Interest Forwarding Strategy in Named *Data* Networks using Thompson Sampling

Nazma Akther

Department of Computer Science and Engineering

United International University

A thesis submitted for the degree of

MSc in Computer Science & Engineering
United International University
Dhaka, Bangladesh

December 2022

Declaration

I, Nazma Akther, declare that this thesis titled,'***Interest* Forwarding Strategy in Named *Data* Networks using Thompson Sampling'** is the outcome of the research carried out by me under the supervision of Dr. Shahid Md. Asif Iqbal and co-supervision of Professor Dr. Mohammad Nurul Huda. I confirm that:

- This work was done wholly or mainly while in candidature for a MSc degree at United International University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at United International University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Nazma Akther
ID: 012203030
Department of Computer Science and Engineering,
United International University, Dhaka-1209, Bangladesh.

Certificate

I do hereby declare that the research works embodied in this thesis entitled **'*Interest* Forwarding Strategy in Named *Data* Networks using Thompson Sampling'** is the outcome of an original work carried out by **Nazma Akther Student ID: 012203030** under our supervision.

I further certify that the dissertation meets the requirements and the standard for the degree of MSc in Computer Science and Engineering to be awarded by United International University (UIU).

1. **Supervisor:**

   Signed: _____

   Dr. Shahid Md. Asif Iqbal
   Associate Professor
   Department of Computer Science and Engineering,
   Premier University, Chittagong, Bangladesh.

2. **Co-Supervisor:**

   Signed: _____

   Dr. Mohammad Nurul Huda
   Professor & Director MSCSE
   Department of Computer Science and Engineering,
   United International University, Dhaka-1209, Bangladesh.

Abstract

Optimizing *Interest* forwarding and *Data* delivery has been among the top dissected problems in *NDN* for the last decade; however, only a few contributions thrive to minimize communication cost and delay concurrently. In *NDN*, a receiver-driven forwarding strategy is considered resource-consuming as the routers incur computation to find the best path to the desired item, specified by an *Interest*'s name. On the other hand, a source-driven forwarding strategy, a scheme that suppresses the sub-optimal sources, experiences increased delay when no source answers in the exploration phase. The confluence of the two strategies can counteract the drawbacks of each one, which, however, has never been investigated. In this work, a reinforcement learning-based, namely Thompson Sampling, strategy is proposed that operates in a receiver-cum source driven fashion to optimize *Interest* forwarding and answering. The proposed method introduces a '*Beam*' concept coupled with adaptive scoped-flooding to optimize *Interest* forwarding, and the sources adopt Thompson Sampling to suppress the sub-optimal responses. When hit by an *Interest*, an optimal source sends back the desired *Data* to the consumer whereas a sub-optimal source remains *Silent*. Together, the '*Beam*' and the scoped-flooding adapt the *Interest* forwarding range based on cache hit/miss ratio. The adaptation optimizes communication cost and delay, and contributes to scheming the proposed strategy resource-savvy. The proof-of-concept implementation in software (simulation) reveals that the proposed system outperforms the counterpart benchmarks by reducing the communication costs and delay in *NDN* (by around 350% and 10%, respectively) without negotiating packet delivery ratio.

Published Papers

The outcome of this research work is recognized in the following esteemed journals:

1. Nazma Akther, Kingshuk Dhar, Shahid Md Asif Iqbal, Mohammed Nurul Huda, and Asaduzzaman, '*Interest* forwarding strategy in Named *Data* Networks (NDN) using Thompson Sampling', Journal of Network and Computer Applications, Elsevier, Vol. 205, 2022, pp. 103458.

Acknowledgements

First and foremost, I'm grateful to Allah (SWT), the Most Gracious and Powerful, for giving me the wisdom, talent, ability, and health I need to pursue an MSc degree in Computer Science and Engineering. It is because of his generosity that I am capable of successfully completing my thesis.

After that, I would like to express my sincere gratitude to Dr. Shahid Md. Asif Iqbal for being the most courteous, helpful, and kind supervisor I could ever have. It has been a long and challenging process, but I am fortunate to work with such a mentor, advisor, and wonderful person. His priceless advice and research ideas gave me the excellent opportunity to do this incredible research work on the topic of Future Internet Architecture called Named *Data* Networking (*NDN*), where NS3-based ndn Simulator technology is new to adopt. I also want to express my gratitude to Dr. Mohammad Nurul Huda for giving me the freedom to pursue my own ideas while I was an MSc student. Furthermore, without his support and inspiration, the journey toward the MSc degree in CSE would have been much more challenging. Again, I am very thankful to them because their reviews and corrections helped me publish my work and complete this outstanding thesis.

I am grateful to Premier University for allowing me to pursue my MSc degree in Computer Science and Engineering as a part-time student. That has momentously aided my development as an experienced researcher and knowledgeable instructor.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Evolution of Named Data Networking

The Internet was developed so that a group of machines or hosts could exchange data [1]. In this host-centric communication model, each connection requires a pair of computers, notably the source and destination [2]. The Internet's enormous range of offerings and versatility has allowed it to enter and reinvent every facet of our daily lives, including banks, travel, finance, governance, education, communication, and so on. Cisco *VNI* [3] predicts that by 2023, more than twenty-nine (29) billion Internet-connected devices, five billion active Internet users, and 300 billion social networking, gaming, and business applications will be downloaded globally onto smartphones. The nature of the content exchanged (downloaded) and the phenomenal expansion of the Internet trigger a significant shift in user attitudes. Security, legitimacy, portability, ubiquity, and other unexpected architectural constraints have arisen due to changes in Internet users' attitudes and the diversity of new applications. Easy access to the Internet and the abundance of content on YouTube, Amazon, Netflix, iTunes, etc., as well as constant demand for these resources, gradually transformed the Internet into a content distribution and retrieval network.

The TCP/IP Internet was initially architected to enable resource sharing between communicating parties in a network. Interconnecting a small group of stationery computers and forwarding user data packets among those devices were

the fundamental requirements in that network [4, 5]. The last decade has endured the evolution and endless versatility of the e-commerce market, digital contents, social platform, cloud storage, smart devices, and hence an exponentially stacking digital information (contents). Content delivery networks (*CDN*) and peer-to-peer (*P2P*) networking are introduced to accelerate the distribution efficiency of such volume; however, such solutions are inappropriate with respect to futuristic applications demand [6], and limited and complex because of the underlying host-centric communication [7]. Peoples' attitude toward the Internet in the same period necessitates transforming the Internet's role from a communication network to a distribution network [8]. Thus, there is a pressing need for the modification of the legacy Internet to align with this radical shift. Information-centric networking (*ICN*) is a promising and clean-state networking paradigm proposed by Van Jacobson in 2006 for architecting the future Internet. *ICN* is based on a content-centric communication paradigm, where each content or chunk is directly addressable, which significantly deviates from the traditional server- or host-centric model. Numerous related solutions, including *DONA* [9], *NetInf* [10], *PSIRP* [11], Content-Centric Networking (*CCN*) [12], *COMET* [13], and others, have been put forth in recent years under this broad *ICN* heading. Among the *ICN* proposals [8, 14, 15], Named *Data* Networking is highly structured and has a broad community [16]. The proposed research is designed in *NDN* tenet.

1.1.1 Content Retrival Process of *NDN* Model

Emphasizing consumers' concern about specific content, *NDN* offers to move the conventional host-centric communication practices into content-centric communication. In general, *NDN* content is segmented into several small, equal-sized chunks. A chunk name adheres to a hierarchical naming structure that contains the content attributes (e.g., a specific video content using a naming pattern like youtube/music/Rock/Jishan/11.

In *NDN* tenet, each router is equipped with a stateful and intelligent forwarding plane upholding the two core principles of *NDN*: forwarding and on-path caching. Together these principles enable content mapping using hierarchical

names and shorten the distance and time of content retrieving. Thanks to content replicas cached or replicated at numerous nearby locations (routers). Any *Data* item is usually divided into several small, equal-sized chunks, requested and retrieved from the provider or identical cached copies independently of its location [8].

An *NDN* router consists of three exclusive data structures: Forwarding Information Base (*FIB*), Pending *Interest* Table (*PIT*), and Content Store (*CS*). Moreover, two distinct types of *NDN* packets, namely *Interest* and *Data*, are commuted for data communication. An *Interest* packet is used to solicit a content while a *Data* packet carries the solicited content. Any item (chunk) requesting *Interest* is uniquely labelled with a name and a random nonce value generated by the requesting application. At the same time, a *Data* packet is signed with a digital signature by the content provider, binding the name with the payload. The Forwarding Information Base (*FIB*) table includes name prefixes and paths to such prefixes that will be searched in turn to forward an *Interest* packet. The forwarding strategy pre-built in each router is responsible for making forwarding decisions. A Pending *Interest* Table (*PIT*) entry tracks the forwarded (and unanswered) *Interest*s and waits for the corresponding item to return. The waiting table aggregates *Interest* forwarding and is used as bread-crumb trails to deliver the returned *Data* packet. Since the nonce value uniquely identifies an *Interest*, the *PIT* can detect and stop *Interest* looping. Requesting and retrieving a content in *NDN* is depicted in Fig. 1.1. By its name, a Content Store (*CS*) holds replicas temporarily to entertain future requests early [8]. *NDN* communi-

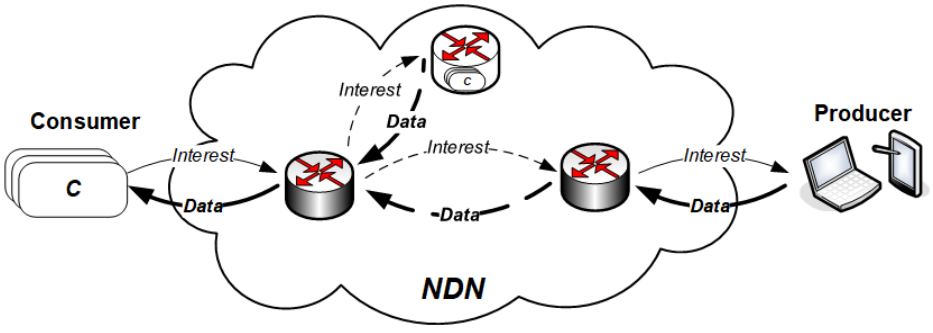


Figure 1.1: Content retrieval process in *NDN*

cation is receiver-driven triggered by the consumers. A consumer disseminates an

Interest requesting the desired content. The intermediate routers receiving the *Interest* use the forwarding strategy module to forward it further until it hits a corresponding content source. A router mainly uses the flooding, or single-route scheme [17] for *Interest* forwarding. However, the content is sent back to the requester along the reverse path, the *PIT* bread-crumb trails. Due to the apparent flaws of the original schemes, numerous schemes have been proposed as described in Chapter 2.

1.1.2 Two-Armed Bernoulli Bandit ($TABB$)

The ($TABB$) [18, 19] is a classical optimization problem where an agent asked to pull one of the two arms is awarded a payout or reward for each action (pull). A payout is considered either a success (1) or a failure (0). When played, an action produces a success payout with a certain probability. The success probabilities, *a.k.a* mean rewards, are unknown but fixed over time and thus can be learned by experimentation. An arm is regarded as a Beta-Bernoulli Bandit if the agent starts with an independent prior belief over the arm's success probability, which is beta-distributed. As observations are accumulated, the distribution is updated, resulting in a posterior distribution which is beta too. The agent must balance between exploration and exploitation to obtain new information and utilize current knowledge, respectively. The objective of learning here is to maximize the cumulative number of successes over a given period. TAB is a sub-class of the renowned Multi-Armed Bandit (MAB) problem.

1.2 Possible Implications

The primary issue that *NDN* will face is the exaggerated amount of data traffic. This is because data-intensive content, like streaming video on YouTube, is so prevalent and there are so many people connected to the Internet. The prominence of such data-intensive traffic, which is often delay- or loss-sensitive, is expected to grow exponentially in the coming days, according to [3]. The proposed forwarding scheme is able to address such constraints effectively by downloading loss-sensitive and delay-sensitive contents such as live streaming, VoIP,

multimedia teleconferencing, and interactive contents, especially in a resource-constrained environment. *Silent-Reply* Armed Beta Bernoulli Bandit ($SRAB^3$) enables content retrieving through non-*Beam* threads in addition to the *Beam* thread, making it resilient to packet loss. Moreover, it leaves the network resource, for example, bandwidth and cache space, by suppressing the sub-optimal *Data* packets, enabling faster content delivery.

1.3 Motivation

NDN forwarding strategies promote optimizing the *Interest* forwarding in order to increase network performance, including overhead and delay [20],[21], [22–30]. Since the size of a *Data* is the primary contributor to transmission overhead and delay, reducing redundant content can significantly improve network performance. The previous results, namely *Data* reply Two-Armed Bernoulli Bandit ($DTABB$) [31], inspire the present work. However, there are basic differences between ($SRAB^3$) and $DTABB$. $DTABB$ suppresses sub-optimal responses at the cost of sending payload-empty or payload-slice *Offer* packets. In contrast, a source can remain mute in $SRAB^3$ while suppressing sub-optimal responses. $DTABB$ often suffers exploration-exploitation phases, which essentially doubles the content download time, while $SRAB^3$ ensures content retrieval at the first phase. The *Offer* packet concept is entirely vanished here, resulting in a relatively reduced communication cost (overhead). Moreover, the *Interest* forwarding is optimized by using a hybrid scoped-flooding.

Optimal *Interest* answering (reciprocating) is formulated as a Two Armed Beta-Bernoulli Bandit (TAB^3) problem here to minimize the communication cost and retrieval delay by suppressing the sub-optimal responses.

1.4 Problem Statement

.Numerous solutions have been urged in the literature to optimize (control) *Interest* forwarding and *Data* delivery, and both have been among the top dissected problems in *NDN*. A group of receiver-driven strategies exploit the optimal path

but probe the available options periodically or event-based to discover a better path; however, only at the cost of compute-intensive policies and enormous control packets [32]. Another group, for example *iNRR* [33], *Discovery* [34], source-driven strategies (*DTABB* [31]), and [20, 35], optimizes the *Data* packets to minimize transmission overhead arguing that a content's size is exceedingly high when compared to an *Interest*. Scoped-flooding [36, 37] is also proposed to reduce communication overhead. However, the strategies, for example source-driven approaches, experience extended delay due to the occasionally encountered exploration-exploitation cycles. To that end, this research proposes a forwarding strategy that optimizes the *Interest* forwarding and content delivery from a receiver-cum-source's viewpoint. Thus, the research questions answered, in this thesis as follows:

- *Q*: How to formulate a source-driven machine learning-based *Interest* answering optimization, equivalently *Data* replying, strategy to retrieve a content from the optimal source?

1.5 Objectives

This research formulates the *Interest* answering strategy by a content source as a Two-Armed Beta-Bernoulli Bandit model, denoted as $SRAB^3$. The model considers a content source as the agent. An agent is asked to react with one of the two actions, namely *Silent* or *Reply*, when the incoming *Interest*s are presented one by one to it. The two options are the two arms of the agent. An agent selects *Reply* if the consumer uses the returned *Data*; otherwise, it chooses *Silent*. The success probabilities of *Silent* and *Reply* are beta distributed. Initially, the source is ignorant about the outcomes of the answers. The efficacy of a reaction is determined by responding to an *Interest* and inspecting the result. An agent uses the outcomes of previous actions to resolve the reaction type for the following *Interest*s. The learning in $SRAB^3$ aims to optimize the communication cost while preserving the content download delay in *NDN*.

Thompson Sampling (*TS*) [38–40], a reinforcement learning algorithm from *MAB* literature, is adopted in the proposed research to solve the *Interest* answer-

ing problem. TS algorithm, specialized in the cases of Beta-Bernoulli Bandit, can effectively resolve the $SRAB^3$ dilemma by enabling a source to learn the optimal response type with only a few *Interest*s reciprocated with the sub-optimal reactions. The proposed TAB^3 problem differs from the *DTABB* [31] in the learning process as it introduces a novel optimization of *Interest* forwarding.

1.6 Contribution

In this paper, we design a novel *Interest* forwarding and answering scheme in *NDN* architecture to select the best path (source) considering both viewpoints, namely receiver, and source, to reduce communication cost and delay. The proposed *Interest* answering scheme learns the optimal response type utilizing Thompson Sampling. The *Interest* forwarding scheme allows a thread of each issued *Interest*, called a *Beam*, to propagate to entire network until it hits a corresponding source, while the rest of the threads are controlled by scoped-flooding.

The key contributions of this study can be summarized as follows:

- A novel hybrid forwarding strategy, namely $SRAB^3$, couples scoped-flooding and the novel *Beam* concept to optimize *Interest* forwarding.
- The matching *Data* is fetched considering receiver-cum-source viewpoint where the sources embrace Thompson Sampling to learn the optimal *Interest* answering. The learning is performed only for the cached items. This research is the first to realize Thompson Sampling for *Interest* answering in *NDN* to the best of the knowledge.
- $SRAB^3$ is realized in the famous ndnSIM simulator [41]. It is contrasted against *DTABB* [31], *Multicast* [41] and *BestRoute* [41] in terms of transmission overhead, latency, cache hit ratio, cache eviction rate, and combined significance varying content popularity, payload size, and content catalogue size.

1.7 Organization of the Thesis

The remaining portion of this thesis is organized as follows. The second chapter begins with a brief overview of notable *ICN* projects. Since the focus of our research contributions is the *NDN* tenet, it also provides a comprehensive overview of the *NDN* architecture. Later in the chapter, the related literature on forwarding strategies in *NDN* is briefly described.

The Chapter 3 describes the preliminaries and the proposed $SRAB^3$ Forwarding Strategy. This chapter demonstrate the methodology regarding *Interest* Forwarding and Answering in *NDN*. Moreover, chapter 3 represents a prominent reinforcement learning algorithm called Thompson Sampling for *Data* response (*Silent* or *Reply*) strategies in *NDN*.

The chapter 4 describes the simulation execution in the famous ndnSIM simulator. Furthermore, performance analysis are demonstrated based on transmission overhead, latency, cache hit ratio, cache eviction rate, and combined significance.

This thesis reaches its conclusion in Chapter 5, which summarizes our contributions and identifies the constraints and future research needs for forwarding in *NDN* and *SDN*-based *NDN* architectures.

Chapter 2

Background and Related Work

2.1 Background

Information-Centric Networking has developed a new communication approach by concentrating on content attributes rather than the precise location of host-centric Internet communication. By prioritizing "what" content is rather than "where" it comes from, *ICN* aims to update the traditional host-centric Internet architecture. With the help of the content-centric approach, a piece of content can become independent of the physical network location, the mode of transportation, the type of storage, and the programs running at the application layer. *ICN* is highly resilient in demanding and dynamic network scenarios thanks to its content-focused design, which also offers better bandwidth demand scalability and increases content retrieval efficiency [42].

This chapter discusses the well-known future Internet architectures available in the literature, emphasizing *Named Data Networking* (*NDN*), where our *Interest* forwarding strategy is proposed and implemented.

2.2 *ICN* Fundamentals

ICN architectures utilize in-network storage for caching and multiparty communication via replication. In 1999, the *ICN* paradigm was launched with the *TRIAD* as the pioneer architecture project [1]. Since then several promising ar-

chitectures, including *DONA* [9], *PURSUIT*(*PSIRP*) [11], *NetInf* [10], Content-Centric Networking (*CCN*) [12],*COMET* [13],*SAIL* [43],*CONVERGENCE* [44], *NDN* [45, 46] and many others, have been discussed in the context of specific projects but leveraging the same paradigm. This section provides a brief summary of important *ICN* proposals.

