APPLICATION OF BLOOM FILTER IN NDN

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Computer Science and Engineering

Submitted By
Harpreet Kaur
(Roll No. 801732017)

Under the supervision of:
Dr. Shalini Batra
Associate Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY
PATIALA – 147004

June 2019

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "*Application of Bloom Filter In NDN*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Computer Science and Engineering* submitted in Computer Science and Engineering Department of Thapar Institute of Engineering and Technology, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Shalini Batra* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Signature:

(Harpreet Kaur)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Dr. Shalini Batra)

Associate Professor,

Computer Science and Engineering

Department

ACKNOWLEDGEMENT

I express my sincerest regards and gratitude to my supervisor Dr. Shalini Batra, Associate Professor, Department of Computer Science and Engineering, Thapar Institute, Patiala for her valuable guidance and suggestions. Without her encouragement and guidance, thesis would not have been materialized. I feel privileged to offer my sincere thanks and owe an enormous deal of gratitude to my supervisor for her guidance and gave me full time to understand the minute details of each and every step, for successful completion of thesis.

I would like to express my gratitude to Dr Ashutosh Mishra, Assistant Professor Department of Computer Science and Engineering, Thapar Institute, Patiala for his kind cooperation and encouragement which helped in the completion of this work.

The generous support of all the staff members of Computer Science Department is greatly appreciated. I would like to express my heartiest thanks to my parents and friends for their help and wishes for the successful completion of this work.

Above all, I express my indebtedness to the "**ALMIGHTY**" for all his blessings and kindness.

Harpreet Kaur
(Roll No 801732017)

ABSTRACT

In this thesis, optimal managing of the NDN router limit caching space and improved NDN data storage, data packet forwarding is proposed, and improvement in routing performance. In addition, the interest packet name searching in CS, PIT, and FIB is given the longest-prefix matching strategy in the NDN router, and the name sizes with different name contents differ. Moreover, some name sizes can reach hundreds of octets and thus will need a large caching space. Therefore, name searching in CS is more complex than that of IP based networks and requires a longer time. Optimizations of caching techniques are compared and data content searching in CS are critical issues is addressed.

The existing NDN caching optimization approaches have low caching efficiency and a slow data-content searching speed. To overcome these limitations, a Bloom-filter-based caching approach is proposed in this project to promote the caching efficiency and data content searching speed.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Introduction to NDN

Communication models are currently based on fast-growing content-type applications. The majority of network services are composed of content-based services.User's currently concentrate on the content itself,together with velocity, quality, retrieval of content and transmission security[1] rather than focus on where the content stored. The content delivery and content safety is also significant parameters.

The NDN conceptis future model of Internet architecture[1, 2]. Named information networking (NDN)[3] is a content-centriccommunication in which routing, forwarding and caching are based on the content.NDN can therefore be used to construct communication systems based on named information instead ofnode-to-node links, thus creating a solid, easy and scalable network. There are two kinds of packetsin NDN:

1. Interest packet: This packet contains the names of desired information/data for subscribers.
2. Data packet: Data packets that contain subscriber information material. A subscriber sends an interest packet to the NDN transport network to investigate prospective target data with
   the name of the information item to receive an information item.

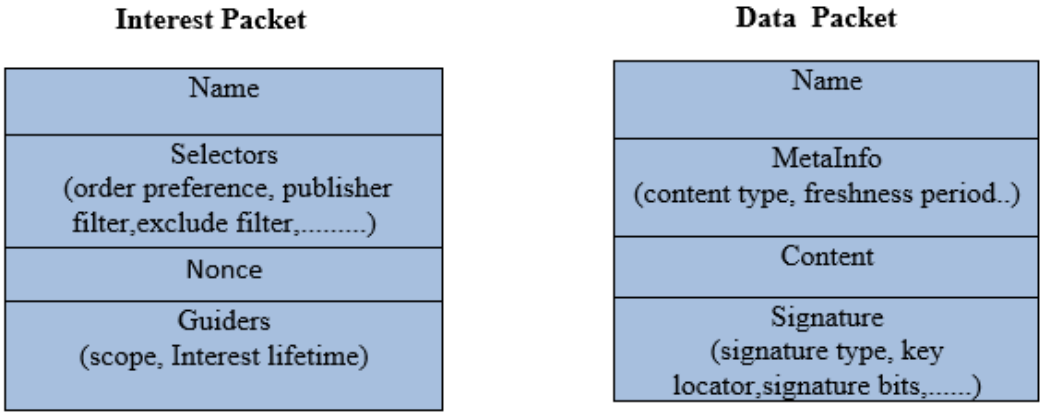

Figure 1.1: Types of NDN Packets

1.1.1 NDN ARCHITECTURE

The required data name is splitintopieces in the Name Data Network and hierarchical names are allocated for their identity and packet forwarding methods are used, hence the use of IP addresses are avoided [4].A subscriber-producer based communication model is used in NDN architecture for the interaction between NDN nodes ,when the request is generated by the subscriber then the content will be supplied in reaction to the associated request. NDN users exchange only two kinds of messages, i.e. data and interest packets . Each NDN router retains three data structures to perform the Interest and Data packet forwarding tasks:

i. Content Store (CS): The data packet is temporary stored in a CS together with other required fields,to fulfil future interests.An NDN router checks the Content Store for matching information when an Interest packet comes ;If it occurs, the Data Packet will be returned to the interface from which the Interest originated.If not, the router will search the name in Pending interest table(PIT) of NDN.If a corresponding entry exists, the incoming interface of this interest is simply recorded in the PIT entry.

ii. Pending Interest Table: An NDN router checks the Content Store for matching information when an Interest packet comes ;If it occurs, the Data Packet will be returned to the interface from which the Interest originated.If not, the router will search the name in PIT of NDN.If a corresponding entry exists, the incoming interface of this interest is simply recorded in the PIT entry. If not, interest packet will be forwarded to FIB.Upon arrival of a data packet, an NDN router will find the corresponding PIT entry and the data has been forward to all downstream interfaces that is listed in PIT entry.After that PIT entry of interest packet will be removed and data packet will stored in Content store for future utilization.

iii. Forwarding Information Base (FIB):It is a routing table that maps interface name component.The interest packet will be forwarded by the routerto the data producer(s) based on information in the FIB or Forwarding Strategy.If a router gets interest from various downstream nodes for the same name, it will only forward the first upstream to the information producer(s).

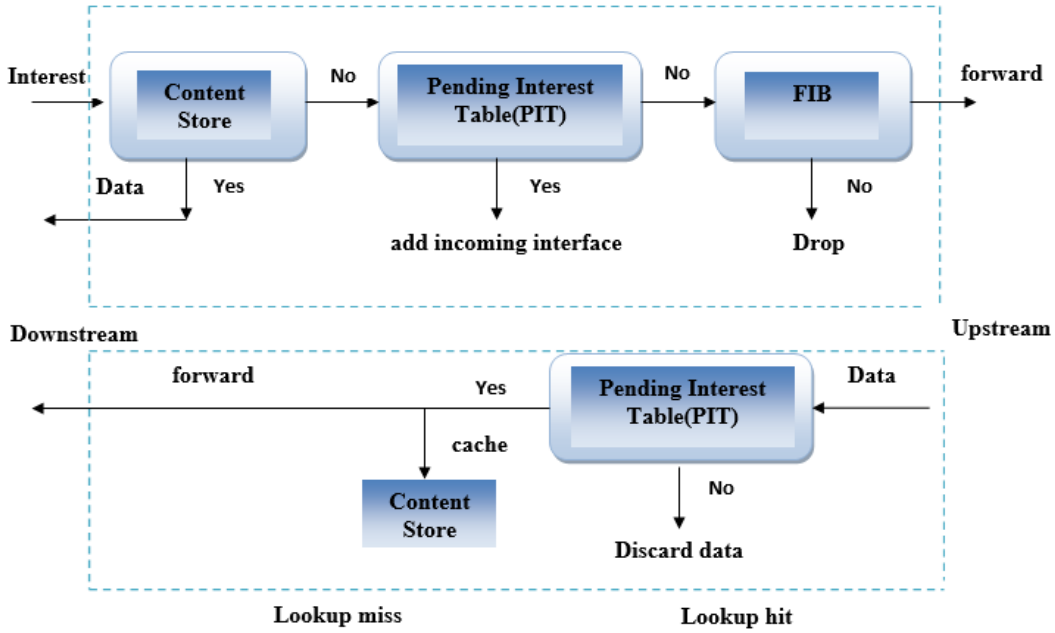

Figure 1.2: Forwarding Process at NDN Node .

In critical situation, interest packets are dropped by Forwarding Strategy . If there is any suspicious DoS attack on interest packet or overloaded of upstream link, forwarding strategy fetches FIB entries for each Interest and decides when and where to forward the Interest.Data packets always follow Interests 's inverse route, and in the case of packet losses, in order to preserve the flow balance, one Interest packet should be available in one data packet on each link.Interests plays a comparable role in managing traffic flow as TCP ACKs in today's Internet to collect big content items that comprise various packets:a fine-grained feedback loop regulated by information consumers.

In NDN architecture no host address or interface addresses are used . Routers only forward the Interest packets in the direction of data producers that is based on the requested names is loaded in the packets, and Data packetsare forwarded to

consumers that is based on its PIT state information,trapped by the requested Interests by consumers at each node. This Interest / Data packet exchange guarantees hop-by-hop control loop symmetry and thus eliminates the need for any other concept of source or destination nodes in data packet transmission, unlike the present end-to-end IP packet distribution model.

The CS stores data generated from publishers. The PIT works with set of rules to handle interest packet generated by its subscribers who are waiting to receive the required data packets. If a data packet is received, the corresponding interest packet will be deleted[5]. Therefore, an optimized caching strategy is critical and open for research in NDN communications community.

