ENHANCED CACHING MECHANISMS FOR NAMED DATA NETWORKING

SADAQ JEBUR TAHER

DOCTOR OF PHILOSOPHY
UNIVERSITI UTARA MALAYSIA
2023

Awang Had Salleh
Graduate School
of Arts And Sciences

Universiti Utara Malaysia

PERAKUAN KERJA TESIS / DISERTASI
(Certification of thesis / dissertation)

Kami, yang bertandatangan, memperakukan bahawa
(We, the undersigned, certify that)

SADAQ JEBUR TAHER

calon untuk Ijazah
(candidate for the degree of) PhD

telah mengemukakan tesis / disertasi yang bertajuk:
(has presented his/her thesis / dissertation of the following title):

"ENHANCED CACHING MECHANISMS FOR NAMED DATA NETWORKING"

seperti yang tercatat di muka surat tajuk dan kulit tesis / disertasi.
(as it appears on the title page and front cover of the thesis / dissertation).

Bahawa tesis/disertasi tersebut boleh diterima dari segi bentuk serta kandungan dan meliputi bidang ilmu dengan memuaskan, sebagaimana yang ditunjukkan oleh calon dalam ujian lisan yang diadakan pada : ***21 September 2022.***
That the said thesis/dissertation is acceptable in form and content and displays a satisfactory knowledge of the field of study as demonstrated by the candidate through an oral examination held on:
21 September 2022.

Pengerusi Viva: *(Chairman for VIVA)*	**Assoc. Prof. Dr. Fauziah Baharom**	Tandatangan *(Signature)*
Pemeriksa Luar: *(External Examiner)*	**Prof. Ts. Dr. Kamarulnizam Abu Bakar**	Tandatangan *(Signature)*
Pemeriksa Dalam: *(Internal Examiner)*	**Ts. Dr. Shahrudin Awang Nor**	Tandatangan *(Signature)*
Nama Penyelia/Penyelia-penyelia: *(Name of Supervisor/Supervisors)*	**Prof. Dr. Osman Ghazali**	Tandatangan *(Signature)*
Nama Penyelia/Penyelia-penyelia: *(Name of Supervisor/Supervisors)*	**Prof. Ts. Dr. Suhaidi Hassan**	Tandatangan *(Signature)*

Tarikh:
(Date) ***21 September 2022***

Permission to Use

In presenting this thesis in fulfilment of the requirements for a postgraduate degree from Universiti Utara Malaysia, I agree that the Universiti Library may make it freely available for inspection. I further agree that permission for the copying of this thesis in any manner, in whole or in part, for scholarly purpose may be granted by my supervisor(s) or, in their absence, by the Dean of Awang Had Salleh Graduate School of Arts and Sciences. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to Universiti Utara Malaysia for any scholarly use which may be made of any material from my thesis.

Requests for permission to copy or to make other use of materials in this thesis, in whole or in part, should be addressed to:

Dean of Awang Had Salleh Graduate School of Arts and Sciences
UUM College of Arts and Sciences
Universiti Utara Malaysia
06010 UUM Sintok

Abstrak

Pertumbuhan pesat Internet telah mengalihkan penggunaan utamanya ke arah pengedaran kandungan, kebanyakannya melalui Rangkaian Data Dinamakan (NDN). Sebagai komponen utama Rangkaian Berpusatkan Maklumat (ICN) untuk Internet masa hadapan, terdapat keperluan mendesak untuk membangunkan strategi pengagregatan yang berkesan untuk NDN. Oleh kerana penghala menghadapi cabaran dengan kekangan memori, kajian ini memperkenalkan Strategi Pengagregatan Alternatif (ACS) untuk menangani penggantian dan pengemaskinian kandungan. ACS menggabungkan dua mekanisme teras iaitu Mekanisme Penggantian Kandungan (CRM) dan Mekanisme Kemas Kini Kandungan (CUM). Kedua-dua mekanisme menggunakan Teori Set Kabur dan Pembuatan Keputusan Pelbagai Atribut Kabur (MADM). Secara khususnya, CRM menggunakan Teknik Keutamaan Pesanan dengan Persamaan kepada Penyelesaian Ideal (TOPSIS), manakala CUM menggunakan Kaedah Pemberatan Tambahan Mudah (SAW). Prestasi ACS dinilai melalui simulasi dan dibandingkan merentasi pelbagai topologi dengan strategi konvensional Masuk-Dulu-Keluar-Dulu (FIFO), Paling-Terkini-Kurang-Digunakan LRU), Paling-Jarang-Digunakan (LFU) dan Strategi Popularity Kandungan Cache (CCP). ACS menunjukkan peningkatan yang ketara dalam nisbah hit, daya pemprosesan dan pengurangan hop dan beban pelayan. Khususnya, peningkatan dalam nisbah pukulan purata ialah 43%, 30%, 21% dan 10% secara berurutan, peningkatan daya pengeluaran ialah 38%, 23%, 16% dan 9%, dan purata bilangan dalam lompat pengurangan ialah 21%, 18%, 14% dan 8%. Akhirnya, dalam purata beban pelayan, ACS mencatatkan peningkatan tertinggi dengan 22% daripada FIFO dan yang terendah dengan 10% daripada CCP. ACS menawarkan pendekatan yang berpotensi untuk pengedaran kandungan Internet di masa hadapan, terutamanya dalam aspek pelaksanaan teknologi IoT.

Kata kunci: Rangkaian Berpusatkan Maklumat, Internet Masa Hadapan, Pembuatan Keputusan Atribut Berbilang, Penggantian Kandungan, Kemaskini Kandungan.

Abstract

The rapid growth of the Internet has shifted its primary use towards content distribution, prominently through the Named Data Network (NDN). As a pivotal component of Information-Centric Networking (ICN) for the Future Internet, there's a pressing need to develop effective caching strategies for NDN. As routers face challenges with memory constraints, this research introduces the Alternative Caching Strategy (ACS) to address content replacement and updating. The ACS integrates two core mechanisms: The Content Replacement Mechanism (CRM) and the Content Update Mechanism (CUM). Both mechanisms employ Fuzzy Set Theory and Fuzzy Multiple Attribute Decision Making (MADM). Specifically, the CRM uses the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS), while the CUM adopts the Simple Additive Weighting (SAW) Method. The performance of ACS is evaluated through simulations and compared across various topologies with the conventional First-In-First-Out (FIFO), Least Recently Used (LRU), Least Frequently Used (LFU), and Cache Content Popularity (CCP) strategies. ACS demonstrated significant improvements in hit ratio, throughput, and reduced hops and server load. Specifically, the enhancements in average hit ratio are 43%, 30%, 21%, and 10% sequentially, throughput enhancements are 38%, 23%, 16%, and 9%, and the averages of reduction hops number are 21%, 18%, 14%, and 8%. Finally, in the average server load, ACS registered the highest enhancement with 22% from FIFO and the lowest with 10% from CCP. The ACS presents a promising approach for the future of Internet content distribution, particularly in the realm of IoT technology deployment.

Keywords: Information-Centric Networking, Future Internet, Multiple Attribute Decision Making, Content Replacement, Content Update.

Declaration Associated with This Thesis

Some of the works presented in this thesis have been published or submitted as listed below.

[1] Sadaq Jebur Taher, Osman Ghazali, and Suhaidi Hassan. “Multi Attribute Decision Making for Content Replacement Mechanism in Named Data Networking”, IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 24, Issue 2, Ser. III (Mar. –Apr. 2022), PP 01-13

[2] **Taher, Sadaq Jebur**, Osman Ghazali, and Suhaidi Hassan. "The Evaluation of Caching Replacement Strategies in Future Networking." The 6th International Conference on Internet Applications, Protocols and Services (NETAPPS2020), 2020

[3] **Taher, Sadaq Jebur**, Osman Ghazali, and Suhaidi Hassan. "A Review on Cache Replacement Strategies in Named Data Network." Journal of Telecommunication, Electronic and Computer Engineering (JTEC) 10, no. 2-4 (2018): 53-57.

[4] **Sadaq Jebur Taher**, Osman Ghazali, and Suhaidi Hassan, “"A Review on Cache Replacement Strategies in Named Data Network" The 5th International Conference on Internet Applications, Protocols and Services (NETAPPS2017), 2017

[5] **Sadaq Jebur Taher**, Osman Ghazali, and Suhaidi Hassan, “Replacement Techniques in named Data Network during Video on demand workload: A Review" Technical Report, UUM/CAS/InterNetWorks/TR2017-17 InterNetWorks Research Laboratory, School of Computing, Universiti Utara Malaysia, 2017.

Acknowledgements

In the name of ALLAH, Most Gracious, Most Merciful:

"*Work; so Allah will see your work and (so will) His Messenger and the believers*;"

(The Holy Quran - AtTawbah 9:105)

Conducting this research marks the end of an interesting and eventful journey. The completion of this dissertation signifies the fulfillment of a long-awaited goal. It could not have been achieved without the professional academic and personal support of the following wonderful and talented people. I will start by extending my deep and sincere gratitude to my supervisors Prof. Dr. Osman B Ghazali and Prof. Ts. Dr. Suhaidi Hassan (School of Computing, Universiti Utara Malaysia) for their tireless encouragement, wisdom and experience, who provided me with constant guidance and constructive criticism throughout all stages of my research. Apart from my devotion, professionalism, daily meetings, and progress inspection were the main reasons for success. Working with them in the field of NDN increased my understanding and improved my intellection. May Allah give them its reward and make them happy.

I must say a huge thank you to the current and past members of InterNetWorks Research Lab whom I enjoyed working with them. I would like to thank the team of the Internet Society's Next Generation Leaders (NGL) program while serving as ISOC Fellow to the Internet Engineering Task Force (IETF). I would also like to thank Dr. Abdul Malek bin Yaakob for guiding throughout the strenuous exercises of Multiple Attribute Decision Making (MADM) that will never be forgotten. Finally, my heartiest gratitude goes to my family, to my father and mother late, to my beloved wife Rasha for her understanding, support, and love, to my sons Jaafer, Osamah and Abhaj, and my daughter Tabarak for being so sweet and loving, to my sister, and to my brothers who always has faith in me and prays for my success.

Dedication

For my family . . .

in memory of my father;

in memory of my mother;

in memory of my father in Law;

my wife ***Rasha****;*

our sons ***Jaafar****,* ***Osamah****, and* ***Abhaj****;*

our daughter ***Tabarak****;*

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ABC	-	Age-based Cooperative
ACS	-	Alternative Caching Strategy
ARC	-	Adaptive Replacement Cache
CAGR	-	Compound Annual Growth Rate
CCP	-	Cache Content Popularity
CDN	-	Content Delivery Network
CHP	-	Content Hit Popularity
CMP	-	Content Miss Popularity
CN	-	Content Networks
CNN	-	Content-Centric Network
COMET	-	COntent Mediator nETworks
CPT	-	Content Popularity Table
CRM	-	Content Replacement Mechanism
CS	-	Content Store
CUM	-	Content Update Mechanism
DNS	-	Domain Name Service
DONA	-	Data Oriented Network Architecture
DRAM	-	Dynamic Random Access Memory
EBA	-	Elimination by Aspects
ELECTRE	-	Elimination et Choice Translating Reality
FIB	-	Forwarding Information Base
FIFO	-	First-In-First-Out police
FP7	-	Framework Program 7
GB	-	Gigabyte
GPL	-	General Public License
IC	-	Information-maximizing Caching

List of Abbreviations

ICC	-	Influenced Closeness Coefficient
ICN	-	Information-Centric Networking
ICNRG	-	Information-Centric Networking Research Group
IETF	-	Internet Engineering Task Force
IOT	-	Internet of Things
IP	-	Internet Protocol
IRTF	-	Internet Research Task Force
ISPs	-	Internet Service Provider service
LAM	-	Linear Assignment Method
LCE	-	Leave Copy Everywhere
LCprob	-	Leaving Copies with Probability
LCUNiP	-	Leaving Copies with Uniform Probability
LFRU	-	Least Frequent Recently Used
LFU	-	Least Frequently Used
LRU	-	Least Recently Used
MADM	-	Multiple Attribute Decision Making
Mbps	-	Megabits per second
MZipf	-	Maldelbrot- Zipf
NAT	-	Network Address Translation
NDN	-	Named Data Networking
NFD	-	NDN Forwarding Daemon
NICC	-	Normalised Influenced Closeness Coefficient
NLSR	-	Named-data Link State Routing Protocol
NS3	-	Network Simulation Version3
OSI	-	Open Systems Interconnection
P2P	-	Peer-to-Peer

List of Abbreviations

PBCE	-	Popularity Based Content Eviction
PIT	-	Pending Interest Table
PPC	-	Popularity Prediction Caching
PPT	-	Popularity and Priority Table
PSIRP	-	Publish Subscribe Internet Routing Paradigm
PURSUIT	-	Publish Subscribe Internet Technology
SAIL	-	Scalable and Adaptive Internet Solutions
SAW	-	Simple Additive Weighting method
SL	-	Server Load
SMART	-	Simple Multiple Attribute Rating Technique
TB	-	Terabyte
TCP	-	Transmission Control Protocol
TFN	-	Triangular Fuzzy Numbers
TLRU	-	Time Aware Least Recently Used
TOPSIS	-	Technique for Order Preference by Similarity to Ideal Solution
TTL	-	Time To Live
UGCs	-	User-Generated Contents
VOD	-	Video-On-Demand

CHAPTER ONE
INTRODUCTION

The rapid increase in the rate of Internet usage in recent times is as a result of its use for content distribution. Its utilisation date back to the 60s and 70s when there was the utilisation of the Internet's point-to-point paradigm at the start of computer networks [1, 2, 3]. There has been a dramatic increase in the amount of Internet traffic resultant from content distribution in the past decade. For instance, in 2019, the traffic accounted from video alone was 80% of the global Internet traffic, and studies have forecast there will be a continuous increase reaching to 83% of global Internet traffic by 2021 [4, 5, 6].

This constant and rapid growth in the amount of Internet traffic regarding content distribution and related phenomena have been a severe challenge to the global Internet scalability. The reason for this inverse effect is partially attributed to the truth of that the Internet is unsuitable for content distribution which happens to be the present major usage case.This is because, the original design of the Internet envisaged its major use case as host-to-host communications. Therefore, there is a general assent that the most efficient approach to solve the problems of content distribution is by adopting globally distributed in-network caches. This is because, a deversity of add-on patches, such as Content Delivery Networks (CDN), Peer to peer overlays, Mobile IP, and Network Address Translation (NAT), which they were not the original design's part, all violate, in different approach, diverse characteristics of the initial Internet architecture [7, 8, 9, 10, 11].

