution service for information-centric networks," in *Proceedings of the ACM SIGCOMM Workshop on Information-centric Networking*, ser. ICN '11. New York, NY, USA: ACM, 2011, pp. 7–12. [Online]. Available: http://doi.acm.org/10.1145/2018584.2018587

[37] A. Eriksson and B. Ohlman, "Dynamic internetworking based on late locator construction," in *2007 IEEE Global Internet Symposium*, 2007, pp. 67–72.

[38] M. D'Ambrosio, P. Fasano, V. Vercellone, and M. Ullio, "Providing data dissemination services in the future internet," January 2008, pp. 5606–5611.

[39] C. Dannewitz, J. Golic, B. Ohlman, and B. Ahlgren, "Secure naming for a network of information," April 2010, pp. 1 – 6.

[40] R. Abassi, R. Abassi, and A. B. C. Douss, *Security Frameworks in Contemporary Electronic Government*, 1st ed. USA: IGI Global, 2018.

[41] I. Seskar, K. Nagaraja, S. Nelson, and D. Raychaudhuri, "Mobilityfirst future internet architecture project," in *Proceedings of the 7th Asian Internet Engineering Conference*, ser. AINTEC '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 1–3. [Online]. Available: https://doi.org/10.1145/2089016.2089017

[42] A. Venkataramani, J. Kurose, D. Raychaudhuri, K. Nagaraja, S. Banerjee, and Z. Mao, "Mobilityfirst: A mobility-centric and trustworthy internet architecture," *ACM SIGCOMM Computer Communication Review*, vol. 44, pp. 74–80, July 2014.

[43] N. Fotiou, D. Trossen, and G. Polyzos, "Illustrating a publish-subscribe internet architecture," *Telecommunication Systems*, vol. 51, pp. 1–13, June 2010.

[44] N. Fotiou, P. Nikander, D. Trossen, and G. Polyzos, "Developing information networking further: from psirp to pursuit," vol. 66, October 2010.

[45] S. Tarkoma, M. Ain, and K. Visala, "The publish/subscribe internet routing paradigm (psirp): Designing the future internet architecture." May 2009, pp. 102–111.

[46] V. Dimitrov and V. Koptchev, "Psirp project – publish-subscribe internet routing paradigm: New ideas for future internet," in *Proceedings of the 11th International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing on International Conference on Computer Systems and Technologies*, ser. CompSysTech '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 167–171. [Online]. Available: https://doi.org/10.1145/1839379.1839409

[47] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Commun. ACM*, vol. 13, no. 7, p. 422–426, July 1970. [Online]. Available: https://doi.org/10.1145/362686.362692

[48] D. Trossen and G. Parisis, "Designing and realizing an information-centric internet," *IEEE Communications Magazine*, vol. 50, no. 7, pp. 60–67, 2012.

[49] H. Park, I. Widjaja, and H. Lee, "Detection of cache pollution attacks using randomness checks," in *2012 IEEE International Conference on Communications (ICC)*, 2012, pp. 1096–1100.

[50] N. Fotiou, G. F. Marias, and G. C. Polyzos, "Information ranking in content-centric networks," in *2010 Future Network Mobile Summit*, 2010, pp. 1–7.

[51] S. Harada, Z. Yan, Y. Park, K. Nisar, and A. A. A. Ibrahim, "Data aggregation in named data networking," in *TENCON 2017 - 2017 IEEE Region 10 Conference*, 2017, pp. 1839–1842.

[52] A. Albarqi, E. Alzaid, F. Alghamdi, S. Asiri, and J. Kar, "Public key infrastructure: A survey," *Journal of Information Security*, vol. 06, pp. 31–37, January 2015.

[53] NDN, "How does NDN differ from Content-Centric Networking (CCN)," https://named-data.net/project/faq/#How_does_NDN_differ_from_Content-Centric_Networking_CCN, July 2020.

[54] A. K. M. M. Hoque, S. O. Amin, A. Alyyan, B. Zhang, L. Zhang, and L. Wang, "Nlsr: Named-data link state routing protocol," in *Proceedings of the 3rd ACM SIGCOMM Workshop on Information-centric Networking*, ser. ICN '13. New York, NY, USA: ACM, 2013, pp. 15–20.