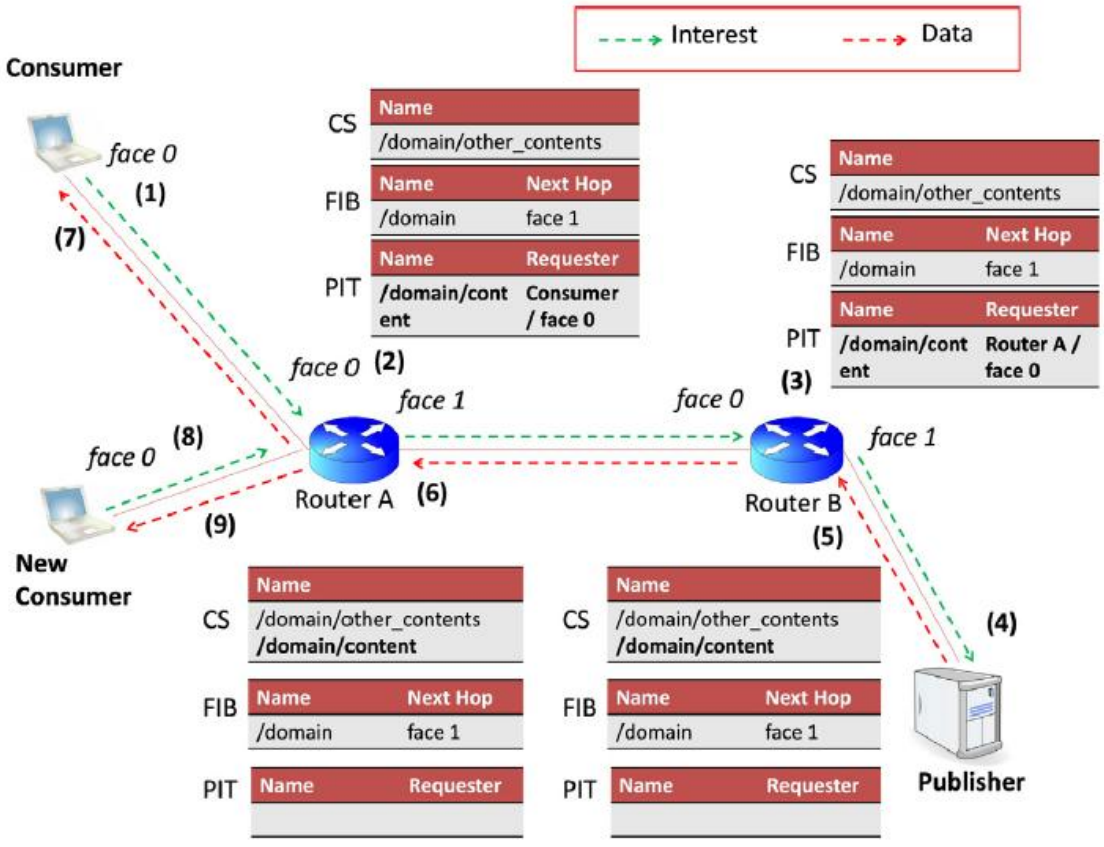

Figure 1.3 A sample NDN communication session.

Description of NDN communication session:

1. The Consumer generates an Interest for the/domain/content.

2. Router A does not find the content in the CS, forwards the request to Router B, according to the FIB, and updates the PIT.
3. Router B does not find the content in the CS, forwards the request to the Publisher according to the FIB, and updates the PIT.
4. The Publisher finds the requested data in its CS and generates the Data packet.
5. Router B caches the data and forwards it to Router A, according to the PIT.
6. Router A caches the data and forwards it to the Consumer, according to the PIT.
7. The Consumer receives the requested content.
8. A new Consumer generates an Interest for the same domain content
9. Router A finds the content in the cache and immediately generates the Data Packet.

1.2 Introduction to Bloom Filter

Bloom filters are probabilistic data structures which works efficiently on huge data sets to check membership query[6].

In 1970 Bloom filter was produced by Bloom. It is commonly used for various reasons today, including web caching, intrusion detection, content-based routing, and databases . A Bloom filter also used for string matching. We can test membership of a string in a set of specified strings using Bloom filter, It is a randomize technique. At first, a set of strings is compressed using this method by calculating various hash functions over each string. Compressed strings are then stored using memory .To find out the membership of string,it can be queried to check whether it is belongs to it or not.A Bloom filter's two significant characteristics that make it a feasible solution for matching strings are as follows:

Scalability: Bloom filter compresses each string with a steady quantity of memory regardless of the length of the initial string. So that with lower memory space, big strings can be stored.In terms of memory utilization, this makes it extremely scalable.

Speed: Using Bloom Filter,the quantity of computation engaged in the detection of a string is steady.Hash function calculation is used to lookups for the respective memory. Efficient hash function scanning can be easily implemented with low resource consumption in hardware.Therefore, a Bloom filter hardware application can match string at elevated speeds. Bloom filters utilize less memory space as compared to other searching techniques to store the packedstrings. The quantity of memory depends on the number of strings being compressed and typically there are few megabits. For example, to store 10,000strings, around 200k bits are required. Nearly FPGAs (Field Programmable Gate Array)are supplied with multi-port integrated memory blocks that are used to build Bloom filters. The true reason for using FPGAs is due to the memory reconfiguration requirement for Bloom filters.

The notions mentioned below helps to understand the variables used in bloom filter along with summary of used symbols.

Table 1.1 -Adopted set of symbols for Bloom Filter

Symbol	Description
S	Data set represented by the *Bloom Filter*
M	Cardinality of *S*
X	Generic element of *S*
Y	Element for which a *membership* query is executed
m	Filter size, expressed in number of bits
k	Number of *hash functions* used to obtain a footprint
$h_i(x)$	i-th hash function executed on element x
p_{fp}	False positive probability
k_{opt}	Optimal number of hash functions
d	Number of bits per cells
p	Number of cells decremented by 1 in a stable *Bloom filter*

A Bloom filter is generally a m-bit long vector to map the IDs of the elements of a S data set.Two fundamental activities are defined by using k autonomous hash function:

1. Mapping of elements
2. Checking membership

The objective is to insert element within the Bloom filter and check unknown elements are respectively part of the represented information set.Figure 1.4 shows a straightforward illustration of how to implement element mapping and membership check activities.The mapping process of the element is carried out through two successive steps. Firstly, an element z's footprint is calculated by performing k hash functions, which are $h_1(z)$, $h_2(z)$ $h_k(z)$. Secondly at positions $h_1(z)$ mod m, $h_2(z)$ mod m…….. $h_k(z)$ mod m, vector bits are set to 1. It should be noted that footprints in one, or more, of their k positions may overlap and thus create collision areas.

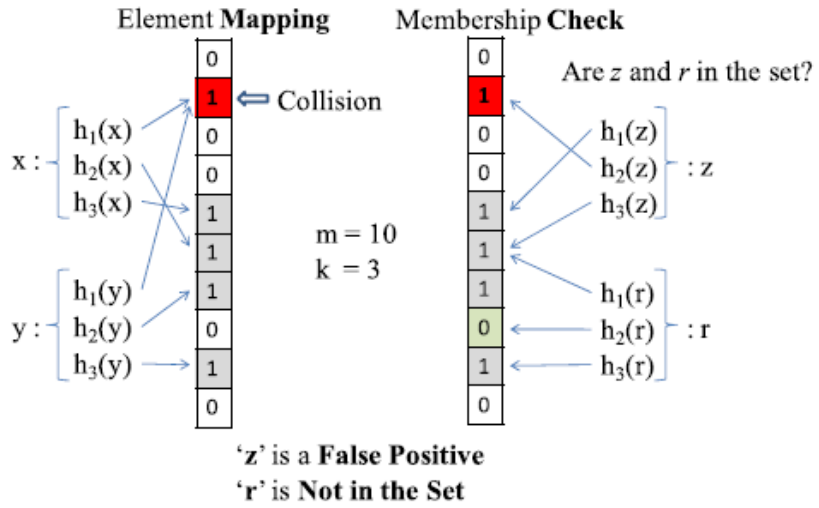

Figure 1.4 Example of a Bloom filter

Instead, its footprint is calculated with the same set of hash functions used for mapping in order to carry out the membership check of an unknown item.The k position identified are then verified: if even a single bit is found to be 0 , y with full confidence is said not to be in the set. Otherwise, the element y is considered as a member of data set S with a fixed false positive probability, p_{fp}, in case all k bits are 1.In fact, if collision areas are present, they can produce an element that is not really present (i.e., false positives), which can be checked for membership.p_{fp} is affected by the filter length according to the mentioned in [7].Therefore, the more bits per element, the less collisions will occur during membership check . However, the optimal choice of filter length according to the following formula has been shown to minimize p_{fp}(1.1) :

$$m = -\frac{M \ln p_{fp}}{(\ln 2)^2} \ldots\ldots\ldots (1.1)$$

In this theory, the ideal number of hash function to be utilized as per equation (1.2):

$$k_{opt} = \frac{m}{M} \ln 2 \ldots\ldots\ldots (1.2)$$

Several Bloom Filter variants have been suggestedso far.In addition, they introduce the option of deleting elements, which allows for dynamic filter management, in different form from the initial application (i.e . the previous one).In such a way, a cell made up of d bits (with d ≥ 1) identifies the single element of vector.The counting Bloom filter [8] is an example, in which case, the k cells associated with the footprint are increased by 1 in every element insertion.Instead, the K cells of the footprint are decreased by 1 when an element is removed.The respective counter is left to the max/min in cases of overflow/underflow of a cell.The Stable Bloom filter[9], which introduces the notion of stability, is another specific feature of the standard Counting Bloom filter.The filter reaches a stable point after certain insertions and deletions, where the proportion of zeros is held constant.

1.3 Thesis Organization

The thesis is organized into following 6 chapters:

- **Chapter 1:** This chapter introduces :Content store(CS), pending interest table(PIT)and Forwarding Information Base(FIB).These are the main three blocks of NDN architecture for performing Interest /Data packets task. This chapter also provides a brief introduction of Bloom Filter.
- **Chapter 2:** This chapter provides a survey of literature in the related area.
- **Chapter3:**In this section problem formulation along with research objectives and researchmethodology used for the proposal are provided.
- **Chapter4:** This chapter described proposed method used for design and implementation of Bloom Filters in NDN.
- **Chapter 5:**This chapter is based on the results obtained from proposed method of BF in NDN.
- **Chapter 6:** Finally, conclusion and future scope of proposed method are provided into this chapter.

Chapter 2

LITERATURE SURVEY

2.1 Literature Survey on NDN

NDN architecture does not use the conventional IP addressing for data routing and the routers only forward the Interest packets in the direction of data producers that is based on the requested names is loaded in the packets, and data packets are forwarded to consumers are cached in Content Store to satisfy the future requests of similar content. As there is limited memory in CS, thus Caching strategies play an essential role in working of NDN router Node.

Caching strategies are classified into two types:

a) caching decision

b) cache replacement decision.