This has led, as an initial short-term solution, to the Content Delivery Networks (CDN) deployment. This widespread deployment has quickly grown in terms of the presence and universally traffic carriage. Thus, the rapid growth direction is forecast for still

continue over the years to come, to the extent that, by year 2019, 62% of Internet traffic will cross a CDN [4]. This leads to a simultaneous huge evolution of the CDN market from USD 6.05 Billion to USD 30.89 Billion (forecast from 2017 to 2022), at a Compound Annual Growth Rate (CAGR) of 32.8% the base year is 2016 [12].

This spiralling increase in content delivery traffic has led to the intensifying of a number of associated phenomena that challenge the reliability and stability of the Internet infrastructure. One of such phenomenon is the flash crowd events, which consists of a sudden and unexpected placing of request by many users, of one or few content items in large volume, from a specified content publisher. This kind of events are now very common and possess the possibility of heavily impacting the operations of the affected content providers. This can only be avoided if adequate countermeasures or proactive capacity planning are initiated [7, 8, 13].

Other solutions, asides CDN, also have been suggested to address the issue challenging global Internet more regularly on the long-term. This include reorganization Internet architecture to its prevalent content distribution use case [7]. Information-Centric Networking (ICN) projects have proven that the aspiration will soon be a success in line with the huge support it has been receiving through European Union FP7 projects, ICN Research Group (ICNRG) at Internet Research Task Force (IRTF) [14]. For example, the ICN paradigm which was recently proposed, has the aim of diverting from the existing Internet architecture whose major abstract is characterized by location-based on repository addresses and transmition to the architecture, whereby the location-agnostic content identifiers is the major abstract. The implementation of this paradigm has been proposed by a number of specific architectures. However, the most prominent one, known as Named Data Networking (NDN) (also known as CCN [15]). NDN means every network router envisages the ubiquitous deployment

of content caches [16, 17, 18].

The distributed caching advantages are not limited to its use as a defensive measure when handling swiftly growing traffic, it also serves as an approach to improve consumer experience through congestion reduction and cache efficiency. As soon as congestion occurs, the retransmission of data is assisted via caching. The retransmission interest will align with the data right on the link where this data lost the packet, as opposed to the original transmitter. In this way, the occurrence of congestion collapse is avoided by NDN which is a possibility in current Internet once lost packet is at the last hop, and bandwidth is regularly consumed by frequent retransmits from the original main host. In fact, the geographical distribution of replicas for contents requested the most, guarantees a possibility of content retrieval by users from nearer positions and then facing the larger throughput and less congestion [19]. Therefore, this proposal provides analytical, design, and implementation of the alternative caching strategy in Named Data Networking.

1.1 Background of the Study

NDN makes the routers have the cache capability of content objects passing by, which is represented as a particular different from current Internet. The routers made current days are equipped with packet buffers and their main usage are for scheduling of packet and line cards queuing. In-router, the distributed caches constitute the cache of in-network. These caches have the ability to achieve remarkable improvements in network performance through the increase of the cache hit rate and a decrease of the upstream traffic. However, in-network caches aim at long term storage of content objects, which is advantageous for the satisfaction of retransmitted requests in respect of frequent duplicate the requests, packet loss and user mobility [17, 20, 21].

It is unrealistic to cache all the traffic of content objects navigating in high-speed and huge-scale through a router. Therefore, the choosing of the content objects to cache is very important to maximize the efficiency of the cache. Intuitively, popular contents should be cache by routers because there is a high probability of these contents being requested again. For this reason, there is a wide study of popularity-based policies when considering the overall cache performance in NDN [17, 22, 23, 24].

The NDN router are all supplied with a Content Store (CS) which stores data packets strategically as they pass through the router. As soon as an interest packet is received, there is an immediate check of the content store by an NDN router. If the store contains a data which has a name aligns under the Interest's name, thus, this data will be return back as an answer immediately. This implies a hit of cache. The basic form of the CS is merely the buffer memory on line cards of current routers. It is worth taking note of that both IP routers and NDN routers of buffer data packets. The characteristic which sets these routers apart is that NDN routers have the ability to reuse data because they have persistent names identification, whereas the cache packets of IP routers is mainly used for packet queuing and scheduling, and does not have the ability to re-use the data after forwarding them. For constant contents, it is considered that NDN achieves nearly optimal data delivery. Caching also benefits dynamic content for example, in the case of multicast, teleconferencing or retransmission the packet when loss data packet occurs. Cache replacement and management are a very key feature for efficiency in cache storage utilization and content delivery [23, 24, 25].

NDN has evolved from the Information Centric Networking (ICN) research area as shown in 1.1, which has also inspired many other Internet architectures [21, 26]. Recently, researchers have analyzed the key features and issues of NDN as the future Internet architecture (FIA) [27, 28, 29, 30, 31, 32, 33].

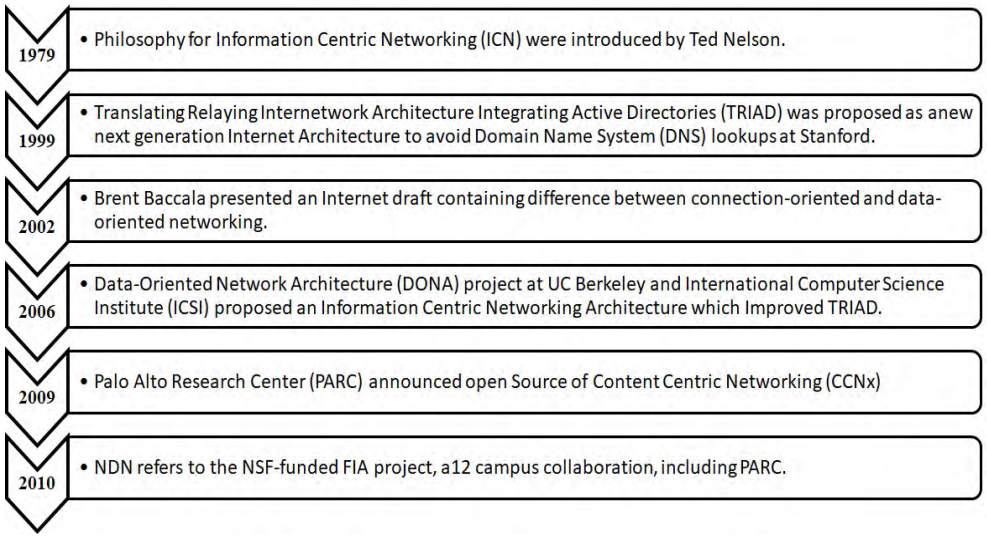


Figure 1.1: NDN Timeline [17, 26, 30]

1.2 Motivation of the Study

In today's context, multimedia has evolved into a feature that is indispensable on the Internet, in regard to the advanced developments encountered in digital media and networking technologies. This is aligned to the increasing popularity gained by animation, audio and video clips on the Internet, and likewise attributed to the Information- Centric Networking (ICN) development within the industry's wider and academic domains [17, 34]. There has been the creation of many distributed multimedia applications, including Internet telephony, Internet video conferencing, Internet collaboration that combines audio, video and whiteboard, Internet TV, on demand streaming or broadcasting, distributed simulation, distance learning, entertainment and gaming, multimedia messaging, etc [14, 35, 36].

There are several funded projects on the design of future content-dependent Internet paradigms. Interestingly, NDN has evolved as the promising filter out of the executed projects. NDN directly handles location-independent names and application generated variable-length for searching and retrieving contents of any consumer who requests, not being limited by their hosting entity [14, 20, 24, 37]. NDN, being the NDN router

are all supplied with a Content Store (CS) which stores data packets strategically as they pass through the router. As soon as an interest packet is received, there is an immediate check of the content store by an NDN router. If the store contains a data which has a name aligns under the Interest's name, thus, this data will be return back as an answer immediately. This implies a hit of cache. The basic form of the CS is merely the buffer memory on line cards of current routers. It is worth taking note of that both IP routers and NDN routers of buffer data packets. The characteristic which sets these routers apart is that NDN routers have the ability to reuse data because they have persistent names identification, whereas the cache packets of IP routers is mainly used for packet queuing and scheduling, and does not have the ability to re-use the data after forwarding them [38, 39].

For constant contents, it is considered that NDN achieves nearly optimal data delivery. Caching also benefits dynamic content for example, in the case of multicast, teleconferencing or retransmission the packet when loss data packet occurs. Cache replacement and management are a very key feature for efficiency in cache storage utilization and content deliveryhe future Internet's forerunner, suggests an infrastructure of network redesign built around the content. This has represented an important moving from the current communication of sender-driven Peer to peer, to a communication of content distribution that is receiver-driven. This new architecture for the network aims to address the current Internet's many issues (content distribution efficiency, security, congestion, etc.) and thus provide users with better communication of network. However, the congestion of the network is mitigated the quality of service when a network is loading more data than it is able to handle [30, 40].

In early 2019, studies revealed that the prevalent social networks Facebook and WhatsApp have together over two billion active users monthly [41, 42]. This implies

that over 25% of the entire Earth's population accesses their Facebook and WhatsApp account at least once every month. The scale of the Internet is so great, that it is almost impossible to grasp and hence is not so reasonable to observe and analyse such information on either a monthly or daily basis. This growth is clearer depicted in Figure 1.2 that appears the data type and its number that is transmitted in an Internet every minute [43].

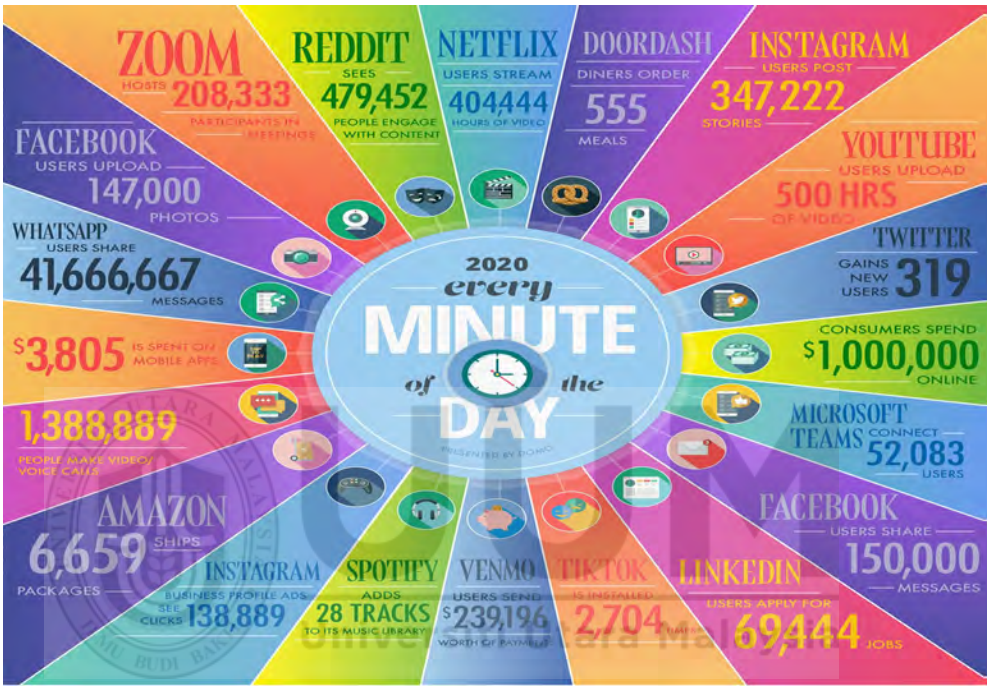


Figure 1.2: What Happens Every Minute on the Internet in 2020 [43]

What happens in an Internet Minute: user share 150,000 messages on Facebook, user apply for 69,444 jobs on LinkedIn, user upload 147,000 photos on Facebook, user shear 41,666,667 messages on WhatsApp, hosts 208,333 participants in meeting on Zoom, users stream 404,444 hours of video on Netflix, users post 347, users upload 500 HRS of video on YouTube, 1,388,889 people make video or voice calls, 6,659 ships packages on Amazon, business profile ADS see clicks 138,889 on Instagram, adds 28 Tracks to its music library on Spotify, 222 stories on Instagram, 319 gains new users on Twitter, connect 52,083 users on Microsoft Teams, diners order 555 meals on DOORDASH, $ 3,805 is spent on MobileApps, users send $ 239,196 worth of payments on Venmo, and consumers spend $ 1 Million online. [43].

Due to these findings, researchers have stressed that the NDN is flexible and supports many of encoding formats allowed by streamer which is a pipeline-based multimedia framework [21]. Thus, publishers are made procure able to sign content object from the input multimedia. This multimedia could either be pre-recorded or live and then stored in node. Whereas, a packet is dropped over a path, then, the used transmission capacity at each of the upstream will links to forward that packet to the point [23, 44, 45]. However, as soon as there is an occurrence of congestion, the retransmission of data is supported via caching as long as the retransferred interest will meet the data right over the lost packet link, and not at the original transmitter. The congestion of network is experienced when demand for a resource is higher that the capacity of the resource. Therefore, the concept of congestion management is likened to achieving cost-savings while preserving subscriber Quality of Experience (QoE). These two objectives most times contradict each other, and there is a need to strike a balance[23, 46, 47, 48]. Thus, in today's Internet, NDN avoids the occurrence of congestion collapse when the lost packet is at the last hop and the bandwidth is mostly consumed by frequent retransfers from the original main host.

1.3 Problem Statement

It has been observed that media file streaming is growing largely in size daily and an inevitable aggravation of the congestion of the Internet is being experienced, with latency now being perceived by the user. For the NDN router, it caches every travelling data packet, and this data packet is fetched again from a physically close distance when a request to the identical data is made. A challenge with NDN is that the node of the intermediate nodes talks directly back to the consumer node and sends messages to the next node in line, therefore, many changes happen incrementally [15, 49, 50, 51, 52, 53, 54].

Specifically, the popular data packet gets the most requests, with a large portion of data packets, which are stored in the cache, being never requested again, and thus leading to an occupied cache [22, 55, 56, 57]. This occupied cache by the data packets not requested, may be stored in the cache forever, except there is an occurrence of some new contents that are of higher popularity and higher total hits. Therefore, cache occupancy may consist of previously popular data but recently used, even though its rarely used. Subsequently, the newly arriving content cannot be saved [57, 58, 59, 60]. The cached data can optionally have a long-freshness-period before its replacement, because the expiration of the freshness-period only means that the producer may have produced newer data [61]. This purpose is indispensable to the dissemination of information that is accessed by a large number of users, such as, disaster-related information or a viral video [60, 62, 63, 64, 65, 66].

There is a lack of cache efficiency and total cost simultaneously increases in the event of unavailability of content in the cache at the time the content is requested, as the content is still fetched and served as soon as possible to the user. The challenge arises in scenarios where routers attain full memory and there is no space to accommodate the contents that are newly arriving. Thus, in order to manage the routers' limited cache size, eviction of the stored contents is initiated [58, 60, 65, 67, 68, 69, 70]. As a consequence, the storage node capacity adopted for caching requested contents is way smaller than the collective total size of the contents that can be requested, since content replacement is unavoidable. For example, a problem will occur when the evicted data packet is requested again in the future and it has a high cost, or stored data packets exist forever in the cache, in addition to fact that data cached may quickly become obsolete as they are transient and require to update [67, 69, 70, 71, 72, 73, 74, 75].