[55] V. Lehman, A. Gawande, B. Zhang, L. Zhang, R. Aldecoa, D. Krioukov, and L. Wang, "An experimental investigation of hyperbolic routing with a smart forwarding plane in ndn," in *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*, June 2016, pp. 1–10.

[56] H. Dai, J. Lu, Y. Wang, and B. Liu, "A two-layer intra-domain routing scheme for named data networking," in *2012 IEEE Global Communications Conference (GLOBECOM)*, December 2012, pp. 2815–2820.

[57] G. Grassi, D. Pesavento, G. Pau, L. Zhang, and S. Fdida, "Navigo: Interest forwarding by geolocations in vehicular named data networking," in *2015 IEEE 16th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, June 2015, pp. 1–10.

[58] A. Kerrouche, M. R. Senouci, and A. Mellouk, "Qos-fs: A new forwarding strategy with qos for routing in named data networking," in *2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–7.

[59] C. Yi, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, "Adaptive forwarding in named data networking," *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 3, pp. 62–67, July 2012. [Online]. Available: http://doi.acm.org/10.1145/2317307.2317319

[60] J. Cao, D. Pei, Z. Wu, X. Zhang, B. Zhang, L. Wang, and Y. Zhao, "Improving the freshness of ndn forwarding states," in *2016 IFIP Networking Conference (IFIP Networking) and Workshops*, May 2016, pp. 189–197.

[61] C. Yi, J. Abraham, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, "On the role of routing in named data networking," in *Proceedings of the 1st ACM Conference on Information-Centric Networking*, ser. ACM-ICN '14. New York, NY, USA: ACM, 2014, pp. 27–36. [Online]. Available: http://doi.acm.org/10.1145/2660129.2660140

[62] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and L. Zhang, "Interest flooding attack and countermeasures in named data networking," in *2013 IFIP Networking Conference*, May 2013, pp. 1–9.

[63] M. Tortelli, L. A. Grieco, G. Boggia, and K. Pentikousisy, "Cobra: Lean intra-domain routing in ndn," in *2014 IEEE 11th Consumer Communications and Networking Conference (CCNC)*, January 2014, pp. 839–844.

[64] COMET, "COntent Mediator architecture for content-aware nETworks (COMET)," http://www.comet-project.org/, January 2010.

[65] CONVERGENCE, "The CONVERGENCE Project," http://www.ict-convergence.eu/, June 2010.

[66] A. Detti, N. Melazzi, S. Salsano, and M. Pomposini, "Conet: A content centric inter-networking architecture," August 2011.

[67] W. K. Chai, N. Wang, I. Psaras, G. Pavlou, C. Wang, G. Garcia de Blas, F. J. Ramon-Salguero, L. Liang, S. Spirou, A. Beben, and E. Hadjioannou, "Curling: Content-ubiquitous resolution and delivery infrastructure for next-generation services," *IEEE Communications Magazine*, vol. 49, no. 3, pp. 112–120, 2011.

[68] T.-x. Do and Y. Kim, "Optimal provider mobility in large-scale named-data networking," *KSII Transactions on Internet and Information Systems*, vol. 9, pp. 4054–2071, October 2015.

[69] Z. Zhang, Y. Yu, H. Zhang, E. Newberry, S. Mastorakis, Y. Li, A. Afanasyev, and L. Zhang, "An overview of security support in named data networking," *IEEE Communications Magazine*, vol. 56, no. 11, pp. 62–68, 2018.

[70] B. Krishnamurthy, C. Wills, and Y. Zhang, "On the use and performance of content distribution networks," in *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, ser. IMW '01. New York, NY, USA: Association for Computing Machinery, 2001, p. 169–182. [Online]. Available: https://doi.org/10.1145/505202.505224

[71] L. Guo, S. Chen, Z. Xiao, and X. F. Zhang, "Analysis of multimedia workloads with implications for internet streaming," 01 2005, pp. 519–528.

[72] S. Sivasubramanian, D. R. Richardson, C. L. Scofield, and B. E. Marshall, "Request routing using network computing components," April 2015.

[73] M. Pathan and R. Buyya, *A taxonomy of CDNs*, January 2008, vol. 9, pp. 33–77.

[74] M. Broadbent, "Opencache: A content delivery platform for the modern internet," Ph.D. dissertation, School of Computing and Communications, Lancaster University, December 2015.

[75] A.-S. K. Pathan and R. Buyya, "A taxonomy and survey of content delivery networks," Grid Computing and Distributed Systems Laboratory, Melbourne, Australia, 2007.