The Table 2.1 depicts the different families ofcaching strategies for decision and replacement:-

Table 2.1 –Family of caching strategies

Layer	Family	Representative Strategies
Caching Decision	Traditional	CEE/LCE [10]
	Static probability	Prob(p) [11]
	Dynamic Probability	ProbCache[12]
		pCASTING
	Topology-based	Betw/EgoBetw[14]
	Cooperative	CINC[15]
Cache Replacement	Traditional	Random Replacement(RR)
		Least Recently used (LRU)
		LFU
	Content-based	MDMR

The caching decision is required to be made wheneveran NDN router receives a data packet which is not yet stored in its CS[13]. It decides either to store or discard the received data packet in its content store; some of the popular strategies are discussed.
A conventional popular memory replacement policy known as Least Recently Used (LRU) which works efficiently and the probability of hit ratio in cache memory increases by storing most recently used data content for more time. The other important cache replacement policy is called as Least Frequently Used (LFU), in this type of replacement strategy, less frequently used contents are removed on first priority.Content cannot be removed from the cache to improve network performance, but one level upstream can be shifted into the cache hierarchy for caching.

Chai *et al.*[14] suggested a caching system operates according to network topology rather than it is used to calculate the probability,simply store the data content at the most central node (i.e. the node with the largest amount of routes between node pairs).Topology oriented methods have the benefit of providing a holistic caching strategy and can enable us to store information in the ideal situation.

2.2 LITERATURE SURVEY ON BLOOM FILTER

In NDN router, utilization of cache memory as well as searching speed of stored contents in the Content Store can be increased by the use of Bloom Filter[1].BFs are widely utilized in databaseapplications and routing .Bloom Filter and its variants were made appropriable and compatible fordiverse applications like network intrusion detection system, Patternmatching, Packet classification, Detection of flooded attacks in internet,distributed caching of web servers, Dictionaries and Database applications.

Fan *et. al.* (2000)[15]Sharing caches between Web proxy is an significant way of reducing Internet traffic. The distribution of caches between web proxies isan essential method to minimize Web traffic and improves the networkbottlenecks. The advantage of cache sharing method is to reduce the overheadof the web servers. In this new protocol, each proxy retains a summary of thecache directory of contributing proxy and ensures the summaries for potentialhits before sending any queries.

Kumar *et. al.* (2006)[16] proposed Space Code Bloom Filter for efficienttraffic flow measurement. A new technique is designed for calculating per flowtraffic for all flows irrespective of their sizes at very high-speed. Themain aim of the SCBF is to estimate demonstration of a multiset. The components in this multiset are characteristics of traffic flow and their multiplicity is the number of packets in the flow. SCBF uses a Maximum probability method to calculate the multiplicity of element in themultiset. With help of parameter tuning, SCBF allocates for attractive tradeoffor connection measurement accuracy, computational complexity and storagecomplexity. SCBF provides high speed data streaming by designing a newmethod called blind streaming.

Mitzenmacher*et.al.* (2005) [17] introduceddictionaries and database applications that frequently require fast, compactand efficient query mechanism. It is also utilized in distributed caching ofweb servers to minimize the traffic of web.

OriRottenstreich*et.al.*(2014)[18]proposed the variable-incrementCounting Bloom Filter. Counting Bloom Filter (CBF) are commonly used in networkingdevice algorithms for efficient membership querying with addition and deletion possibilities.However, they use considerable memory. CBFs execute fast set demonstrations to maintainmembership queries with inadequate error and support element deletions,though it utilizes large amount of memory. A fresh overall technique is used to improve the effectiveness of CBFs and their variants based on varying_increments.Not like CBFs, the hashed counters are increased with a hashed v ariable increase instead of a unit increase. Afterthat, to query an element, the correct value of a counter is taken. This technique is focused on two easy systems.. The false positive rate and reduced probability of overflow can always be achieved by this technique in practical schemes than CBF.

Tian*et. al*. [19] designed basket Bloom Filter for membership queries.BF is extensively employed in databases, networks, and distributed systems.A new Bloom Filter called Basket Bloom Filter (BBF) is designed. The BBF manages many items in a data set based on the cost of invalidating the query by grouping the items into different baskets. The full cost function of invalidating the request is defined.The

genetic algorithm is used to locate the optimum number of hash functions for all baskets to reduce the total query invalidation cost.

Data aggregation designed by Rajagopalan and Varshney (2006) isthe operation carried out in the intermediate nodes which joins a lot ofmessages and sends out the combined results.

Adam Kirsch*et.al.*[20]planned distance-sensitive-Bloom Filters. A Bloom filter is a spatial data structure which responds set membership queries with a certain risk of false positive results .Bloom Filter generalizations problems were scheduled to react to form queries, where closeness is calculated using a suitable metric.Such a data structure would be used for various natural networking and database applications.Using locality-sensitive hash functions as a building block, the appropriate data structures are designed to examine the results of a natural method under the Hamming metric.

Hao *et. al.* (2007)[21]proposed high accuracy Bloom Filter usingpartitioned hashing. The significance of actions like content of packet inspection, packet classification based on non-IP headers, preserving flowstateare resulted in enhanced interest in the networking applications of Bloom Filters.It takes place because BFs offer simple technique for hardwareimplementation of set-membership queries. Though the tradeoff is in BFs, itoffers probabilistic test and membership queries resulted in false positives.The key aim of the method is to minimize the false positive probability which compared to existing schemes. This is performed by increasing a partitionedhashing method that leads to the alternative option of hash functions that can sets lesser bits in the BF bit vector. The reduction in fill factor of the bitvector is transformed to lower false positive probability.

Geneiatakis et al. (2009)[22]briefed utilizing Bloom Filter fordetecting floodingattacks against SIP based services. Many threats–attackswith the service provider takes the flooding attacks on the signalling level aresame as TCP servers. However, it is created at the application level of theInternet architecture. It identifies the flooding attacks against VoIP architectures which creates the Session Initiation Protocol (SIP) as theirsignalling protocol, This is the main aim to the design and

implementation of thesuitable detection method. Particularly, BF based monitor is designed and anew metric called session distance is designed to offer an effective protectionscheme against flooding attacks.

Xiao *et. al*. [23] presented Bloom Filters for multi-attributesfor network services. Bloom Filters are not appropriate for various newnetwork applications. It supports network services like illustration andquerying of items, containing multiple attributes different from a single attribute.An efficient demonstration and querying of multi-attribute items are designedusing BFs in exact manner.

Chen *et. al.*[24] proposed a Bloom filter based approach in data dissemination protocol for wireless sensor networks. As Bloom filters leads to compact storage of the data items, this protocol efficiently identifies the version differences among data items with the same key. -The results demonstrated that, this protocol outperformed the existing schemes with low energy cost, reduced delay of updating new items with improved reliability.

Dharmapurikar*et. al.*[25]proposed a new approach to packet classification which combines architectural and algorithmic techniques. This proposed cross product algorithm was fast but has significant memory overhead due to the extra rules needed to represent the cross products. It reduced the memory requirement without compromising on performance. This technique reduced the unnecessary accesses to the off-chip memory are avoided by filtering them through on chip Bloom filters.

Chapter 3

PROBLEM STATEMENT

3.1 Motivation

In NDN the data content that is frequently requested ,store in Content Store of NDN router. Due to limited size of memory in NDN router, the data content needs to be replaced frequently and Caching Information Table (CIT) updated accordingly.The requested interest packet requires matching the existing contents in Content Store.

The current NDN optimization methods for caching has limited storage effectiveness and searching speed of data content in CIT is slow. For addressing these issues the use of Bloom-filter-based searching method is suggested to encourage caching effectiveness and information content search velocity in order to overcome these constraints.

3.2 Problem description

NDN is content based network to delivers the data, requested from users without using the conventional Internet Protocols. The data content frequently requested by consumer stored in the Content Store of NDN router for future use. Thus searching and caching efficiency are crucial aspect in NDN router. Because storage space of cache memory is limited due to this some replacement policies like LRU , LFU are apply to take decision for caching data in cache memory.

Bloom filter compresses each string with a steady quantity of memory regardless of the length of the initial string. So that with lower memory space, big strings can be stored.In terms of memory utilization, this makes it extremely scalable .Therefore, a Bloom filter hardware application can match string at elevated speeds.Bloom filters utilize less memory space to store the packed strings,which will increase the

efficiency of searching and caching data content in caching information table (CIT) located in NDN router.

3.3 Objective

The main focus of our thesis work is to implement Bloom filter in NDN router's Content Store.The key objective of this work are listed below:

1. Bloom filters utilize less memory space to store the packed strings,which will increase the efficiency of searching and caching data content in caching information table (CIT) located in NDN router.
2. The comparison of searching and caching efficiency of Bloom filter in NDN router will be done in this work.
3. The results of simulation of NDN Node network using ndnSIM and ns3 will be compared with the implementation of BF.
4. The results will be obtained with Visualizer and R-studio for comparison of performance of NDN network.
5. The comparison of number of satisfied Interest packed to verify the increase in searching efficiency of Bloom filter in NDN router will be done in this work.
6. The comparison of packed data rate handling for NDN router with used of BF and without BF case will be done.
7. The simulation of 7 Nodes NDN network topology using ndnSIM will be done in this work.

3.4 Methodology

Major aim of this work is comparison of the existing proposed NDN router with the application of BF enhance the performance by increasing the searching and efficiently caching of data contents in Content Store of NDN router.

Below is the methodology that will be followed to achieve the same:

1. Use of ndnSIM integrated with standard ns3 open source software will be used for simulation of NDN architecture.
2. For recording the simulation parametersof NDN nodes in txt files, the different type of standard functions CS trace command, delay rate trace of Data Packets and L3 and L2 commands are in program code used in ndnSIM.
3. For generation of results graphically the RScriptcode is used for plotting the various plots.
4. Visualizer option is also used for simulation to verify the data rates of point to point links between NDN nodes.
5. The codes are written in C++ and the code of Bloom Filter will be appended in the standard library of ndnSIM.

Chapter 4

DESIGN AND APPLICATION OF BLOOM FILTER IN NDN

4.1 Software Requirement

As networks grow bigger and more complicated, it becomes more critical to simulate them with high precision and scalability. Although large-scale network test beds are created for research, simulation programs still play a very important role because they are more scalable, quickly installable and reproducible than the real test beds of the network.