It is assumed that a certain amount of cost is incurred when data packets are fetched

from their provider. It is up to the cellular operator, the data packet provider, or both, on how this cost is defined. However, if the condition were switched and the data packet is not cached, fetching of the data packet has to be from its provider which directly translates to the payment of certain costs. Moreover, if the cache has attained its maximum capacity and is filled up, there needs to be an allocation of cache capacity for the new data packets, through eviction of some data packets from the cache. Bear in mind that a wrong selection of the data packets being evicted may impose an increase in the total cost incurred in the long run, specifically in a situation where there is a future request for the already evicted data packet.

To relate the aforementioned explanation to a certain scenario, we consider an empty network at initial time = 0. At this time, all Consumer nodes will have no problem sending interest packets out as they are small and the network allows for a large number of interest packets. Then the intermediate nodes will use their forwarding strategies to forward the interest packets across the shared link towards the correct publishers in accordance with the sought out prefixes. As the network has no former traffic at this time, all data is at the publisher nodes and the network will have closer characteristics to the traditional IP-based networks. The problem encountered in the cache is as shown in Figure 1.3 below.

For the interest packet traffic, the Consumer1, Consumer3 and Consumer4 packet streams do not share CS yet. Hence, the Publisher1 and Publisher2 starts to answer every interest packet with a data packet sent the same route back towards the respective consumer. Each node has CS and PIT to keep control of all data packets and interest packets received and sent out. For this reason, the rate of data packets arriving at the intermediate node will either equal the number of interest packets sent to the publisher node, or it will equal the available bandwidth from the publisher to the intermediate

node. Where the maximum data packets received will be capped by the bandwidth and the minimum will be equal to the interest packet frequency. This will be the first shaping of the network capacity and may create congestion as the number of interest sent upstream can be greater than the capacity to send data back downstream. This means that that at the intermediate node, there will now be arrival of interest packets and data packets that both share a CS.

Therefore, this study will design a content replacement mechanism and a content update mechanism to obtain a new model of alternative caching strategy depending on both cache popularity and cache priority. This study will further validate and verify the performance of the alternative cache strategy for NDN.

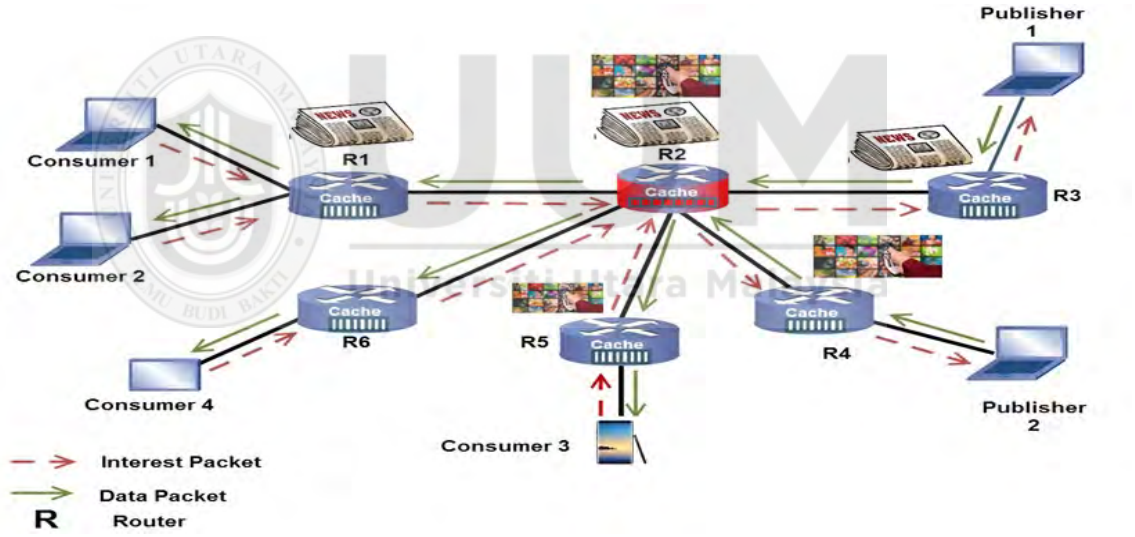


Figure 1.3: Research Problem

1.4 Research Questions

In addressing NDN cache management, the following research questions have to be answered:

i. How to evict content items in the content store in order to increase cache hit ratio

and reduce server load?

ii How to keep data fresh in the content store to increase throughput, and decrease average number of hops ?

iii. What is the impact of the proposed strategy on throughput, server load, cache hit ratio, average number of hops ?

1.5 Research Objectives

The purpose of this research is to design a new cache management strategy in NDN to achieve better performance in terms of throughput, reduce server load, cache hit ratio, bandwidth, and low latency, during content replacement. This aim can be further clarified by considering the following specific research objectives:

I. To design a Content Replacement Mechanism (CRM) based on the properties of Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) and its flexibility of content popularity and priority, with the use of on-path caching to improve cache hit ratio, and decrease server load.

II. To design a Content Update Mechanism (CUM) based on the characteristics of Simple Additive Weighting (SAW) and its coherence to content update, by using information freshness and on-path caching to increase throughput and decrease average number of hops.

III. To propose the deterministic and consistent case scenarios of content generation by utilizing satisfactory statistical distribution, and then comparative the performance evaluation of the current NDN caching approaches and the proposed new strategy.

1.6 Research Scope

This study aims to provide a wide outline of ICN and particularly the architecture of NDN. The focus is on the design of mechanisms appropriate for content prioritisation and increasing cache hit ratio and decrease server load during content popularity. In addition, the proposed strategy increases throughput and reduce the average number of hops to cache efficiency and the total cost. Moreover, the alternative caching strategy is designed to take into consideration the most demanding data. Examples of multimedia applications that are used and/or requested by millions of consumers every day, are distance learning, interactive games, VoD, patient monitoring systems, remote robotic agents, and so on. The scope of this research is presented in Figure 1.6.

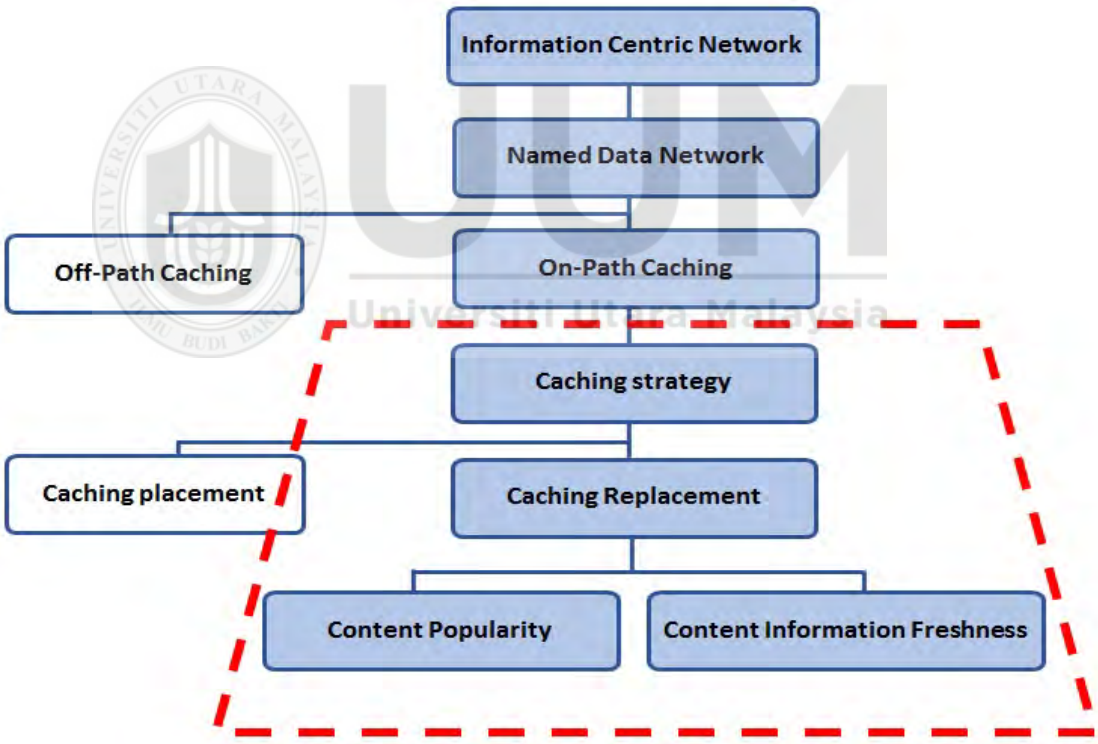


Figure 1.4: Scope of Research

In the given Figure 1.4, the ICN caching is contained two parts, i.e., on-path caching and off-path caching. The proposed caching replacement strategy is under the on-path caching. While Caching strategy is also provides two parts which are caching

placement strategy and caching replacement strategy. Caching placement stratey is out of the scope. This proposal focuses about cache replacement strategy which contains content popularity and content prioritisation schemes. The alternative caching strategy is based on both of popularity and prioritisation schemes.

1.7 Significance of the Research

This study would have a actual contribution to demand more Internet multimedia requests when the node has lower data congestion. Consequently, the advantage of multimedia requests would impact both publishers and customers of Internet multimedia positively [49]. In other words, the new strategy will be more gains for ISPs. This leads to give revenue for the host of multimedia indirectly, when the frequent and long buffering of the multimedia requests are mitigated. Moreover, the study would contribute to the academic sector through the integration of the previous works on NDN, multimedia, and Internet of Things (IoT).

Caching affects the nature of high information reachability to end-users of the Internet. The current impact of users demanding multimedia would mitigate the problems of data flooding, packet loss, bandwidth utilization, and in locations that are not needed. When the idea of NDN caching is standardized, the entire NDN research community will benefit from its implementation through:

1. The novel caching strategies in NDN that will enhance the manageability through careful content popularity, content prioritisation and content information freshness in a content store,

2. The alternative caching strategy that can be used in NDN approach designs with marginal configuration complexity,

3. The suitable simulation tools by including the proposed ideas through programming.

1.8 Thesis Organization

The organisation of this thesis as follows:

Chapter One begins with the general introduction of the study, then discusses the motivation and the background of the Study. After that, a comprehensive overview of the proposal is presented with respect to the these points: problem statement, research questions, research objectives, research scope, research Steps, significance of the research and research contributions.

Chapter Two provides an extensive review of previous, related literature and presents existing works on NDN by discussing the basic architecture and attributes of NDN. A critical review of previous works on the categories for the cache replacement strategies in NDN are highlighted and the information freshness in NDN is also discussed in detail. A critical review of previous works on the Multiple Attribute Decision Making (MADM) are highlighted.

Chapter Three focuses on the design of a conceptual model of alternative caching strategy and methodology approach to realize the objectives of this research.

Chapter Four is related to the modeling of alternative caching strategy and its two mechanisms for content replacement and content update, named CRM and CUM, respectively. The chapter discusses the design motivation of ACS along with its verification and validation.

The chapter also describes two techniques of fuzzy set theory, based on Multiple Attribute Decision Making (MADM), called TOPSIS and SAW.

Chapter Five evaluates the performance of ACS and its comparison with the FIFO, LRU, LFU, and CCP through simulations. The simulation metrics for evaluation are cache hit ratio, throughput, server load, bandwidth and average number of hops.

Chapter Six expresses the conclusion as well as the contributions and limitations of the research work introduced in this thesis, and then suggests further directions for future studies in line with some pivotal NDN research challenges.

CHAPTER TWO
LITERATURE REVIEW

In accomplishing the stated objectives of this study, the literature highlighted in this chapter serves as part of the requirements needed to achieve the research goals. This chapter gives detailed discussions of pertinent literature which are interesting in this study's context. This chapter covers the architecture of NDN, in-network Caching, and Cache Replacement Strategies. The chapter further focuses on information freshness in the NDN and Multiple Attribute Decision Making (MADM). Then the discussion of the replacement strategies through the analysis and the selection of replacement strategies.

2.1 NDN Architecture

Fetching of content by NDN is conducted by names, where the implementation of these names is a part of the process in facilitating the search for content and/or content retrieval. Naming schemes are independent of the network, and also application-specific [14, 15]. Therefore, the names required to retrieve a global data is expected to be globally unique, and this is ensured by NDN as consumers can retrieve data content using specified hierarchical data names [14, 46, 76] . This affirms the capability of the NDN architecture to represent future Internet architecture in ensuring accurate and swift delivery of online requests [46, 77].

The consumer initiates an NDN communication in the form of an interest packet (see Figure 2.1 (a)). When an interest packet reaches to a node or a content provider produces a valid requested content, a data packet is issued for that interest packet (see Figure 2.1 (b)). Content Networks (CNs) are established in the data packet and interest packet. The data packet return back by tracing the path of the interest packet

in inverse to arrive at the consumer. The node returning a cached copy of the content interested is referred to as 'publisher'.

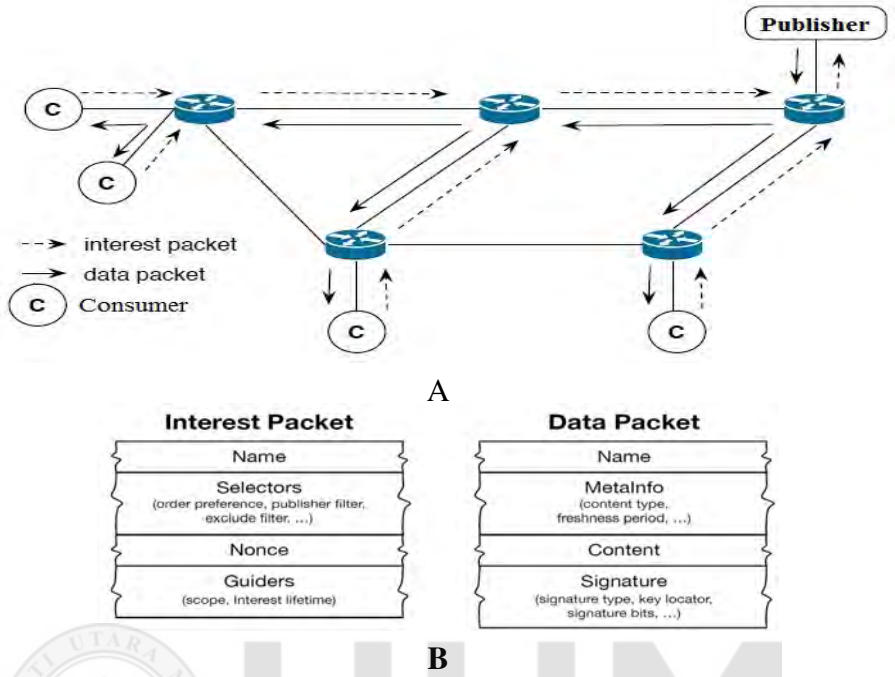


Figure 2.1: Packets in NDN Architecture [15]

2.2 In-network Caching

The NDN package having the name and signature of data comprises of requested or retrieved independent projects. Thus, receiving data packets can be cached by the router in the content store, and this can be used in meeting demands in the future. The content store has a lot of similarities to the buffer cache in IP routers, although there is no reuse of IP routers after forwarding of package to the destination, while the routing devices in NDN can be reused. In certain scenarios, NDN handles storage channels and network that are like to it, in terms of data retrieval. When considering content delivery and statistical context, the achievement of NDN covers almost the entire optimal data delivery and maximizes the advantage of caching in the event where there is retransmission or multicast after packet loss [38, 76].