[76] H. Nielsen, R. T. Fielding, and T. Berners-Lee, "Hypertext Transfer Protocol – HTTP/1.0," RFC 1945, May 1996. [Online]. Available: https://rfc-editor.org/rfc/rfc1945.txt

[77] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig, "Comparing dns resolvers in the wild," 01 2010, pp. 15–21.

[78] G. Ma, Z. Chen, J. Cao, Z. Guo, Y. Jiang, and X. Guo, "A tentative comparison on cdn and ndn," in *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, October 2014, pp. 2893–2898.

[79] M. Mangili, F. Martignon, and A. Capone, "A comparative study of content-centric and content-distribution networks: Performance and bounds," in *2013 IEEE Global Communications Conference (GLOBECOM)*, December 2013, pp. 1403–1409.

[80] A. Lertsinsrubtavee, P. Mekbungwan, and N. Weshsuwannarugs, "Comparing ndn and cdn performance for content distribution service in community wireless mesh network," in *Proceedings of the AINTEC 2014 on Asian Internet Engineering Conference*, ser. AINTEC '14. New York, NY, USA: ACM, 2014, pp. 43:43–43:50. [Online]. Available: http://doi.acm.org/10.1145/2684793.2684800

[81] H. Yuan and P. Crowley, "Experimental evaluation of content distribution with ndn and http," in *2013 Proceedings IEEE INFOCOM*, April 2013, pp. 240–244.

[82] Y. Wang, K. Lee, B. Venkataraman, R. L. Shamanna, I. Rhee, and Sunhee Yang, "Advertising cached contents in the control plane: Necessity and feasibility," in *2012 Proceedings IEEE INFOCOM Workshops*, 2012, pp. 286–291.

[83] H. Farahat, R. Atawia, and H. S. Hassanein, "Robust proactive mobility management in named data networking under erroneous content prediction," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, December 2017, pp. 1–6.

[84] H. Farahat and H. S. Hassanein, "Supporting consumer mobility using proactive caching in named data networks," in *2016 IEEE Global Communications Conference (GLOBECOM)*, December 2016, pp. 1–6.

[85] ——, "Proactive caching for producer mobility management in named data networks," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, June 2017, pp. 171–176.

[86] A. Marandi, T. Braun, K. Salamatian, and N. Thomos, "Bfr: A bloom filter-based routing approach for information-centric networks," in *2017 IFIP Networking Conference (IFIP Networking) and Workshops*, 2017, pp. 1–9.

[87] S. Bayhan, L. Wang, J. Ott, J. Kangasharju, A. Sathiaseelan, and J. Crowcroft, "On content indexing for off-path caching in information-centric networks," in *Proceedings of the 3rd ACM Conference on Information-Centric Networking*, ser. ACM-ICN '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 102–111. [Online]. Available: https://doi.org/10.1145/2984356.2984372

[88] A. Afanasyev, X. Jiang, Y. Yu, J. Tan, Y. Xia, A. Mankin, and L. Zhang, "Ndns: A dns-like name service for ndn," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, 2017, pp. 1–9.

[89] C. Dannewitz, D. Kutscher, B. Ohlman, S. Farrell, B. Ahlgren, and H. Karl, "Network of information (netinf) - an information-centric networking architecture," *Comput. Commun.*, vol. 36, no. 7, pp. 721–735, April 2013. [Online]. Available: http://dx.doi.org/10.1016/j.comcom.2013.01.009

[90] L. Saino, I. Psaras, and G. Pavlou, "Hash-routing schemes for information centric networking," in *Proceedings of the 3rd ACM SIGCOMM Workshop on Information-Centric Networking*, ser. ICN '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 27–32. [Online]. Available: https://doi.org/10.1145/2491224.2491232

[91] V. Sourlas, I. Psaras, L. Saino, and G. Pavlou, "Efficient hash-routing and domain clustering techniques for information-centric networks," *Computer Networks*, vol. 103, pp. 67 – 83, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128616300998

[92] D. Trossen and G. Parisis, "Designing and realizing an information-centric internet," *IEEE Communications Magazine*, vol. 50, no. 7, pp. 60–67, 2012.