4.1.1 Modeling with ns3

The primary objective in ns3 was to achieve realistic modes. In terms of implementation, the designers produced it close to the real software. It is created using C++ and makes it easier for simulations to use the C language. It also makes it very practical to debug. The fundamental models used in ns3 are outlined below:

1. Node: It is used for implementation of routers, switches, hubs, pcs, or laptops.
2. Device:Physical devices in ns3 for example Ethernet and IEEE 802.11 standard for Wireless devices is used. It is also used to provide the connectivity between two nodes to provide a channel of communication as per IEEE standards.
3. Communication Channel: This acts as a physical links which connects to one device to another network devices and channel can be used for point-to-point connectivity or in broadcasting mode.
4. Protocols:These are the standard protocol used for Internet connectivity. The packet uses the defined protocol layers before sending a packet or getting a packet.
5. Packets:These are the primary communications units in networks and these packets contain data and content along with overhead information.

ns3 is intended to enable users to readily obtain customized logs and statistics. It's called the subsystem of tracing. Users can edit this subsystem without altering

anything of the program to get the required outputsand logs. To collect the run data, a metadata is described. It is possible to dump the data, outputs and settings in different formats. To draw plots and graphs, dumped logs are used. External scripts and instruments can also be used to evaluate them from distinct angles. The program can also draw graphical user interfaces. The program can operate simulation visually if required.

4.1.2 ndnSIM

The ndnSIM simulator (ndnSIM) is an environment for simulating named data networks in an open source simulation implementation. It's a ns3 extension. It has the following characteristics:

1. An open source project to support and help other scientists in conducting NDN experiments.
2. Almost all NDN installations and characteristics are supported.
3. It has instruments for analyzing results such as measuring traffic, analyzing packets(Data/Interest)comes from consumer or producer etc.
4. It can support big simulation experiments without any trouble.
5. It enables users to experiment with the network layer. It is easy to simulate routing, caching, forwarding and congestion.

ndnSIM is NDN architecture compliant. It is created as a protocol for the network layer. It can therefore operate on various link-layer protocols, such as CSMA, point-to-point, and wireless. Using object-oriented programming, ndnSIM is created with modular framework. Modular conduct allows users to alter required components without influencing other components. The minimum requirement of operating system for ndnSIM 2.29 and ns3 are following :

1. Ubuntu 16.04
2. Python higher or equal version than 2.6
3. Boost Libraries higher or equal version than 1.54
4. Updated Packet Configuration
5. Open SSL
6. Lib-crypto++
7. Lib-Sqlite3
8. Minimum hardware Requirement : Core 2 Dou

9. Minimum hardware Requirement :4GB RAM

NdnSIM is a modular open source NS-3-based Named Data Networking simulator that performs all NDN fundamental activities such as Forwarding interest base(FIB), Pending Interest Table (PIT), Content Store (CS). Following are some steps followed on Ubuntu 16.04 to install ndnSIM.

4.2 Installation of Prerequisites

Following modules / libraries are necessary in order to run ndnSIM.

1. sudo apt-get update
2. sudo apt-get install python-software-properties
3. sudo add-apt-repository ppa:boost-latest/ppa
4. sudo apt-get update
5. sudo apt-get install libboost-all-dev
6. sudo apt-get install libssl-dev
7. sudo apt-get install build-essential
8. sudo apt-get install libsqlite3-dev libcrypto++-dev
9. sudo apt-get install python-dev python-pygraphviz python-kiwi
10. sudo apt-get install python-pygoocanvas python-gnome2
11. sudo apt-get install python-rsvg ipython
12. sudo apt-get install doxygen graphviz python-sphinx python-pip
13. sudo pip install sphinxcontrib-doxylink sphinxcontrib-googleanalytics

4.2.1 Downloading ndnSIM source

The following commands create ndnSIM directory and download ndnSIM package from GitHub.

1. sudo apt-get install git (if not already installed)
2. mkdir ndnSIM
3. cd ndnSIM
4. git clone https://github.com/named-data/ndn-cxx.git ndn-cxx
5. git clone https://github.com/named-data-ndnSIM/ns-3-dev.git ns-3
6. git clone https://github.com/named-data-ndnSIM/pybindgen.git pybindgen
7. git clone --recursive https://github.com/named-data-ndnSIM/ndnSIM.git ns-3/src/ndnSIM

4.2.2 Compiling and running ndnSIM

Following commands will compile and install ndn-cxx library and compile NS-3 with ndnSIM module as shown in Figure 4.2 and Figure 4.4

1. cd ndnSIM/ndn-cxx
2. ./waf configure --enable-shared --disable-static
3. ./waf
4. sudo ./waf install
5. sudo ldconfig
6. cd ndnSIM/ns-3
7. ./waf configure --enable-examples
8. ./waf

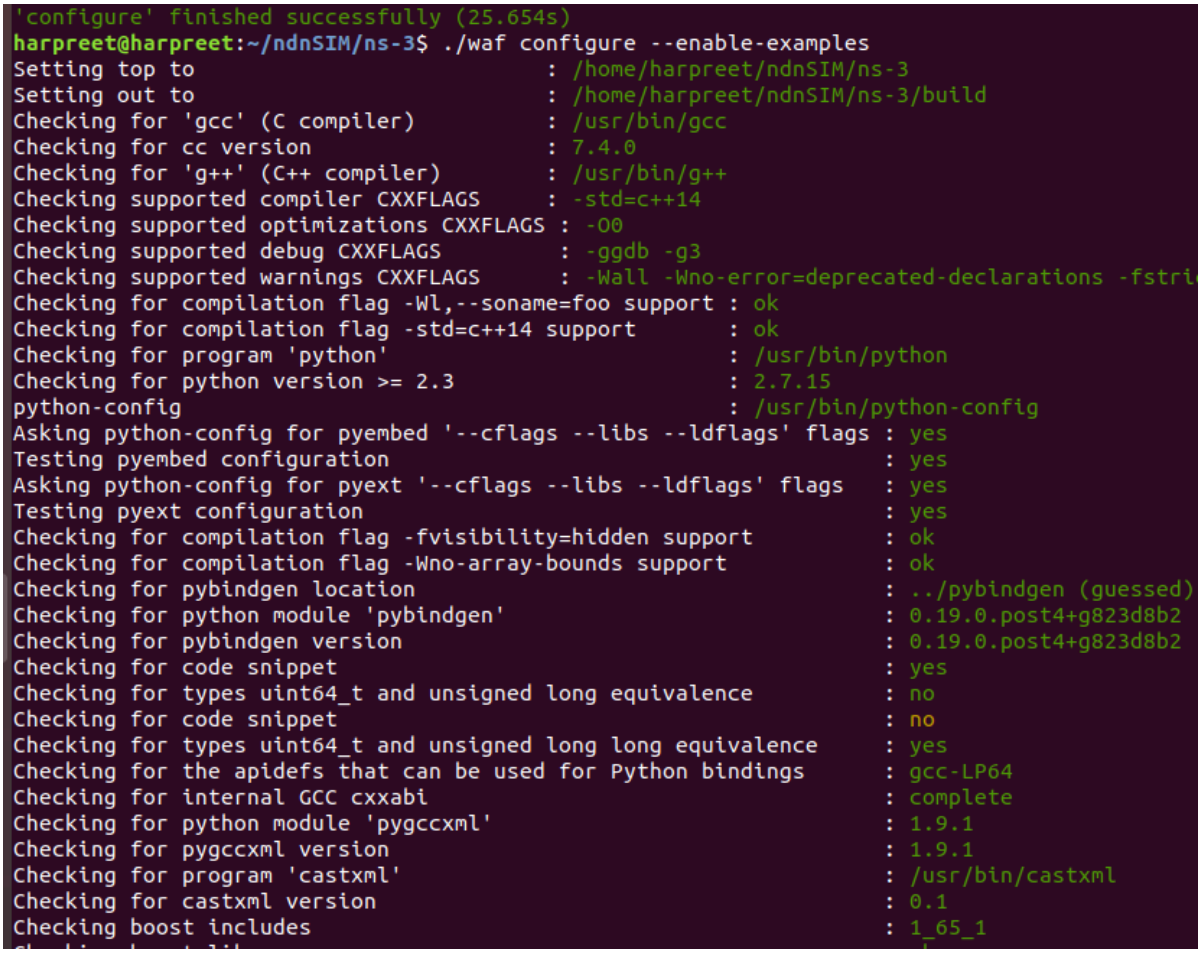

Figure 4.1 ndnSIM snapshot ./waf configure --enable-examples

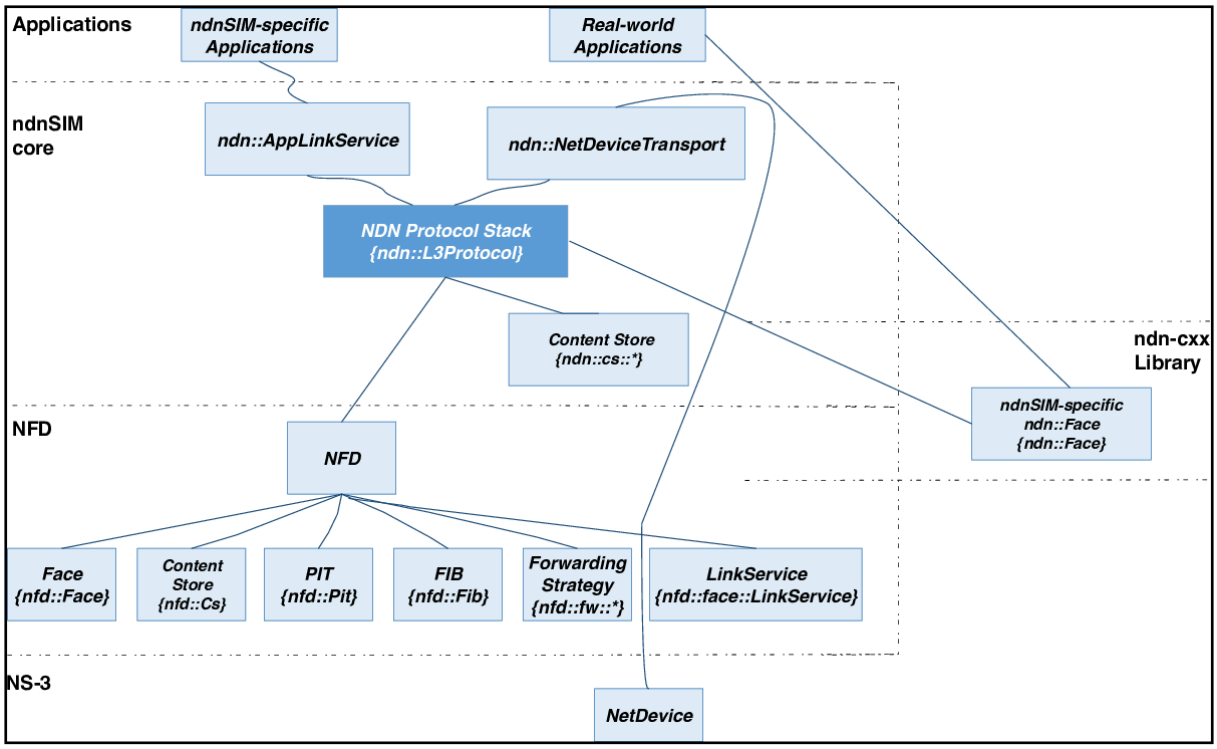

Figure 4.2 ndnSIM structure

The definitions of the elements in Figure 4.2 are given below.