Additionally, there is a raise of consumer name unlike that of IP when caching data, thus allowing packet headers' privacy concerns to be examined and loading data consumers [76, 78]. Controlling which data is requested is not facilitated by naming and caching of data in NDN networks. However, in the absence of destination addresses, the determination of who requests it becomes difficult. Therefore, a radically different kind of privacy protection of existing IP networks is offered by the NDN. Many researchers, when considering the concept of caching have affirmed this in the network, such as the foundational advantage of building ICN in context [78]. Irrespective of the fact that the NDN has the ability to supporting CDN architectures of TCP/IP that are more powerful, NDN make provision of a wide variety of other functions such as data that lead to many insurance gains, and displays strong and significant advantages [76, 79].

In-network storage or content caching [14] at intermediate nodes is essential in supporting the basic concept of an NDN delivery model that is content-centric, peer-to-peer, and low in costs. There are many advantages to caching in NDN, such as, when contents produced by other nodes are cached, it helps the contents to fully dissociate from their producers. Furthermore, it allows the overhead to move to the producer side. Additionally, since it allows multiple copies of the same content to be available in the network, it helps avoid a single point of failure. It likewise provides the dynamic contents with high benefits in case of retransmission or multicast due to packet loss. While significantly lowering the data distribution latency and network load [14, 15, 20].

Cache decision strategy is used to determine if the data packet is cached within the intermediate routers or not. To cache an strategy efficiently, two important concerns have to be answered, which includes, where the content should be cached and which

content should be replaced first. Thus, caching can be broadly classified into cache replacement (decision for the content to be cached at the router) and cache placement (decision for the content to be placed at the network or not).

2.2.1 Cache Placement

Leave Copy Everywhere (LCE) is the main cache placement strategy that is often built into the architecture of NDN [14, 20, 80, 81, 82, 83]. In the LCE strategy, a data packet is cached in all the existing routers between consumer and the producer (or provider). This strategy is capable of inducing significant cache redundancy. In other words, it can cache the same content at multiple nodes, therefore reducing the cached contents' diversity in the whole network. Some of the current cache placement strategies that are used to reduce cache redundancy include Leaving Copies with Uniform Probability (LCUniP) and Leaving Copies with Probability (LCProb) [80]. LCProb uses the caching probability 1/(hop count) to cache the content at the router while LCUniP uses uniform probability to cache the content at the router.

Popular contents should be stored at the network edge in order to raise the cached contents' utility in the whole network, lessen the content downloading latency, and improve the cache diversity of the whole network [84, 80, 85, 86]. An effective coordination scheme [80] for the routers is also needed to reduce cache redundancy and improve cache diversity. Cache coordination strategy can be categorised into two types; explicit and implicit. For the explicit scheme, cache network topology, content access frequency, and the cache state information of the network must be known beforehand in order to make placement decisions regarding the content. Further categorisation of this scheme is of three kinds:

1- Path coordination, where decision making includes all the cache nodes found along

the path between the producer/provider and consumer.

2- Global coordination, where decision includes all the cache nodes.

3- Neighbourhood coordination, involving all the cache node's neighbours.

For implicit cache coordination, it is not required that each cache node know the other cache nodes' state information. They also only exchange minimal information with the others before the content placement decision is taken.

2.2.2 Cache Replacement

A very popular strategy for cache replacement is the Least Recently Used (LRU) strategy, which refers to content that is discarded due to being least recently accessed [87, 88, 89]. The popularity of this strategy is due to its well known impressive performance, and also its increased chances of obtaining a cache hit as it possesses the property of being able to store the most recent data for a longer period of time. Another important cache replacement strategy is the Least Frequently Used (LFU) strategy. This strategy is also considered important due to its feature of being able to identify less frequently used contents and dispose them first [89, 90]. When considering the decision timing of the cache, its response can be attributed to content replacement result and content arrival at the router. Note that, in order to improve network performance, there should not be deletion of any cache content. However, for caching, one can initiate a one-level upstream movement within the cache hierarchy. If one decides to further classify cache replacement, it is done with respect to either content popularity or content prioritization schemes. This implies that the contents that are replaced first, are contents having low popularity or priority. However, when considering the approaches adopted by both LRU and LFU

strategies, it is observed that LRU does not consider content popularity, but rather takes into consideration content timestamp, while for the contents in LFU, distribution frequency is calculated. This results in inaccuracy when computing the degree of the newly incoming content [91, 92]. Table 2.1 presents comparison between different caching schemes respectively which are based on popularity or prioritization schemes as explained below:

Content Popularity Scheme In recent times, the approach adopted to deal with challenges encountered in cache replacement strategies when considering content popularity, is to maintain a data structure displaying content names with their popularities available at the router's cache [19, 91]. In order to ensure that there is an improvement in the performance of the network, an important role is played by the cache replacement for a dynamic network. Although when considering strategies such as LRU and LFU, which are founded on access-time-patterns, they lack the ability to optimally utilize the popularity of the contents within the NDN network. Hence, to achieve the aim of using an efficient cache strategy, the NDN network requires a content popularitybased cache replacement strategy [93]. Whereas, cache performance improvement can be accomplished via the strategy proposed by Ran et al. [91], such that cache replacement is dependent on the content popularity. This strategy is referred to as Cache Content Popularity (CCP).

When considering data structure, a Content Popularity Table (CPT), responsible for information storage, was added on the Content Store (CS). Specific information such as content name, cache hit, and instant/past popularity are stored. The CCP will then compute the present popularity of the periodic content with respect to cache hit, and content access frequency, and thereafter, initiate a replacement of the least popular content with the other contents existing in the store. A comparison of this new

proposed strategy to the LRU and LFU strategies, shows a better performance of the proposed strategy. In addition, for CCP table update, the suggested manner has the highest CPU time consumption, and this property is not ideal for large-scale networks.

Content Prioritisation Scheme Lately, through the utilization of the knapsack problem, content priority adoption was initiated to enable access to cached content [94]. In a situation where a content is assigned priority, the objective of this priority is to determine the content to be exchanged first. In a case where there a quick meet-up of two mobile nodes, an interchange of their high priority of contents is done during their encounter time. Content prioritization plays a very important duty in application performance of extremely effective networks. It is known that high priority content has a lot of network availability in comparison to low priority contents. Moreover, it is also known that contents of low priority possess high access latency. Thus, the major point to consider is how to determine the priority of each content. The two major priorities are setting the factors which are the content demand and the general information that is generated or swapped within the nodes.

Ming et al. [67] suggested an algorithm aimed towards priority for ICN nodes with respect to content age, and applied using an age-based policy in order to avoid extensive calculations during the computations. If the two conditions, age of content has expired and node's cache memory is full, are satisfied, a replica is replaced by other object(s). A strategy was proposed by Dron et al. [94] for content prioritization in-network, which can be adopted for cache replacement. The authors also laid emphasis on the advantages of information delivery data naming, thus maximising the cached content within the ad-hoc networks. Thereafter, a categorization of every item is made to either be cold or hot. Therefore, contents in the hot category are exchanged first in the event that an encounter occurs. In order to determine what contents to be

categorised in the hot category, Dron et al. [94] developed the knapsack problem. For this problem, the criterion for classification is that the items within the knapsack must be able to maximise their utility through all the replies to the user query. Then the categorization follows that items that are utilized-maximising are labelled hot. Three performance metrics are used to analyse the information-maximising strategy, which include responses per query (throughput), coverage per query (amount of space occupied by a query response in case of a sensor), and average delays (latency).

Table 2.1: *Comparison Between Different Caching Replacement Schemes*

Auther	Caching Strategy	Year	Replacement Scheme	Architecture	Replacement speed
Zhang, Yuanzun and Tan, Xiaobin and Li, Weiping	**PPC**	2017	Content popularity	ICN	Medium
Bilal, et al. [95]	**LFRU**	2017	Content popularity	ICN	Medium
Lal, Kumari Nidhi [50]	**PBCE**	2017	Content popularity	CCN	Medium
Bilal, et al. [96]	**TLRU**	2014	Content prioritization	ICN	Fast
Ming, Zhongxing [67]	**ABC**	2014	Content prioritization	ICN	Fast
Ran, et al. [91]	**CCP**	2013	Content popularity	NDN	Medium
Dron, et al. [94]	**IC**	2013	Content prioritization	NDN	Fast

2.3 Cache Replacement Strategies

The cache replacement strategies have an important role in NDN. Replacement strategies play a very important role in the achievement of the highly sophisticated mechanism of the cache. This replacement strategy conducts the eviction of data packet from the cache and afterwards, create a new space for the incoming data packet. However, due to the limited size of the cache, it is unable to provide storage for the entire requested data packet. For this reason, cache replacement strategies need to make room for new data packet [88, 97]. Thus, the cache replacement strategy a cache

administrator chooses can significantly impact global network traffic as well as local resource utilization.

The research into cache replacement strategies is very active. A search in the literature reveals the existence of several cache replacement strategies that have been proposed for use in-network. Replacement strategies can be grouped into several broad categories depending on the key features of the workload and/or cached document used to make a replacement. The researchers [98, 99] are underlined that the key features are :

(1) Recency: time of the last reference to the data packet,

(2) Frequency: number of requests to a data packet,

(3) Size: size of the data packet,

(4) Cost of fetching the data packet: cost to fetch a data packet from its publisher, and

(5) Access latency of data packet.

The table 2.2 offers the replacement decision based on the key feature(s). This table is representing the existing state of affairs in this area.

Table 2.2: *Features Used by Replacement Policies*

No.	Replacement Decision	Feature				
		Size	Recency	Frequency	Cost	Latency
1	Popularity Prediction Caching (PPC)	X	-	X	-	-
2	Least Frequent Recently Used (LFRU)	-	X	X	-	-
3	Popularity Based Content Eviction (PBCE)	-	-	X	-	X
4	Time Aware Least Recently Used (TLRU)	-	X	-	-	X
5	Age-Based Cooperative Caching (ABC)	-	-	X	X	-
6	Cache Replacement Policy Based On Content (CCP)	-	X	-	-	-
7	Information-maximizing Caching (IC)	-	X	-	-	-
8	Adaptive Replacement Cache (ARC)	-	X	X	-	-
9	Least Frequently Used (LFU)	-	-	X	-	X
10	Least Recently Used (LRU)		X	-	-	X
11	First-In-First-Out (FIFO)	-	-	-	-	-

2.3.1 Popularity Prediction Caching

Popularity prediction caching (PPC) refers to a caching replacement scheme (chunklevel) within the network that makes predictions of the future popularity state for video chunks. Record and ranking of content popularity in PPC is based on the number of historical request. Although, this statistical method approach for chunk-level popularity has its shortcomings, most especially with respect to video applications. The role of PPC is basically for the prediction and caching of the most popular chunks in the future, while evicting the chunks with least popularity in the future via linear complexity. Thus, PPC is dependent on previous popularity, consistency content and occupied cache [72].

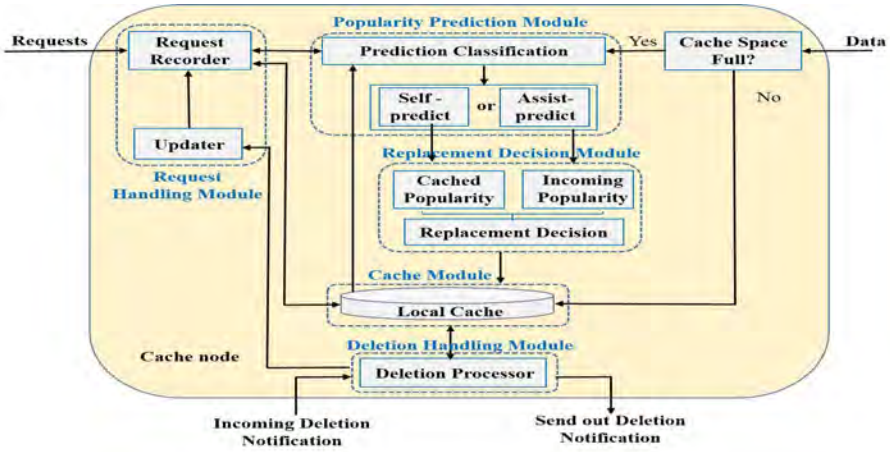


Figure 2.2: Popularity Prediction Caching

2.3.2 Least Frequent Recently Used

This strategy is developed based on LFU approximation and a partitioned LRU, and referred to as the Least Frequent Recently Used (LFRU) eviction policy. It is an efficient content eviction policy, and its implementation approach follows a division of the cache into two partitions, namely the privileged partition and the unprivileged partition. In the privileged partition, the LRU replacement policy is adopted while the unprivileged partition adopts an approximated LFU (ALFU). This ALFU documents the access history of the content for a limited time window (W). Thus, LFRU follows the approach of evicting stored content, in the event that the newly arriving content satisfies a qualifying condition. This implies that LFRU is dependent on previous popularity, but is independent of consistency content and occupied cache [95].

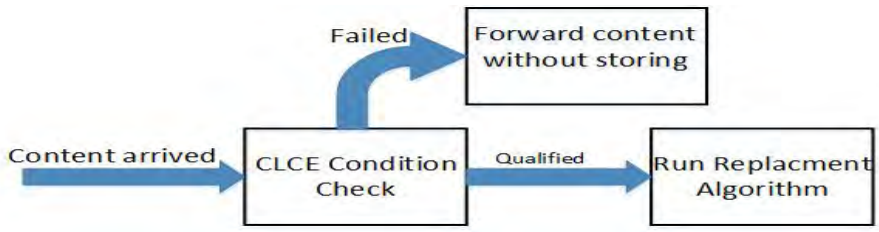


Figure 2.3: Least Frequent Recently Used

2.3.3 Popularity Based Content Eviction

The Popularity Based Content Eviction (PBCE) scheme for CCN evaluates content's local popularity via the adoption of a betweenness-centrality concept. Before a content is cached, the CR has to make certain decisions as regarding eviction. This decision is a determining factor to select which content should be evicted and which content should remain in the CS. After a CR has been selected for content caching using the aforementioned betweenness-centrality approach, one content from the CS is thereafter chosen by PBCE with respect to its lower instantaneous hit rate. After the content is selected, PBCE initiates a calculation of the local popularity and universal popularity of the selected content, coined LP(Pop(Ci)) and UP(Cj) respectively, for the content that is arriving newly in the CS.