[93] B. Krishnamurthy, C. Wills, and Y. Zhang, "On the use and performance of content distribution networks," in *Proceedings of the 1st ACM*

SIGCOMM Workshop on Internet Measurement, ser. IMW '01. New York, NY, USA: ACM, 2001, pp. 169–182. [Online]. Available: http://doi.acm.org/10.1145/505202.505224

[94] X. Jiang and J. Bi, "ncdn: Cdn enhanced with ndn," in *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, April 2014, pp. 440–445.

[95] S. Guo, H. Xie, and G. Shi, "Collaborative forwarding and caching in content centric networks," in *NETWORKING 2012*, R. Bestak, L. Kencl, L. E. Li, J. Widmer, and H. Yin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 41–55.

[96] L. Wang, M. Hoque, C. Yi, A. Alyyan, and B. Zhang, "Ospfn: an ospf based routing protocol for named data networking," January 2012.

[97] J. Torres, I. Alvarenga, R. Boutaba, and O. C. M. B. Duarte, "Evaluating cros-ndn: a comparative performance analysis of a controller-based routing scheme for named-data networking," *Journal of Internet Services and Applications*, vol. 10, December 2019.

[98] V. G. Vassilakis, M. F. Al-Naday, M. J. Reed, B. A. Alzahrani, K. Yang, I. D. Moscholios, and M. D. Logothetis, "A cache-aware routing scheme for information-centric networks," in *2014 9th International Symposium on Communication Systems, Networks Digital Sign (CSNDSP)*, 2014, pp. 721–726.

[99] N. Aloulou, M. Ayari, M. F. Zhani, and L. Saidane, "A popularity-driven controller-based routing and cooperative caching for named data networks," in *2015 6th International Conference on the Network of the Future (NOF)*, September 2015, pp. 1–5.

[100] K. Shilton, J. Burke, C. Duan, and L. Zhang, "A world on ndn: Affordances and implications of the named data networking future internet architecture," Tech. Rep., 2014.

[101] V. Sivaraman and B. Sikdar, "Hop-count based forwarding for seamless producer mobility in ndn," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, December 2017, pp. 1–6.

[102] D. Gao, Y. Rao, C. H. Foh, H. Zhang, and A. V. Vasilakos, "Pmndn: Proxy based mobility support approach in mobile ndn environment," *IEEE Transactions on Network and Service Management*, vol. 14, no. 1, pp. 191–203, March 2017.

[103] S. Hassan, A. Habbal, R. Alubady, and M. Salman, "A taxonomy of information-centric networking architectures based on data routing and name resolution approaches," *Journal of Telecommunication, Electronic and Computer Engineering*, vol. 8, pp. 99–107, January 2016.

[104] S. K. Fayazbakhsh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. Maggs, K. Ng, V. Sekar, and S. Shenker, "Less pain, most of the gain: Incrementally deployable icn," in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, ser. SIGCOMM '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 147–158. [Online]. Available: https://doi.org/10.1145/2486001.2486023

[105] G. Rossini and D. Rossi, "Coupling caching and forwarding: Benefits, analysis, and implementation," in *Proceedings of the 1st ACM Conference on Information-Centric Networking*, ser. ACM-ICN '14. New York, NY, USA: ACM, 2014, pp. 127–136. [Online]. Available: http://doi.acm.org/10.1145/2660129.2660153

[106] W. Wong, Liang Wang, and J. Kangasharju, "Neighborhood search and admission control in cooperative caching networks," in *2012 IEEE Global Communications Conference (GLOBECOM)*, 2012, pp. 2852–2858.

[107] R. Chiocchetti, D. Perino, G. Carofiglio, D. Rossi, and G. Rossini, "Inform: A dynamic interest forwarding mechanism for information centric networking," in *Proceedings of the 3rd ACM SIGCOMM Workshop on Information-Centric Networking*, ser. ICN '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 9–14. [Online]. Available: https://doi.org/10.1145/2491224.2491227

[108] R. Chiocchetti, D. Rossi, G. Rossini, G. Carofiglio, and D. Perino, "Exploit the known or explore the unknown? hamlet-like doubts in icn," *ICN'12 - ACM Proceedings of the Information-Centric Networking Workshop*, August 2012.

[109] G. Rossini and D. Rossi, "Evaluating ccn multi-path interest forwarding strategies," *Computer Communications*, vol. 36, no. 7, pp. 771 – 778, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140366413000261

[110] O. Akinwande, "Interest forwarding in named data networking using reinforcement learning," *Sensors*, vol. 18, p. 3354, October 2018.