2.2.1 Data-Oriented Network Architecture (*DONA*)

The methodology used by *DONA* is information routing at the elevated amounts of inter domain architecture. It involves a complete overhaul of Internet naming and name resolution. As a result, Resolution Handlers (RHs), a layer focused on information, are created above the Internet. The names of *DONA* are arranged according to principals. All services, as well as other named entities (hosts, domains, etc.) are affiliated with a principal, coupled with a public-private key pair. A principal enters the names into its local Resource Handler (RH). A content router known as a RH is connected to other RHs hierarchically to provide name resolution service. All RHs are updated with the location of a specific content object as a result of the network propagation of REGISTER messages from the producers. To retrieve specific information, interested users send a FIND message to the connected RH. The RH directs the appropriate resource handler to handle the FIND request. The content object is then sent back to the requesting client via the reverse path of the FIND message or directly via a different route.

The RH network in *DONA* is used to implement on-path caching. When a RH chooses to cache an item, it front-forwards the message to the succeeding RH and changes the FIND message's source network address to its own self-network address. It has the opportunity to accept the Data in its cache because the requested content is now being returned through the current RH.

2.2.2 The Publish-Subscribe Internet Routing Paradigm(*PSIRP*)

The *PSIRP* project aims to create and assess an information-centric architecture for the future Internet. The goal is to develop a new type of internetworking that

supports innovative applications, new business opportunities, and availability, security, and mobility in addition to providing the functionality, flexibility, and performance that are desired. This is a collaboration of two projects: the *PSIRP* and *PURSUIT* internet routing paradigms. As a continuation of the *PSIRP* [11, 47, 48],the *PURSUIT* [49] is an EU FP7 funded project that was started in the year 2010. The publish-subscribe protocol stack replaces the traditional send-receive-based Internet communication in *PURSUIT*.

Publishers serve content objects to the PSI network in the *PURSUIT (PSIRP)* architecture. Subscribers act as consumers who express their interest in a specific item by sending subscription request messages. The Rendezvous Point (RP) is a critical component in the architecture, as it accomplishes the mapping between a subscriber interest and the respective content item. *PURSUIT* names, like *DONA*, are flat and identified by a set of attributes (scope ID, rendezvous ID). The rendezvous part of a name represents the specific information piece , whereas the scope part resembles the availability or reachability of a content object. The scope of a content, i.e. authorization, access rights, reachability, availability, and so on, can be physical, such as only available at a specific campus location, or logical, such as only available to a specific group of users. Subscription requests for unpublished content can be held by RPs for a specified period of time. This subscription option decouples the subscription and publication functions spatially, and clients can issue subscription curiosity for data items that have yet to appear in the network.

2.2.3 Network of Information (*NetInf*)

In this architecture, name-based routing and name resolution are two different methods for retrieving a content, also known as a Named Data Object (*NDO*). As the first order networking primitive, named data objects (*NDO*s) are accessible through the networking strategy known as NetInf. The transmission of (*NDO*) requests and the corresponding objects is the main function of NetInf nodes. In order to PUBLISH, GET, and SEARCH for (*NDO*s), the message-based NetInf protocol provides both requests and responses. Using the PUBLISH message, a source announces the contents' availability according to the model set up in

the local network to ensure that the routing protocols can reveal the object's routing information or perhaps the name resolution service (NRS) can register the proper name-location binding. The name-based routing method is fairly simple. According to the routing protocol, the request (GET message) for the desired content is sent straight to its source locations. Using name-based routing as well, the contents are transmitted from the sources to the requester end.

The name resolution approach, on the other hand, prompts that the user explore the available finders of the named object by communicating their interest (a SEARCH message) to the NRS. After that, NRS provides a list of finders that the user can use to find the artifact from the leading source. A smooth transition between the conventional location-dependent Internet architecture and a completely name-based Internet architecture is made possible by the new *NetInf* model. It is possible to implement *NetInf* by launching the two models separately or by combining them into a hybrid model that allows for hop-by-hop swapping between the two models [2].

Contrary to earlier ICN proposals, *NetInf* supports caching of both request messages and objects. Once a GET message reaches a node, the node can decide whether to perform a per-request NRS lookup or perform request aggregation using a waiting table. On the other side, a registered copy of a data object may be directly retrieved from the NRS or may be explored using other methods to duplicate the object in a node. Additionally, a copy may be requested using a cache-aware *NetInf* transport protocol while traveling to a location where copies are known to be kept [50].

2.2.4 Content-Centric Networking (*CCN*)

In a *CCN* architecture, a name refers to a data object. A client who wants a specific object can request it by sending an *Interest* message with the name of the data item they want. The client broadcasts the *Interest* message to the network, which is routed and forwarded until it reaches a node that holds a replica of the object whose name relates to the *Interest*. The source node, which is the one holding the replica, creates a *Data* packet carrying the replica of the desired item and sends it back to the requester. To get to the client, the *Data* packet follows

the footprint of the *Interest*. An on-path intermediate router may keep a copy of the *Data* as the packet is disseminated back to the client in order to satisfy subsequent requests *(Interests)* from other clients for exactly the same *Data*.

The Content-Centric Networking (*CCN*) project, which Van Jacobson first presented in front of the public in 2006, served as the inspiration for *NDN* [45, 46]. It is a part of a significant collaborative research project carried out by various academic and industrial institutions. However, both designs share a few fundamental concepts, such as retrieving content based on content name, having an *Interest* specify a request, and delivering content via a *Data* packet. Despite the fact that the *NDN* architecture and *CCN* architecture initially conveyed typical design patterns, the *NDN* architecture has now differentiated from the *CCN* architecture and is developing in a distinct direction. For example, when forwarding *Interest* to sources and searching for matching content in caches, an *NDN* router looks for the longest prefix match [51]. In contrast, to make the forwarding decisions convenient, *CCN* accomplishes route lookup using precise matching. *CCN* and *NDN* are highlighted among *ICN* proposals in the literature due to their broad and enthusiastic research community and open-source availability.

2.2.5 Named Data Networking (*NDN*)

NDN is a completely new network architecture, but its design principles are focused primarily on the excellence of today's Internet. One of the five initiatives bankrolled by the Future Internet Architecture (*FIA*) Program of the American National Science Foundation (*NSF*) is Named Data Networking (*NDN*)[45, 46]. The goal of the *NDN* project is to explore Jacobson's ground-breaking suggestion of a radical switch from the traditional host-centric network architecture (IP) to a content-centric network architecture (*NDN*). The hourglass design of the TCP/IP Internet has a thin waist at the network (IP) layer, where only the most basic functionality is required to establish universal interconnection.The narrow waist concept was a significant factor in the Internet's explosive growth as a communication network, which decoupled the development of lower and upper-layer technologies. Nevertheless, IP was invented to establish an end-to-end

communication link in which a packet is named and routed by the IP addresses of the communication endpoints.

Currently, the Internet plays a dominant or significant role in acting as a distribution network that is more general than its counterpart. Addressing distribution challenges through a point-to-point communication network is complicated and error-prone. The *NDN* endeavors to make the slim waist's role more ubiquitous, so that packets can always be labeled and routed by an object's name rather than communication endpoints.The hourglass shape of the *NDN* architecture is preserved, but the thin waist is changed to concentrate on the data itself rather than where it is. To be more precise, *NDN* modifies the semantics of network communication from sending a packet to a specific destination node to retrieving data recognized by a particular name. (Fig. 2.1).

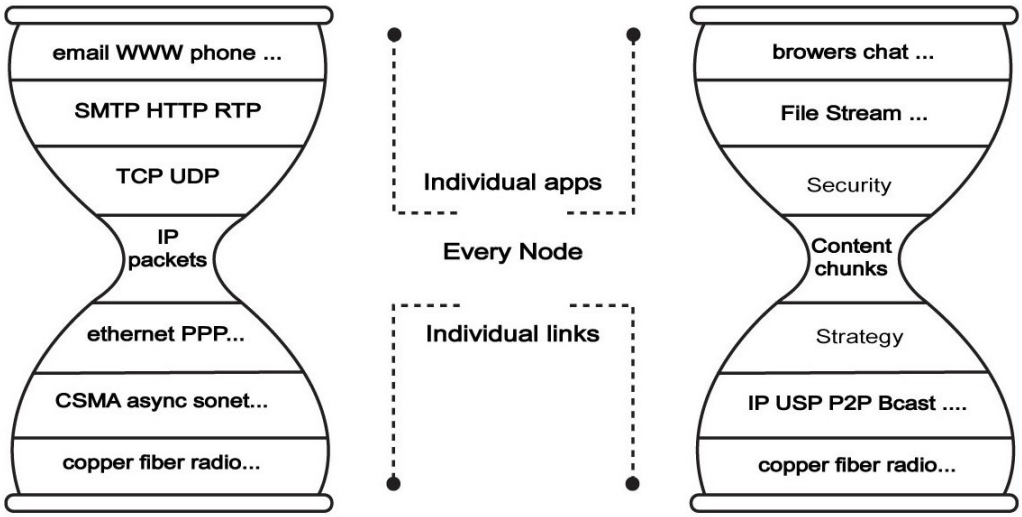


Figure 2.1: *IP* and *NDN* hourglass architectures

2.3 *NDN* Architecture Overview

NDN is a network-layer communication protocol with a receiver-driven architecture. The *NDN* architecture, similar to IP, mainly focuses on the thin waist. Since *NDN* uses data names rather than IP addresses to deliver and retrieve

content, it offers a new set of minimal functionalities. The new set introduces significant differences in IP and *NDN* data delivery operations. This section introduces the various concepts and aspects used in *NDN* and their roles in the overall architecture.

2.3.1 *NDN* Packets

A piece of data is recognized and retrieved using its given name in accordance with the semantics of the *NDN* services. In *NDN*, any communication between two entities is carried out as receiver-driven, and the name can refer to any network entity. There are primarily two packet types used for *NDN* communications: *Interest* and *Data*. Figure 2.2 depicts the packet formats of the two unique packet types. Each type of packet carries a name that designates the specific

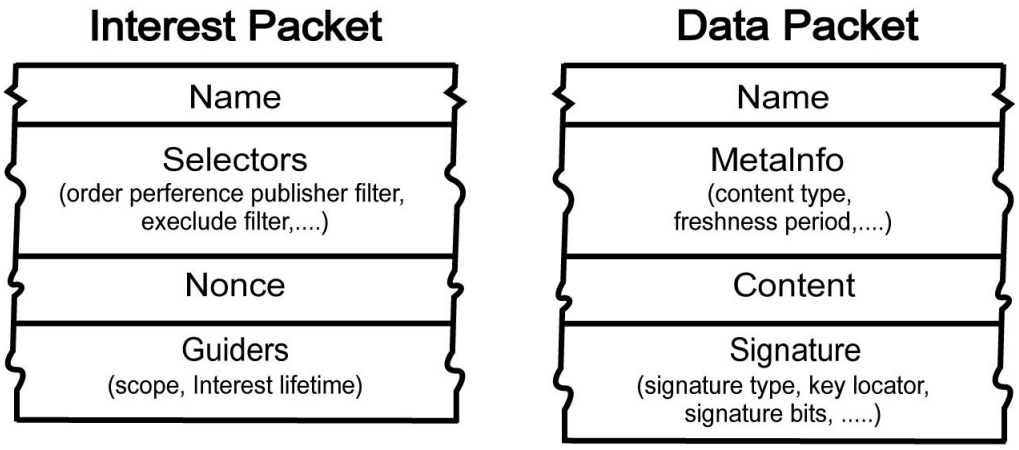


Figure 2.2: Packets in *NDN* architecture

data item that can be transmitted in the single *Data* packet. A consumer must send an *Interest* packet to the network with the name of the data they want to receive. A consumer issues an *Interest* packet, a tiny envelope resembling the TCP acknowledgement messages, to request the desired item. The consumer generates an *Interest* packet and transmits it over the network after inserting the preferred data name into it. The name is the only required component of an *Interest* packet, but it also contains other crucial fields that help with *Interest* matching and forwarding. An *Interest* message can only be identified by its name

and the nonce field, a value that is generated at random. Detecting and avoiding *Interest* looping is the main goal of having a nonce value. The selector field describes the desired data more precisely. The lifetime and scope fields indicate how much time remains before the *Interest*is timed out and dropped.

Once the corresponding *Interest* occurs, a network node holding a copy of the desired item can produce the *Data* packet. The node may function as either a producer or a caching router. To respond to the corresponding requesting *Interest*, the source node produces and transmits a *Data* packet. An NDN router can cache a *Data* packet to satisfy future requests. The *Data* packet includes the name, some arbitrary data, some optional information (like meta-data), and the cryptographic signature that links the content to the name and secures the data. In order to facilitate packet processing, the name is the first field in a *Data* packet. The signature is placed at the end of the packet because the computation of the cryptographic signature encompasses all preceding elements.

2.3.2 *NDN* Naming

The routers just use*NDN* data names in an Interest to direct the request to probable content sources. *NDN* names are opaque to the network, which means that routers are only aware of the boundaries between the components of a name but not its meaning. Because of this, naming schemes can develop independently of the network and each application is able to select the naming scheme that best suits its requirements. Each distinct piece of a data object contained within a single *Data* packet [52] can be identified by a name. Each *NDN* name is made up of multiple constituent elements that are organized hierarchically. A name component is typically an arbitrary-length string (human-readable textual representation). The delimiter "/" both delimits and concatenates the name constituents. A name component's hierarchy is also delineated by the same delimiter. In Fig. 2.3, the current binary encoding and application-layer conventions for content naming are displayed. The version maker is encoded as RS pursued by a numerical version number in the textual representation v. The segmentation marker is simultaneously set as 00, followed by an integer number that could represent a frame number, block number, or byte number for any particular segment of the

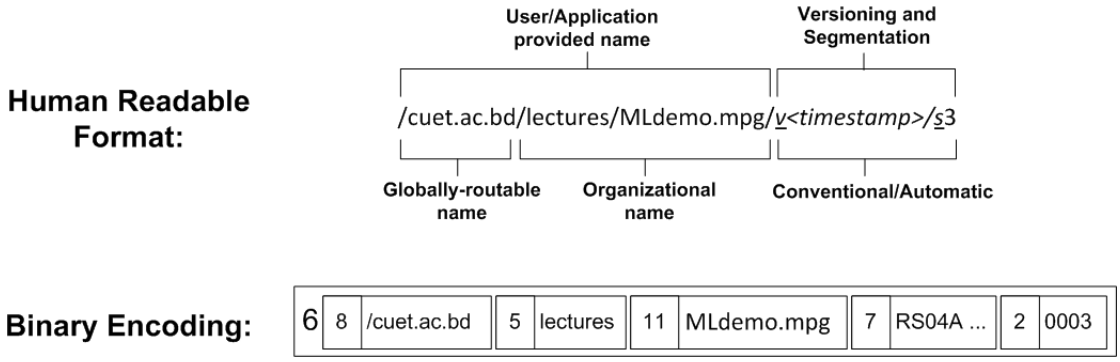


Figure 2.3: Example *NDN* name format, from [52].

video content[52]. Furthermore, the application can incorporate any contextual information and data element relationships into the names.

Any application can define the naming format that best suits its needs thanks to the hierarchical naming structure, which also allows naming schemes to develop on their own. The application can also include any contextual information and relationships between the data elements in the names. The autonomous content hosting system is represented by the hierarchical name tree in Fig. 2.4.

To fetch any arbitrary data, consumers must really be able to generate the name of the required data chunk without prior knowledge of the content's name. Based on shared information, the producer and consumer can use deterministic algorithms to generate the same name. Otherwise, using *Interest* selectors in conjunction with the longest prefix matching, one or more rounds can be used to retrieve the desired piece [2]. According to Zhang *et* al. [46], retrieving a data packet only requires a minimal set of selectors and a minimal amount of naming information. The hierarchical tree shown in Fig. 2.4 represents a portion of the name tree associated with the convention shown in Fig. 2.3. For example, a student seeking the foremost version of the MLdemo.mpg. video lecture from CUET could request /cuet.ac.bd/lectures/MLdemo.mpg/v1 with the *Interest* selector set to "leftmost child" and receive a Data packet named /cuet.ac.bd/lectures/MLdemo.mpg/v1/s1 identifying the first segment. The same client can make subsequent requests that incorporate details from the first *Data* packet and the naming scheme used by the publishing application [46].

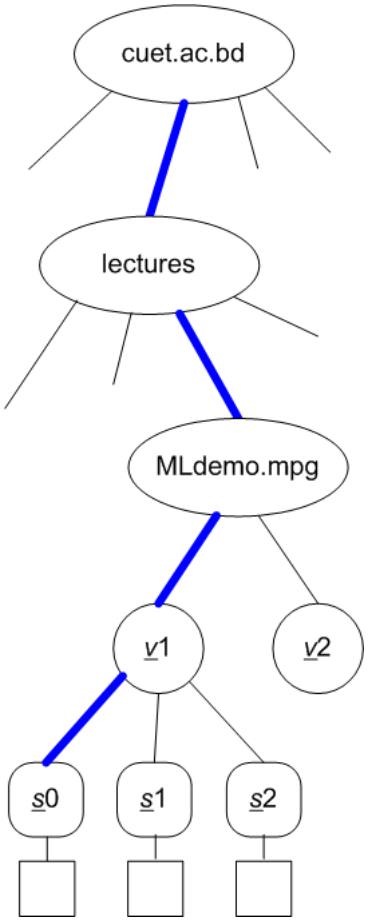


Figure 2.4: *NDN* node data structure, from [52].

In order to retrieve data globally, only names that are globally unique are required. Aside from these names, other names do not need to be globally unique. Local communication names are significant in the local context and involve only constrained routing (or local broadcast) to locate the matching Data. Ultimately, Data naming is the most important component of the *NDN* paradigm. Thanks to it, *NDN* can support a variety of network functionalities, including the simultaneous multicasting of the same *Data* to multiple users, the distribution of the same content at various times, user mobility, and content retrieval even in the presence of intermittent connectivity (delay-tolerant networking).

2.3.3 *NDN* Data Structures

Each *NDN* router maintains three major data structures to bring out the aforementioned *Interest* and *Data* packet: the Content Store (*CS*), the Pending *Interest* Table (*PIT*), and the Forwarding Information Base (*FIB*) [53–55]. The three structures are critical components of the *NDN* forwarding engine. Figure 2.5 depicts the data structures and forwarding engine.

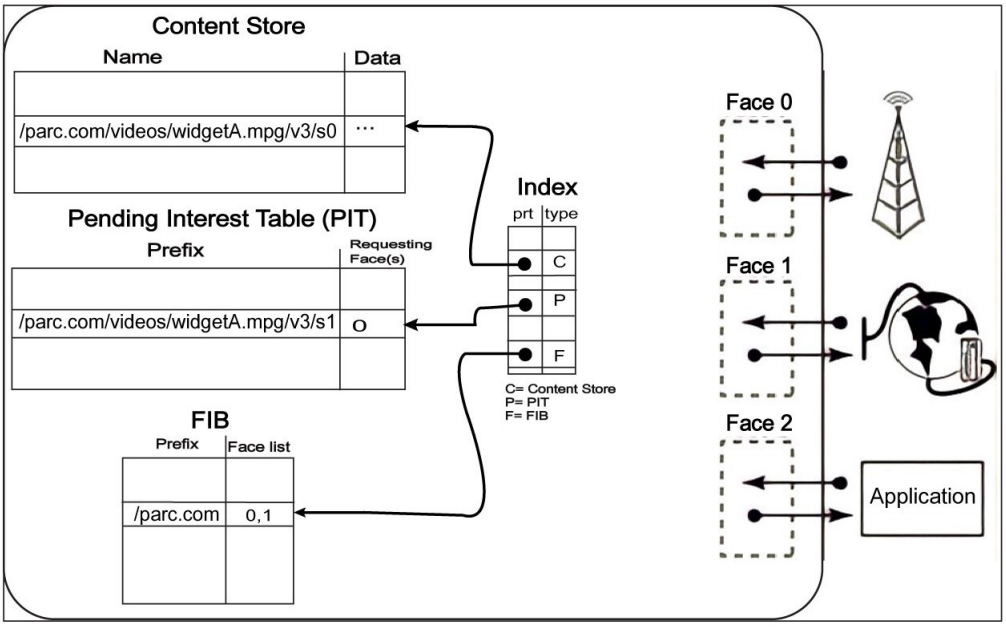


Figure 2.5: *NDN* forwarding engine model, from [52].