If the condition $\mid UP(Cj) - LP(Pop(Ci)) \mid \geq 0$ is satisfied, then the newly arriving content replaces the selected content. Whereas, in the case where the condition is not satisfied, no task will be executed, rather, the content that is newly arriving will be forwarded directly to the client. For a content is cached using betweenness-centrality, its popular context will continually be cached within a given number of CRs. This enables the achievement of maximum performance with respect to cache hit ratio, although some congestion is experienced in these CRs. In order to avoid this congestion, a threshold value of time Tc is set. This implies that after CR R3 performs content caching, then the CR R3 will not execute any other caching in the duration of time Tc ms as shown in Figure 2.4. During that period of time, the next CR possessing high value of betweenness-centrality will be chosen for content caching [50].

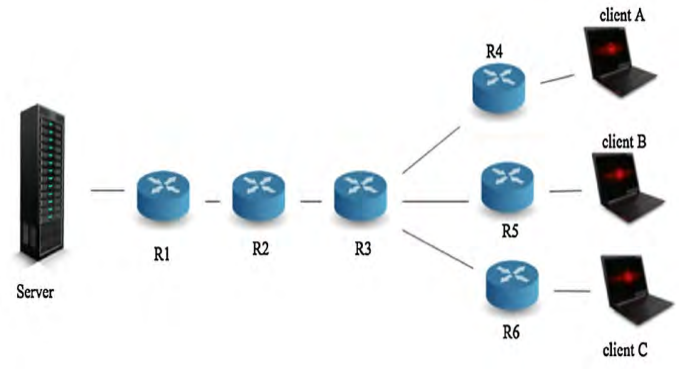


Figure 2.4: Popularity Based Content Eviction [50]

There can be an incorporation of all these factors into the replacement decision. Depending on these factors, the replacement strategies can be classified based on six categories: content priority based strategy, content popularity based strategy, content priority popularity based strategy, Recency/Frequency based polices, Recency-based polices, and Frequency-based polices.

2.3.4 Time Aware Least Recently Used (TLRU)

TLRU is a priority-based content life time aware eviction strategy, which is an extension of the simple LRU. LRU as a commonly deployed cache management strategy, maintains contents in cache with respect to their access time. In a case where free cache space is unavailable and there is an arrival of a new content, the least recently used content is deleted to create space for the newly arriving content. LFU strategy on the other hand, makes a sorting of the queue by access frequency. The arrival of a new content when the cache space is full brings about a replacement of the lowest frequency content of the queue. Nevertheless, as a result of the nature of LRU and LFU, it is possible to evict the most popular content by a least popular one. In overcoming this problem, [96] proposed the Time Aware Least Recent Used (TLRU) strategy.

In this strategy, the time stamp of a content that just arrived is being calculated locally by a cache node. If the average request time is smaller than the time stamps of the stored contents, the arriving content is cached. A storage of the content is made by TLRU if there is available space in cache, otherwise there is an application of the simple LRU on the cached contents, so as to create space for the arrival of new content. TLRU is proposed for the removal of contents that are relatively less popular, unlike LFU and LRU which are eviction strategies that are non-protective. In this strategy, as shown in Figure 2.5, a change of the eviction order could be made at any instant of time without there being a corresponding change in the contents stored in cache. From the given figure, the arrow represents one state transition of First-In-First-Out (FIFO) for some contents.

Findings from research affirm that when comparing TLRU to LRU or LFU, TLRU outperforms both of them in the event that there is available space in cache, otherwise the simple LRU strategy is used in creating space for the newly arrived content. However, referring to Cisco Visual Networking Index [100], the extreme popularity of the delivery of multimedia applications on the Internet, comes with the forecast that the accumulation of Multimedia formats will account for about 86 percent of the user traffic worldwide. As a result of this spiking increase in the population of Internet users and their demand for multimedia applications, there is a high chance of filling CR caches in a network quickly. When such a case comes to pass, the TLRU will not be effective for eviction of conThere can be an incorporation of all these factors into the replacement decision. Depending on these factors, the replacement strategies can be classified based on six categories: content priority based strategy, content popularity based strategy, content priority popularity based strategy, Recency/Frequency based polices, Recency-based polices, and Frequency-based polices.tent anymore. This new proposed strategy performs better compared to LFU and LRU. Additionally, the

suggested manner does not mention about Update Data for Name Content and depends on recency in the end.

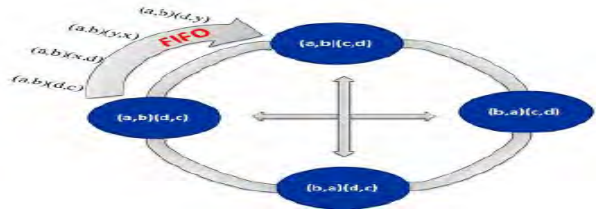


Figure 2.5: Time Aware Least Recently Used [100]

2.3.5 Age-based Cooperative Caching in Information-Centric Networks (ABC)

Where an object's TTL on a router's cache depends on object's popularity and router's location. The objective of the age-based cooperative scheme is to perform a dynamic configuration of content age in an implicit cooperation manner within the ICN routers. The scheme's principle has the following features [67].

• Every router has an age. The age of a replica in each router determines the lifetime of the replica.

• The age of the replica is obtained when it is added to the cache and a removal from the cache is initiated when the age expires.

• Routers implicitly collaborate through age modification. Age guidance is done using content location and popularity, which obey the following rules:

• The closer a replica is to the network edge, the longer age it has.

• The more popular a replica is, the longer age it has.

A mini network detailing intuitively how the scheme works is shown in Figure 2.6. In Figure 2.6, two routers (R1 and R2) make the connection between one client and one server, where the server serves two contents such that one is more popular than the other. The process follows that the client successfully makes a request of the two contents with respect to their popularity. Although, to make it simpler, the researcher follows the assumption that both routers have the storage capacity of one replica (Table 2.11 The researcher then starts at the initial phase with the condition that the two routers have no content cached (Figure 2.6(a)). Later on, the client will request for the popular content, where both routers have the content cached, and this leads the system to Figure 2.6(b). Based the approach followed by our scheme, when considering the same content, R1 assigns it a longer age than R2. Then after some time has passed, there is an expiration of the popular content in R2, but there is still room for it to be cached by R1 (Figure 2.6(c)). In the event that the less popular is required at this time, then R2 will automatically cache the replica. Therefore, there is a migration of the system to Figure 2.6(d), which is an optimal state, due to the fact that there is a minimization of aggregated network delay and publisher load. Due to the condition imposed by the researcher such that the popular content is provided with longer age over the less popular one, the chances of the popular content being replaced is less, which tends towards system maintenance in this state.

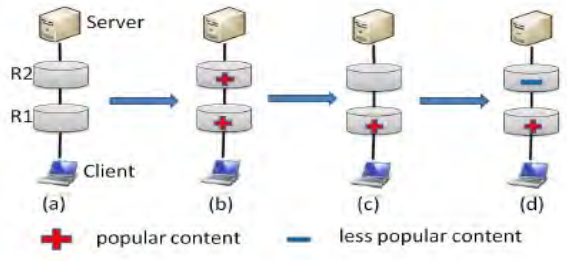


Figure 2.6: Age-Based Coopopularity Prediction Caching Perative Scheme [67]

This leads to the situation where all network devices have the ability to cache contents.

Furthermore, the performance of the origin server may be impacted by validation requests from widespread caches, or even overwhelm it.

2.3.6 Cache Content Popularity (CCP)

This is a cache replacement strategy based on content popularity. Content routing is adopted for the determination of the state of cache replacement strategy by considering the size of the remaining space available on the current repository. The process follows that after a data packet yet to be stored is received by a content router, there is a need to compute the calculation of the difference between the threshold value of the cache size and the numerical value of cached contents to obtain the value of the cache space size remaining. In a scenario where the data packet is larger than the rest of the space, then replacement will occur. On the basis of CCP, there will be a replacement of the content having the minimum popularity and its record in the Content Popularity Table (CPT) will be deleted simultaneously. Then the newly arrived content will be cached in Content Store (CS) and its own record in the CPT will be set up at the same time [91]. This is shown in Figure 2.7 below. This new proposed strategy performs better compared to LFU and LRU. Additionally, the suggested manner does not mention about Update Data for Name Content and its popularity depends on the hit rate like LFU strategy.

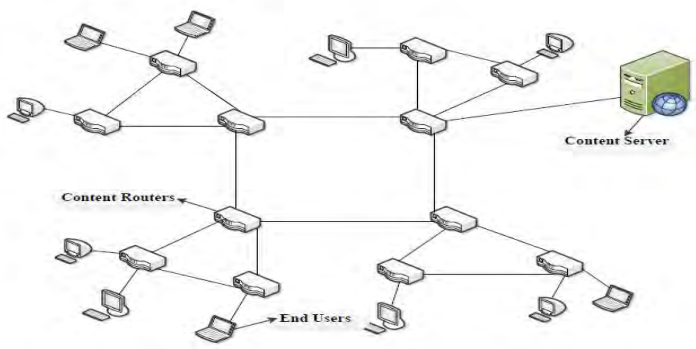


Figure 2.7: The Topology of the Network for CCP [91]

2.3.7 Information-maximizing Caching

Through the process of assigning hierarchical names to data, NDN gives network devices the room to understand relations between content, where the concept of similarity is one of such important relation. Delivering similar content items has sub-additive utility, and having a proper name-space design (that is, similar objects have similar names) makes it possible to conduct a clustering of objects with respect to their names. For instance, every branch at a user-selected level in the name tree can define a cluster, and this correspondence between names and clusters is all that is needed for optimization of utility by name-based replacement policies, as discussed subsequently [94].

Collection of sensor data by the application might be initiated from three different locations which are identified by the top-level portion of the name, where each location makes up a cluster. The collection of multiple sensor readings is carried out in each location, and the readings are more similar or redundant, therefore carrying less information jointly. A partial redundancy can be identified by the fact that the readings have the same name prefix in common. In such a case, the caching problem is treated as the famous knapsack problem, which serves as a very suitable comparison. Definitions to describe what benefit is measure by are most times ambiguous, because it is expected that the benefit should be dynamic and decreasing with reference to the expiration times of the objects. In the event where the expiration time is not a factor, then recency information would need to be considered by the benefit. Bearing in mind that while considering recency information, there should also be a consideration of how recently the last request to the object being considered occurred, which leads to more space overhead for objects not in our cache. This further adds more complexity to the entire process [93, 94].

2.3.8 Adaptive Replacement Cache (ARC)

The fundamental thought behind ARC in Figure 2.8 is to keep up two LRU arrangements of pages [101]. One rundown, say L1, contains pages that have been seen just once "as of late", while the other rundown, say L2, contains pages that have been seen at any rate twice "recently". The things that have been seen twice inside a brief timeframe have a low between entry rate, and, henceforth, are considered as "high-recurrence". Subsequently, the researcher consider L1 capturing "recency" while L2 as catching "recurrence". the researcher attempt to keep these two records to generally a similar size, in particular, the reserve measure 'c'. Together the two records recall precisely double the quantity of pages that would fit in the store. At the end of the day, ARC keeps up a reserve registry that recalls twice the same number of pages as in the store cache. Whenever, ARC picks a variable number of latest pages to keep from L1 and from L2. The exact number of pages drawn from the rundown L1 is a tunable parameter that is adaptively and constantly tuned [101, 102, 103].

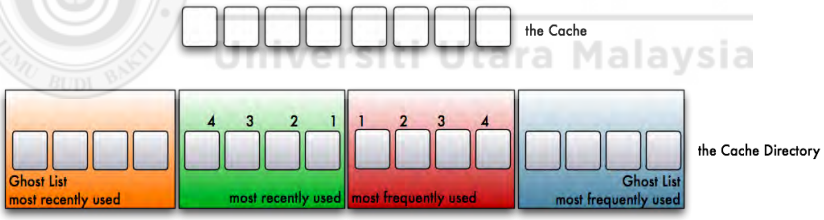


Figure 2.8: Adaptive Replacement Cache

This empowers a slick component of, let's accept read through a lot of documents for log record preparing, requiring each page just once. A typical slightest as of late utilized based reserve would stack every one of the pages in the store, accordingly ousting all most of the time which got to pages. As they get to them just once, they simply top off the store without presenting any focal points.

2.3.9 Least Frequently Used (LFU)

LFU as a cache replacement strategy also follows the concept of removing requests. Although, in this case, the least frequently used request is removed, to be replaced by a new request in the free space available in the cache [89, 90, 92, 98, 99]. This strategy also possess that advantage of simplicity and ease of use, similar to LRU. The concept follows maintaining a counter for counting the frequency of the request, while replacing less frequent request with new incoming ones.

Table 2.3: *Example of LFU Multimedia Replacement Strategy*

Requests	*M7*	*M0*	*M1*	*M2*	*M0*	*M3*	*M0*	*M4*	*M2*	*M3*
Part3			M1,1	M1,1	M1,1	M3,1	M3,1	M3,1	M2,1	M2,1
Part2		M0,1	M0,1	M0,1	M0,2	M0,2	M0,3	M0,3	M0,3	M0,3
Part1	M7,1	M7,1	M7,1	M2,1	M2,1	M2,1	M2,1	M4,1	M4,1	M3,1

For a clear understanding of the LFU strategy, the researcher assumes having "10" requests as Multimedia pages, that are arranged sequentially as (M7, M0, M1, M2, M0, M3, M0, M4, M2, and M3) based on the time of request for each object. This is presented in Table 2.3 above, which gives an example of how LFU caching replacement strategy is being implemented. It is further suggested that the three parts of cache are empty when initializing, and the frequency count has an initial value staring at zero for all requests (frequency of all requests=0). Thus, for instance, the first three requests (M7, M0, M1) are inserted in the three parts of cache (Part 1, Part 2, and Part 3) sequentially because these parts of cache are empty.

The request (M2) is saved instead of (M7), based on three reasons: (1) The three parts of cache saved three requests of Multimedia (M7, M0, M1), which makes the cache full. (2) All the three parts of cache have the same value of frequency count (frequency=1). (3) The request (M7) is the request that was done before the requests M0 and M1. Note that the last request (M3) gives another case that saves the request

(M3), instead of the request (M4). This case occurs as a result of three reasons:

(1) The cache is becoming full. (2) The frequency count is different (frequency of M4=1, frequency of M0=3, and frequency of M2=1). Therefore, the request (M0) must not be replaced with another request due to its high frequency. (3) The request (M4) is the request that was done before the request (M2).