[111] I. V. Bastos and I. M. Moraes, "A forwarding strategy based on reinforcement learning for content-centric networking," in *2016 7th International Conference on the Network of the Future (NOF)*, 2016, pp. 1–5.

[112] L. Wang, S. Bayhan, J. Ott, J. Kangasharju, A. Sathiaseelan, and J. Crowcroft, "Pro-diluvian: Understanding scoped-flooding for content discovery in information-centric networking," in *Proceedings of the 2Nd ACM Conference on Information-Centric Networking*, ser. ACM-ICN '15. New York, NY, USA: ACM, 2015, pp. 9–18. [Online]. Available: http://doi.acm.org/10.1145/2810156.2810162

[113] O. Ascigil, V. Sourlas, I. Psaras, and G. Pavlou, "Opportunistic off-path content discovery in information-centric networks," in *2016 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, June 2016, pp. 1–7.

[114] E. J. Rosensweig and J. Kurose, "Breadcrumbs: Efficient, best-effort content location in cache networks," in *IEEE INFOCOM 2009*, 2009, pp. 2631–2635.

[115] O. Ascigil, V. Sourlas, I. Psaras, and G. Pavlou, "A native content discovery mechanism for the information-centric networks," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, ser. ICN '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 145–155. [Online]. Available: https://doi.org/10.1145/3125719.3125734

[116] Y. Chen, R. H. Katz, and J. D. Kubiatowicz, "Scan: A dynamic, scalable, and efficient content distribution network," in *Pervasive Computing*, F. Mattern and M. Naghshineh, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 282–296.

[117] L. Wang, S. Bayhan, J. Ott, J. Kangasharju, A. Sathiaseelan, and J. Crowcroft, "Pro-diluvian: Understanding scoped-flooding for content discovery in information-centric networking," in *Proceedings of the 2nd ACM Conference on Information-Centric Networking*, ser. ACM-ICN '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 9–18. [Online]. Available: https://doi.org/10.1145/2810156.2810162

[118] H. Qian, R. Ravindran, G. Wang, and D. Medhi, "Probability-based adaptive forwarding strategy in named data networking," in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, 2013, pp. 1094–1101.

[119] D. Nguyen, M. Fukushima, K. Sugiyama, and A. Tagami, "Efficient multipath forwarding and congestion control without route-labeling in ccn," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, 2015, pp. 1533–1538.

[120] D. Posch, B. Rainer, and H. Hellwagner, "Saf: Stochastic adaptive forwarding in named data networking," *IEEE/ACM Transactions on Networking*, vol. 25, no. 2, pp. 1089–1102, 2017.

[121] G. Carofiglio, M. Gallo, L. Muscariello, M. Papalini, and Sen Wang, "Optimal multipath congestion control and request forwarding in information-centric networks," in *2013 21st IEEE International Conference on Network Protocols (ICNP)*, 2013, pp. 1–10.

[122] K. Lei, J. Wang, and J. Yuan, "An entropy-based probabilistic forwarding strategy in named data networking," in *2015 IEEE International Conference on Communications (ICC)*, June 2015, pp. 5665–5671.

[123] K. Lei, J. Yuan, and J. Wang, "Mdpf: An ndn probabilistic forwarding strategy based on maximizing deviation method," in *2015 IEEE Global Communications Conference (GLOBECOM)*, December 2015, pp. 1–7.

[124] N. Aloulou, M. Ayari, M. F. Zhani, L. Saidane, and G. Pujolle, "Taxonomy and comparative study of ndn forwarding strategies," in *2017 Sixth International Conference on Communications and Networking (ComNet)*, March 2017, pp. 1–8.

[125] H. Salah and T. Strufe, "Evaluating and mitigating a collusive version of the interest flooding attack in ndn," in *2016 IEEE Symposium on Computers and Communication (ISCC)*, June 2016, pp. 938–945.

[126] H. Dai, Y. Wang, J. Fan, and B. Liu, "Mitigate ddos attacks in ndn by interest traceback," in *2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, April 2013, pp. 381–386.

[127] D. Chuan-zhen, Z. Yan, and L. Ju-long, "A leading routing mechanism for neighbor content store," vol. 137, May 2014, pp. 159–173.