2.3.4 Content Store (*CS*)

The Content Store of a node is the location where data packets are temporarily stored before being forwarded to the next node to process incoming requests. In an *NDN* node, the only component that permits in-network caching is a CS. Typically, the CSs can only preserve a tiny portion of the enormous amount of content that is available. The content name is used to index and look up an entry in the CS table.

2.3.5 Pending *Interest* Table (*PIT*)

A user can request and retrieve content pseudonymously via *NDN*, which means that neither a user address nor a name is stamped on the *Interest* packet to identify or discover the requesting consumer. Instead, the consumer receives a response to the returned *Data* packet by following the routers' Pending Interest Table (*PIT*) trails. When an *Interests* is forwarded and waiting for the response (*Data*), it is recorded in a *PIT* entry. A router continuously monitors incoming interfaces for pending or dissatisfied Interests, which are attempted to push forward to reach potential data source(s) but have yet to be satisfied. When matching Data is received, these records are used to send the *Data* packet to the requesting consumer (s). The *Data* name is used to index an entry in this waiting table. A unique *PIT* entry is created for each requested name. A forwarded Interest's name, list of incoming interfaces, and list of outgoing interfaces are all recorded in each entry, which is a three-element tuple. A *PIT* entry also includes a list of nonce values that have been discovered for that name. Because of these tuples, the *PIT* becomes stateful, allowing similar *Interests* to be forwarded only once.

For future reference, an incoming interface will always keep track of the longest *Interest* lifetime it has received, and will be removed from the *PIT* entry once that lifetime has expired. An outgoing interface, on the other hand, records the precise moment that the *Interest* is forwarded through this interface so that, once the relevant *Data* packet arrives, Round Trip Time (RTT) can be anticipated for the relevant name prefix stored in the FIB [56, 57]. The forwarding state in *PIT* is depicted in Figure 2.6. The data plane effectively becomes a stateful component by holding onto the *PIT* entry until its fate is decided. This allows *NDN*'s data plane to be responsive in the event of network outages and efficient in utilizing network resources.

2.3.6 Forwarding Information Base (*FIB*)

Despite having many similarities to the *FIB* of the IP router, the *FIB* of the *NDN* router differs from it in two key ways. First, unlike an *NDN FIB*, an IP *FIB* entry typically maintains a single best next-hop instead of a ranked

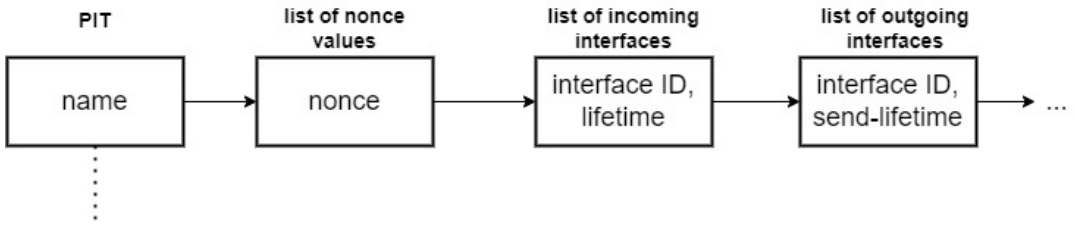


Figure 2.6: Forwarding state in *PIT*, from [57].

list of multiple interfaces. Second, an IP *FIB* entry contains only the next-hop information, whereas an *NDN FIB* entry facilitates adaptive forwarding decisions by collecting information from both the routing and forwarding planes. As shown in Fig. 2.7, when a name prefix is announced during routing, a new entry is added to the *FIB*. Each *NDN* node has a Forwarding Strategy module, which is responsible for deciding where to send an incoming *Interest* packet based on data from the Content Store (CS), the Pending *Interest* Table (*PIT*), and the Forwarding Information Base (*FIB*).

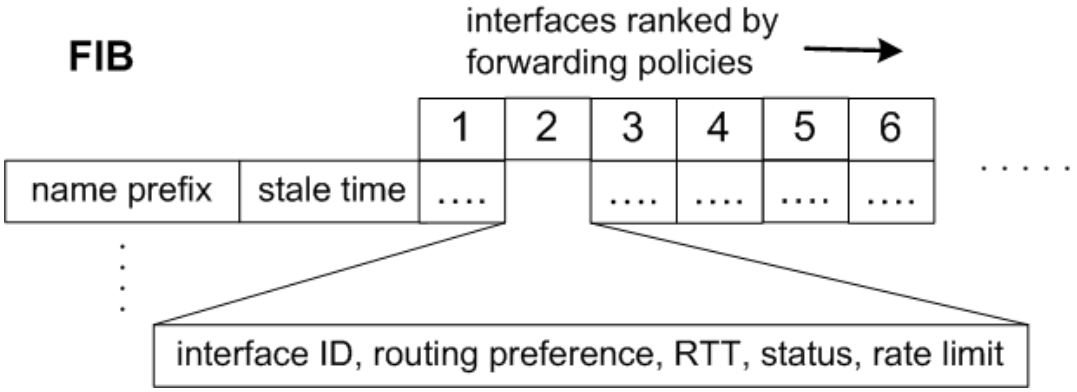


Figure 2.7: Forwarding state in *FIB*, from [57].

2.3.6.1 *NDN* Forwarding Process

In *NDN*, each time an *Interest* or a *Data* packet reaches an intermediate node, the node processes the packets and decides whether to forward them in accordance with the procedure shown in Fig. 2.8 A router first searches the *CS* for an appropriate match when an *Interest* packet arrives. If a lookup hit occurs,

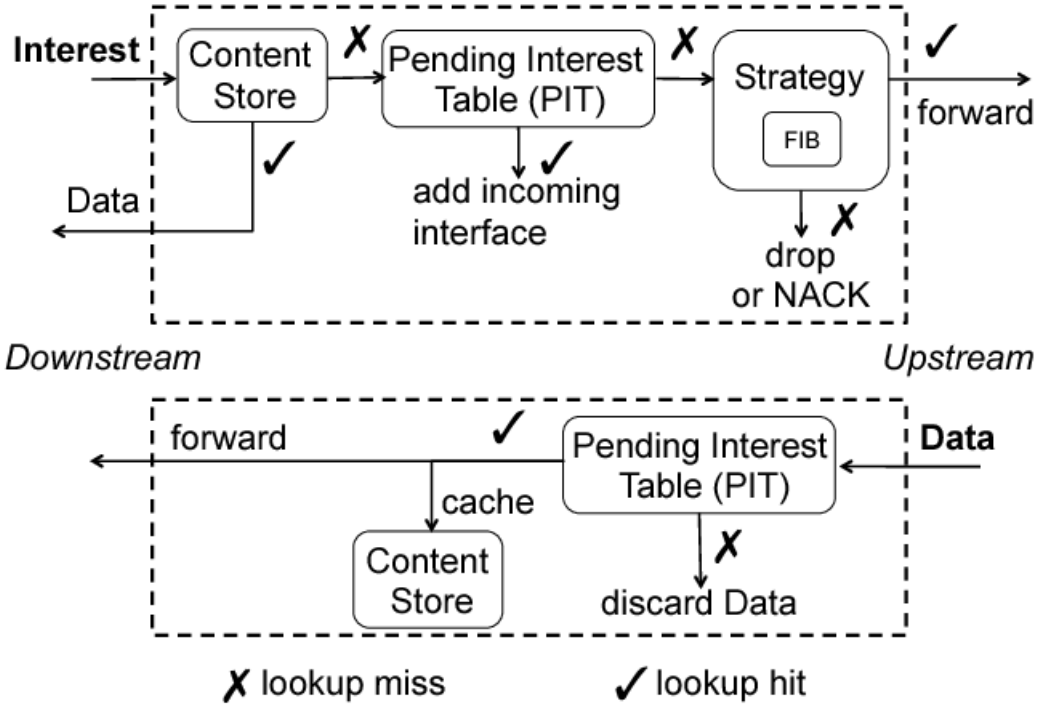


Figure 2.8: Forwarding state in *FIB*, from [46].

indicating that the *Interest* can be pleased from the local cache, the intermediate node returns the requested *Data* through the incoming interfaces of the *Interest*. If the router is unable to satisfy the request locally, it performs a *PIT* lookup. If a matching *PIT* entry exists, the *Interest* is dropped if the same *Interest* is looped back through a different interface. In contrast, if the *Interest* comes from a different client accessing through a different interface, the arriving interface for the *Interest* is added to the list of requesting interfaces in the associated *PIT* entry, and the *Interest* is then suppressed. A router creates a new *PIT* entry for the arriving *Interest* in the event of a *PIT* lookup failure, which means the *Interest* name doesn't really reside in the *PIT*. The router then performs a *FIB* lookup to forward the soliciting *Interest*. If the corresponding name doesn't match any *FIB* entries, the *Interest* is dismissed. Otherwise, as determined by the forwarding strategy module, the requesting *Interest* is pushed forward in accordance with the interfaces of the matched *FIB* entry.

The first thing an intermediate router does after receiving a Data packet is run a *PIT* lookup. If the router discovers an unsatisfied interest with the same

name already waiting in the *PIT* table, it resembles the data packet to all the interfaces through which the interest originated, traces the interest as satisfied, and deletes the *PIT* entry. As a result, the matched Data packet always follows the exact same route as the soliciting Interest, but in the reverse way. The packet is suppressed if there are no entries that match the arriving Data packet because it is assumed to be an unsolicited packet. Each Interest that a consumer issues has a lifetime associated with it, and if a timeout occurs before it is satisfied, any Interest that is still awaiting in the *PIT* table is eliminated. Once a Data packet is received, a router inserts it into the network cache in accordance with its caching policy. Since future content requests or retransmitted Interests can be satisfied directly from the router's cache rather than having to travel up to the producer, in-network caching, which is built into *NDN*, speeds up the content retrieval process.

2.3.7 IP and *NDN* Routing

A router in a TCP/IP architecture forwards a data packet based on its source and destination Internet Protocol addresses. A data plane and a routing plane make up an IP router. The intelligent routing plane is in full control of broadcasting hosts' and connected networks' IP addresses throughout the network. On the basis of the received advertisements, the routing plane is also in responsible of calculating the path(s) to networks. The plane broadcasts routing announcements across the entire network using routing protocols like Distance Vector or Link State, BGP, OSPF, etc. Using the received advertisements, each router determines the best path (next hop), stores the paths in its FIB, and then computes the next hop. Additionally, the address-based routing must be able to recognize the address space, reveal (hide) public (private) addresses, and handle NAT problems, among other things. As a consequence, the IP Routing Information Base (RIB), which is similar to the *NDN FIB* table in appearance, includes a single best next-hop and relevant information. Therefore, the RIB table must be updated whenever there are network changes like topology or policy changes. This is the responsibility of the IP routing plane. Thus , the routing plane is stateful and adaptive. While the forwarding plane is stupid, only able to travel in one

direction, and stateless, it completes all the major tasks. Moreover, the data plane only uses one interface to forward packets that strictly follow the RIB table [57]. As a result, in IP networks, the intelligent plane is completely responsible for packet delivery.

When a network change occurs, for example a route failures, the router swaps routing updates in order to recover from the failure and achieve consistency. The interval between the moment of network failure detection and the moment at which all routers receive an update is known as the routing convergence time. When the network has recovered, packet delivery can resume with a short convergence time to reduce packet loss. Nevertheless, attaining fast convergence in large-scale networks is difficult, particularly when routing stability and scalability are taken into consideration.

Scalability is necessary to support large-scale networks made up of numerous nodes or routers, IP prefixes, and links, while routing stability is necessary to handle applications that are affected by RTT fluctuation. Therefore, ensuring rapid convergence, stability, and scalability all at once is not simple. Distance-vector routing protocols, for example, guarantee higher scalability but slower convergence by using the hop distance that separates two routers (without understanding the network topology) as a path metric to route and forward packets. Link-state routing protocols, on the other hand, can converge quickly but have poor stability and constrained scalability because each router is conscious of the whole topology.

Since IP doesn't support multi-path routing by default and requires routing to ensure loop-free paths, it is less preferred when it comes to multi-path routing. As a result, the routing plane is accountable for ensuring loop-free paths through fast convergence when looping packets are detected, which is resource intensive and CPU-consuming.

2.3.7.1 *NDN* Routing Plane

NDN automatically solves the issues with IP addresses by routing and forwarding a Data packet in accordance with its name. Because the *NDN* namespace is infinite, there is no address exhaustion problem. Additionally, there is no NAT

traversal issue because the routers convey *Data* toward the requesters by following the trails in the *PIT* rather than requiring a host to reveal its private address in order to offer content. Eventually, in LAN, there is no longer a requirement for address management and allocation [46]. In contrast to IP, which announces address prefixes that cover the content the router is willing to serve, *NDN*'s routing plane announces name prefixes. The announcement of name prefixes is initiated by the content providers, and the routes to those prefixes are calculated directly by the other routers. A routing protocol is used to distribute the announcement throughout the network, and each router that receives it develops its own *FIB* [52]. The routing plane, without a doubt, populates and updates the *FIB*. Link-state, OSPF, and other routing protocols that function for IP should also function for *NDN*. The *NDN* routing plane should be designed differently from the current routing plane because *NDN* has an intelligent and adaptive data plane, which is described later. The current routing plane only supports the forwarding plan by populating the *FIB*.

2.3.7.2 *NDN* Forwarding Plane

In *NDN*, the data plane has two responsibilities: sending consumer *Interest* throughout the network and pulling the relevant *Data* across the same path but in the other direction. To keep the forwarding state-full, the routers that are forwarding the *Interest* maintain an entry in their *PIT*. As a result, *NDN* helps to make adaptive forwarding possible by symmetrically exchanging interest-data and maintaining a forwarding state at routers [56, 57]. An *NDN* router can assess packet delivery performance, such as round-trip time and throughput, using the forwarding state data (*PIT* entries) and by looking into the data packets that are returning. In addition to identifying link failures or congestion, a router can also identify issues that cause packet losses. Assume, for instance, that the requested *Data* is not retrieved within a reasonable amount of time via the same interface as the *Interest* was forwarded. If so, the router observes that as a potential issue: a connection dropped because of loops, link failures, or congestion and so on.

Decisions about *NDN* forwarding are thus flexible to the state of the network. So, the forwarding plane can select from a variety of alternative routes to avoid the

problematic interface. The forwarding plane's ability to quickly detect and recover from link failure means that *NDN* routing does not need to converge quickly to reflect network adjustments. Furthermore, routing scalability and stability can be greatly enhanced by addressing network failures at the forwarding plane, which frees the routing plane from transient failures.

The idea of a *Nack* message was first initiated by Yi *et* al. [56, 57], which a node creates and retransmits to the downstream node when it rejects an incoming *Interest*. In addition to the same name as the incoming *Interest*, *Nack* is accustomed to quickly notifying network problems by sending them a code along with a message explaining how they originated about. Without having to wait for the routing convergence, the forwarding plane can immediately take, local action to recover from network failures [58]. The forwarding plane can prevent *Interest* looping since each *Interest* is tagged with a nonce field (a random number) that, along with the *Interest* name, uniquely defines the *Interest*. Moreover, the corresponding *Interest* cannot loop either because the *Data* packets follow the same routes but in the reverse direction. Because of this, an *NDN* router can forward an Interest across the many interfaces without generating loops. The *FIB*, which supports *NDN*'s inherent support for multipath forwarding, contains a ranked list of multiple interfaces for each entry. An incoming *Interest* is forwarded by the interface(s) that the forwarding strategy module chooses based on forwarding information, routing information, and forwarding policies established by the operator [2].

2.3.7.3 Establishing the *FIB* table

To make the content stored on the producer application widely available for retrieval in *NDN*, the producer application records a content's prefix with the local forwarder within the same host machine. After that, the content producer makes announcements about its accessibility for remote *NDN* forwarders to request and download. Manual configuration, dynamic name announcement protocols (like NLSR), or opportunistic data discovery strategies (like the access strategy) are some of the ways to announce the availability of content [2]. The trade-offs

associated with each method vary, depending on factors like usability, complexity of implementation, and communication expense.

Jacobson *et* al. [59] invented the Automatic Prefix Propagation (*APP*) protocol, which is used on the producer machine to allow the local *NDN* forwarder to instantaneously propagate local prefix registrations to the gateway router's remote forwarder. Local prefix registration will start this propagation if two conditions are met: (1) there is an active remote *NDN* gateway; and (2) the local forwarder has a private key and the corresponding *NDN* certificates that match or contain the locally registered namespace. Figure 2.9 depicts the procedure.

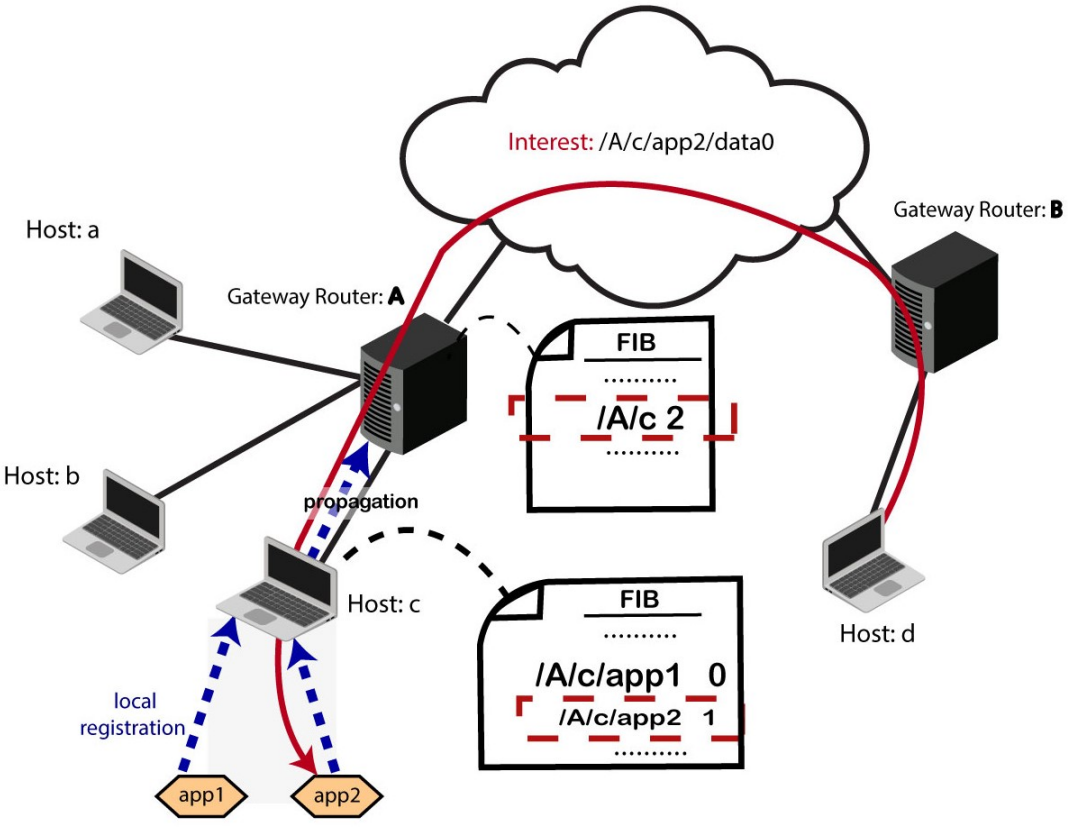


Figure 2.9: Advertising a producer application's *Data* to *NDN* forwarders, from [59]

In the event that there are multiple candidates for the keys/namespace, the key with the relatively short namespace is chosen. This allows the forwarder to combine multiple local registrations into a single remote action, which reduces

communication and bookkeeping costs. Forwarding Deamon (*NFD*) intermittently refreshes the registration after the initial prefix dissemination, as long as the prefix is required to register locally or there are *NDN* gateways configured.