Based on the fact that it is expected for a network to adequately serve thousands of popularity prediction cachingrequest per second, it becomes very important that the overhead required to achieve this is constrained to a minimum. As a result of this, there should be an eviction of requests that are used less frequently. Thus, in making a choice of which requests to retain between the frequently used and the less frequently used, the former is selected over the latter, based on their proven usefulness over time [97]. Although, some arguments have evolved over time to disagree with this approach of choice, because it is inferred that in future, there is a possibility that the frequently used requests may not be required. However, this situation is hardly experienced in a lot of scenarios. For example, it is known that static requests of heavily used Multimedias are most times requested repeatedly by individual users of the request. In such a case, to achieve maximum use of the cache, the LFU cache replacement strategy evicts the least frequently used requests or resources.

2.3.10 Least Recently Used (LRU)

One of the cache replacement strategies that is easy to use and simple to understand is the Least Recently Used (LRU) replacement strategy. This strategy is implemented using the concept of timestamp, and the process involves a removal of the least recently used object, that is, the object that has not been made use of in a long time. The removal initiated when the maximum size of the cache is exceeded, involves a new

object being introduced in replacement of least recently used evicted object [87, 88, 89, 98, 99, 104]. Whereas, when considering a case where the cache size has not attained its maximum size, the object is inserted in the cache.

Table 2.4: *Example of LRU Multimedia Replacement Strategy*

Requests	*M7*	*M0*	*M1*	*M2*	*M0*	*M3*	*M0*	*M4*	*M2*	*M3*
Part3			M1	M1	M1	M3	M3	M3	M2	M2
Part2		M0	M0	M0	M0	M0	M0	M0	M0	M3
Part1	M7	M7	M7	M2	M2	M2	M2	M4	M4	M4

In order to clearly understand LRU strategy, the researcher assumes having "10" requests as Multimedia pages. These requests are arranged sequentially as (M7, M0, M1, M2, M0, M3, M0, M4, M2, M3), based on the time of request for each object as shown in the Table 2.4 above. The table presents an example of how LRU caching replacement strategy works. In addition, the researcher suggest that the three parts of cache are empty when starting. For instance, the first three requests (M7, M0, M1) are inserted in the three parts of cache (Part 1, Part 2, and Part 3) sequentially, because these three parts of the cache are empty.

The request (M2) is saved instead of the request (M7), for two reasons:

(1) The three parts of cache have saved three requests of Multimedia (M7, M0, M1), and that makes the cache full.

(2) The request (M7) is the least recently used (that is, M7 is the request that was initiated before the requests M0 and M1).

2.3.11 First-In-First-Out (FIFO)

Another cache replacement strategy is the First-In-First-Out strategy, with the acronym FIFO. The strategy operates by setting and maintaining a finite-sized queue of tags T. Therefore, it is possible to model a concrete k-way FIFO cache set S, as k-tuple cache tags, being arranged from left to right in descending order (that is, starting from last-in to first-in) [90, 98, 99, 105]. Note that there will be no change in the cache set when a cache hit occurs, there can be an incorporation of all these factors into the replacement decision. Depending on these factors, the replacement strategies can be classified based on six categories: content priority-based strategy, content popularity based strategy, content priority popularity based strategy, Recency/Frequency-based polices, Recency-based polices, and Frequency-based polices. However, a new tag is inserted at position 0 when a cache miss occurs. In the event of a cache miss, the others are shifted to the right after the new tag is inserted, and the tag at the rightmost position is evicted. Modelling the effect of cache set when evaluating a cache block denoted by the tag 'T' ', involves adopting the update function US: $S \times T \rightarrow S$ [106].

Table 2.5: *Example of FIFO Multimedia Replacement Strategy*

Requests	*M7*	*M0*	*M1*	*M2*	*M0*	*M3*	*M0*	*M4*	*M2*	*M3*
Part3			M1	M1	M1	M1	M0	M0	M0	M3
Part2		M0	M0	M0	M0	M3	M3	M3	M2	M2
Part1	M7	M7	M7	M2	M2	M2	M2	M4	M4	M4

To better understand the FIFO strategy, the researcher assumes having "10" requests as Multimedia pages, that are arranged sequentially as (M7, M0, M1, M2, M0, M3, M0, M4, M2, and M3) based on the time of request for each object (details are given in Tabl 2.5 above). Table 2.5 presents an example of how FIFO caching replacement strategy works, with the assumption that the three parts of cache are empty when starting. For example, the first three requests (M7, M0, M1) are inserted in the three parts of the cache (Part 1, Part 2, and Part 3) sequentially because these three parts of

cache are empty. On the other hand, the request (M2) is saved instead of the request (M7), because of two reasons:

(1) The three parts of the cache saved three requests of Multimedia (M7, M0, M1), making the cache full. (2) The request (M7) is the request that was initiated first before the requests M0 and M1. Likewise, the sixth request (M3) is saved instead of the request (M0), for two reasons:

(1) The three parts of cache have saved three requests of Multimedia (M2, M0, M1), and this makes the cache full. (2) The fifth request (M0) is not saved but only matched with the previous cache that is (M2, M0, M1). Therefore, in this case the fifth request (M0) is already saved. FIFO cache replacement strategy is advantageous over other cache replacement strategies for a number of reasons [106]. Firstly, caches utilizing FIFO have been proven to consume lower energy, mostly in comparison to other caches utilizing the LRU replacement strategy [107]. Another advantage of FIFO is in its design simplicity, which makes it a cache replacement strategy that is not expensive to implement. As advantageous as the FIFO cache replacement strategy is, it is also burdened by certain shortcomings. One of the disadvantages encountered by FIFO in static analysis, deals with its inability and difficulty to achieve extraction of may information [105].

2.4 Information Freshness in the NDN

As discussed in the previous section, scalability and performance of a communication network is improved through caching. In the process of caching, cache replacement is a very important phase and it plays a very vital role in serving the cache's motive [64, 88, 97]. A cache is of no advantage if the data in its storage is stale or invalid. In a case where this happens, high congestion is experienced in the intermediate node as a result

of the stale data, likewise, latency and delay occurs which leads to overhead instead of being beneficial. The freshness of a cache is analysed based on its content age and it signifies the period of time the cached value can be utilized without a validation of the original data source [108].

In 2009, Content-Centric Networking (CCN) was proposed by Jacobson et al. [15]. where a recommendation of a version number field was made in content name format, but with no specific mention to cache consistency. Another study by Lixia Zhang et al. [14] presented Named Data Networking (NDN) design, and proposed a suggestion for the use of Time To Live (TTL) approach (a widely adopted concept in DNS) in resolving the issue of cache consistency. When adopting the TTL approach, a conservative TTL is set by the origin server for its content. In most cases, even on expiry of the TTL, the content still remains valid. The cache has the ability to send a request to the origin server for content validation. If an unmodified reply is received, the content's TTL can be refreshed. Whereas for the server invalidation approach, such as Lease, the relevant cache is notified by the origin server of the information regarding the modification of the content.

All these approaches discussed have the same basic mode of operation, such that the origin server has two roles, which are to serve content and to validate/invalidate caches. This is achievable in traditional network due to the fact that there is limited number of caches (e.g. CDN surrogate servers or web caches) and most importantly, these caches have a controlled manner behaviour. Although, there is a dramatic situation change in NDN where every network device has the ability to cache contents. The performance of the origin server may be impacted by the validation requests from the widespread caches, or even overwhelm it [108, 62, 63].

Communication in NDN is driven by receivers i.e., data consumers, through the exchange of two types of packets, which are Interest and Data. These two carry a name identifying a piece of data, with the possibility of being transmitted in one data packet [17, 38, 109, 110]. At all events, MetaInfo appear highlight for information freshness as shown in Figure 2.9 [21, 61].

- The optional Freshness-Period gives an indication of the period of time a node should wait after this data arrival before marking it “non-fresh”. Bear in mind that the “nonfresh” data is still valid, and the Freshness-Period expiration simply refers to the fact that the producer may have produced newer data. The encoded value is number of milliseconds.

- If the data packet satisfies, Freshness-Period > 0, it should be initially considered by a node as “fresh”. Then after the data has stayed in the node for Freshness-Period milliseconds, it will now be marked as “non-fresh”. If the Data does not have a Freshness- Period or if, Freshness-Period = 0, it must be marked as “non-fresh”immediately.

- If an Interest contains Must-Be-Fresh element, “non-fresh” data must not be returned by the node in response to this Interest. The effect it has is same as the case such that “non-fresh” data is non-existent. This implies that there is a possibility the Interest is matched by another Data in the store, or, if that fails, then it is forwarded to other nodes. On arrival of an an exact duplicate of the “non-fresh” Data packet with a positive Freshness-Period value at the node, the node re-mark the data packet as “fresh” for the required period of time [61].

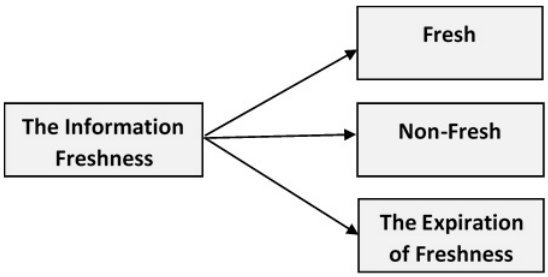


Figure 2.9: The Information Freshness [61]

When caching is adopted, there will be a lot of duplicates of the same content within the network. For this reason, in a situation where the origin content copy changes sometime, the information freshness among all the copies has to be documented and taken into consideration. For contents with no modification after publishing, for example most movies, the information freshness does not arise as a problem. Likewise, for contents requiring real time update, such as stock index, the information freshness is not required either due to the fact that contents will not be cached at all. However, occasionally changing contents, like blog and software, the most suitable approach is caching them while simultaneously incorporating the information freshness mechanism [108, 58, 61, 62, 63, 67, 71, 85].

2.5 Multiple Attribute Decision Making (MADM)

Zadeh (1965) and Bellman and Zadeh (1970) proposed fuzzy sets to address the issues of uncertain or linguistic information and furthermore, serve as a generalization of conventional set theory. Considering the successful applications of fuzzy sets in the field of automatic control, it has been recently been incorporated into MADM for handling MADM problems in cases or scenarios of subjective uncertainty. Figure 2.10 details the holistic development of MADM..

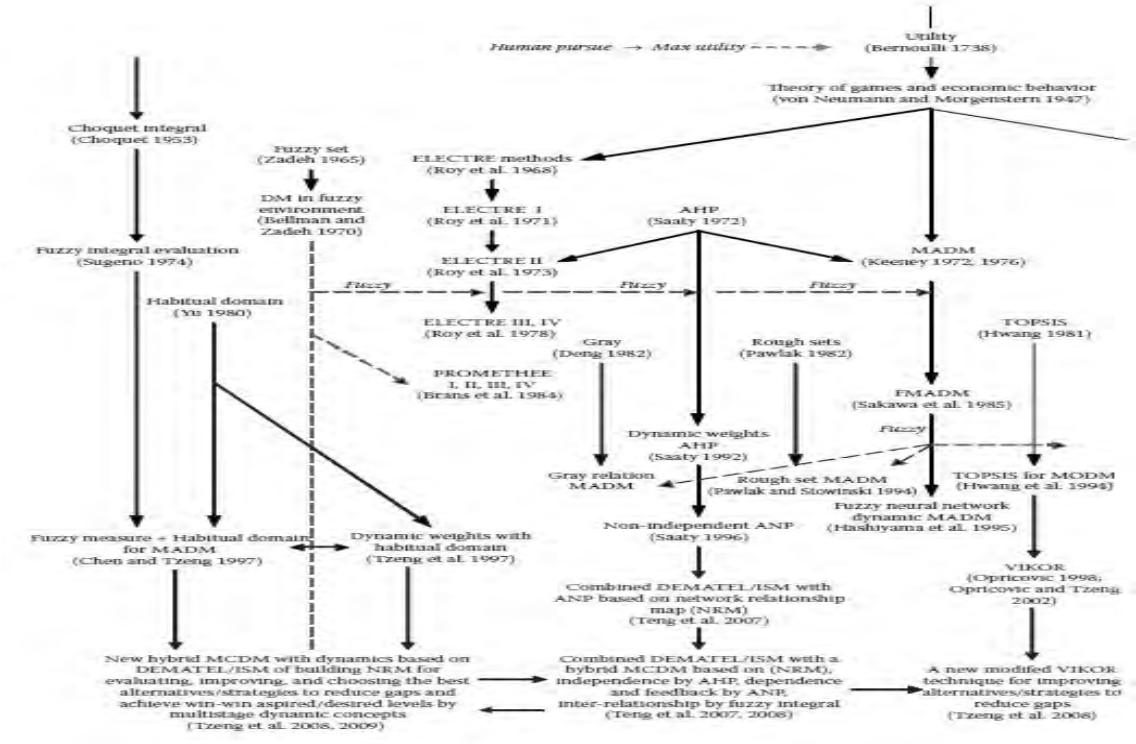


Figure 2.10: Development of Multiple Attribute Decision Making [111]

Considering problems that involve selecting from a fixed number of alternatives, MADM rises to the challenge as an approach to be employed in solving the problems. In arriving at a choice from the alternatives, the method works by specifying the manner in which an attribute information should be processed. Both intra-attribute and interattribute comparisons are required by MADM methods. Appropriate explicit tradeoffs are also involved in the method.

There are four major categories in every decision table (or decision matrix) of MADM methods. These categories include alternatives, attributes, attribute weight or relative importance, and performance measures of alternatives with respect to each attribute. A sample of the general outlook of the decision table is given in Table 2.6. From the table, all categories are evident, as seen in alternatives, $A_{i|\{i=1,2,...,N\}}$, attributes, $B_{j|\{j=1,2,...,M\}}$, attributes' weights, $w_{j|\{j=1,2,...,M\}}$ and performance

measures of alternatives,$m_{ij|\{i=1,2,...,N\};\{j=1,2,...,M\}}$. The presence of the decision table information and a decision-making method, sets the task for the decision maker in finding the best alternative, and likewise ranking the whole set of alternatives. Note that before a consideration of all possible attributes in the decision problem, there must be a normalization of all the decision table elements to define them in terms of the same units [112, 111, 113, 114, 115].

Table 2.6: *Decision Table in MADM Methods*

Alternatives	Attributes					
	B_1 (w_1)	B_2 (w_2)	B_3 (w_3)	– (-)	– (-)	B_M (w_M)
A_1	m_{11}	m_{12}	m_{13}	-	-	m_{1M}
A_2	m_{21}	m_{22}	m_{23}	-	-	m_{2M}
A_3	m_{31}	m_{32}	m_{33}	-	-	m_{3M}
-	-	-	-	-	-	-
-	-	-	-	-	-	-
A_N	m_{N1}	m_{N2}	m_{N2}	-	-	m_{NM}

Several MADM techniques are available for attribute selection in making a decision. These includes but not limited to: 1) Maximin, 2) Maximax, 3) Dominance, 4) Conjunctive/ Satisficing method, 5) Lexicographic method, 6) Disjunctive method, 7) Lexicographic semi corder, 8) Linear Assignment Method (LAM), 9) Elimination by Aspects (EBA), 10) Simple Additive Weighting method (SAW), 11) Elimination et Choice Translating Reality (ELECTRE), 12) Technique for Order Preference by Similarity to Ideal Solution (TOPSIS), 13) Weighted product distance from target. Specifically, TOPSIS and SAWare the two techniques of MADM that are explained in this study.