a) Named Data Networking Forwarding Daemon contains the followings:
   - Forwarder is the main class and has PIT,CS and FIB tables,and faces. Forwarding pipelines are implemented in forwarder.
   - Face is an abstraction that includes the communication rules to send or receive network packets.
   - Link services enables the translation between link layer packets and network layer packets.
   - Transport provides the link service of a face with packet delivery service.
   - CS is a cache table that keeps the data packets.
   - PIT keeps track of the interest packets.
   - FIB is used to forward interest toward the sources.

b) AppLinkService: It is an implementation of Link Service for ndnSIM application and enables the communication with applications.

c) NetDeviceLinkService:It is an abstraction to provide communication with other nodes.

d) L3_Protocol: the primary objective of L3_Protocol is to initialize NFD example for each scenario node.It also offers tools for tracing.

e) Content Store(CS) caches the contents of the information.

f) Many standard applications in ndnSIM are the built-in consumer and producer NDN applications. They produce network traffic and sink it down. Using parameters, these apps can be configured.

g) Trace helpers are helpful to collect simulation data and all the results are log into to a text file.

Figure 4.3 ndnSIM snapshot ./waf

4.3 Programming in ndnSIM

In ndnSIM[26]is an open source software based on the NS-3 network simulator[27]. Bloom filter module is created in ndnSIM and NDNstack that can be installed within a simulated node.This includes all the structures, characterizing in NDN node. In

addition, ndnSIM makes possible to generate traffic which helps to simulate the scenario of NDN architecture[26].

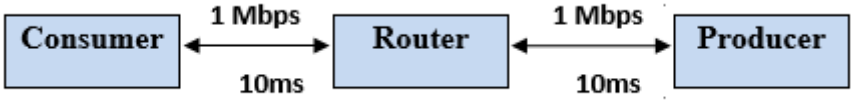

Figure 4.4 NDN with 3 nodes

4.3.1 Program for NDN for 3 Nodes

```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/ndnSIM-module.h"

namespace ns3 {

int
main(int argc, char* argv[])
{
  // setting default parameters for PointToPoint links and
channels

Config::SetDefault("ns3::PointToPointNetDevice::DataRate",
StringValue("1Mbps"));
  Config::SetDefault("ns3::PointToPointChannel::Delay",
StringValue("10ms"));
  Config::SetDefault("ns3::DropTailQueue::MaxPackets",
StringValue("20"));

  // Read optional command-line parameters (e.g., enable
visualizer with ./waf --run=<> --visualize
  CommandLine cmd;
  cmd.Parse(argc, argv);

  // Creating nodes
  NodeContainer nodes;
  nodes.Create(3);

  // Connecting nodes using two links
  PointToPointHelper p2p;
  p2p.Install(nodes.Get(0), nodes.Get(1));
  p2p.Install(nodes.Get(1), nodes.Get(2));

  // Install NDN stack on all nodes
  ndn::StackHelper ndnHelper;
  ndnHelper.SetDefaultRoutes(true);
  ndnHelper.InstallAll();

  // Choosing forwarding strategy
  ndn::StrategyChoiceHelper::InstallAll("/prefix",
"/localhost/nfd/strategy/broadcast");

  // Installing applications

  // Consumer
```

```
  ndn::AppHelper consumerHelper("ns3::ndn::ConsumerCbr");
  // Consumer will request /prefix/0, /prefix/1, ...
  consumerHelper.SetPrefix("/prefix");
  consumerHelper.SetAttribute("Frequency",
StringValue("10")); // 10 interests a second
  consumerHelper.Install(nodes.Get(0));
// first node

  // Producer
  ndn::AppHelper producerHelper("ns3::ndn::Producer");
  // Producer will reply to all requests starting with
/prefix
  producerHelper.SetPrefix("/prefix");
  producerHelper.SetAttribute("PayloadSize",
StringValue("1024"));
  producerHelper.Install(nodes.Get(2)); // last node

  Simulator::Stop(Seconds(20.0));

  Simulator::Run();
  Simulator::Destroy();

  return 0;
}

} // namespace ns3

int
main(int argc, char* argv[])
{
  return ns3::main(argc, argv);
}
```

4.4 The flowchart of packet

When a new interest packet is generated by subscriber which is connected to NDN router, The requested content of interest packet will be matched in content store[30]. If the data content already cached in Content Store then the data packet sent back to the requested subscriber, otherwise the request along with the interest packet is stored in PIT. And if the requested data packet already stored in PIT then it is forwarded to FIB. After accomplishment of Data packet to subscriber, the corresponding entry in PIT and FIB is cleared.[31]

When data packet generated by publisher caches into the Content Store of NDN router[32]. Then PIT forwards the data packet to requested interface based on the interface information of interest packet. If PIT does not have any information regarding data packet, the packet will be discarded.

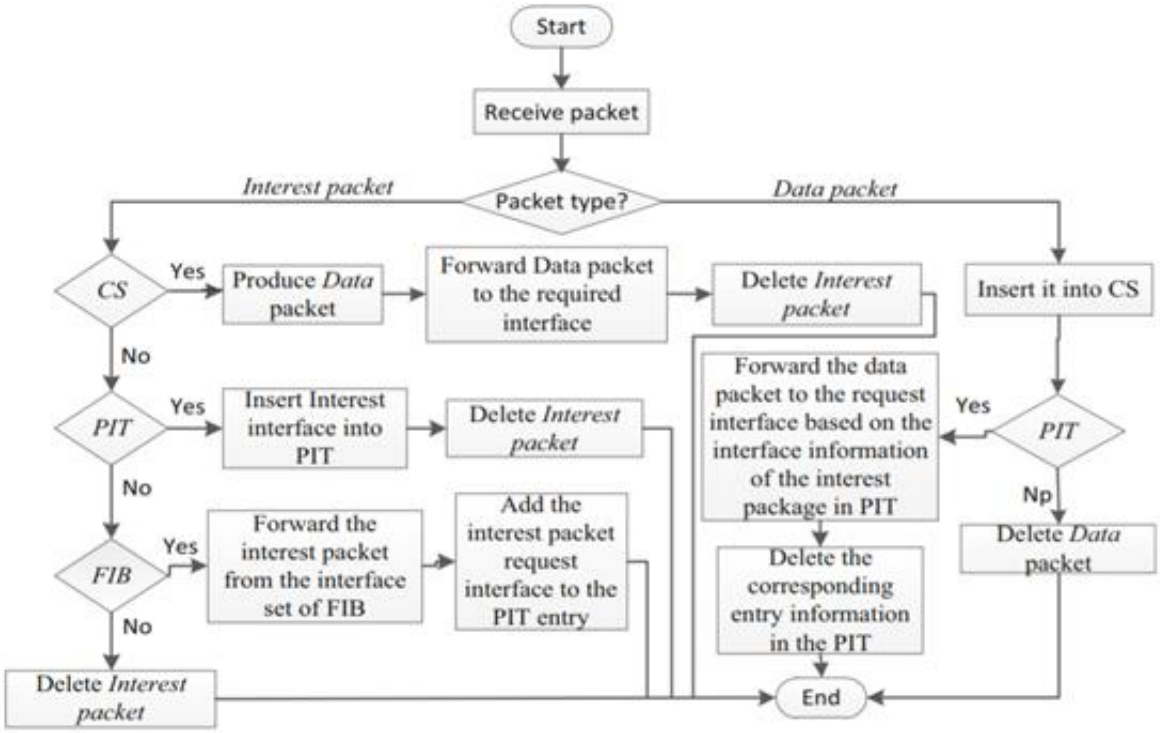

Figure 4.5: Packet forwarding flow chart.

4.5 Bloom Filter Implementation in NDN Router

The data content stored in the Content Store of NDN router is to be matched with the content requested in the interest packet by the consumer, thus the speed of searching is critical performance parameter for NDN router and also the cache memory is effectively utilised with the advantages of using bloom filter. To check the effectiveness of bloom filter in NDN router in this thesis, it is implemented by appending the standard functions of Content Store which are defined in C++ code in NDN library functions and the Bloom filter code is defined in the standard functions-matchesData(),isStale(), and hasData() functions in CS entry library code.