2.3.8 *NDN* Forwarding Strategies

The majority of current *NDN* forwarding techniques fall into one of two categories: [60]: flooding-based strategies [20, 61–68] and single-route-based strategies [60, 69, 70]. The flooding-based strategies hit multiple sources via different paths, and the contents are returned to the requester in the opposite direction [28, 71–78]. It provides a guarantee that the information will be retrieved because Interest is forwarded to all available outgoing interfaces. As a result, the requester receives multiple copies of a given piece of content; however, aside from the first copy, the others are discarded as being unnecessary [20]. The strategy is simple, but the redundant copies introduce significant traffic (transmission) overhead, which consumes valuable network bandwidth, wastes node resources (energy), causes network congestion, and reduces network performance, particularly in dense networks [60, 79].

2.4 Related Work

Coupling forwarding and caching in *NDN* is crucial to realize the essence of name-based forwarding and benefit from in-network caching. The ideal Nearest Replica Routing (*iNRR*) proposed in [33] couples forwarding and caching to assume that the routers are equipped with an "oracle" that keeps track of the underlying network cache to forward an *Interest* to the nearest possible content replica. However, realizing such an "oracle" to use prior information for content retrieving is not feasible [80]. Hence, the authors proposed a more practical approach, a.k.a exploration-exploitation, to retrieve the desired content. The two-episode process significantly reduces the traffic load but experiences a more prolonged delay while downloading a content from the nearest copy. Iqbal *et* al. proposed *Discovery* [34], *DTABB* [31], and *SRP* [32], which simultaneously explored and exploited the

best replica to retrieve the desired item. The simultaneous strategies minimize the delay and traffic load successfully.

Yet again, the strategies proposed by Iqbal *et* al. are source-driven, where the optimal source responds with the entire payload and the sub-optimal sources reciprocate with payload-free or payload-slice *Offer* packets to minimize the delay. The source-driven approach saves a lot of bookkeeping and resources (CPU time and space) as nodes need not update forwarding paths frequently, and a router maintains information only for the cached contents. However, the source-driven approach occasionally suffers from increased communication overhead when sub-optimal sources respond with complete payload and extended delay when no sources answer with the entire content.

On the other hand, the receiver-driven approach needs to maintain and update routing information (FIB) frequently for all the encountered prefixes to forward an *Interest* to the best source. For instance, the *Multicast* [81] strategy proposed replicating an incoming *Interest* to all the outgoing interfaces suggested by the *FIB*, excluding the arrival interface of the *Interest*. The best-route process [81] needs an agile *FIB* to select the lowest cost available upstream interface to forward an incoming *Interest*. The adaptive smoothed RTT-based forwarding strategy (ASF) [82] measures the $SRTT$ of all the outbound interfaces and chooses the one with the lowest measured $SRTT$ among those offered by the underlying routing protocol. However, ASF needs to probe the alternative interfaces at regular intervals to gather forwarding information in advance.

A group of forwarding strategies [21, 83–89] utilize additional control packets, for instance, Probe *Interest*s, Probe *Data* (a.k.a *Interest* and *Data* ants), *HELLO* packets, Virtual *Data* packets, etc., to obtain prior forwarding information to keep the *FIB* updated. The control packets are issued periodically or event-triggered to collect the routing information early. A small probing period may generate tremendous network overhead on the Internet, which already hosts tons of materials, and the volume is increasing at lightning speed. In contrast, a higher probing interval may cause the *FIB* entry to become stale.

Another group of forwarding policies, such as [60, 79, 90–94], uses the optimal path, determined so far, to forward an incoming *Interest*, however, probes the alternative options (interfaces) periodically or opportunistically to find the

better path (if any). For instance, *INFORM* [91] alternates between exploration and exploitation based on the Q-learning algorithm [95]. During the exploration phase, an incoming *Interest* is forwarded using the best outgoing interface, according to the *FIB*, learned so far, and on an arbitrarily selected interface. The exploitation capitalizes on the other phase and employs the best path only to forward an arriving *Interest*. The best-route strategy examines each outgoing interface's performance and chooses the best interface by learning and depending on the result of earlier decisions. However, best-route is not agile [96] and adopts *NACK* packets [60, 97] while learning different path metrics. Lei *et* al. proposed an entropy-based probabilistic forwarding (*EPF*) [79] to select an interface based on multi-attribute decision making problem (*MADM*). *EPF* claimed to achieve better load balance, however, at the cost of computing selection probabilities of every interface at the time of decision making (outgoing interface selection).

A third group of forwarding strategies apply Q-learning to select the optimal interface, however, only at the cost of increased overhead packets [80, 91, 96, 98–100] or enlarging the size of regular *Interest* and *Data* packets in [99, 100]. A few proposals, such as *IFS-RL* [96], *NDNFS-RLRNN* [80], employ a computation-intensive neural network in the learning process to select the best outgoing interface(s).

Single-route-based strategies, which sacrifice extra control packets and/or processing overhead, choose the best outgoing interface and discover and update path information. Table 2.1 presents a comparative study of the forwarding strategies in *NDN*.

Overall, the forwarding strategies investigating the available alternatives aim to optimize the *Interest* forwarding to improve network performance, i.e., communication overhead and retrieval latency. However, the size of a *Data* packet plays the dominating role in terms of network performance. The research question, Q, is answered in chapter 3 by $SRAB^3$, a reinforcement learning-based forwarding strategy that uses Thompson Sampling in the learning process where a router performs learning only for the cached contents. The proposed strategy aims to optimize *Data* communication by suppressing the replies from sub-optimal sources. However, it differs from the previous works as it never encounters the use of an *Offer* packet and hence avoids occasional prolonged delays, unlike [34]. $SRAB^3$

Table 2.1: An overview of forwarding strategies in *NDN*

Reference Strategies	***Forwarding Approach***	***Interest Forwarding***	***Source Reply***	***Path Metric***	***Probing Rate***	***Control Packets***	***Learning***
Multicast [81]	Receiver-driven	Flooding	*Data*	–	–	–	–
iNRR [33]	Receiver-driven	Scoped-flooding	*Data*/*Offer*	Hop count	–	–	–
DTABB [32]	Source-driven	Flooding	*Data*/*Offer*	Acknowledgement	–	*Interests*	Greedy,UCB1,Gradient Bandit
SRP [31]	Source-driven	Flooding	*Data*/*Offer*	Hop count	–	–	–
Discovery [34]	Receiver-driven	Scoped-flooding	*Data*/*Offer*	Hop count, #*Interests*	–	–	–
$SRAB^3$	Receiver-cum-source-driven	Scoped-flooding	*Data*/*No Reply*	interface rank, Acknowledgement	–	*Interests*	Thompson Sampling
BestRoute [81]	Receiver-driven	Single	*Data*	SRTT	–	–	–
ASF [82]	Receiver-driven	Single	*Data*	Interface rank	–	–	–
INFORM [91]	Receiver-driven	Single	*Data*	Delivery time	Event-triggered	–	Q
DIVER [88]	Receiver-driven	Single	*Data*	Hop count	Event-triggered	*Interests*	–
AC-QOS-FS [60]	Receiver-driven	Single	*Data*	Interface rank	Periodic	*Interests* and *Data*	–
Persistent [86]	Receiver-driven	Single	*Data*	Delay and path loss	Event-triggered	*Interests* and *Data*	–
APFS [83]	Receiver-driven	Single	*Data*	Hop count	Event-triggered	*Interests* and *Data*	–
Bloom filter-based [87]	Receiver-driven	Single	*Data*	Bandwidth	Periodic	*Interests* and *Data*	–
LiteNDN [89]	Receiver-driven	Single	*Data*	Routing costs, Hops	Periodic	*Data*	–
IPI [94]	Receiver-driven	Single	*Data*	Average Pending *Interests*	Event-triggered	–	–
AFSndn [100]	Receiver-driven	Single	*Data*	Delay	Event-triggered	–	Q
DRL-based [98]	Receiver-driven	Single	*Data*	RTT	Event-triggered	–	Deep-Q
IFS-RL [96]	Receiver-driven	Single	*Data*	interface RTT and usage	Event-triggered	–	RL + NN
NDNFS-RLRNN [97]	Receiver-driven	Single	*Data*	RTT	Event-triggered	–	RL + RNN
fwd^{pro} [101]	Receiver-driven	Single	*Data*	interface utilization ratio,RTT,NACK ratio	Event-triggered	–	–
DQN-AF [102]	Receiver-driven	Single	*Data*	#Packets,RTT,SRTT,RTO	Event-triggered	–	Deep-Q

introduces the Beam concept, an Interest thread, to ensure content download and to avoid the need for *Offer* packets.

2.5 Summary

This chapter includes a brief discussion of the important *ICN* architectures such as *DONA*, *PURSUIT*, *NetInf*, *CCN*, and novel inovation *NDN*. The proposed architectures all use the same design paradigm, which is content-centric rather than host-centric, although having different methods. Since the proposed study is developed and improves to the *NDN* tenet, the architectural components and guiding principles of the *NDN* are discussed extensively. The next chapter discusses the $SRAB^3$ forwarding strategies.

Chapter 3

$SRAB^3$ Forwarding Strategy

This chapter commences with a brief picture of Two-Armed Beta-Bernoulli Bandit (TAB^3) followed by a formal problem definition where a content source's reaction to a corresponding *Interest* is articulated as a TAB^3 problem.

3.1 System Model, Assumptions, and Problem Formulation

3.1.1 Beta-Bernoulli Bandit

Assuming that there are K possible actions (arms) that an agent can try, and action $k \in \{1, 2, ...K\}$, when played, yields a binary payout, i.e., either a success or failure. The general definition corresponds to Multi-Armed Bernoulli Bandit and reduces to $TABB$ when $K = 2$. Action k produces a success with probability $\theta_k \in [0, 1]$ and a failure with probability 1–θ_k. Each θ_k can be interpreted as the success probability or mean reward of action k. The success probabilities of the k arms $(\theta_1, \theta_2, ..., \theta_K)$ are unknown to the agent but fixed over time, and therefore can be learned by playing or experimenting. Let the agent begin trying the options with an independent prior belief over each θ_k. Beta-distribution [103] for these priors beliefs are considered with shape parameters $\alpha \in (\alpha_1, ..., \alpha_K)$ and $\beta \in (\beta_1, ..., \beta_K)$. Thus, the prior probability density function, denoted as $p(\theta_k)$,

of action k is defined as eq. 3.1:

$$\begin{aligned} p(\theta_k) &= \frac{\Gamma(\alpha_k + \beta_k)}{\Gamma(\alpha_k)\,\Gamma(\beta_k)} \theta_k^{\alpha_k - 1} (1 - \theta_k)^{\beta_k - 1} \\ &= \frac{1}{B(\alpha, \beta)} \theta_k^{\alpha_k - 1} (1 - \theta_k)^{\beta_k - 1} \end{aligned} \tag{3.1}$$

where Γ is the gamma function [104]. The beta function, B, is a normalization constant to ensure that the sum of the probability is 1. A random variable X having beta distribution with parameters α and β, hereafter, will be denoted by $X \sim Beta(\alpha, \beta)$.

As the agent starts playing and gathers observations, the distribution is updated according to Bayes' rule. The conjugacy characteristics of beta distribution makes it convenient to work with, and an arm's posterior distribution also follows beta with the shape parameters updated after each observation as eq. 3.2:

$$(\alpha_k, \beta_k) = \begin{cases} (\alpha_k, \beta_k), & x_t \neq k \\ (\alpha_k, \beta_k) + (r_t, 1 - r_t), & x_t = k \end{cases} \tag{3.2}$$

where x_t and r_t are the action taken and corresponding payout, respectively, at turn t. The shape parameters, *a.k.a* pseudo-counts, α, and β, are incremented by one with each earned success or faiulre, respectively. When $\alpha_k = \beta_k = 1$, the prior $p(\theta_k)$ is uniform over [0, 1]. Notably, the corresponding parameters of the selected action only are updated after each round. A particular beta distribution with parameters has mean $\frac{\alpha_k}{\alpha_k + \beta_k}$ and the distribution becomes concentrated as the arm is played more, alternatively as the sum $(\alpha_k + \beta_k)$ grows.

Thompson Sampling algorithm is specialized for the Beta-Bernoulli Bandit case. At any round t, it generates an estimate of the success probability, $\hat{\theta_k}$, randomly sampled from the posterior distribution, a beta distribution with parameters α_k and β_k. The action x_t with the highest estimate $\hat{\theta_k}$ (success probability) is then selected. The reward for the selection r_t is observed and the distribution parameters α_{xt} and β_{xt} are updated. The agent's objective is to earn the maximum number of successes over T trials by playing the k arms. The value of T is exceedingly high compared to the number of components (in our case 2). The agent needs to balance between playing (exploiting) the current best arm at hand

and discover (explore) the alternative options (arms) to maximize the cumulative payout (number of successes).

The exploration-exploitation trade-off between the available reactions leaves the agent (source) with the following contradictory dilemma: should it continue to exploit the reaction option that so far produced the maximum payouts, or should the alternative reaction option be sampled (investigated) to discover more about its success probability? Premature exploitation of the present best arm may cease the agent from exploring the optimal reaction. In contrast, sticking needlessly to the sub-optimal option defers the optimum success that can be achieved by trying the optimal response type. $SRAB^3$ proposes to use *TS*, a reinforcement learning algorithm, to maximize the cumulative payouts and thus solve the defined TAB^3 problem of a content source.

3.1.2 Justification for Thompson Sampling

Reinforcement learning algorithms such as *Greedy*, $\epsilon - Greedy$, and $UCB1$ for solving *MAB* problems require an agent to start the learning with an initial exploratory phase where the agent equally (almost) samples the available arms in round-robin fashion or chooses the arm randomly with uniform probability [31, 105]. As learning is executed only for the cached contents in $SRAB^3$ and the caching duration of any item is highly dependent on the underlying network's caching capacity, the popularity of the content, cache admission policy, etc., determining the length of the initial exploratory phase is non-trivial. A large initial phase may evict a content before or shortly after the exploratory phase ends without reaping the learning profit. On the contrary, if the phase is too small, the inferior action may be sampled more in the exploitation phase by virtue of estimated values.

However, Thompson Sampling follows a different tactic; instead of merely refining the estimated values, it extends this to construct a probability distribution from the obtained rewards and samples the distribution to choose an action. Thus, there is no initial exploratory phase to begin with. Therefore, it is easy to understand and implement.

Again, advanced reinforcement learning such as Q-learning is not considered to solve the $SRAB^3$ problem as a learning agent here assumes a single state only. The advanced learning algorithms are feasible when an agent switches among multiple states and the problem has a goal state to reach.

3.1.3 Problem Formulation

In the proposed strategy, a content source, temporary custodian or permanent host, is considered an agent that can reciprocate to an incoming *Interest* in precisely one of the two possible ways. A content source assumes a discrete-time model for the corresponding arriving *Interest*s. For each *Interest* that hits, alternatively at each time slot, the custodian decides between reacting with one of the two actions, namely, *Reply* or *Silent*. The *Reply* action corresponds to the source answering with the matching *Data* (content). In contrast, the *Silent* option resembles that the content holder selects ignoring the *Interest* or restraining from sending the *Data* back. This differs from the source-driven approaches where a sub-optimal source must send back an *Offer* when it decides not to answer with a *Data*. A source can assume the *Silent* role as the content download is ensured by the *Interest* forwarding policy of the proposed strategy.

A content source can learn the proper action only by playing (experimenting). So, a sub-optimal source may respond with a *Reply* and hence redundant *Data* may arrive at the user. On arrival of the initial *Data* at the consumer, an acknowledgement message, denoted as *AckNode*, is sent to all the content sources stroke by the corresponding *Interest*. However, apart from the first *Data* arrived at the requester, subsequent replies are suppressed. The *AckNode* message contains the node ID that served the *Data*. Each source receiving the acknowledgement message employs it to learn the payout or reward received for the action.

The number of corresponding *Interest*s striking a source is represented by I and presented to the source one after another. The source must react to each hitting *Interest* i, where $i \in I$, by *Silent* or *Reply*. *Silent* and *Reply* actions are considered as the two arms (a) of $SRAB^3$ such as $a \in A = \{S, R\}$, where S and R denote reacting with *Silent* and *Reply*, respectively. For an i offered at time t, a source pulls one of the two arms denoted by a_t, and when pulled, the arm produces

a binary payout b_t, represented as success or failure, drawn independently of the past. As there are only two possible actions and an action's corresponding payout distribution is unknown, it is assumed that the two actions probabilities are (θ_S, θ_R), indicating i is assigned to arm S and arm R with probability θ_S and θ_R, respectively. Formally, $A : \{a : \theta_S \geq 0, \theta_R \geq 0, \theta_S + \theta_R = 1\}$. After reacting to i with an action a, the source examines the *AckNode* message send by the corresponding user and determines the binary payout for that particular action. The model aims to find a sequential strategy for pulling arms that maximizes the cumulative payout earned, i.e., attain the maximum number of observed successes from the I *Interest*s. Notably, reacting with *Reply* always may generate excessive traffic load while sticking to *Silent* every time may be inferior in terms of delay.

The above assumptions lead $SRAB^3$ to be defined by random variables $X_{a(t),t}$ for $a \in \{R, S\}$ and $t \geq 1$. If action a is sampled successively yielding payouts (successes or failures) $X_{a(1),1}$, $X_{a(2),2}$, ... where the payouts are picked independently from an unknown but fixed beta Bernoulli distribution. The independence also holds for successes or failures across arms. Thus, $X_{R,m}$ and $X_{S,n}$ are independent and generally not identically distributed for each $m, n \geq 1$. The random variables can be represented as $X_{R,m} \sim Beta(\alpha_{Rm}, \beta_{Rm})$ and $X_{S,n} \sim Beta(\alpha_{Sn}, \beta_{Sn})$, respectively. At each round, the source generates estimates of the success probabilities, $\hat{\theta_R}$ and $\hat{\theta_S}$, randomly sampled from the corresponding posterior distributions. The reaction having the higher value is chosen.

An action selection strategy (optimizer) in a content source chooses the action between R and S to react to the next *Interest* i based on the success probability estimates. Let, $\hat{\theta_*}$ be the mean reward if the optimal reaction is picked by the process every time while answering the initial I *Interest*s. Therefore,the regret of the process after reciprocating I is defined by

$$\hat{\theta_*} * I - \mathbf{E}\left[\sum_{t=1}^{I} Beta(\alpha_{at}, \beta_{at})\right]$$

$$= \hat{\theta_*} * I - \mathbf{E}\left[\sum_{t=1}^{I} X_{a(t),t}\right]$$

where $\mathbf{E}\left[\sum_{t=1}^{I} Beta(\alpha_{at}, \beta_{at})\right]$ is the expected sum of mean estimates. The regret is considered as the expected loss indicating that the process is not perfect; in other words, it does not always choose the best action. The agent aims to minimize the regret loss by maximizing $\mathbf{E}\left[\sum_{t=1}^{I} X_{a(t),t}\right]$ while reacting to the I *Interest*s.