2.5.1 Classification by Solution

This type of classification involves an initial categorization of the methods based on a certain property. The property to be considered is the type of information received

from the decision maker. This information could take certain forms; no information given, information on attributes are given, or information on alternatives are given. In the scenario where no information is given, the suitable MADM techniques to be adopted in this category are maximin, maximax and dominance. On the other hand, when considering a case where information is given, another factor is put into consideration. This factor represents the prominent characteristic of the received information from the decision maker. Thus, this characteristic is used in grouping the methods further.

The information signifies a standard level of each attribute, it may be the ordinal preference of attributes, or a cardinal preference of the attributes, it may also be of the marginal rate of substitution between the attributes. If the information given is a standard level of each attribute, then the conjunctive or disjunctive method may be adopted. Whereas Elimination by Aspects (EAB) and Lexicographic method would apply if it is an ordinal preference of attributes. In the case of a cardinal preference of the attributes, SAW, ELECTRE and TOPSIS methods apply, and the hierarchical tradeoffs method applies if the information is of the marginal rate of substitution between the attributes [114, 115].

Asides from making a classification of MADM methods based on the type of information received from the decision makers, and the prominent characteristic of the information (as in Figure 2.11), there are other possible classification schemes. According to Hwang, another approach for classification of MADM methods is according to the solution being targeted, as shown in Figure 2.12. To implement this classification follows thus, if the decision maker targets a solution to screen, then the conjunctive, disjunctive or dominance method is appropriate. In the case where the decision maker requires a solution targeted at evaluating, prioritizing

and selecting, some of the appropriate methods are maximin, ELECTRE, SAW, or TOPSIS. However, there are some situations wherein the decision maker aims at a solution to initially screen, then going further to evaluate, prioritize and select. In such a case, one of the methods suitable for screening can be adopted at the screening stage, and another suitable technique can be chosen to evaluate, prioritize, and select.

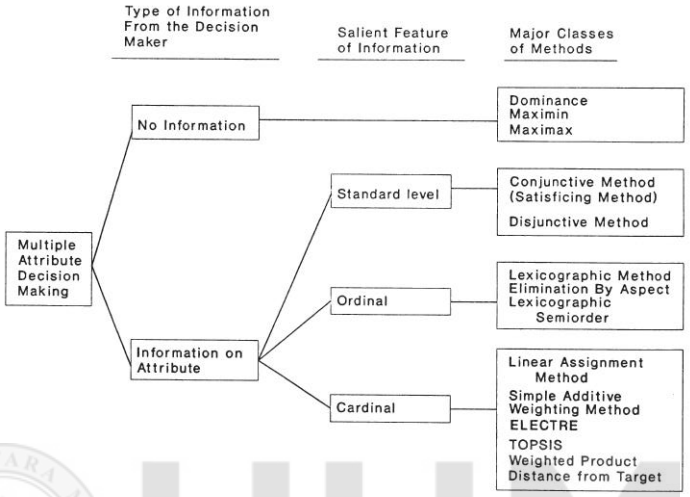


Figure 2.11: A Taxonomy of Multiple Attribute Decision Making Methods [114]

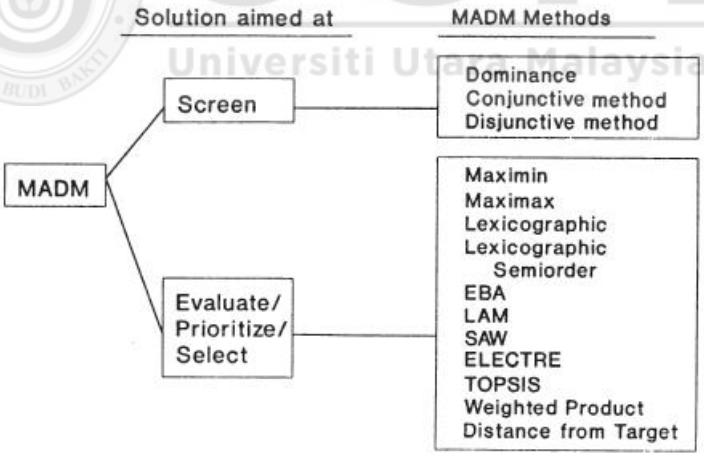


Figure 2.12: Multiple Attribute Decision Making Methods Classified [115]

2.5.2 Fuzzy Set Theory

A background knowledge of fuzzy set theory is introduced, since the system model introduced in this study is founded on a fuzzy approach for MADM. Fuzzy set theory is a very appropriate approach for modeling uncertain (imprecise) data [116], and is

the basis for MADM approach. MADM involves making a choice or selection among the action contents in the presence of multiple attributes, which are in most cases conflicting [117, 115]. In the proposed strategy developed in this study, the contents are fuzzy because of its similarity in attributes. Hence, the need for the adoption of fuzzy set theory with the specific selection of Triangular Fuzzy Numbers (TFNs) to be adopted.

Zadeh [94, 116] first introduced the concept of fuzzy set theory as a means of addressing the presence of vagueness in the human thought. In definition, a class of objects having continuum membership grades is referred to as a fuzzy set. For these said objects, a set is defined to assign each object with a membership grade over the interval [0;1], with the set being characterized by a membership function [118].

When adopting fuzzy set theory, we consider the use of linguistic variables, which refers to a category of variables having linguistic terms as their values. The use of the linguistic terms have been justified based on the intuitive ease of use when expressing subjective concepts and/or qualitative imprecision of the assessments of a decision maker [117, 115, 118].

The linguistic terms that represent the consequents of rules in this study is shown in Figure 2.13 and named “Attributes Level”. It is represented by fuzzy set categories, “Very Low (VL)”, “Low (L)”, “Medium Low (ML)”, “Medium (M)”, “Medium-High (MH)”, “High (H)” and “Very High (VH)”.

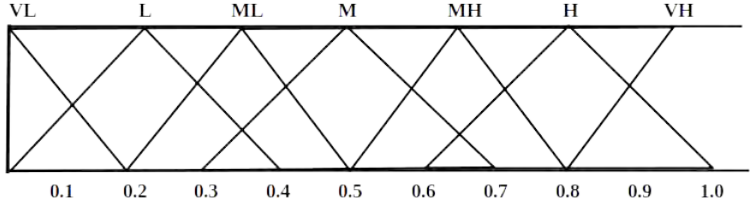


Figure 2.13: Scaling System for Linguistic Terms.

Adopting variants of fuzzy numbers (triangular, trapezoidal, hexagonal, etc) based on the situation or scenario being considered is mostly the most probable approach. However, studies such as [114, 115, 119, 118].have grounded that triangular fuzzy numbers (TFNs) are more convenient to work with in applications. This is as a result of the computational simplicity of TFNs, and also its usefulness in information processing and promoting representation, in a fuzzy environment. Thus, this study utilizes the adoption of TFNs in the developed fuzzy TOPSIS and fuzzy SAW methods.

Definition 1: A fuzzy set A on the universe of discourse X has a membership function $\mu_A : X \rightarrow [0,1]$ mapping every element of X to a value within the interval (0,1). This value, lying between 0 and 1, is called the degree of membership or membership value, and it quantifies the membership grade of the element on the universe of discourse X, to the fuzzy set $\tilde{A}$.

In our proposed strategy for content replacement, fuzzy triangular values as in Figure 2.14 are briefly defined for the purpose of providing a basic knowledge of the concept, before its adoption in the fuzzy MADM approach.

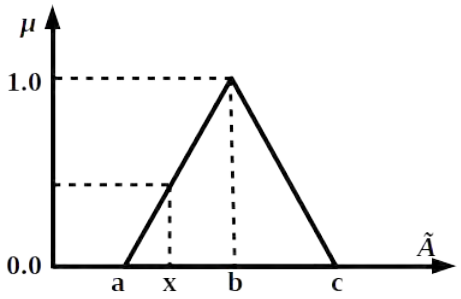


Figure 2.14: Fuzzy Triangular Number [7]

Definition 2: The membership function $\mu_A(x)$ is defined as

$$\mu_A(x) = \begin{cases} \frac{x-a}{b-a}, & x \in [a,b]; \\ \frac{c-x}{c-b}, & x \in [b,c]; \\ 0, & otherwise, \end{cases} \tag{2.1}$$

where a denotes the lower values of the fuzzy number support and c denotes the corresponding upper values, with $a \leq b \leq c$.

Definition 3: The fuzzy triangular numbers $\tilde{x}$ and $\tilde{y}$ follow the definition [114, 115, 118]:

$$\tilde{x} = (a^x, b^x, c^x) \tag{2.2}$$

$$\tilde{y} = (a^y, b^y, c^y) \tag{2.3}$$

where a denotes the lower values of the fuzzy number support and c denotes the corresponding upper values, with $a \leq b \leq c$.

The transformation for the fuzzy membership functions and its representation in terms of its importance of criteria and the rating of the alternative is detailed in Table 2.7.

Table 2.7: *Transformation for Fuzzy Membership Functions*

Linguistic Terms	Code	Membership functions
Very Low	VL	(0, x, 0.2)
Low	L	(0, x, 0.4)
Medium Low	ML	(0.2, x, 0.5)
Medium	M	(0.3, x, 0.7)
Medium High	HM	(0.5, x, 0.8)
High	H	(0.6, x, 1.0)
Very High	VH	(0.8, x, 1.0)

2.5.3 Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) Method

TOPSIS method as developed by Hwang and Yoon (1981) is a method based on the concept of distance. The rule of the method follows that, the chosen alternative should have the shortest Euclidean distance from the ideal solution, and the farthest distance from the negative ideal solution. In the database where all the satisfying solutions comprises, the ideal solution refers to the hypothetical solution where all attribute values correspond to the maximum attribute values. Whereas within the same database, the hypothetical solution where all attribute values correspond to the minimum attribute values is the negative ideal solution. In a nutshell, it can be generalized that TOPSIS chooses a solution that is farthest from the hypothetically worst but closest to the hypothetically best. A general overview of the main procedure for implementing TOPSIS method in selecting the best alternative is described in the following steps:

Step 1: Define the objective and identify the appropriate evaluation attributes,

Step 2: Develop a matrix based on all available information on attributes. The matrix being referred to in this step is the decision table defined in Table 2.6. An alternative is allocated to each row of this matrix, while individual attributes are allocated to each

column, Therefore, an element $m_{ij}$of the decision table defines the j^{th} attribute value in original real values (that is, non-normalized units and form), for thei^{th}alternative. Note that if the objective value is not available, which implies a subjective attribute, a scale of a ranked value judgement is adopted. Table 2.8, gives a type of scale that may be adopted for this purpose. As soon as a scale representation of the subjective attribute is given, then the researcher calculate the attribute's normalized values which is assigned for different alternatives using similar approach as that of the objective attributes. This leads to the third step.

Table 2.8: *Value of Selection Attribute*

Subjective measure of attribute		Assigned value
Exceptionally low		0.0
Extremely low		0.1
Very low		0.2
Low		0.3
Below average		0.4
Average		0.5
Above average		0.6
High		0.7
Very high		0.8
Extremely high		0.9
Exceptionally high		1.0

Step 3: Obtain the normalized decision matrix, R_{ij}. This decision matrix can be represented as

$$R_{ij} = m_{ij} / \left[\sum_{j=1}^{M} m_{ij}^2 \right]^{1/2} \tag{2.4}$$

Step 4: Assign weights/relative importance to each attributes with respect to the objective. A set of weightsw_j (j=1, 2, ….., M) such that $\sum w_j$ =1 may be decided

upon.

Step 5: Obtain the weighted normalized matrix V_{ij}. This weighted normalized matrix is obtained from multiplying each element of the matrix column R_{ij} with its associated weight w_j. Therefore, each element of the weighted normalized matrix V_{ij} are obtained by calculating:

$$V_{ij} = w_j R_{ij} \tag{2.5}$$

Step 6: Obtain the ideal and negative ideal solutions. Recall that the ideal solution is the best solution, and the negative ideal solution is the worst solution. Each solution are as defined below:

$$V^{+} = \left\{ \left(\sum_i^{\max} V_{ij} / j \in J \right), \left(\sum_i^{\min} V_{ij} / j \in J' \right) / i = 1, 2, ..., N \right\},$$

$$V^{+} = \left\{ V_1^{+}, V_2^{+}, V_3^{+}, ..., V_M^{+} \right\} \tag{2.6}$$

$$V^{-} = \left\{ \left(\sum_i^{\max} V_{ij} / j \in J \right), \left(\sum_i^{\min} V_{ij} / j \in J' \right) / i = 1, 2, ..., N \right\},$$

$$V^{-} = \left\{ V_1^{-}, V_2^{-}, V_3^{-}, ..., V_M^{-} \right\} \tag{2.7}$$

where J = (j = 1, 2, . . . , M) /j and J′ = (j = 1, 2, . . . , M) /j are associated with beneficial attributes and non-beneficial attributes respectively.

V_j^+ indicates the ideal value (i.e. best value) of the considered attribute from all values of the attribute for different alternatives. Note that in the case of beneficial attributes (which are those attributes of which higher values are required for the given application), V_j^+ specifies the higher value of the attribute. Whereas in the case of nonbeneficial attributes (referring to those attributes of which lower values are required for the given application), V_j^+ specifies the lower value of the attribute. Similarly, the negative ideal is indicated by V_j^-. This is otherwise known as the worst value of the considered attribute from all values of the attribute for different alternatives. For V_j^-, considering the case of beneficial attributes, V_j^- indicates the lower value of the attribute, while in the case of non-beneficial attributes, V_j^- indicates the higher value of the attribute.

Step 7: Determine the separation measures. The determination of the separation of each alternative from the ideal one is given by the Euclidean distance in Equations 3.5 and 3.6.

$$S_i^+ = \left\{ \sum_{J=1}^{M} \left(V_{ij} - V_j^+ \right)^2 \right\}^{0.5}, \quad i = 1, 2, 3, ..., N \qquad (2.8)$$

$$S_i^- = \left\{ \sum_{J=1}^{M} \left(V_{ij} - V_j^- \right)^2 \right\}^{0.5}, \quad i = 1, 2, 3, ..., N \qquad (2.9)$$

Step 8: Compute the relative closeness of each alternative to the ideal solution, P_i, This is expressed as follows.

$$P_i = S_i^- / \left(S_i^+ + S_i^-\right) \tag{2.10}$$

Step 9: Generate in descending order, a set of alternatives, with respect to the value of P_i. P_i indicates the feasible solutions that are most preferred and least preferred. It also could be referred to as the composite or overall performance score of alternative A_i.