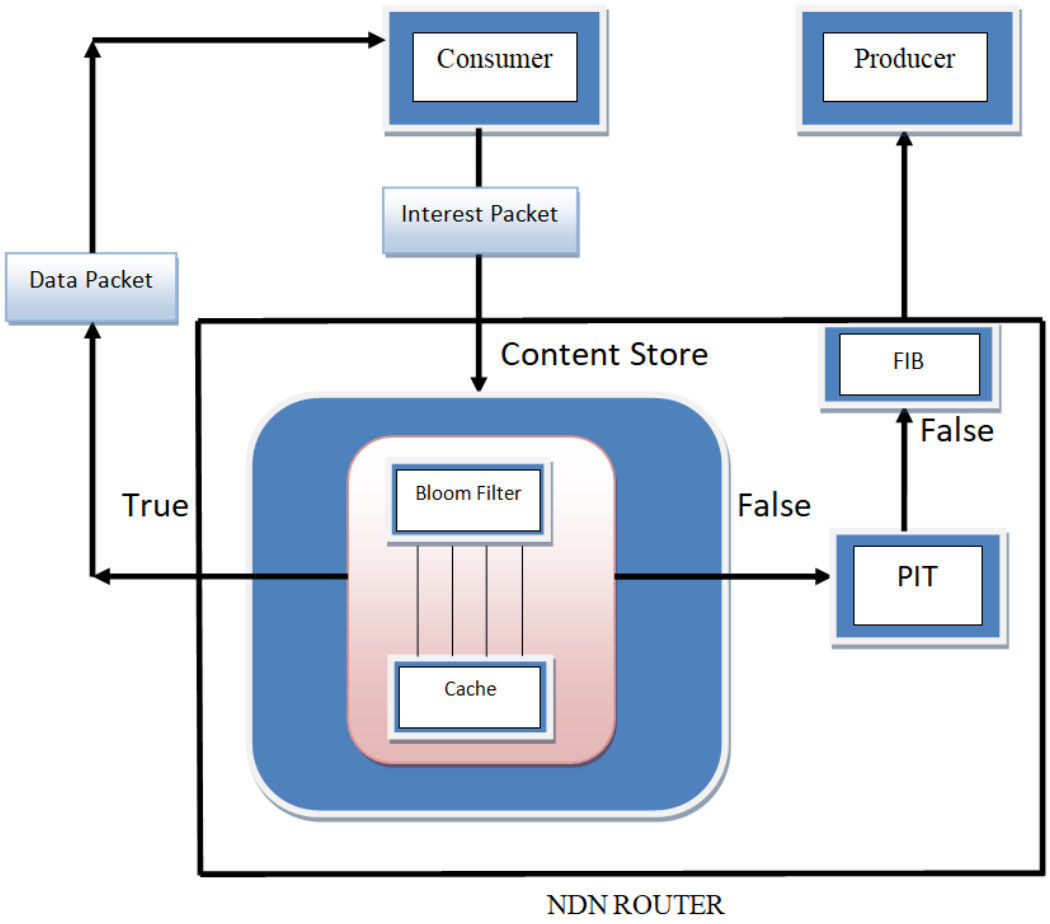

Fig 4.6 Proposed packet flow Bloom Filter in NDN Content Store

The following are packet flow in the proposed Bloom Filter implementation in NDN router as shown in the figure 4.6 :-

1. The interest packet is received in NDN router connected to Consumer Node, the requested content is matched in Content Information Table to check the available data in Content Store by utilising the bloom filter membership check function.
2. The bloom filter performs the searching with respect to the requested data content in the existing stored data in Content Store of NDN router. The bloom filter search more efficient that other searching techniques, so it reduces the time for searching operation in NDN Content Store.
3. If the membership check function of Bloom Filter returns true value means the information is already stored in content store. Thus the same is accessed from the

cache memory of the content store are routed to the Consumer Node from which the Interest Packet was requested. But one major drawback of it ,is that after returning the true value by Bloom Filter it is the possibility of generating the false positive rate.

4. The other possibility is that if the membership check function of Bloom Filter returns false value and if there is no similar entry recorded earlier in Pending Interest Table, thenew entry is recorded in Pending Interest Table. The routing information is alsoupdated in Forward Information Block w.r.t each PIT entry.
5. The required content information is also forwarded to nearest router on the basis of FIB entry.
6. In the event of receiving of a Data Packet from connected router in upstream or Publisher Node directly connected to router, the new content is stored in cache memory using insert member function of Bloom filter and the corresponding entry in PIT is cleared and Content information table is updated accordingly. The contents are substituted in accordance with the LRU rule in cache memory.

Chapter 5

TESTING AND RESULTS

To verify the effectiveness and efficiency of Bloom Filter in caching and searching techniques in Content Store of NDN router using ndnSIM[29] , the network topology of 7 Nodes is used. There are four Nodes defined as Consumers, two Nodes as NDN Routers and one Producer Node as shown in Fig. 5.1

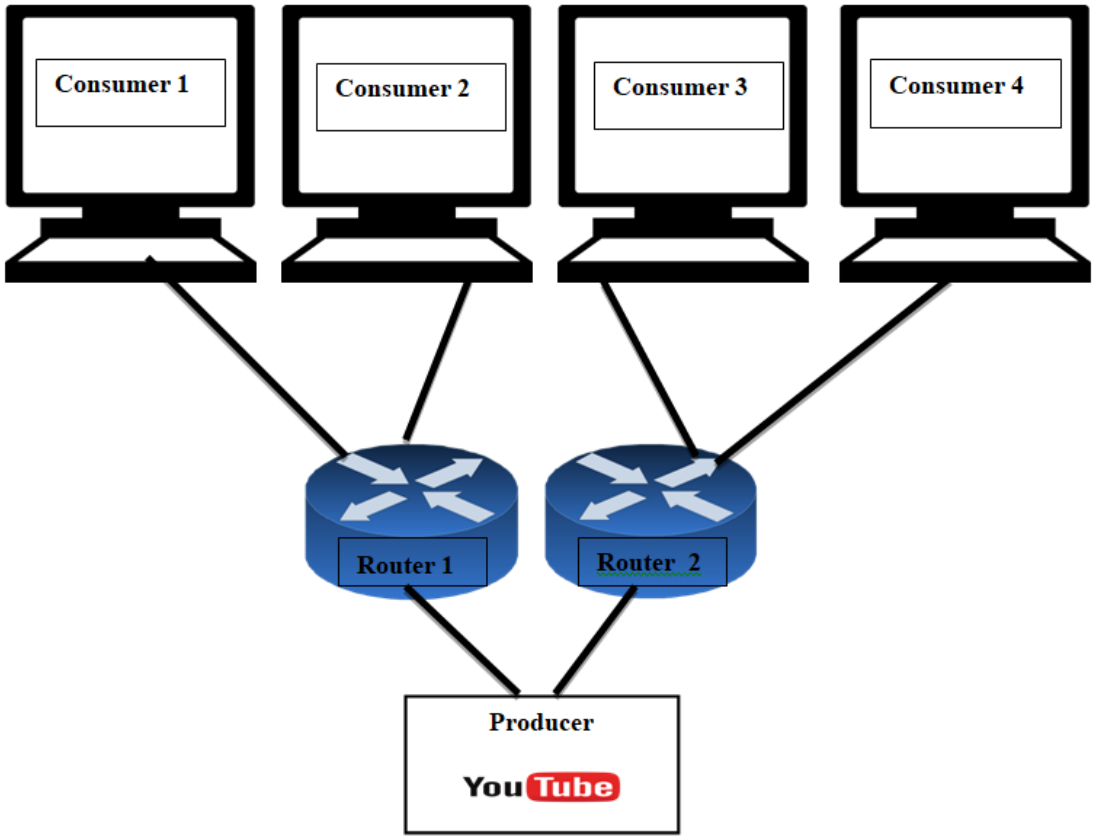

Figure 5.1: NDN network topologyused for Testing NDN router

Each consumer node generates 150 interest packet per second during the simulation. Data content is being generated by Producer Node which is connected to both routers and the size of each data payload is assumed as 1024 bytes. The bandwidth of each link connected between NDN nodes is taken 10 Mbps and which are defined using point to point link attributes in ns3 standard function.

The topological links parameters along with the nomenclature are defined in the program code as perthe table given below:-

Table 5.1: NDN Node link parameters

Name of Node (from)	Name of NDN (to)	Capacity	Delay
Consumer-1	Router-1	10 Mbps	1 ms
Consumer-2	Router-2	10 Mbps	1 ms
Consumer-3	Router-1	10 Mbps	1 ms
Consumer-4	Router-2	10 Mbps	1 ms
Router-1	Producer	10 Mbps	1 ms
Router-2	Producer	10 Mbps	1 ms

The code is compiled along with visualizer and the 7 Node topology tree with instantaneous data rates on each link between each NDN node is projected as shown in the figure below :-

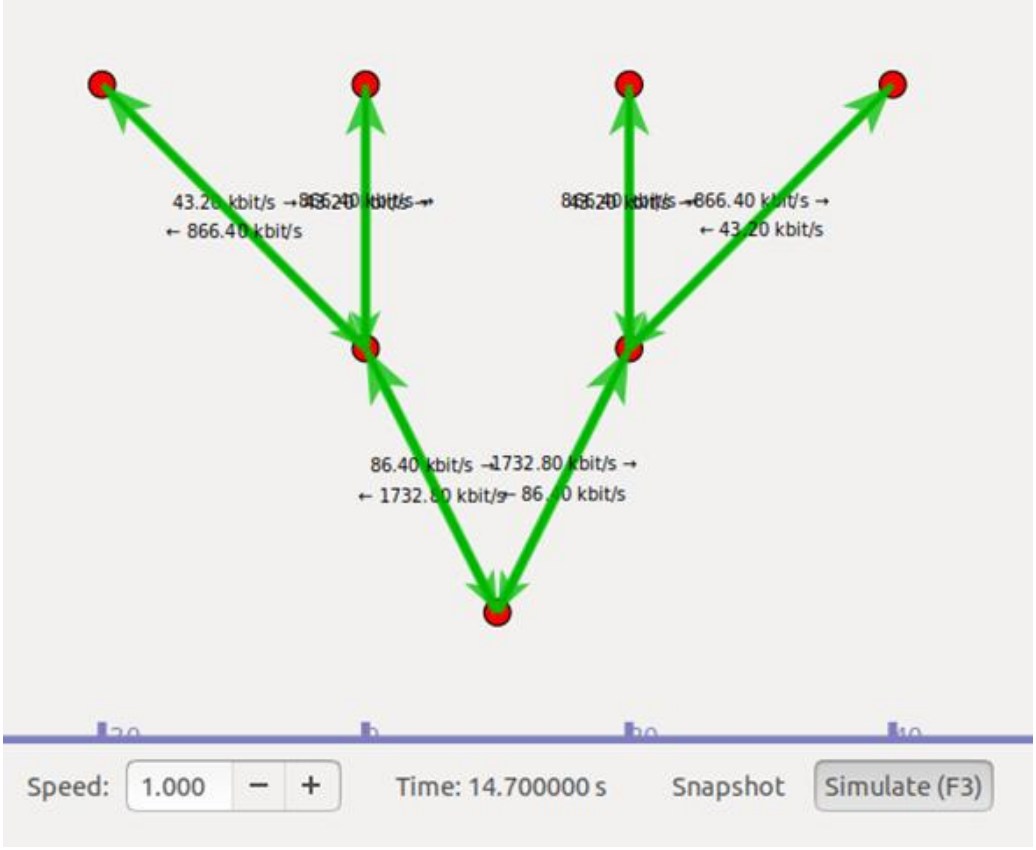

Figure 5.2: Instantaneous data rates on NDN Node at each link

To compare the performance of NDN routers using Bloom Filter for matching the requested content, stored in cache of NDN router with default searching techniques defined in the NDN project, the packet flow rates during simulation are recorded with the help of L3 and L2 rate trace commands in which the data rates, the data drops due to congestion or over flow of data and the average delay are recorded. The CS trace command records the Cache Hits and Cache Miss during the simulation in each router. The above mentioned parameters are recorded in txt files during the simulation and the comparison of results for a default NDN router implemented with bloom filter with cache of CS is shown in this chapter.