3.2 *SRAB*^{3} Strategy

This section introduces the various components of *SRAB*^{3} followed by the principal concepts and algorithms used to commute *Interest*s and *Data* to retrieve a content.

3.2.1 *SRAB*^{3}

SRAB^3 uses a scoped-flooding where an *Interest* is disseminated in the ring centered around the user; however, a single copy, denoted as *Beam*, is permitted to travel further until it strikes a corresponding source. Upon stroke by the corresponding *Interest*, a custodian reacts either by sending a *Reply* (matching *Data*) or remaining *Silent* (or mute). A *Data* carrying the entire payload is delivered along the reverse path to the consumer. The consumer, upon arrival of a matching *Data*, issues a *AckNode* message which is communicated to all the custodians that were hit by the requesting *Interest*.

3.2.1.1 Example Scenarios

Fig. 3.1 corresponds to the *Interest* forwarding and content delivery in *SRAB*^{3}. The requester R issues an *Interest* to retrieve content M which is scoped-flooded (green-dotted arrow) in the network, forming a ball around R. A distinct thread of the *Interest* (brown-dotted arrow), called *Beam* (B), crosses the ring and is conveyed further to hit a corresponding replica. The sources inside the circle react to the *Interest* with the reaction learned according to Thompson Sampling. At the same time, the replica stroke by the *Beam* must reciprocate by sending back item M to R. The presence of B ensures the content download and avoids the

exploration then exploitation problem of *DTABB* while downloading the desired content.

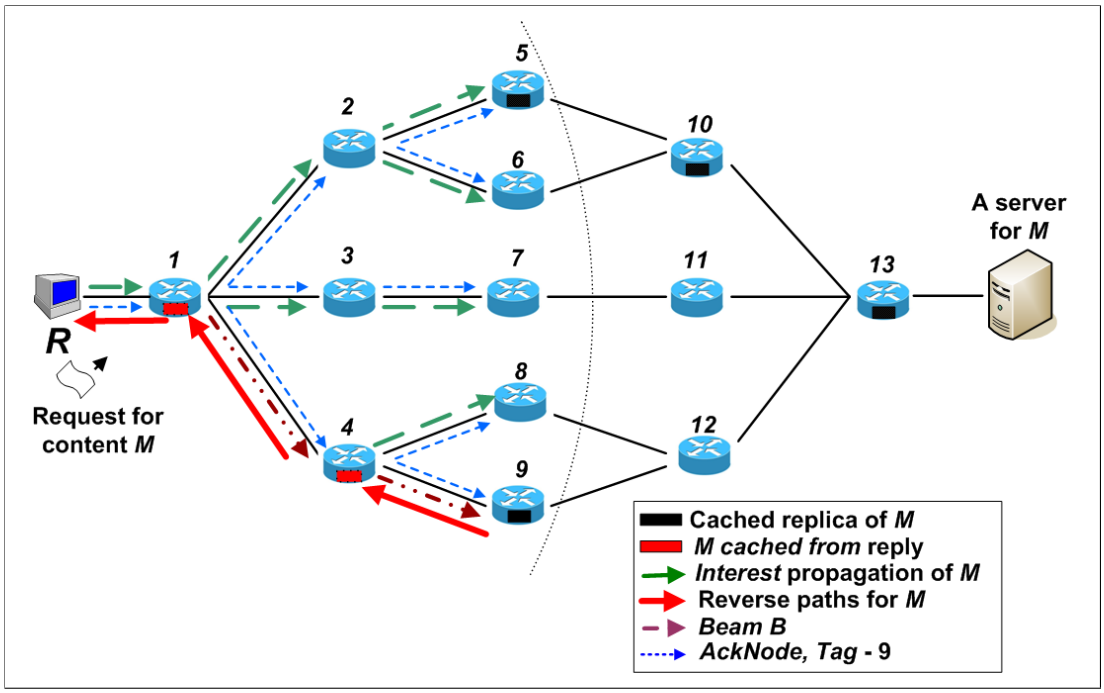


Figure 3.1: Optimizing *Interest* forwarding in *SRAB*^{3} - an optimal scenario

Fig. 3.1 corresponds to the situation when the *Beam* hits the optimal source within the ring, and Fig. 3.2 resembles the scenario when *B* strikes a sub-optimal source and an optimal source inside the circle reciprocates with *Data* (red-dotted arrow). The user sends an acknowledgement (blue-dotted arrow) for the *Data* arrived first.

3.2.1.2 *NDN* Packet Header Extension

SRAB^3 extends the headers of default *Interest* and *Data* to control the scoped-flooding and content delivery. A couple of new fields, designated as *Beam* and *Scope*, are added to the default *Interest* header. The first field is used to indicate the thread of the *Interest* that is free to travel outside the ring. The second field controls the radius of the *Interest* flooding circle (number of hops), similar to the

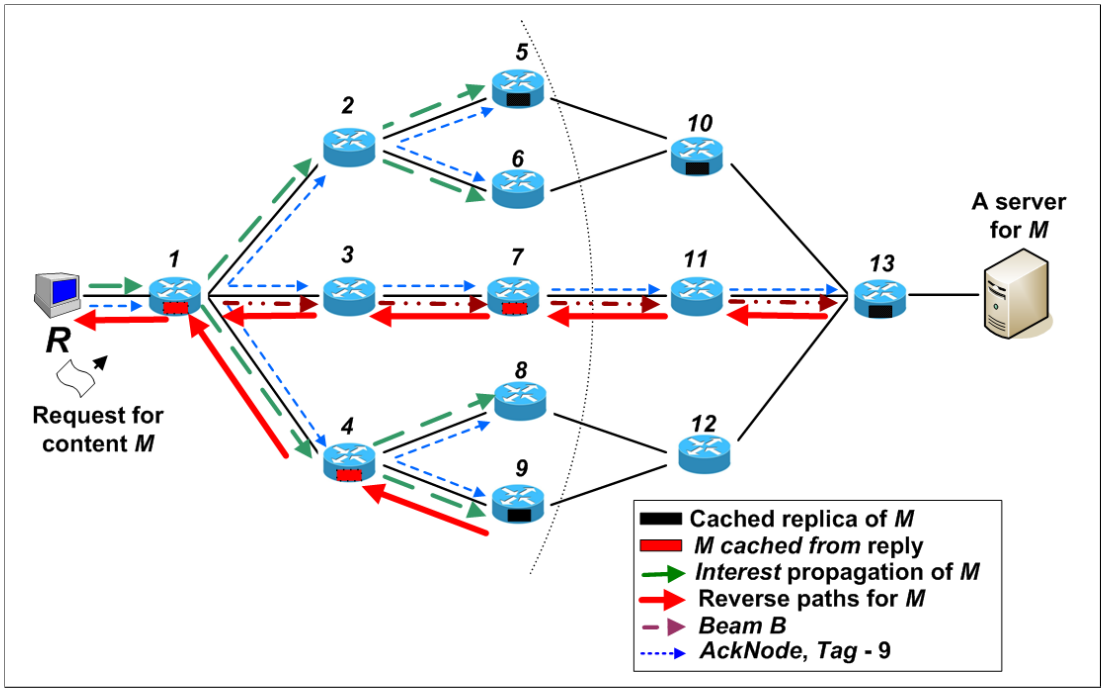


Figure 3.2: Optimizing *Interest* forwarding in $SRAB^3$ - a sub-optimal scenario

IP packet's *TTL* field. The two fields are initialized by the consumer and updated by the forwarders as the *Interest* is pushed forward.

The default *Data* header is extended to include a field, denoted as *Tag*, to piggyback the responding node ID or a pre-defined value. The *Tag* value is set according to the outcome of Thompson Sampling at the sources. The requester uses the value while sending the acknowledgement (*AckNode*). Finally, the *Interest* message name is modified by adding a random number, for instance, /prefix/rand/00, to by pass *PIT* waiting.

3.2.1.3 Introduction to *AckNode* Message

On the reception of the matching *Data*, the corresponding requester recognizes the event by sending an acknowledgement packet, namely *AckNode*, to the content sources hit by the requesting *Interest*. The *AckNode* is issued once (on the arrival of the first *Data*), as subsequent receptions of redundant *Data* are suppressed. The *AckNode* message is tailored by modifying the *Interest*. A tag, for instance,

"A", is assigned to the name of an *AckNode* message, for example, /prefix/00/A, to distinguish the *Interest* as an *AckNode*. Moreover, an *AckNode* message header is extended by a new field, namely *Tag*, used for learning purpose by the sources. An *Interest* and it's subsequent *AckNode* messages are forwarded using the same *FIB* entries; however, no *PIT* entry is inserted for any *AckNode* message. The *AckNode* message hits all the sources stroke by the corresponding *Interest*.

3.2.1.4 Payout Determination

In a source, say Z, the reinforcement learning, decision at time t for prefix p is denoted as TS_t^p. TS_t^p indicates the type of reaction Z takes to entertain the current *Interest* for p, and hence, $TS_t^p = R$ or $TS_t^p = S$. When $TS_t^p = S$, Z remains mute while entertaining the *Interest* and waits for the corresponding acknowledgement to arrive. If $TS_t^p = R$, Z places its ID in the *Tag* field of the returned *Data* packet and waits for the complementary *AckNode* message. If $TS_t^p = S$ but the *Interest* thread is the *Beam*, the source ignores TS_t^p and reacts with *Data*; however, it puts the pre-defined value, such as $Tag = 10000$, in *Data* in such case. The payout determination process in depicted in Fig. 3.3. Also, the table shows the *SucxFail* table, introduced later, update operation.

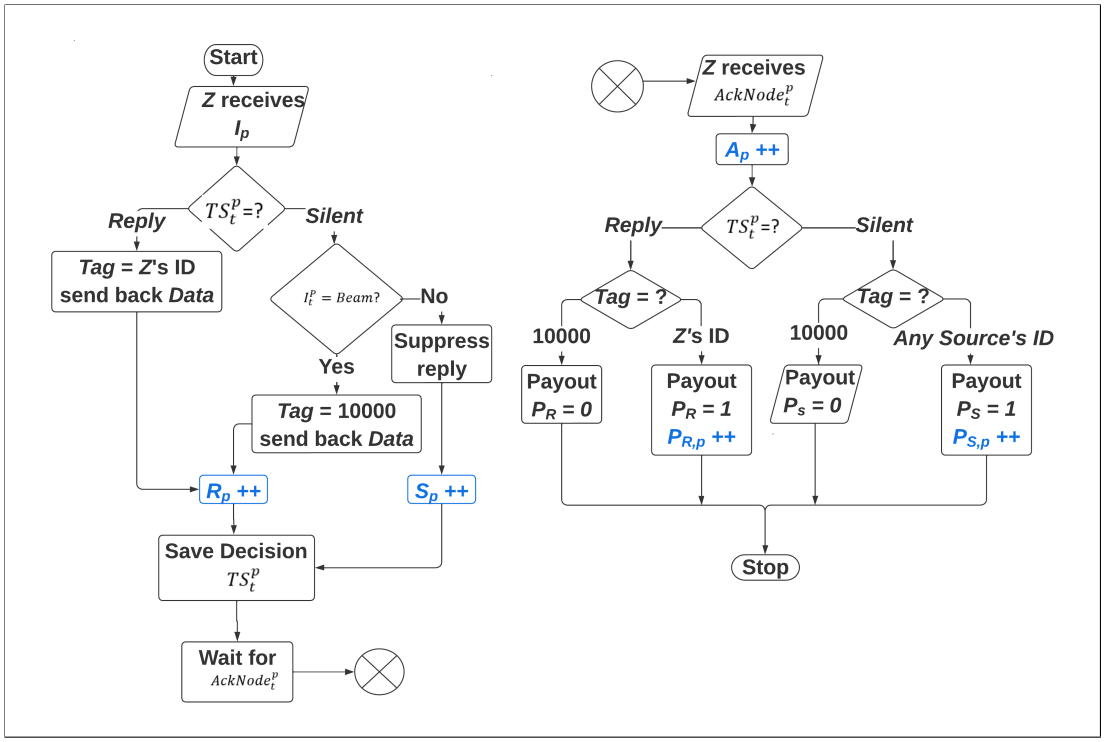


Figure 3.3: Payout determination and *SucxFail* table update operation at a source Z

Once reacted, the resultant binary payout is either a success (1) or failure (0) and decided upon receipt of the *AckNode*. The payout for last reaction to *Interest* for p, R or S, are denoted as P_R and P_S, respectively. If $TS_t^p = S$, and *AckNode* is piggybacking $Tag = 10000$, then the $P_S = 0$. This means that the content p was served by a sub-optimal source (hit by the *Beam* but $TS_t^p = S$). Alternatively, if $Tag \neq 10000$, remaining *Silent* last time was successful ($P_S = 1$). On the contrary, if $TS_t^p = R$, and the *AckNode* piggybacks the source's ID, then $P_R = 1$; otherwise, $P_R = 0$. For simplicity of discussion, the success payouts for R and S are represented as P_R^S and P_S^S, respectively, and, the the corresponding failure rewards are denoted as P_R^F and P_S^F, respectively. Note that, for *Reply* (R), $P_R^S = \alpha_R$, and $P_R^F = \beta_R$; and that for *Silent* (S), are $P_S^S = \alpha_S$, and $P_S^F = \beta_S$

3.2.1.5 Introduction to *SucxFail* Table

A content source maintains a *Data*base, namely *SucxFail*, to record its reactions to the corresponding *Interest*s and the payouts received for the responses. The corresponding table entry is updated every time a source reacts with a *Reply* or *Silent* action and on the arrival of a complementary *AckNode* message. The custodian also remembers the reaction to the last *Interest* served. The last reaction and the corresponding *AckNode* message together determine the fate (payout) of that particular reaction. A content source computes the number of successes and failures of a response using the fields of a *SucxFail* entry. The pattern of the *SucxFail Data*base with exemplary numbers is portrayed in Table 3.1. At any moment,

Table 3.1: *SucxFail* Table

Prefix(p)	***Interface (f)***	***Number of Reply*** (R_p)	***Number of Silent*** (S_p)	***Number of*** $P^S_{R,p}$	***Number of*** $P^S_{S,p}$	***Number of AckNode*** (A_p)
audio/one	x	20	5	18	4	25
audio/one	y	30	28	15	24	58
audio/two	x	25	3	25	0	28
audio/two	y	30	10	20	5	40

the numbers of observed successes and failures for each reaction type in a Z_p can be represented as a 4-elements vectors equal to the number of *Interest*s arrived for p (I_p) on an interface f. Thus, at any time epoch, $P^{S,f}_{R,p} + P^{F,f}_{R,p} + P^{S,f}_{S,p} + P^{F,f}_{S,p} = I^f_p$. Alternatively, $\alpha^f_{R,p} + \beta^f_{R,p} + \alpha^f_{S,p} + \beta^f_{S,p} = I^f_p$. All are non-negative integers.

Every time p is cached, an entry is entered in the *SucxFail* table, and the entry is continuously update until p is evicted from the cache. If Z_p answers with $R(S)$ in response to an i_p on interface f ($f \in F$; F is the number of physical interfaces in S_p), and receives the corresponding *AckNode* message on f, it increments the number of successes $\alpha^f_{R,p}(\alpha^f_{S,p})$ or failures $\beta^f_{R,p}(\beta^f_{S,p})$ by one.

3.2.1.6 Optimizing *Interest* Forwarding

A consumer opting for the desired content issues an *Interest* headed towards to the gateway router. The new fields are initialized as $Beam = 1$ and $Scope = 0$. The gateway router responds immediately by sending back the matching *Data* if found in its *CS*. A consumer receiving the desired item from the gateway router

does not acknowledge it by an *AckNode*. In case of a *CS* miss, the gateway node sets the *Scope* field as eq. 3.3:

$$Scope = \frac{MaxDist * Miss/(Hit + Miss)}{RadCntrl} \tag{3.3}$$

After that, the gateway, like *Multicast* [106], finds the outgoing interfaces from the *FIB*. $Beam = 1$ is used for the *Interest* thread (the *Beam*) outgoing through the best interface, while $Beam = 0$ is set for other threads. The gateway suppresses the non-*Beam* threads if $Scope = 0$.

In eq. 3.3, the *MaxDist* is the distance (hops) between the nodes that are at maximum distance from each other, and $SRAB^3$ assumes that each gateway router has prior knowledge of that value. The *Hit* and *Miss* represent the number of cache hits and misses, respectively, in the gateway. The maximum value of the ratio $MissRatio = Miss/(Hit + Miss)$ is 1. A higher ratio indicates a higher value of *Scope*, and the *Interest* is forwarded deep in the network to hit sources. On the contrary, a smaller value of the ratio denotes that the *Interest* is confined to a small ring. *RadCntrl* is the network parameter, $1 \leq RadCntrl \leq MaxDist$, used to control the scope (depth) of the *Interest* forwarding explicitly.

An intermediate node (for a cache miss) receiving the *Beam* thread updates the *Beam* field of the *Interest* as the gateway node and *Scope* as $Scope = Scope - 1$. For a cache hit, the router finds the *TS* decision for the *Beam*, populates the *Tag* of the matching *Data* as described in section 3.2.1.4, returns the *Data*, and waits for the corresponding *AckNode* message to arrive. The *Beam* ensures content retrieval. Notably, any router forwarded a regular *Interest* thread (non-*Beam*), inserted a *PIT* entry for the same and then received the *Beam* originated from other consumers, bypasses the *PIT* entry, and forwards the *Beam* thread.

On the other hand, if the thread is not the *Beam*, and a *CS* miss occurs, the forwarder drops the *Interest* if $Scope = 0$. Otherwise, it updates *Scope* as $Scope = Scope - 1$, keeps $Beam = 0$, and frontwards the thread. In case of *CS* hit, *TS* decision and the *Data Reply*ing are executed as in section 3.2.1.4. The router waits for the *AckNode* message to determine the payout for the action sampled.

The communication protocol of $SRAB^3$ is depicted in Fig. 3.4 and Fig. 3.5 . Fig. 3.4 represents how *Interest*s and *Data* are commuted and Fig. 3.5 resembles how an *AckNode* message is communicated.