[128] Y. Zhang, Z. Xia, A. Afanasyev, and L. Zhang, "A note on routing scalability in named data networking," in *2019 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2019, pp. 1–6.

[129] H. Khelifi, S. Luo, B. Nour, H. Moungla, Y. Faheem, R. Hussain, and A. Ksentini, "Named data networking in vehicular ad hoc networks: State-of-the-art and challenges," *IEEE Communications Surveys Tutorials*, vol. 22, no. 1, pp. 320–351, 2020.

[130] S. Gao, H. Zhang, and B. Zhang, "Energy efficient interest forwarding in ndn-based wireless sensor networks," *Mobile Information Systems*, vol. 2016, pp. 1–15, 04 2016.

[131] M. Sepahkar and M. Khayyambashi, "Improving energy efficiency in information-centric mobile ad-hoc networks using places of interest while respecting privacy," *International Journal of Communication Systems*, vol. 32, p. e3945, 03 2019.

[132] J. Barrachina, P. Garrido, M. Fogue, F. Martinez, J.-C. Cano, C. Calafate, and P. Manzoni, "Road side unit deployment: A density-based approach," *IEEE Intelligent Transportation Systems Magazine*, vol. 5, pp. 30–39, July 2013.

[133] A. Festag, A. Hessler, R. Baldessari, L. Le, W. Zhang, and D. Westhoff, "Vehicle-to-vehicle and road-side sensor communication for enhanced road safety," 2008.

[134] J. Cha, J. Choi, J. Kim, S. Min, and Y. Han, "A mobility link service in ndn face to support consumer mobility service," in *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, 2017, pp. 444–449.

[135] J. Cha, J.-H. Choi, J.-Y. Kim, Y.-H. Han, and S.-G. Min, "A mobility link service for ndn consumer mobility," *Wireless Communications and Mobile Computing*, vol. 2018, pp. 1–8, July 2018.

[136] C. Huang-Fu, C. Chen, and Y. Lin, "Location tracking for wave unicast service," in *2010 IEEE 71st Vehicular Technology Conference*, 2010, pp. 1–5.

[137] A. Suwannasa, S. Puangpronpitag, and W. Phongsiri, "A novel authentication scheme for v2i communication based on wave unicast services," *International Journal of Distributed Sensor Networks*, vol. 2013, pp. 1–10, December 2013.

[138] Z. Zhou, X. Tan, H. Li, Z. Zhao, and D. Ma, "Mobindn: A mobility support architecture for ndn," in *Proceedings of the 33rd Chinese Control Conference*, 2014, pp. 5515–5520.

[139] J. Lee, D. Kim, M. Jang, and B. Lee, "Proxy-based mobility management scheme in mobile content centric networking (ccn) environments," in *2011 IEEE International Conference on Consumer Electronics (ICCE)*, 2011, pp. 595–596.

[140] R. Ravindran, S. Lo, X. Zhang, and G. Wang, "Supporting seamless mobility in named data networking," in *2012 IEEE International Conference on Communications (ICC)*, 2012, pp. 5854–5869.

[141] H. Nakazato, S. Zhang, Y. J. Park, A. Detti, D. Bursztynowski, Z. Kopertowski, and I. Psaras, "On-path resolver architecture for mobility support in information centric networking," in *2015 IEEE Globecom Workshops (GC Wkshps)*, 2015, pp. 1–6.

[142] Y. Rao, H. Zhou, D. Gao, H. Luo, and Y. Liu, "Proactive caching for enhancing user-side mobility support in named data networking," in *2013 Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, 2013, pp. 37–42.

[143] Libraries / ndn platform. [Online]. Available: https://named-data.net/codebase/platform/

[144] Ndn common client libraries (ndn-ccl). [Online]. Available: https://named-data.net/codebase/platform/ndn-ccl/

[145] ndn-cxx: Ndn c++ library with experimental extensions. [Online]. Available: https://github.com/named-data/ndn-cxx

[146] T. V. Smith, A. Afanasyev, and L. Zhang, "ChronoChat on Android," NDN, Technical Report NDN-0059, Revision 1, Apr. 2017.

[147] D. Coomes, A. Gawande, N. Gordon, and L. Wang, "Android multimedia sharing application over ndn," in *Proceedings of the 5th ACM Conference on Information-Centric Networking*, ser. ICN '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 210–211. [Online]. Available: https://doi.org/10.1145/3267955.3269014

[148] Nlsr - named data link state routing protocol. [Online]. Available: https://github.com/named-data/NLSR

[149] Z. Zhu and A. Afanasyev, "Let's chronosync: Decentralized dataset state synchronization in named data networking," in *2013 21st IEEE International Conference on Network Protocols (ICNP)*, 2013, pp. 1–10.