The L3 trace logs results for the number of satisfied interest packetsin standard NDN node and nodes with Bloom Filter at an intervals of 0.5 second shown in the table below:

Table 5.2 : Comparison of Number of Satisfied Interest Packets

Name of NDN Node	Simulation Time (sec)	No. Of Satisfied Interests Packets without Bloom Filter	No. Of Satisfied Interests Packets Bloom Filter implementation in Content Store
Router 1	0.5	120	159
Router 1	1	118	155
Router 1	1.5	238	316
Router 1	2	236	315
Router 1	2.5	144	192
Router 1	3	143	190
Router 1	3.5	286	381
Router 1	4	148	197
Router 1	4.5	148	198
Router 1	5	297	396

From the table above, the performance of NDN router implemented using bloom filter is able to search quickly and thus the average rate of interest packets satisfied in 0.5 sec is greater in the case of NDN bloom filter router.

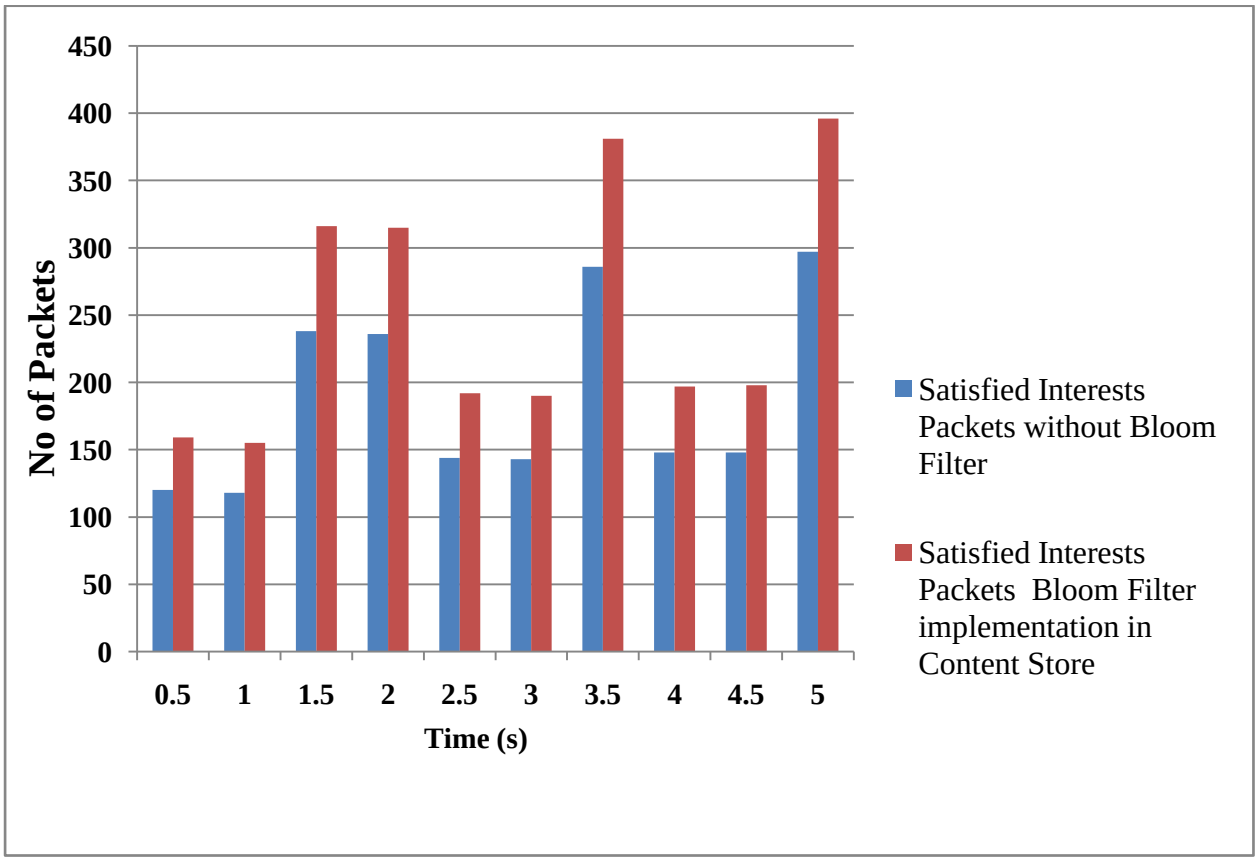

Figure 5.3: Comparison of Number of Satisfied Interest Packets

The results for Number of Satisfied Interest Packets are compared in the bar graph as shown in the figure 5.3, there are about 60 additional interest packets satisfied in each 0.5 second interval of time in NDN bloom filter router than default NDN router.

The total packet data rate (kilobits per sec) is recorded for the same seven Nodes NDNscenarios and the simulation for 20 seconds is used for comparing the overall data packet rate of all the NDN nodes to compare the performance of proposed NDN bloom filter router with the default NDN router.

The packet rate graph is plotted in consideration to four types of data packets that are InData , InInterest , OutData and OutInterest data packet rates using R studio script code program.

The comparison of data packet rate is shown for the NDN default router and according to this rate graph shows that the Outdata packet rate is about 2500 kbits per second and InData rate is about 1250 kbits per second for default NDN router network.

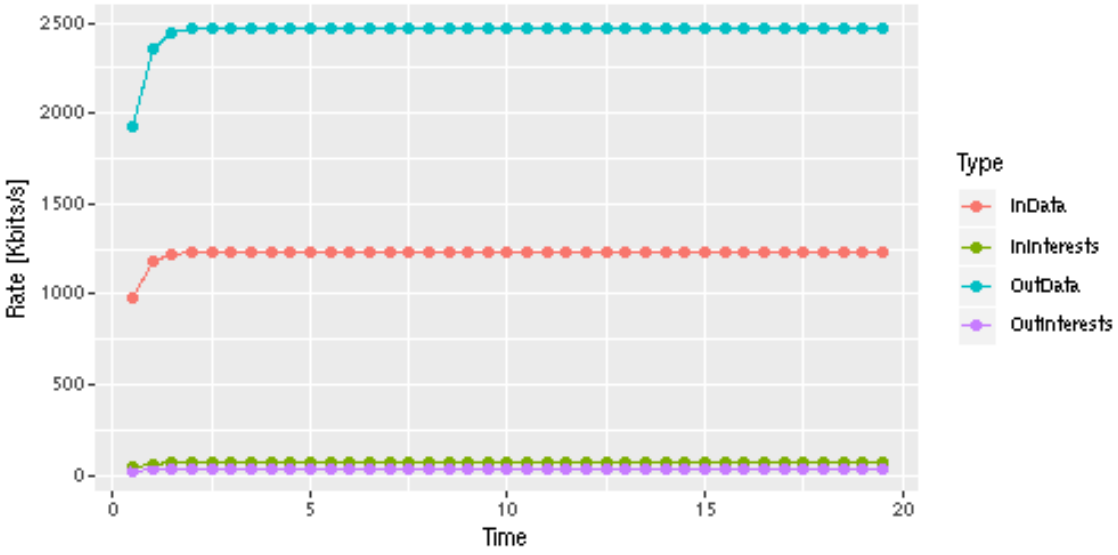

Figure 5.4: Packet data flow rate of default NDN Node

However, the packet data rate for proposed Bloom filter based router network recorded the Outdata packet rate near to 3200 kbits per second and InData rate is about 1550 kbits per second. Thus, the packet data rate performance of NDN router network is enhanced with the use of bloom filter in the content store.

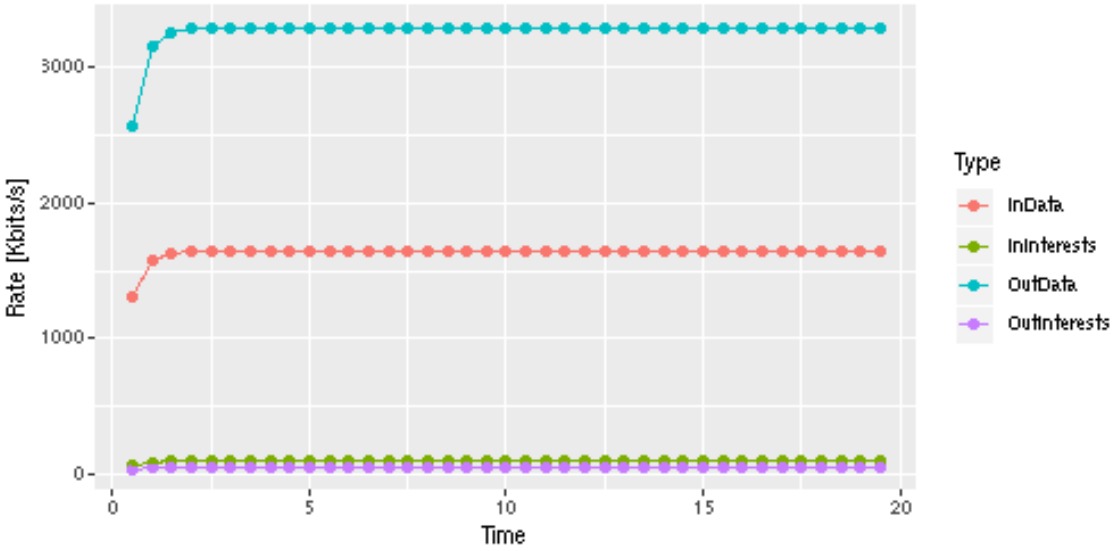

Figure 5.5: Packet data flow rate of NDN Nodes with BF

The delay trace command is also used in the program to record the average delay of data packets received by each consumer Nodes in micro seconds as shown below table :-

Table 5.3: Average Delay of Data Packets during simulation

S NO	Name of Consumer Node	Average of Delay in Receiving Data Packets (µs)	Average of Delay in Receiving in Data Packets with Bloom Filter in Routers (µs)
1	Consumer-1	5817.683417	4697.62881
2	Consumer-2	5793.394944	4877.62881
2	Consumer-3	5617.683417	4767.62881
3	Consumer-4	5790.394944	4865.62881

The bar graph shown below in the figure 5.6 is the comparison of the average delay of data packets received by each consumer Nodes for both cases for NDN routers.