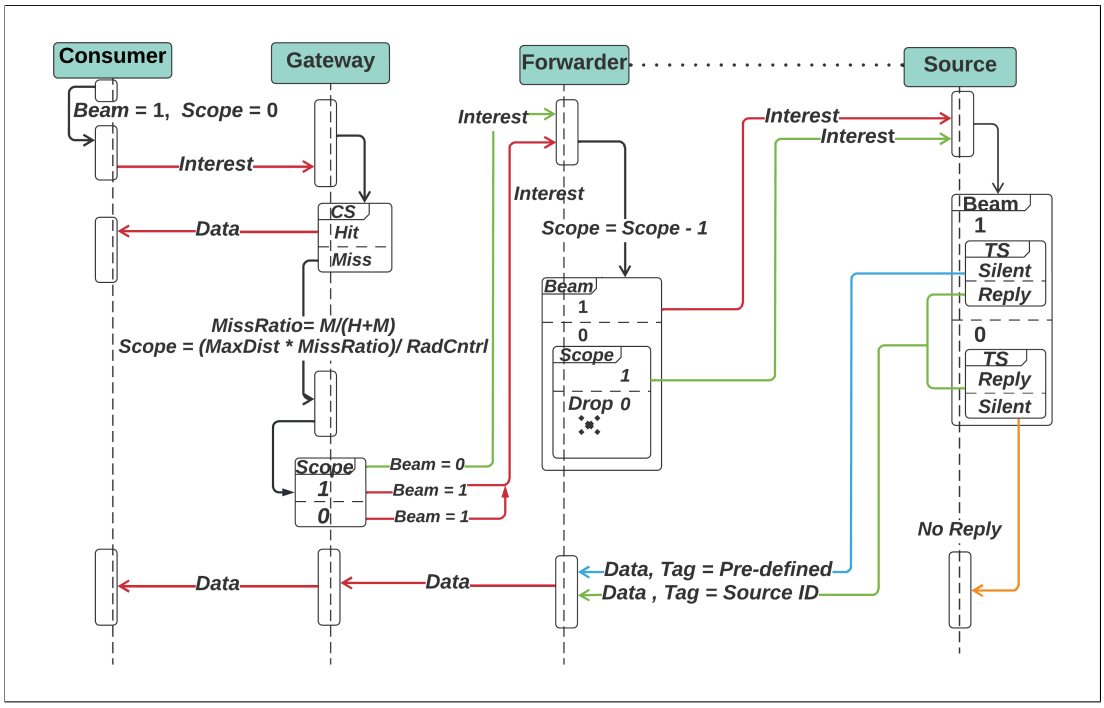


Figure 3.4: *Interest* forwarding and answering communication protocol in $SRAB^3$

The *Interest* update operations performed at the routers are trivial, and therefore the nodes are not loaded with computation burden, except computing *Scope* at the gateways. However, the *Interest* answering or *Data* delivery causes computation at the sources and for cached items only. The overhead incurred by *AckNod*e message is counteracted by suppressing the sub-optimal *Data* replies. The policy saves a lot of *CPU* cycles in compared to the strategies that compute the best path for each *Interest* forwarded.

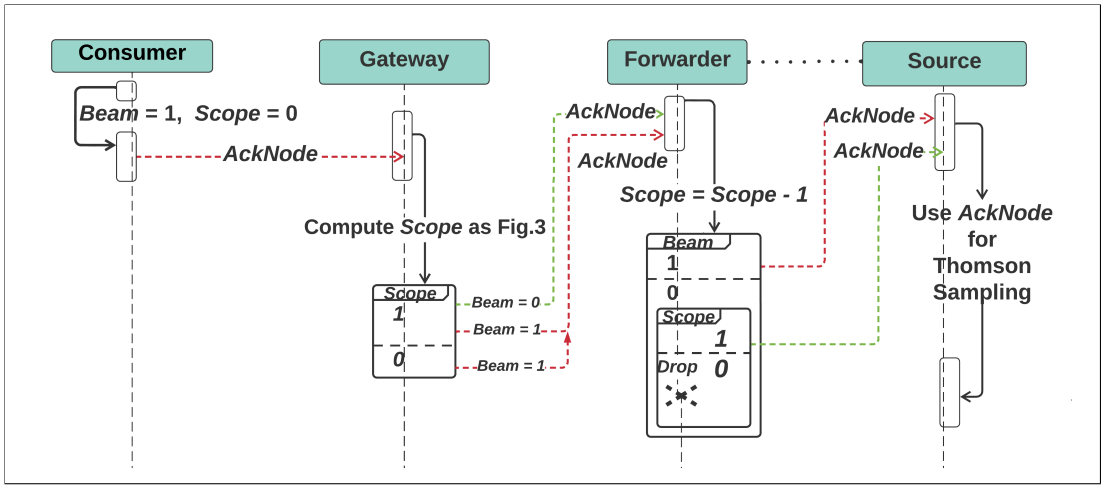


Figure 3.5: *AckNode* communication protocol in $SRAB^3$

As each source contains a learning agent that uses Thompson Sampling to decide the response type when a non-*Beam Interest* strikes, there is a possibility that all the sources stroke by the non-*Beam Interests* stay silent. $SRAB^3$ ensures content retrieval, even in such cases, through the *Beam* thread facilitating the corresponding consumer to avoid content miss. As proposed, a source requested by an *Interest*'s *Beam* thread must reciprocate with the *Reply* action, and the *Scope* of an *Interest* controls only the forwarding of its non-*Beam* threads; the *Beam* thread is free to spread in the entire network. However, in a dynamically changing network topology, where the path pursued by the *Beam* thread breaks due to any node or link failure and the sources stroke by the non-*Beam* threads decide to remain mute, the consumer has to re-transmit the *Interest* to retrieve the content. The likelihood of such scenarios is negligible. The default strategies [41], too, use re-transmission to tackle similar situations.

3.2.1.7 Thompson Sampling algorithm for *Interest* Answering

The proposed strategy intends to minimize the number of *Reply* actions, vice versa, it aims to maximize the number of *Silent* reactions. *TS* is an online learning algorithm that records the number of successes and failures of actions to estimate success probabilities, and at any time point, there is at least one action whose success probability is higher. The algorithm samples a statistically

possible success probability for each action and selects the one that yields the best estimate. If, action R has been sampled $(\alpha^f_{R,p} + \beta^f_{R,p})$ times prior to step t, yielding responses $\alpha^f_{R,p}$, $\beta^f_{R,p}$ (number of successes and failures, respectively), then the estimated success probability for action R at the tth time step computed according to eq. 3.4:

$$\hat{\theta}^f_{R,t} = X^f_{R,t} = Beta(\alpha^f_{R,t}, \beta^f_{R,t}) \tag{3.4}$$

Similarly, the estimated success probability for action S at t^{th} time slot can be calculated as eq. 3.5:

$$\hat{\theta}^f_{S,t} = X^f_{S,t} = Beta(\alpha^f_{S,t}, \beta^f_{S,t}) \tag{3.5}$$

TS algorithm for $SRAB^3$ problem is given in Algorithm 1.

Algorithm 1 Thompson Sampling for $SRAB^3(R,\ S,\ \alpha,\ \beta,\ f)$

0: **for** $t = 1, 2, ...$ **do**
0: # sample success probability:
0: **for** $a = R,\ S$ **do**
0: Sample $\hat{\theta}^f_{a,t} = X^f_{a,t} = Beta(\alpha^f_{a,t}, \beta^f_{a,t})$
0: **end for**
0: # choose and apply action as:
0: $a_t \leftarrow \underset{a}{} \hat{\theta}^f_{a,t}$
0: React with a_t and observe payout $P_{a,t}$
0: # update parameters:
0: $(\alpha_{a,t}, \beta_{a,t}) == (\alpha_{a,t} + P_{a,t},\ \beta_{a,t} + 1 - P_{a,t})$
0: **end for**=0

Chapter 4

Simulation and Results

The default *Interest* and *Data* packets [41] and structure of the *CS* are modified to implement $SRAB^3$. The *CS* modification is mainly to realize the *SucxFail* table used for Thompson Learning. As the consensus for the evaluation of *NDN* is still evolving [80], the simulation scenario considered here is comparable to much of the related literature, especially in terms of workload and network topology considered [31, 34]. In the remainder of this section, the simulation scenario, assumptions adopted, selection of parameters, and results are analyzed.

4.1 Performance Metrics

Since $SRAB^3$ attempts to balance between flooding and best-route, contrasting against flooding (*Multicast* [41]) and singe-route (*BestRoute*, built in [41]) will assess its suitability in terms of minimizing transmission overhead and reducing delay. Also, $SRAB^3$ is compared against *DTABB* [31] to evaluate its performance in suppressing the sub-optimal responses using the *Silent* action. Essential performance metrics, such as transmission overhead [100], cache hit ratio [100], latency [100], cache-eviction rate [107], and combined significance [108], are considered to compare the performance of the strategies.

The transmission overhead is the sum of the network packets, such as *Interest*, *Data*, and *AckNode* messages transmitted for all requests sent. Cache hit ratio is ratio of the *Interest*s satisfied from the local cache to the total *Interest*s arrived. Cache eviction rate is the ratio of the number of *Data* packets evicted or replaced

from the cache to the number of cache misses. The performance varies inversely with the value. Latency denotes the time required to request and download an item, averaged over all received packets. The combined significance is measured as the ratio of the packet delivery ratio (PDR) to the product of latency and transmission overhead. The combined metric indicates how resource-consuming or resource-savvy an scheme is.

Since we are proposing an *Interest* forwarding strategy that optimizes the *Data* delivery to reduce communication costs, such performance metrics are able to measure the efficiency of the proposed approach. In other words, the performance metrics can denote to what extent $SRAB^3$ reduces communication overhead without sacrificing packet delivery ratio and increasing retrieval delay. Similar metrics (or equivalent) are also considered in the literature [31, 32, 35, 80] to measure such performance of a forwarding strategy.

4.2 Simulation Topology

The *BRITE* [109] Internet topology generator is undertaken to generate a scale-free random *NDN* network topology. The arbitrary topology forms a graph of 50 nodes grouped in two autonomous systems (AS) at the top level and 25 nodes at the bottom level. The interconnections between high-level nodes and low-level nodes are generated arbitrarily using the Barabasi-Albert model [109]. The inter-AS link and the intra-AS link bandwidths are arbitrarily picked from a range of 10 Mbps to 100 Mbps. The default Barabasi-Albert scale-free topology generally encompasses a single link connecting the two ASs at the top level. Two additional edges, with link bandwidths of 100 Mbps, are connected between the ASs to allow multiple paths between those systems. A bottom-level AS node having a single physical link to the network is designated as a leaf node. A node is defined as a non-leaf node if it has more than one link to the rest of the network. Each point in the resulting graphs is obtained by repeating the corresponding simulation ten times and then averaging all runs for accuracy. In each repetition, an arbitrary network topology having an equal number of routers but different connectivity among those is assumed. Such randomness conceals many cases that prevail in real-world scenarios.

4.3 Simulation Environment

A total of five servers are configured to host a content so that an *Interest* can reach the content sources through multiple paths. Optimization of *Interest* forwarding can be realized in such scenarios. The popular Zipf-mandelbrot content popularity model [33, 110], as implemented in the ndnSIM, is assumed when a client opting for the hosted items generates *Interest*s. A catalogue of 500 to 2000 content chunks or items is considered for the hosted content, which are requested by the 100 clients. Each server and every client is attached to an arbitrary leaf node (access router). The servers follow the pre-installed producer application in ndnSIM to serve user *Interest*s.

SRAB^3 is contrasted against one single-path forwarding strategy, such as *BestRoute*, one multipath flooding-based forwarding strategy, such as *Multicast*, and one controlled-flooding strategy, for example, *DTABB*. The ndnSim has the *Multicast* and *BestRoute* algorithms pre-installed, and we implemented *DTABB* and *SRAB*^{3} in ndnSIM.

Since *SRAB*^{3} is a hybrid scheme proposing to optimize *Interest* forwarding and *Data* delivery, comparing it against those different strategies can aid in quantifying its performance in terms of reducing traffic load and retrieval delay. A separate traffic load, varying the payload size, is considered to compare the proposed and the reference strategies. So, the *Interest* requesting rate of the clients is kept common, such as 10 *Interest*s per second, for all scenarios.

4.4 Caching and Cache Size

For completeness of evaluation, (*LCE*, *LRU*) [41] are considered as the cache admission and cache eviction policies, respectively. Each simulation starts with each router having empty cache size. Though, leave copy everywhere (*LCE*) admits and thus evicts content aggressively, it is used to evaluate the performance of any proposed forwarding strategies. Additionally, each router is equipped with a constant cache size of 100 items, which means that any router can cache from 5% to 20% (in Group *III* as in Table 4.1) of the items requested by the clients.

Each client is configured to issue *Interest*s from the start of every simulation run, and each run is continued for 100 simulation seconds. The rationale for selecting the simulation time is that after the first 100 seconds, the rate of change of the performance metrics is negligible. All performance points are obtained through the mean, and standard deviation of 10 randomised simulation runs.

Four groups of simulations, each for a different parameter, are conducted to evaluate the proposed and the default strategies. Since $RadCntrl$ is a network exploration parameter and can be configured based on the network scenario or preference. The first group is executed to observe the performance of $SRAB^3$ when different ranges of the network is explored, i.e., varying $RadCntrl$ in eq. 3.3. In this experiment, the values of other control parameters are as given in Table 4.1. Constant value, $RadCntrl = 3$, is used for rest of the simulation groups. Secondly, the size of a *Data* payload is varied while keeping the popularity parameter and catalogue size fixed, for instance, $\alpha = 0.7$ and 500 chunks, respectively. The payload size varies from 512 bytes to 2048 bytes to gauge the strategies in network scenarios ranging from low to high traffic load. Thirdly, performance is tested against content popularity where the content distribution parameter α is varied while considering a constant *Data* payload size, for instance, 1024 bytes and a constant catalogue, such as 500 items. Content caching performance depends on the content popularity parameter, so the parameter is varied from $\alpha = 0.2$ to $\alpha = 1.0$ to evaluate the strategies in cache replacement scenarios ranging from frequent to infrequent. Finally, the catalogue size is varied from 500 to 2000 content chunks to assess the strategies when user requests range from common to diverse contents, while keeping the popularity parameter ($\alpha = 0.7$) and payload size (1024 bytes) unchanged. We consider an empty payload size for each *Offer* packet and greedy learning policy in *DTABB*. An empty-payload *Offer* corresponds to comparatively lower transmission overhead but increased latency. Table 4.1 gives a chart of the values of different parameters used in the three simulation groups.

4.5 Performance against Network Exploration

This experiment investigates the impact of the network exploration parameter ($RadCntrl$) of $SRAB^3$ scheme. The parameter denotes the maximum portion

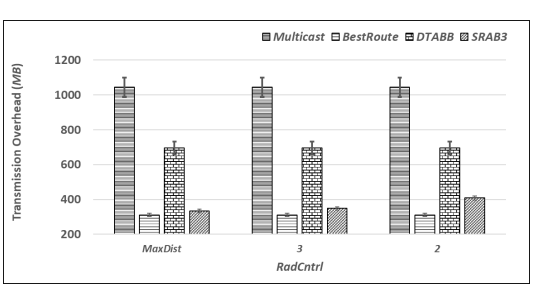
Table 4.1: Simulation parameters

Parameter	***Group I***	***Group II***	***Group III***	***Group IV***
Content popularity parameter (α)	0.7	0.7	0.2, 0.7 and 1.2	0.7
Data payload size (bytes)	1024	512, 1024, and 2048	1024	1024
Catalogue size (items)	500	500	500	500, 1000, and 2000
Caching capacity, Client size	100, 100	100, 100	100, 100	100, 100
RadCntrl	3, 2, and $MaxDist$	3	3	3
Offer payload size (bytes) ($DTABB$)	0	0	0	0

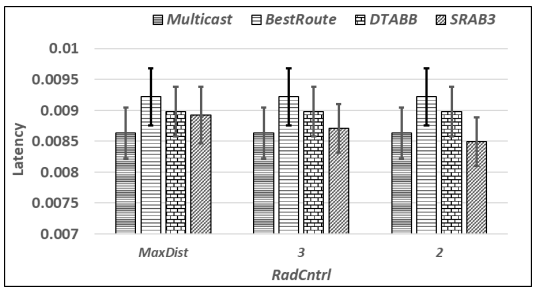
of the entire network explored by a gateway router to fetch a content. Notably, *RadCntrl* determines only the perimeter. The effective range is controlled by the remaining part of eq. 3.3.

In Fig. 4.1, it is evident that the transmission overhead of $SRAB^3$ surges as increasing portion of the network is queried, i.e., as the value of *RadCntrl* declines from $MaxDist$ to down. A lower *RadCntrl* value permits the non-*Beam* threads to exploit a better replica and hit more content sources by exploring an increased portion of the network. The increased exploration ultimately brings more *Data* in the network resulting in improved content availability, and hence reduced latency (4.1b); but, only at the cost of increased communication overload (4.1a). The increased communication cost is much lower than that of *Multicast* and *DTABB*, thanks to the *Silent* action for suppressing the sub-optimal *Data* packets.

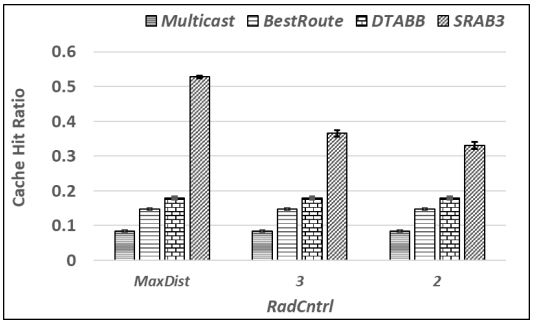
$SRAB^3$ performance converges to that of *BestRoute*, higher latency but lower communication cost, when the *RadCntrl* value is maximum. At this point, the non-*Beam* threads can hit only a few sources resulting in a desired transmission overhead performance improvement; however, the latency performance is on the lower side as the content is retrieved primarily through the *Beam* thread. The performance metrics changes steadily, as expected, while the *RadCntrl* value increases. The transmission overhead performance gap between $SRAB^3$ and *BestRoute* narrows down from 31% to 7% when *RadCntrl* changes from lowest ($RadCntrl = 2$) to highest ($RadCntrl = MaxDist$). Correspondingly, the latency performance difference broadens from 4% to 8%. The latency performance $SRAB^3$ is approximately similar to *Multicast* and *DTABB*; however, the later achieves it at the cost of exceedingly higher overhead (two to three times higher than the overhead of $SRAB^3$). The latency performance becomes better



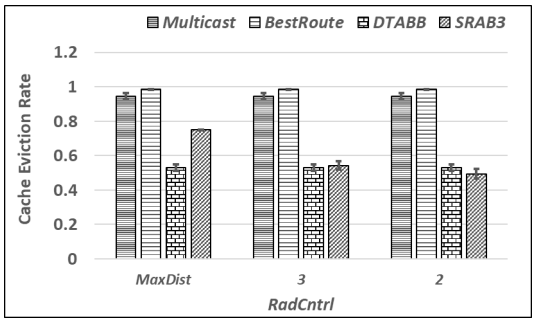
(a) Transmission overhead



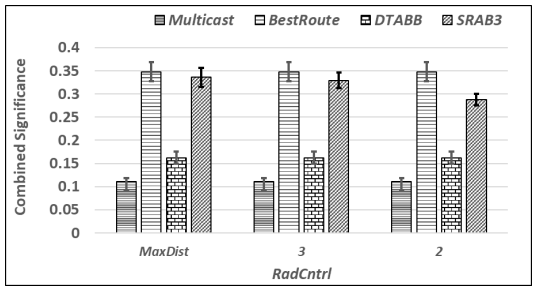
(b) Latency



(c) Cache hit ratio



(d) Cache eviction rate



(e) Combined significance

Figure 4.1: Performance evaluation and comparison against *RadCntrl*

in $SRAB^3$ as the suppressed *Data* replies reduce frequent cache admission and cache eviction, resulting in more cache hit. Notably, the comparatively high deviation of the latency metric from the average values, whenever found, can be explained by the randomization of the network topology.

A source in *DTABB* can respond with a *Data* or an *Offer* (*a.k.a* meta-data) packet when an exploratory *Interest* strikes. A consumer may need to issue an explicit *Interest* to retrieve a content when no source responds to the initial *Interest* with *Data*, alternatively when all sources respond with an *Offer* packet to that exploratory *Interest*. Thus, it shows a higher transmission overhead due to a few sub-optimal sources, in addition to the best source, responding with the *Data* packet, especially during the initialization rounds of the greedy learning process. The phase also is liable for the slightly increased latency experienced by *DTABB* as a consumer receiving only the *Offer* packets in reply to an early exploratory *Interest* must retrieve the content sending an explicit request. It is worth mentioning that $SRAB^3$ avoids the need for *Offer* packets through the *Beam* thread, and hence the corresponding exploration-exploitation phases.