[150] Chronosync: synchronization library for distributed realtime applications for ndn. [Online]. Available: https://github.com/named-data/ChronoSync

[151] Nfd - named data networking forwarding daemon. [Online]. Available: http://named-data.net/doc/NFD/

[152] Mini-ndn: A mininet based ndn emulator. [Online]. Available: https://github.com/named-data/mini-ndn

[153] Ns-3 based named data networking (ndn) simulator. [Online]. Available: https://ndnsim.net/

[154] Mininet: Rapid prototyping for software defined networks. [Online]. Available: https://github.com/mininet/mininet

[155] C. Cabral, C. Esteve Rothenberg, and M. Magalhães, "Mini-ccnx: fast prototyping for named data networking," 08 2013, pp. 33–34.

[156] Ndn essential tools. [Online]. Available: https://github.com/named-data/ndn-tools

[157] A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM: NDN simulator for NS-3," NDN, Technical Report NDN-0005, October 2012. [Online]. Available: http://named-data.net/techreports.html

[158] ns-3 — a discrete-event network simulator for internet systems. [Online]. Available: https://www.nsnam.org/

[159] S. Mastorakis, A. Afanasyev, and L. Zhang, "On the evolution of ndnsim: An open-source simulator for ndn experimentation," *SIGCOMM Comput. Commun. Rev.*, vol. 47, no. 3, p. 19–33, September 2017. [Online]. Available: https://doi.org/10.1145/3138808.3138812

[160] 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.

[161] ——, "ndnSIM 2: An updated NDN simulator for NS-3," NDN, Technical Report NDN-0028, Revision 2, November 2016.

[162] ndnsim mobile simulation package. [Online]. Available: https://github.com/4th-ndn-hackathon/ndnSIM-Mobile-Simulation-Package

[163] T. Liang, J. Pan, M. A. Rahman, J. Shi, D. Pesavento, A. Afanasyev, and B. Zhang, "Enabling named data networking forwarder to work out-of-the-box at edge networks," in *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2020, pp. 1–6.

[164] Y. Elkhatib, M. Mu, and N. Race, "Dataset on Usage of a Live & VoD P2P IPTV Service," in *Proceedings of the IEEE International Conference on Peer-to-Peer Computing*, 2014.

[165] (2018, October) Ndn testbed - named data networking. [Online]. Available: https://named-data.net/ndn-testbed

[166] W. A. Simpson, D. T. Narten, E. Nordmark, and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)," RFC 4861, 2007. [Online]. Available: https://rfc-editor.org/rfc/rfc4861.txt

[167] J. Kempf, J. Arkko, B. Zill, and P. Nikander, "SEcure Neighbor Discovery (SEND)," RFC 3971, 2005. [Online]. Available: https://rfc-editor.org/rfc/rfc3971.txt

[168] S. M. Kerner, "Mobile Internet Traffic Growing Fast," http://www.enterprisenetworkingplanet.com/netsp/mobile-internet-traffic-growing-fast.html, Febuary 2019.

[169] G. Tyson, N. Sastry, R. Cuevas, I. Rimac, and A. Mauthe, "A survey of mobility in information-centric networks," *Commun. ACM*, vol. 56, no. 12, pp. 90–98, December 2013. [Online]. Available: http://doi.acm.org/10.1145/2500501

[170] G. Pravin, K. Girish, and D. Ghorpade, "Mobile ad hoc networking : Imperatives and challenges," *International Journal of Computer Applications*, January 2010.

[171] Q. Xu, T. Mak, J. Ko, and R. Sengupta, "Vehicle-to-vehicle safety messaging in dsrc," in *Proceedings of the 1st ACM International Workshop on Vehicular Ad Hoc Networks*, ser. VANET '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 19–28. [Online]. Available: https://doi.org/10.1145/1023875.1023879

[172] J. Li, H. Wu, B. Liu, J. Lu, Y. Wang, X. Wang, Y. Zhang, and L. Dong, "Popularity-driven coordinated caching in named data networking," in *2012 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, October 2012, pp. 15–26.