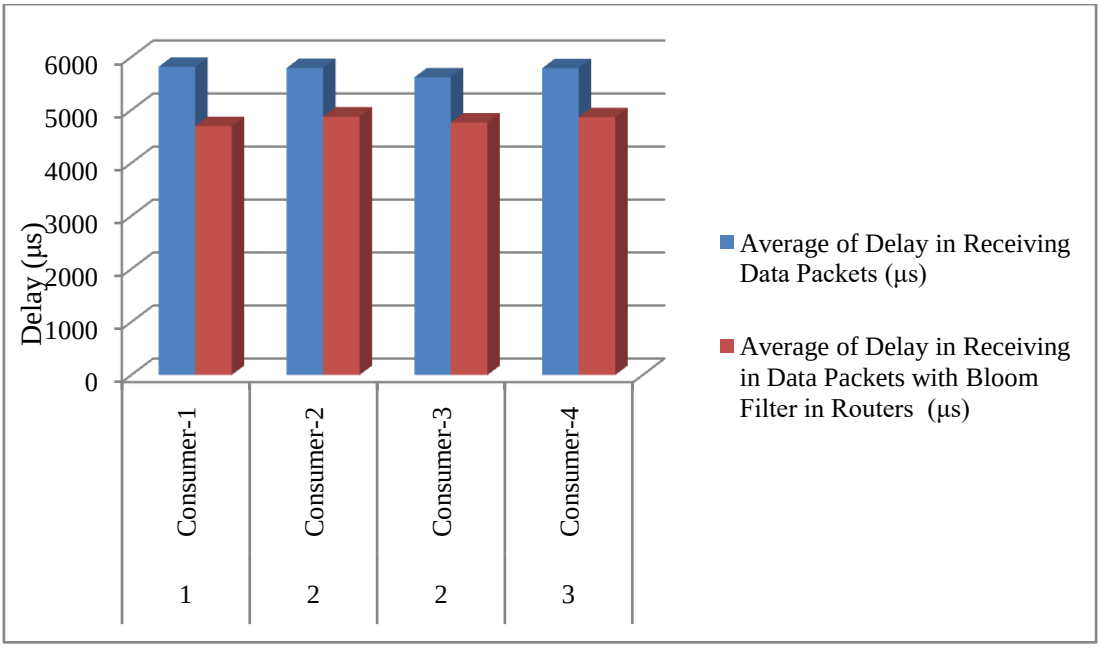

Figure 5.6: Simulation for 25 sec has been plotted using RScript

Thus from the above comparisons of different parameters of NDN nodes, the performance of NDN router with bloom filter is better. Thus it can be inferred that use of Bloom filter increases the efficiency ofcaching and searching in content store of NDN router which enhances the performance of NDN network.

Chapter 6

CONCLUSION

6.1 Conclusion

In this thesis, the application of Bloom Filter in NDN router's Content Store for caching and searching of data contents is proposed. The comparison of results have been done and the results shows considerable improvement in the packet rate handled by NDN router. The overall efficiency of NDN architecture is also enhanced. As modern Bloom filter's false positive rate are very low which can prove beneficial in the working of NDN router as the content information table has to be matched with the interest packet received from the subscriber and the same has been shown in the results.NDN architecture is the best approach for modern data content applications and IOTs(Internet on things) because this is more reliable and robust against malicious packets,as every data packets consists of the signature of the producer. Thus combination of Bloom filter searching technique with NDN can improve the next generation of internet.

6.2 Scope of Future Work

As we know that research is about continuous improvement of techniques in various fields. Similarly there is a scope of improvement in this proposed work to further increase the searching efficiency of NDN router.The content replacement policies may be improved further considering the location of router in network topology as well as the probability of the requested data information.

The combination of probabilistic and bloom Based for caching the data in the Content Store may be tested in the future.

The scalability and the performance of this proposal can be tested for huge network topologies on NDN testbeds which are current live for NDN based real time applications.The NDN project is in development phase and many changes are implemented from time to time, thus scope of improvement is possible .

REFERENCES

[1]. M. Awais and M. A. Shah, "Information-centric networking: A review on futuristic networks," *2017 23rd International Conference on Automation and Computing (ICAC)*, 2017.

[2]. 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.

[3]. M. M. S. Soniya and K. Kumar, "A survey on named data networking," *2015 2nd International Conference on Electronics and Communication Systems (ICECS)*, 2015.

[4]. V. Jacobson, D. K. Smetters, J. D. Thornton, M. Plass, N. Briggs, and R. Braynard, "Networking named content," *Communications of the ACM*, vol. 55, no. 1, p. 117, 2012.

[5]. S. Shailendra, S. Sengottuvelan, H. K. Rath, B. Panigrahi, and A. Simha, "Performance evaluation of caching policies in NDN - an ICN architecture," *2016 IEEE Region 10 Conference (TENCON)*, 2016.

[6]. B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[7]. S. Tarkoma, C. E. Rothenberg, and E. Lagerspetz, "Theory and Practice of Bloom Filters for Distributed Systems," *IEEE Communications Surveys & Tutorials*, vol. 14, no. 1, pp. 131–155, 2012.

[8]. L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache," *ACM SIGCOMM Computer Communication Review*, vol. 28, no. 4, pp. 254–265, 1998.

[9]. F. Deng and D. Rafiei, "Approximately detecting duplicates for streaming data using stable bloom filters," *Proceedings of the 2006 ACM SIGMOD international conference on Management of data - SIGMOD 06*, 2006.

[10]. G. Zhang, X. Wang, Q. Gao, and Z. Liu, "A Hybrid ICN Cache Coordination Scheme Based on Role Division between Cache Nodes," *2015 IEEE Global Communications Conference (GLOBECOM)*, 2015.

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

[12]. I. Psaras, W. K. Chai, and G. Pavlou, "Probabilistic in-network caching for information-centric networks," *Proceedings of the second edition of the ICN workshop on Information-centric networking - ICN 12*, 2012.

[13]. S. Eum, K. Nakauchi, M. Murata, Y. Shoji, and N. Nishinaga, "Catt," *Proceedings of the second edition of the ICN workshop on Information-centric networking - ICN 12*, 2012.

[14]. W.K Chai, D. He, I. Psaras, "Cache in information-centric networks," *Proceedings of the 11th International IFIP TC 6 Conference on Networking*, vol. 7289,pp. 27–40, 2012)

[15]. L. Fan, P. Cao, J. Almeida, and A. Broder, "Summary cache: a scalable wide-area Web cache sharing protocol," *IEEE/ACM Transactions on Networking*, vol. 8, no. 3, pp. 281–293, 2000.

[16]. A. Kumar, J. Xu, J. Wang, O. Spatschek, and L. Li, "Space-code bloom filter for efficient per-flow traffic measurement," *Ieee Infocom 2004*.

[17]. A. Broder and M. Mitzenmacher, "Network Applications of Bloom Filters: A Survey," *Internet Mathematics*, vol. 1, no. 4, pp. 485–509, 2004.

[18]. O. Rottenstreich, Y. Kanizo, and I. Keslassy, "The Variable-Increment Counting Bloom Filter," *2012 Proceedings IEEE INFOCOM*, 2012.

[19]. X.-M. Tian, D.-F. Zhang, K. Xie, C.-Q. Shi, and X.-B. Yang, "Algebra Operations on Counting Bloom Filters," *Chinese Journal of Computers*, vol. 35, no. 12, p. 2598, 2012.

[20]. A. Kirsch and M. Mitzenmacher, "Less hashing, same performance: Building a better Bloom filter," *Random Structures and Algorithms*, vol. 33, no. 2, pp. 187–

218, 2008.

[21]. F. Hao, M. Kodialam, and T. V. Lakshman, “Building high accuracy bloom filters using partitioned hashing,” *Proceedings of the 2007 ACM SIGMETRICS international conference on Measurement and modeling of computer systems - SIGMETRICS 07*, 2007.

[22]. D. Geneiatakis, N. Vrakas, and C. Lambrinoudakis, “Utilizing bloom filters for detecting flooding attacks against SIP based services,” *Computers & Security*, vol. 28, no. 7, pp. 578–591, 2009..

[23]. B. Xiao and Y. Hua, “Using Parallel Bloom Filters for Multiattribute Representation on Network Services,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 1, pp. 20–32, 2010.

[24]. T. Chen, D. Guo, Y. He, H. Chen, X. Liu, and X. Luo, “A Bloom filters based dissemination protocol in wireless sensor networks,” *Ad Hoc Networks*, vol. 11, no. 4, pp. 1359–1371, 2013.

[25]. S. Dharmapurikar, H. Song, J. Turner, and J. Lockwood, “Fast packet classification using bloom filters,” *Proceedings of the 2006 ACM/IEEE symposium on Architecture for networking and communications systems - ANCS 06*, 2006.

[26]. Ndnsim.net. (2019). *ndnSIM Documentation — ndnSIM documentation*. [online] Available at: http://ndnsim.net/current/ [Accessed 18 Jun. 2019].

[27]. M. Varvello, D. Perino, and J. Esteban, “Caesar,” *Proceedings of the second edition of the ICN workshop on Information-centric networking - ICN 12*, 2012.

[28]. S. Mastorakis, A. Afanasyev, and L. Zhang, “On the Evolution of ndnSIM,” *ACM SIGCOMM Computer Communication Review*, vol. 47, no. 3, pp. 19–33, 2017.

[29]. S. Shailendra, S. Sengottuvelan, H. K. Rath, B. Panigrahi, and A. Simha, “Performance evaluation of caching policies in NDN - an ICN architecture,” *2016 IEEE Region 10 Conference (TENCON)*, 2016.

[30]. X.-L. Wei, M. Chen, J.-H. Fan, G.-M. Zhang, and Z.-Y. Lu, “Architecture of the Data Center Network,” *Journal of Software*, vol. 24, no. 2, pp. 295–316, 2013.

[31]. S. Mastorakis, A. Afanasyev, and L. Zhang, “On the Evolution of ndnSIM,” *ACM SIGCOMM Computer Communication Review*, vol. 47, no. 3, pp. 19–33, 2017.