In general, the cache hit ratio of $SRAB^3$ (Fig. 4.1c) is higher than *DTABB*, *BestRoute* and *Multicast* by more than 3.0, 3.5 and 6.5 times, respectively, which decreases as the value of *RadCntrl* declines. More *Interests* are transmitted for lower value of *RadCntrl* resulting in more cache miss (low cache hit). Nevertheless, Thompson Learning suppresses *Data* replies causing lower cache eviction rate (Fig. 4.1d) compared to the *Multicast* (by around 25%, 75%, and 90%) and *BestRoute* (by approximately 30%, 80%, and 100%). *DTABB* also suppresses *Data* replies in exchange of *Offer* packets. As *Offer* packets are never cached, it enjoys a lower cache eviction rate than the default strategies. The *Data* suppression also minimizes the communication cost in $SRAB^3$ and *DTABB* compared to *Multicast*. The combined significance of $SRAB^3$, as depicted in Fig. 4.1e, indicates that, it can perform remarkably well enough such that further exploration (below $RadCntrl < 3$) becomes less cost-effective.

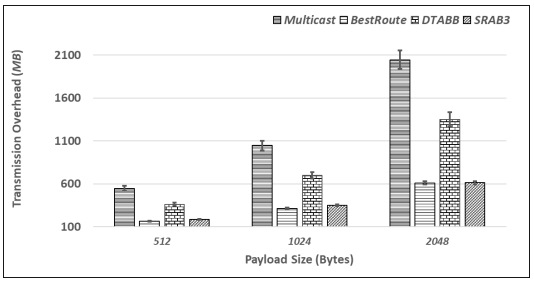
4.6 Performance against Payload Size

The performance test is executed, Fig. 4.2, to reveal that the overhead incurred by $SRAB^3$ (due to exploration) declines comparatively as the payload size increases. Since, the volume of *Interest* and *AckNode* packets remains constant as the payload size increases, the added overhead of $SRAB^3$ becomes comparatively negligible. Besides, Thompson Sampling suppresses the sub-optimal *Data* replies, alleviating network bandwidth consumption which aids in delivering the other optimal *Data* packets faster.

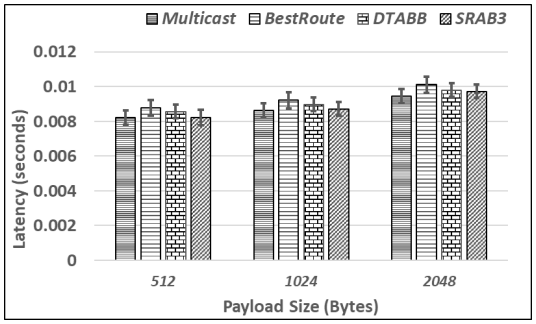
The transmission overhead difference between the $SRAB^3$ and *BestRoute* (Fig. 4.2a) declines from approximately 13% to 8% as the payload size increases from 512 bytes to 2048 bytes. The transmission overhead of *DTABB* and *Multicast*, as they explore the entire network, remains around two and three times higher, respectively, than that of $SRAB^3$ and *BestRoute*. However, the cache hit ratio (Fig. 4.2c) and cache eviction rate (Fig. 4.2d) performances of $SRAB^3$ remain higher than *Multicast* (by approximately 4.5 times and 75%, respectively), *BestRoute* (by nearly 2.5 times and 85%, respectively), and *DTABB* (by approximately 2 times and similar, respectively). Content retrieval latency (Fig. 4.2b) changes linearly *w.r.t* the size of *Data* payload. This performance of $SRAB^3$ is superior to *BestRoute* by just above 5%, and roughly identical to *DTABB* and *Multicast*. Again, the combined significance of *Multicast* is almost three times less than $SRAB^3$, the combined significance of *DTABB* is 20%-60% less than $SRAB^3$, and that of *BestRoute* becomes approximately similar to $SRAB^3$ when the payload size is higher, such as 2048 bytes.

4.7 Performance against Content Popularity

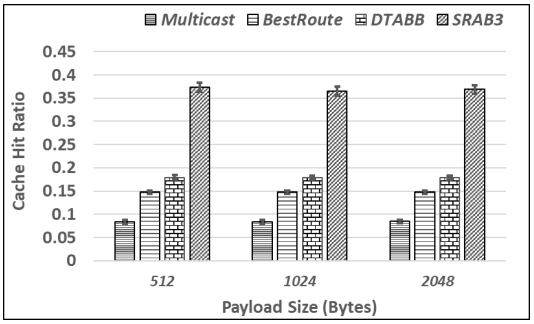
The Zipf-mandelbrot law is versatile for modelling popularity in *ICN* [33, 110, 111]. The distribution is characterized by the skew parameters, namely α and q. There is no thumb rule for parameter value settings; however, the following values of $\alpha \in [0.6, 2.5]$ and $q \in [0, 50]$ are common [80]. In our tests, we consider the parameter values, such as $\alpha \in [0.2, 1.2]$ (keeping $q = 0.7$, same as the default



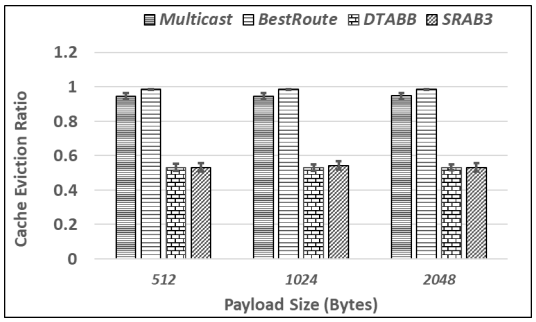
(a) Transmission overhead



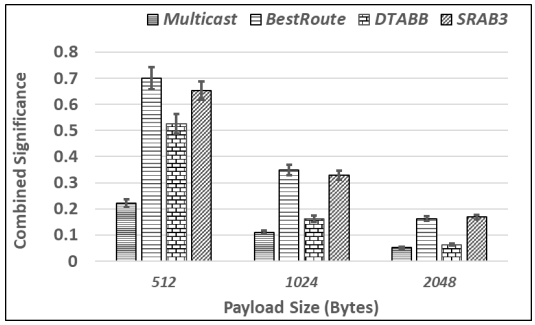
(b) Latency



(c) Cache hit ratio



(d) Cache eviction rate



(e) Combined significance

Figure 4.2: Performance evaluation and comparison against payload size

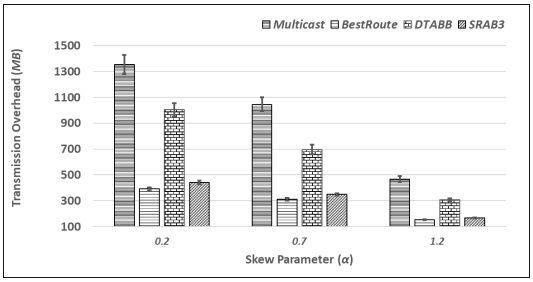
value in ndnSIM) which are realistic regarding the catalogue size explored for the experiment.

Fig. 4.3 represents impact of the content popularity model on the performance of the three competing forwarding strategies. It is observed that the in-network caching becomes effective as the skew parameter, for the same catalogue size (500), becomes closer to the higher value. The higher value of α repeats user requests on a narrower portion of the content catalogue. Therefore, the forwarding strategies perform better in all metrics, as the value of α surges.

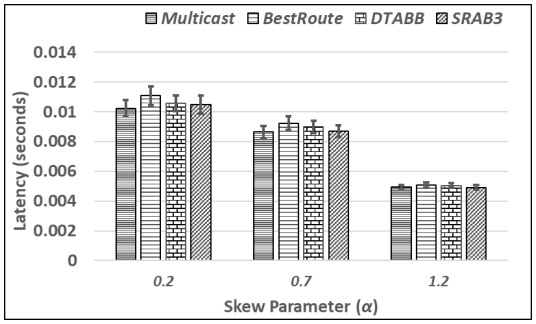
When the value of α rises, $SRAB^3$ produces a three times performance improvement compared to *Multicast*, around two times performance gain than $DTABB$, and only 10% performance degradation when contrasted against *BestRoute*, in terms of communication cost as depicted in Fig. 4.3a.

The results suggest that *Multicast* and $DTABB$ despite exploring the entire network, show identical performance as $SRAB^3$ in terms of latency, while *BestRoute* has the higher latency (around 5%) as it follows a single route (Fig. 4.3b). Thompson Sampling aids $SRAB^3$ to minimize the transmission overhead and scoped-flooding explores the required part of the network to better exploit the in-network caching compared to the reference strategies. The *Silent* action suppresses the sub-optimal responses leaving the cache space for other packets, thereby, increases the cache hit ratio by one to three times than *BestRoute* and two to six times compared with *Multicast*, as given in Fig. 4.3c. The *Offer* packets in $DTABB$ aid in achieving a higher cache hit than *Multicast* and *BestRoute*. However, $SRAB^3$ outweighs $DTABB$, by approximately 2.5-1.5 times, in terms of cache hit as more sources respond with *Data* packets that occupy the cache space. As *Silent* action stops certain *Data* packets from travelling in the network, the cache eviction rate performance of $SRAB^3$, as exhibited in Fig. 4.3d, becomes superior to *Multicast* and *BestRoute* by 1.5-2 times, and roughly equal to $DTABB$. So, a content can live in a cache for longer, which increases the cache hit ratio as a byproduct.

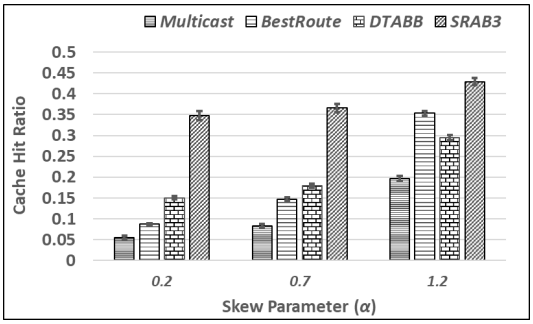
Notably, the packet delivery ratio (PDR) of all the participating forwarding strategies of all groups of simulation is higher than 0.99, and hence, the combined significance is primarily determined by transmission overhead and latency. The combined significance performance of $SRAB^3$ is superior to *Multicast* and



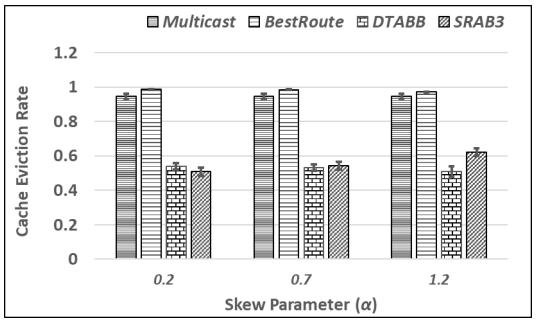
(a) Transmission overhead



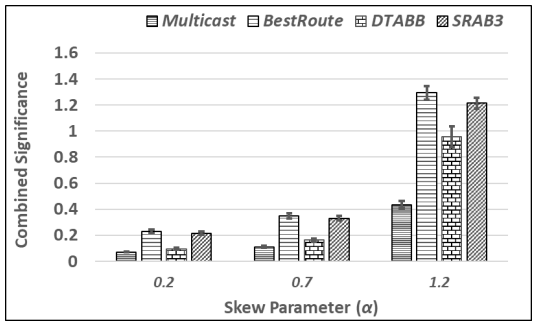
(b) Latency



(c) Cache hit ratio



(d) Cache eviction rate



(e) Combined significance

Figure 4.3: Performance evaluation and comparison against skew parameter

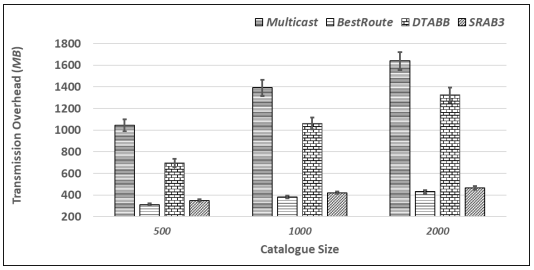
DTABB by around three and two times, respectively, and inferior to *BestRoute* only by 7%, which is constant for all values α meaning that $SRAB^3$ can perform identically for any value of α (Fig. 4.3e).

4.8 Performance against Catalogue Size

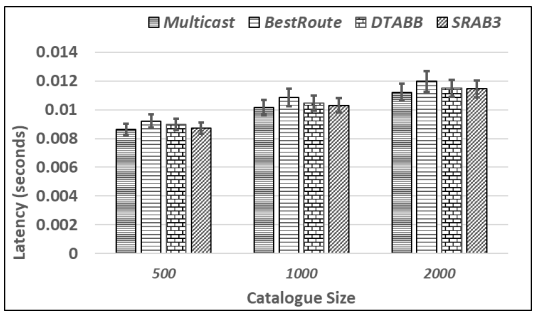
Fig. 4.4 shows the performance of the forwarding strategies when the ratio of the cache size to the catalogue size varies. As the catalogue size inclines, diverse contents are requested by the clients which essentially diminishes the in-network caching benefit. In general, the performance gap diminishes as this ratio declines (catalogue size increases). It is observed that $SRAB^3$ can find the contents more cost-effectively compared to the reference strategies.

Multicast emits the highest transmission overhead (due to flooding) while *BestRoute* is the best performer in terms of network packets. $SRAB^3$ emits slightly higher, around 9%-12%, transmission cost than *BestRoute* but around three times and two times less than the flooding strategy and *DTABB*, respectively. However, $SRAB^3$ achieves lower latency despite injecting significantly lower overhead than *Multicast* and *DTABB*. As $SRAB^3$ spares certain *Data* packets from caching, it exploits in-network caching and cache eviction more effectively than the counterparts for all the scenarios. In terms of cache hit rate, it shows a significance performance improvement of two to five times compared with *BestRoute*. Also, $SRAB^3$ outperforms *Multicast* by 4.5–10 times and *DTABB* by around two times as the catalogue size increases. Since $SRAB^3$ adjusts (adapts) the flooding ring according to the cache hit (miss) rate, it can find the content sources more efficiently. The same reason holds for shorter latency as $SRAB^3$ can hit nearer sources (if any) through the other threads. Notably, the latency performance of $SRAB^3$ is never lower than *BestRoute* as it ensures a content download through the *Beam*. $SRAB^3$ achieves similar latency performance as *Multicast* and *DTABB* but at the cost of much less overhead.

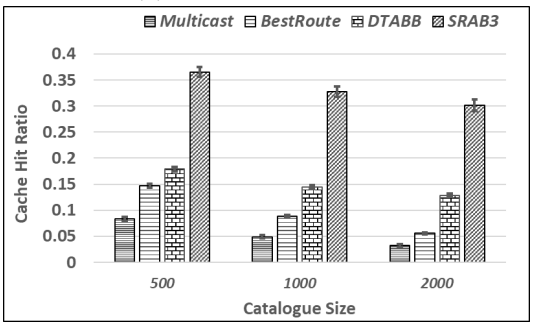
The slightly higher communication overhead incurred by $SRAB^3$ compared to *BestRoute* is due to the scoped-flooding, additional *AckNode* packets, and increased header size of *Interest* and *Data* packets. Nevertheless, $SRAB^3$ retrieves



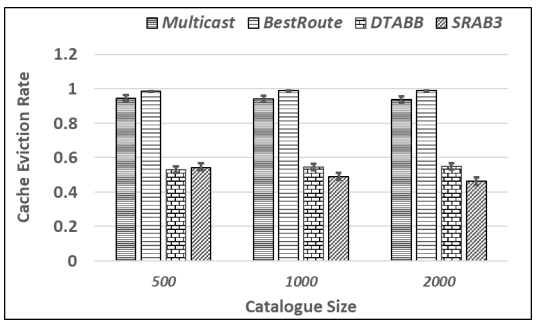
(a) Transmission overhead



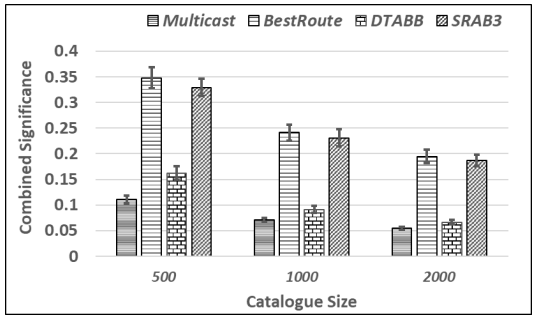
(b) Latency



(c) Cache hit ratio



(d) Cache eviction rate



(e) Combined significance

Figure 4.4: Performance evaluation and comparison against catalogue size

the content from nearer sources. The transmission overhead is less than *Multicast* and *DTABB* due to the scoped flooding and *Silent* action as suggested by Thompson Sampling.

The combined significance indicates that *Multicast* is the most resource-intensive scheme, meaning that it consumes higher network bandwidth and nodes CPU compared to the others. Although $SRAB^3$ pumps slightly more overhead than *BestRoute*, it is almost as resource-savvy as the single-route strategy for higher catalogue size.

Chapter 5

Conclusions and Future Works

5.1 Conclusions

This thesis investigates a receiver-cum-source-driven forwarding strategy, namely $SRAB^3$, which employs Thompson Sampling (a reinforcement learning algorithm) to control *Interest* forwarding and answering in *NDN*. The proposed strategy retrieves a content not only through the best link from a receiver's perspective, it retrieves the same content from an alternative (better) source considering the sources' viewpoint. The alternative sources to be explored and exploited is adapted dynamically based on network dynamics, for instant, cache hit/miss ratio.

$SRAB^3$ is contrasted against two legendary but opposite forwarding strategies: one that uses *Interest* flooding (*Multicast*) and other that uses single- or best-link suggested by the routing protocol (*BestRoute*). Moreover, $SRAB^3$ is evaluated against its main and recent counterpart, for instant, *DTABB*. The performance of the forwarding strategies is evaluated under three different network scenarios, such as varying catalogue size, payload size, and content popularity distribution.

Simulation results suggest that when the clients request diverse contents, $SRAB^3$ can achieve better transmission overhead performance than *Multicast* and *DTABB* (by at most 3.5 and 2.5 times, respectively) and latency performance than *BestRoute* (by around 10%). Furthermore, the results also indicate that for larger payload size the transmission overhead gap between $SRAB^3$ and *Multicast*

and *DTABB* increases and that of between *BestRoute* diminishes. However, the performance of $SRAB^3$ can be optimized, such as trade-off between communication overhead and latency, by adjusting the value of the *RadCntrl* parameter.

5.2 Limitations

A few flaws encountered by $SRAB^3$ are listed below:

- In this work, the value of the *RadCntrl* parameter is determined heuristically, which may increase the redundant replies.
- The number of *AckNode* messages increases proportionally with the number of sources in the network, causing increased network overhead.
- The implementation of *SucxFail* table in the sources consumes memory space.

5.3 Future Work

One promising research dimension is to extend $SRAB^3$ for large-scale deployment of *NDN*, for instant *SDN*-based *NDN*. Currently, multiple sources can send *Data* back to the consumer (occasionally), which is accountable for increased overhead. The *SDN*-based extension will be exploited to ensure *Data* reply from only a single source. Furthermore, the proposed strategy will be assessed for resource constrained environments with the presence of denial of service attacks.

Bibliography

[1] 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, 2013. 1, 9

[2] A. Kerrouche, "Routing named data in information-centric networks," Ph.D. dissertation, Université Paris-Est, 2017. 1, 12, 17, 26

[3] Cisco annual internet report (2018–2023) white paper. 2020. [Online]. Available:. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/whitepaper-c11-741490.html 1, 4

[4] 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, 2012. 2

[5] 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, 2013. 2

[6] R. A. Rehman and B.-S. Kim, "Lomcf: Forwarding and caching in named data networking based manets," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 10, pp. 9350–9364, 2017. 2

[7] P. Wendell and M. J. Freedman, "Going viral: flash crowds in an open cdn," in *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, 2011, pp. 549–558. 2

[8] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014. 2, 3

[9] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, 2007, pp. 181–192. 2, 10

[10] B. Ahlgren, M. D'Ambrosio, M. Marchisio, I. Marsh, C. Dannewitz, B. Ohlman, K. Pentikousis, O. Strandberg, R. Rembarz, and V. Vercellone, "Design considerations for a network of information," in *Proceedings of the 2008 ACM CoNEXT Conference*, 2008, pp. 1–6. 2, 10

[11] S. Tarkoma, M. Ain, and K. Visala, "The publish/subscribe internet routing paradigm (psirp): Designing the future internet architecture." in *Future Internet Assembly*, 2009, pp. 102–111. 2, 10, 11

[12] C. Tsilopoulos, G. Xylomenos, and Y. Thomas, "Reducing forwarding state in content-centric networks with semi-stateless forwarding," in *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE, 2014, pp. 2067–2075. 2, 10

[13] Comet. [Online]. Available:. [Online]. Available: http://www.comet-project.org 2, 10

[14] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, 2007, pp. 181–192. 2

[15] D. R. Cheriton and M. Gritter, "Triad: A new next-generation internet architecture," 2000. 2

[16] J. Li, R.-c. Xie, T. Huang, and L. Sun, "A novel forwarding and routing mechanism design in sdn-based ndn architecture," *Frontiers of Information Technology & Electronic Engineering*, vol. 19, no. 9, pp. 1135–1150, 2018. 2

[17] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, B. Zhang, G. Tsudik, D. Massey, C. Papadopoulos *et al.*, "Named data networking (ndn) project," *Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC*, vol. 157, p. 158, 2010. 4

[18] T. A. Kelley, "A note on the bernoulli two-armed bandit problem," *The Annals of Statistics*, vol. 2, no. 5, pp. 1056–1062, 1974. 4

[19] O.-C. Granmo, "Solving two-armed bernoulli bandit problems using a bayesian learning automaton," *International Journal of Intelligent Computing and Cybernetics*, 2010. 4

[20] H. Yan, D. Gao, W. Su, and H.-C. Chao, "A forwarding strategy of counteracting redundancy data in named data networking," *International Journal of Communication Systems*, vol. 28, no. 18, pp. 2289–2310, 2015. 5, 6, 28

[21] H. Qian, R. Ravindran, G.-Q. 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)*. IEEE, 2013, pp. 1094–1101. 5, 29

[22] B. Hao, G. Wang, M. Zhang, J. Zhu, L. Xing, and Q. Wu, "Stochastic adaptive forwarding strategy based on deep reinforcement learning for secure mobile video communications in ndn," *Security and Communication Networks*, vol. 2021, 2021. 5

[23] Y. Mordjana, B. Djamaa, and M. R. Senouci, "A q-learning based forwarding strategy for named data networking," in *2021 International Conference on Networking and Advanced Systems (ICNAS)*. IEEE, 2021, pp. 1–6.

[24] S. Ryu, I. Joe, and W. Kim, "Intelligent forwarding strategy for congestion control using q-learning and lstm in named data networking," *Mobile Information Systems*, vol. 2021, 2021.

[25] H. Li, "Icn routing strategy based on reinforcement learning and neural network," *Internet Technology Letters*, vol. 4, no. 5, p. e266, 2021.

[26] N. Dutta, "A bargain game theory assisted interest packet forwarding strategy for information centric network," *Journal of Network and Computer Applications*, p. 103546, 2022.

[27] C. Pu, ": Mcdm-based interest forwarding and cooperative data caching for named data networking," *Journal of Computer Networks and Communications*, vol. 2021, 2021.

[28] S. Shi, J. Li, B. Han, H. Wu, Y. Ma, Q. Dong, and H. D. Schotten, "Multipath forwarding strategy for named data networking based on pending interests and available bandwidth," in *2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*. IEEE, 2021, pp. 731–737. 28

[29] M. R. Senouci, A. Habbouchi, and Y. Mordjana, "Bandit algorithms for robust forwarding in named data networking," in *International Conference on Computing Systems and Applications*. Springer, 2022, pp. 211–221.

[30] F. Abdi, M. Ahmadi, and M. Ghanem, "La-mdpf: A forwarding strategy based on learning automata and markov decision process in named data networking," *Future Generation Computer Systems*, vol. 134, pp. 22–39, 2022. 5

[31] S. M. Iqbal, M. M. Hoque *et al.*, "A source-driven reinforcement learning-based data reply strategy to reduce communication overhead in named data networks (ndn)," *Cluster Computing*, pp. 1–27, 2021. 5, 6, 7, 28, 31, 35, 48, 49

[32] S. M. A. Iqbal and M. M. Hoque, "A source-driven probabilistic forwarding and caching strategy in ndn and sdn-based ndn," *International Journal of Communication Systems*, p. e5093. 6, 28, 31, 49

[33] G. Rossini and D. Rossi, "Coupling caching and forwarding: Benefits, analysis, and implementation," in *Proceedings of the 1st ACM Conference on Information-Centric Networking*, 2014, pp. 127–136. 6, 28, 31, 50, 55

[34] S. M. A. Iqbal *et al.*, "Adaptive forwarding strategies to reduce redundant interests and data in named data networks," *Journal of Network and Computer Applications*, vol. 106, pp. 33–47, 2018. 6, 28, 30, 31, 48

[35] L. Huo, "Packet-level-based traffic aggregation to optimize ndn content delivery," *International Journal of Communication Systems*, vol. 33, no. 12, p. e4473, 2020. 6, 49

[36] 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*, 2015, pp. 9–18. 6

[37] M. Badov, A. Seetharam, J. Kurose, V. Firoiu, and S. Nanda, "Congestion-aware caching and search in information-centric networks," in *Proceedings of the 1st ACM Conference on Information-Centric Networking*, 2014, pp. 37–46. 6

[38] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, no. 2, pp. 235–256, 2002. 6

[39] O. Chapelle and L. Li, "An empirical evaluation of thompson sampling," *Advances in neural information processing systems*, vol. 24, pp. 2249–2257, 2011.

[40] S. Agrawal and N. Goyal, "Analysis of thompson sampling for the multi-armed bandit problem," in *Conference on learning theory*. JMLR Workshop and Conference Proceedings, 2012, pp. 39–1. 6

[41] S. Mastorakis, A. Afanasyev, and L. Zhang, "On the evolution of ndnSIM: an open-source simulator for NDN experimentation," *ACM Computer Communication Review*, Jul. 2017. 7, 46, 48, 50

[42] Information-centric networking: Baseline scenarios. [Online]. Available:. [Online]. Available: https://datatracker.ietf.org/doc/rfc7476/ 9

[43] Sail. [Online]. Available:. [Online]. Available: http://www.sail-project.eu 10

[44] The convergence project. [Online]. Available:. [Online]. Available: http://www.ict-convergence.eu 10

[45] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, B. Zhang, G. Tsudik, D. Massey, C. Papadopoulos *et al.*, "Named data networking (ndn) project," *Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC*, vol. 157, p. 158, 2010. 10, 13

[46] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014. x, 10, 13, 17, 22, 25

[47] N. Fotiou, D. Trossen, and G. C. Polyzos, "Illustrating a publish-subscribe internet architecture," *Telecommunication Systems*, vol. 51, no. 4, pp. 233–245, 2012. 11

[48] D. Trossen and G. Parisis, "Designing and realizing an information-centric internet," *IEEE Communications Magazine*, vol. 50, no. 7, pp. 60–67, 2012. 11

[49] N. Fotiou, P. Nikander, D. Trossen, and G. C. Polyzos, "Developing information networking further: From psirp to pursuit," in *International Conference on Broadband Communications, Networks and Systems*. Springer, 2010, pp. 1–13. 11

[50] 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, 2012. 12

[51] Z. I. Kiss, Z. A. Polgar, C. Vinti, M. Varga, A. B. Rus, and V. Dobrota, "Network coding-based congestion control at network layer: Protocol design and evaluation," *International Journal of Computer Networks and Communications*, vol. 3, no. 1, pp. 119–138, 2011. 13

[52] 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*, 2009, pp. 1–12. x, 16, 17, 18, 19, 25

[53] Domain names - implementation and specification. [Online]. Available:. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc1035 19

[54] H. Yan, D. Gao, W. Su, and H.-C. Chao, "A forwarding strategy of counteracting redundancy data in named data networking," *International Journal of Communication Systems*, vol. 28, no. 18, pp. 2289–2310, 2015.

[55] H. Qian, R. Ravindran, G.-Q. 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)*. IEEE, 2013, pp. 1094–1101. 19

[56] C. Yi, A. Afanasyev, L. Wang, B. Zhang, and L. Zhang, "Adaptive forwarding in named data networking," *ACM SIGCOMM computer communication review*, vol. 42, no. 3, pp. 62–67, 2012. 20, 25, 26

[57] C. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, and L. Zhang, "A case for stateful forwarding plane," *Computer Communications*, vol. 36, no. 7, pp. 779–791, 2013. x, 20, 21, 24, 25, 26

[58] 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*, 2014, pp. 27–36. 26

[59] V. Jacobson, J. Burke, L. Zhang, T. Abdelzaher, B. Zhang, P. Crowley, J. A. Halderman, C. Papadopoulos, and L. Wang, "Named data networking next phase (ndn-np) project may 2015-april 2016 annual report," *Named Data Networking (NDN)*, 2016. x, 27

[60] 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)*. IEEE, 2016, pp. 1–7. 28, 29, 30, 31

[61] M. Kuai, X. Hong, and R. R. Flores, "Evaluating interest broadcast in vehicular named data networking," in *2014 Third GENI Research and Educational Experiment Workshop*. IEEE, 2014, pp. 77–78. 28

[62] S. Ahdan, H. Situmorang, and N. R. Syambas, "Effect of overhead flooding on ndn forwarding strategies based on broadcast approach," in *2017 11th International Conference on Telecommunication Systems Services and Applications (TSSA)*. IEEE, 2017, pp. 1–4.

[63] Y.-T. Yu, R. B. Dilmaghani, S. Calo, M. Sanadidi, and M. Gerla, "Interest propagation in named data manets," in *2013 international conference on computing, networking and communications (ICNC)*. IEEE, 2013, pp. 1118–1122.

[64] 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)*. IEEE, 2016, pp. 1–7.

[65] 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*, 2015, pp. 9–18.

[66] M. Amadeo, A. Molinaro, and G. Ruggeri, "E-chanet: Routing, forwarding and transport in information-centric multihop wireless networks," *Computer communications*, vol. 36, no. 7, pp. 792–803, 2013.

[67] S. Dash, B. J. Sahu, N. Saxena, and A. Roy, "Flooding control in named data networking," *IETE Technical Review*, vol. 35, no. 3, pp. 266–274, 2018.

[68] A. Ioannou and S. Weber, "A survey of caching policies and forwarding mechanisms in information-centric networking," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2847–2886, 2016. 28

[69] D. H. Lee, K. Thar, D. Kim, and C. S. Hong, "Efficient parallel multi-path interest forwarding for mobile user in ccn," in *2016 international conference on information networking (ICOIN)*. IEEE, 2016, pp. 390–394. 28

[70] A. Z. Khan, S. Baqai, and F. R. Dogar, "Qos aware path selection in content centric networks," in *2012 IEEE International Conference on Communications (ICC)*. IEEE, 2012, pp. 2645–2649. 28

[71] M. Alhowaidi, D. Nadig, B. Ramamurthy, B. Bockelman, and D. Swanson, "Multipath forwarding strategies and sdn control for named data networking," in *2018 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*. IEEE, 2018, pp. 1–6. 28

[72] Y. Zhang, J. Liu, T. Huang, J. Chen, and Y. Liu, "Multi-path interests routing scheme for multi-path data transfer in content centric networking," 2013.

[73] D. H. Lee, K. Thar, D. Kim, and C. S. Hong, "Efficient parallel multi-path interest forwarding for mobile user in ccn," in *2016 international conference on information networking (ICOIN)*. IEEE, 2016, pp. 390–394.

[74] D. Rossi and G. Rossini, "Caching performance of content centric networks under multi-path routing (and more)," *Relatório técnico, Telecom ParisTech*, vol. 2011, pp. 1–6, 2011.

[75] A. Udugama, S. Palipana, and C. Goerg, "Analytical characterisation of multi-path content delivery in content centric networks," in *2013 Conference on Future Internet Communications (CFIC)*. IEEE, 2013, pp. 1–7.

[76] X. Hu, S. Zheng, G. Zhang, L. Zhao, G. Cheng, J. Gong, and R. Li, "An on-demand off-path cache exploration based multipath forwarding strategy," *Computer Networks*, vol. 166, pp. 107 032–107 050, 2020.

[77] A. Udugama, X. Zhang, K. Kuladinithi, and C. Goerg, "An on-demand multi-path interest forwarding strategy for content retrievals in ccn," in *2014 IEEE network operations and management symposium (NOMS)*. IEEE, 2014, pp. 1–6.

[78] G. Zhang, H. Li, T. Zhang, D. Li, and L. Xu, "A multi-path forwarding strategy for content-centric networking," in *2015 IEEE/CIC International Conference on Communications in China (ICCC)*. IEEE, 2015, pp. 1–6. 28

[79] 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)*. IEEE, 2015, pp. 5665–5671. 28, 29, 30

[80] O. Akinwande, "Interest forwarding in named data networking using reinforcement learning," *Sensors*, vol. 18, no. 10, p. 3354, 2018. 28, 30, 48, 49, 55

[81] A. Afanasyev, J. Shi, B. Zhang, L. Zhang, I. Moiseenko, Y. Yu, W. Shang, Y. Huang, J. P. Abraham, S. DiBenedetto *et al.*, "Nfd developer's guide," *Dept. Comput. Sci., Univ. California, Los Angeles, Los Angeles, CA, USA, Tech. Rep. NDN-0021*, 2014. 29, 31

[82] 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)*. IEEE, 2016, pp. 1–10. 29, 31

[83] B. Li, M. Ma, and X. Yang, "Advanced perceptive forwarding in content-centric networks," *IEEE Access*, vol. 5, pp. 4595–4605, 2017. 29, 31

[84] C. Li, K. Okamura, and W. Liu, "Ant colony based forwarding method for content-centric networking," in *2013 27th international conference on advanced information networking and applications workshops*. IEEE, 2013, pp. 306–311.

[85] A. Kerrouche, M. R. Senouci, A. Mellouk, and T. Abreu, "Ac-qos-fs: Ant colony based qos-aware forwarding strategy for routing in named data networking," in *2017 IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–6.

[86] P. Moll, J. Janda, and H. Hellwagner, "Adaptive forwarding of persistent interests in named data networking," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*, 2017, pp. 180–181. 31

[87] A. Marandi, V. Hofer, M. Gasparyan, T. Braun, and N. Thomos, "Bloom filter-based routing for dominating set-based service-centric networks," in *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2020, pp. 1–9. 31

[88] I. V. Bastos and I. M. Moraes, "A diversity-based search-and-routing approach for named-data networking," *Computer Networks*, vol. 157, pp. 11–23, 2019. 31

[89] M. Abdelaal, M. Karadeniz, F. Dürr, and K. Rothermel, "litendn: Qos-aware packet forwarding and caching for named data networks," in *2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2020, pp. 1–9. 29, 31

[90] 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, 2016. 29

[91] 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*, 2013, pp. 9–14. 30, 31

[92] 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)*. IEEE, 2015, pp. 1–7.

[93] C. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, and L. Zhang, "A case for stateful forwarding plane," *Computer Communications*, vol. 36, no. 7, pp. 779–791, 2013.

[94] I. Araujo, A. Silva, A. Klautau, and N. Linder, "Qos in forwarding strategies for icn: New algorithm and experimental evaluation," *Journal of Communication and Information Systems*, vol. 34, no. 1, pp. 275–286, 2019. 29, 31

[95] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018. 30

[96] Y. Zhang, B. Bai, K. Xu, and K. Lei, "Ifs-rl: An intelligent forwarding strategy based on reinforcement learning in named-data networking," in *Proceedings of the 2018 Workshop on Network Meets AI & ML*, 2018, pp. 54–59. 30, 31

[97] O. Akinwande and E. Gelenbe, "A reinforcement learning approach to adaptive forwarding in named data networking," in *International Symposium on Computer and Information Sciences*. Springer, 2018, pp. 211–219. 30, 31

[98] J. Lv, X. Tan, Y. Jin, and J. Zhu, "Drl-based forwarding strategy in named data networking," in *2018 37th Chinese Control Conference (CCC)*. IEEE, 2018, pp. 6493–6498. 30, 31

[99] B. Fu, L. Qian, Y. Zhu, and L. Wang, "Reinforcement learning-based algorithm for efficient and adaptive forwarding in named data networking," in *2017 IEEE/CIC International Conference on Communications in China (ICCC)*. IEEE, 2017, pp. 1–6. 30

[100] M. Zhang, X. Wang, T. Liu, J. Zhu, and Q. Wu, "Afsndn: A novel adaptive forwarding strategy in named data networking based on q-learning," *Peer-to-Peer Networking and Applications*, vol. 13, no. 4, pp. 1176–1184, 2020. 30, 31, 48

[101] C. Pu, "Adaptive forwarding strategy based on mcdm model in named data networking," 2020. 31

[102] Y. A. B. de Sena, K. L. Dias, and C. Zanchettin, "Dqn-af: Deep q-network based adaptive forwarding strategy for named data networking," in *2020 IEEE Latin-American Conference on Communications (LATINCOM)*. IEEE, 2020, pp. 1–6. 31

[103] A. K. Gupta, *Beta Distribution*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 144–145. 33

[104] E. Artin, *The gamma function*. Courier Dover Publications, 2015. 34

[105] K. Avrachenkov and P. Jacko, "Ccn interest forwarding strategy as multi-armed bandit model with delays," in *2012 6th International Conference on Network Games, Control and Optimization (NetGCooP)*. IEEE, 2012, pp. 38–43. 35

[106] S. Mastorakis, A. Afanasyev, and L. Zhang, "On the evolution of ndnsim: An open-source simulator for ndn experimentation," *ACM SIGCOMM Computer Communication Review*, vol. 47, no. 3, pp. 19–33, 2017. 44

[107] A. Ioannou and S. Weber, "A survey of caching policies and forwarding mechanisms in information-centric networking," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2847–2886, 2016. 48

[108] A. T. Prodhan, R. Das, H. Kabir, and G. C. Shoja, "Ttl based routing in opportunistic networks," *Journal of Network and Computer Applications*, vol. 34, no. 5, pp. 1660–1670, 2011. 48

[109] A. Medina, I. Matta, and J. Byers, "Brite: a flexible generator of internet topologies," 2000. 49

[110] G. Carofiglio, M. Gallo, L. Muscariello, and D. Perino, "Modeling data transfer in content-centric networking," in *2011 23rd International Teletraffic Congress (ITC)*. IEEE, 2011, pp. 111–118. 50, 55

[111] L. Muscariello, G. Carofiglio, and M. Gallo, "Bandwidth and storage sharing performance in information centric networking," in *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*, 2011, pp. 26–31. 55