2.5.4 Simple Additive Weighting Method (SAW)

The simple additive weighting method (SAW) is also known as the weighted sum method [120] This method remains the simplest, and widely used MADM method. Suitable case for adopting SAW is when the decision attributes can be expressed in identical units of measure, such as, only pounds, only dollars, only seconds, and so on. In SAWmethod, individual attributes are assigned a weight (where the total sum of all weights equals 1), and each alternative is evaluated in respect to each attribute. Equation 3.8 gives the overall performance score of an alternative.

$$P_i = \sum_{j=1}^{M} w_i m_{ij} \tag{2.11}$$

On availability of attribute objective values, normalized values are computed by

calculating $\left(\mathrm{m_{ij}}\right)\mathrm{K}/\left(\mathrm{m_{ij}}\right)\mathrm{L}$, where $\left(m_{ij}\right)K$ denotes the attribute measure for the K^{th} alternative, and $\left(m_{ij}\right)L$ represents the attribute measure for the L^{th} alternative, having the highest attribute measure in comparison to all alternatives considered. As aforementioned, the attributes can be beneficial or not (non-beneficial). Thus, this ratio is valid for beneficial attributes only (e.g., profit), which refers to an attribute whose higher measures are more required for the given decision-making problem. Distinctively, non-beneficial attributes, such as cost, refers to the situation where the lower measures are required, and the calculation of the normalized values is given by $\left(\mathrm{m_{ij}}\right)\mathrm{L}/\left(\mathrm{m_{ij}}\right)\mathrm{K}$.

Relaxing the constraint that $\sum w_j = 1$, then Equation 3.8 can be used. This results in the method known as the Simple Multiple Attribute Rating Technique (SMART).

$$\mathrm{P_i} = \left[\sum_{\mathrm{j=1}}^{\mathrm{M}} \mathrm{w_i m_{ij}}\right] / \sum_{\mathrm{j=1}}^{\mathrm{M}} \mathrm{w_j} \tag{2.12}$$

where X_{ij} denoted the result of the i^{th} alternative about the j^{th} attribute with a comparable numerical scale, and W_j represents the weight of the j^{th} attribute.

SAW method has certain requirements, notable advantages and disadvantages.

Requirement: Only comparable and numerical attributes can be used, and assigning of weights (relative importance) to attributes is done by the decision maker.

Advantages: SAW is the most broadly used and best known method. This MADM technique is simple, easy to use and straightforward to understand. In addition, the

tradeoff among attributes is compensatory.

Disadvantages: If attributes are complementary, that is, the presence of a high score on one attribute simultaneously occurs with a corresponding high score on another attribute, then a violation of the assumption of separable utility of each attribute is experienced in the computed score [120].

2.6 The Discussion of Replacement Strategies

This section contains two subsections. The first one elaborates on the analysis of ten replacement strategies by explaining the description and the disadvantages of these ten strategies. In addition, what time these strategies are born? The second subsection explains the available strategies and with which strategies are compared to get out with the best one.

2.6.1 Replacement Strategies Analysis

The replacement strategies that are based on priority is the fastest, but it loses content popularity. While the strategies that are based on popularity was kept the content popularity for a long time and it gives higher latency. Therefore the strategies which are based on both popularity and priority are the best in NDN. Therefore, the best replacement strategy must evict the less popular and low priority content. The popularity and priority schemes are explained in detail in subsection 2.2.2. As shown in the table above, this study also focuses on popularity, content consistency, and occupied cache. According to popularity, the replacement strategies that are depending on previous popularity are PPC, LFRU, CCP, ARC, and LFU. While PBCE and ABC replacement strategies are depending on current popularity. The strategies are not depending on popularity but depend on priority are PBCE, TLRU, ABC, IC, ARC, and LRU.

Table 2.9: *The Disadvantage of Replacement Strategy Decision*

Replacement Strategy	Brief Description	Disadvantage	Year
PPC	Use popularity-based on an approximation of LFU and a functions	Previous popularity, not priority, consistency content and occupied cache	2017
LFRU	Use popularity-based on an approximation of LFU and a partitioned LRU	Previous popularity, not priority, not consistency content and occupied cache	2017
PBCE	Use popularity-based on an approximation of LRU	Current popularity, priority, not consistency content and occupied cache	2017
TLRU	Use priority- based content lifetime factor to remove Data packet.	Not popularity, priority, not consistency content and not occupied cache	2014
ABC	Use popularity-based on age content	Current popularity, not priority, not consistency content and occupied cache	2014
CCP	Use popularity-based hit rate factor to remove Data packet.	Previous popularity, not priority, not consistency content and occupied cache	2013
IC	Use priority- based timestamp factor to remove Data packet.	Not popularity, priority, consistency content and not occupied cache	2013
ARC	Attempt to combine both spatial and temporal locality together maintaining their characteristics of the previous two classes.	Previous popularity, priority, not consistency content and not occupied cache	2003
LRU	Use recency factor as the primary factor to remove web object	Not popularity, priority, consistency content and not occupied cache	1993
LFU	Use object popularity (or frequency count) as the primary factor to remove web object	Previous popularity, not priority, not consistency content and occupied cache	1997

Moreover, the strategies which are depending on consistency content are PPC, IC, and LRU. While LFRU, PBCE, TLRU, ABC, CCP, ARC, and LFU strategies aren't depending on consistency content. In addition, The PPC, LFRU, PBCE, ABC, CCP, ARC, and LFU replacement strategies provide occupied cache. This leads to conclude that IC, TLRU, and LRU strategies are the best because don't have occupied cache but they are poor to the popularity.

2.6.2 Selection of Replacement Strategies Evaluation

The strategies evaluation (in descending order of performance) and those that performed best are presented. Table 2.10 shows a list of studies in which the performance of cache replacement strategies was assessed and presents references to cache replacement strategies in the literature.

Table 2.10: *References to Cache Replacement Strategies in Literature*

Available Replacement Strategies	**Strategies Studied**	**Champion Strategies**
Popularity prediction caching in ICN	PPC, CCP, LRU, LFU, FIFO	PPC, CCP, LFU,LRU
Least Frequent Recently Used [95]	LFRU, LRU	LFRU
Popularity Based Content Eviction [50]	PBCE, LRU	PBCE
Time Aware Least Recently Used [96]	TLRU, LRU	TLRU
Age-based Cooperative Caching in Information-Centric Networks [67]	ABC, LRU, FIFO	ABC, LRU
Cache replacement policy based on content popularity [91]	CCP, LRU, LFU	CCP, LRU
Information-maximizing Caching [94]	IC,LRU	IC
ARC [121, 102, 101, 103]	ARC,LRU	ARC
LRU [87, 88, 89, 99, 98, 104], LRU-Threshold [122], LOG2-SIZE [123], and HLRU[124]	LFU,FIFO	LFU
LFU [98, 99, 90, 89, 92], LFU-Aging [125], and LFU-DA [125]	LRU, FIFO	LRU

There are scores of cache replacement strategies. It would be unrealistic and counterproductive to assess the performance of all existing strategies. For this reason, it makes sense to reduce the field of replacement strategies. Two criteria are used in the study to select the strategies:

1- The strategy is considered a "champion" strategy. That is, it has been demonstrated that it outperforms other policies in previous studies.

2- The strategy has been widely studied or is popular.

A quick reorganization of the information in Table (References to cache replacement policies in the literature) is shown in Table 2.11. This table contains a list of those strategies assessed for their performance in the studies with the number of references of each one and the frequency with which they ended up as best performers.

Table 2.11: *Number of References and Champion Strategies in Literature*

strategy	Number of references	Frequency
PPC	1	0
LFRU	1	0
PBCE	1	0
TLRU	1	0
ABC	1	0
CCP	2	1
IC	1	0
ARC	1	0
LFU	3	2
LRU	9	4
FIFO	4	0

The selection of the strategies to be evaluation is based on the information contained in Table 2.11. For strategies that require the use of parameters, the parameters will be chosen following the results from past comparisons existing in the literature. The parameters that result in the best performance will be selected. According to these criteria, the strategies selected for evaluation are:

1- Cache Content Popularity (CCP).

2- Least Recently Used (LRU).

3- Least Frequently Used (LFU).

4- First-In-First-Out strategy(FIFO)

2.7 Summary

This chapter has discussed previous studies on the architecture of NDN, and in-network Caching, Cache Replacement Strategies, and MADM. This chapter can explain:

1- The role played by a node of the NDN, the ways in which caching can improve performance by reducing latency and internal traffic, and the special problems presented by caching systems in the network, such as variable object sizes, temporal locality, and long-term access frequency.

2- The metrics commonly used to gauge the performance of cache replacement, especially the cache hit ratio, server hit reduction ratio, Aggregated network delay, average server load, average throughput, and byte hit rate. The tradeoff that exists between these two metrics, and the relevance of the tradeoff for this research.

3- It defines the information freshness in NDN.

4- The role of MADM by defining the attribute of the replacement strategies.

5- The analysis of replacement strategies was cleared in detail.

6- A description of existing cache replacement policies and the factors taken into consideration by those strategies to work. Also, it's a possible role in balancing cache hit ratio, server hit reduction ratio, Aggregated network delay, average server load, average throughput, and byte hit rate.

CHAPTER THREE
RESEARCH METHODOLOGY

This chapter discusses the methodology adopted in carrying out the research. The chapter presents the approaches used in executing the research and thus demonstrates how these approaches achieve the objectives of the research. In presenting the discussions of this chapter, several subsections have been outlined. The outline of the chapter starts by discussing the reasons for why needing an alternative Caching Strategy (ACS)in Section 3.1. Then, highlighting the research steps and research methodology options available in Section 3.2, with an inclusion of the justification for the selected approach in this research. In Section 3.3, the conceptual model of the alternative caching strategy is explained, while Section 3.4 discusses Multiple Attribute Decision Making (MADM) methods adopted in the evaluation of communication network performance mentioned in Section 3.5. The last section of this chapter (Section 4.5) presents a summary of the research methodology.

3.1 The Need for an Alternative Caching Strategy

Due to its limited cache, a short-run cache attains its maximum capacity after a certain time frame, and upon the entrance or arrival of new content, a challenge is encountered, as the cache has already reached its full capacity. This implies that the content in the router is more popular than the incoming content or it can said that the expiration content has high popularity. Hence, the need for an alternative caching strategy which must have the following characteristics:

• Reduction of miss rate or increment in hit rate. That is, the cache replacement mechanism should prioritise the preservation of the storage of the most popular items available in the cache.

• Keeping of more up-to-date items. This means that the cache update mechanism must have knowledge of content availability with respect to its expiration time.

• To retain more documents which must travel on congested paths, in cases where the information of heavy congestion of certain network paths is known.

• The computational complexity of the alternative strategy must be as simple as possible.

Therefore, an alternative caching strategy depends on two mechanisms which are the replacement mechanism and the update mechanism. For the replacement mechanism,less popular and low priority content is replaced first, while the update mechanism depends on popular and priority content.

The Popularity Concept: The issues related to content popularity have been addressed through the maintenance of a data structure displaying content names and the corresponding available popularities of these content names at the router's cache. This popularity depends on the number of hits for each item in the content router and the number of a miss for each item that is not in the content router. In other words, the popularity depends on cache hit and cache miss according to the content router.

The Priority Concept: This concept pre-fetches popular content during idle time (data readiness for later use). For example, different episodes of one TV series usually share quite similar popularity change patterns. Therefore, if one TV series is popular in the local network, the local cache server can pre-fetch the new episode of the TV series as soon as it is released or during idle time. Thus, latency and bandwidth can be saved when this TV episode is requested during a busy time. Therefore, priority

reduces latency and bandwidth.

From the rigorous literature review and analysis seen in previous chapters, the cardinality or the magnitude of the attributes were chosen to be eight. The eight selected attributes serve as the level of measurement for the alternative caching strategy. The main input attribute considered in the content selection are broadly classified into userrelated attributes and content-related attributes, as described below.

User-benefit attributes are the parameters which include cache hit and cache miss. When referring to a cache hit, this occurs in cases when the data requested by the consumer can be found in a cache, and is used for current hit rate (current popularity content). Whereas, a cache miss occurs when the data requested by the consumer cannot be found in a cache, and is used for previous miss rate (previous popularity for content. On the other hand, content store-cost attributes are the attributes related to the data content resources. They include cost time (the amount of time needed to fetch a content), freshness period of content (the period for content to stay in the content router), and content size (the size for each content which is saved in the content router).

3.2 Research Operational Framework

To fulfill the aim of this research, the following research steps were carried out as are illustrated in Figure 3.1.

Step1 : Explain of the study and extensive analysis of literature leading to the formulation of research problem. This phase involves performing a critical review of empirical studies, to identify the strengths and weaknesses of the replacement caching strategies in NDN that leads to

developing the conceptual model.

Step2 : Develop the proposed conceptual model describing the expected requirements to achieve an improved situation for the alternative cache strategy and actualize the research objectives. The conceptual model generally consists of suppositions which present an overview to reduce the complexity of the proposed model.

In the first step, the study focuses on issues pertaining to the research subject. These issues include, the basis of the research and its motivation, research questions, objectives, problems and important areas to be addressed, and areas of contribution. In response to these issues, this study gives adequate background information and sets the research context that are studied thoroughly and analytically. In addition, extensive analysis and evaluation of the strengths and weaknesses of the previous replacement caching strategies are presented. The output of this step is a set of potential enhancements to the original replacement caching strategies. These enhancements are required in order to mitigate the congestion and produce cache efficiency. A step further is also taken in identifying the research challenges in NDN caching that should be taken into consideration while analysing caching strategies.

The second step is to develop a conceptual model of alternative caching strategy to describe the expected requirements to achieve an improved situation. In science, conceptual models are easy and simple ways to portray or explaining certain concepts, thus reducing complex ideas and presenting them in a manner that is easy to understand. Conceptual models tend to be simplified because that is what makes them useful. Although, its simplicity is accompanied by the shortcoming of not always being perfectly accurate.

Furthermore, based on the above steps, this research will continue to find out the objectives by flow the phases below. Each phase (phase one, phase two, and phase three) will represent the objective sequentially (Objective one, objective two, and objective three) in order to achieve the alternative caching strategy for NDN.

Phase One:

Step1 : Design a Content Replacement Mechanism (CRM) with the use of on-path caching to improve cache hit ratio, bandwidth, and low latency of the caching. This new design that remains the popular contents for mitigating congestion and growing cache efficiency by including these processes:

- The content with minimum popularity will be replaced with new content in CS and delete its record from Content Table, then save the new content record.
- The content popularity ranking will be depended on time or hit or size.

Step2 : Use Fuzzy Multiple Attribute Decision (MADM), Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) Method.

Phase Two:

Step1 : Design a Content Update Mechanism (CUM) by using information freshness and on-path caching to increase throughput and reduce server load. The design process includes:

- The content with content ranking will be replaced with new content in CS and delete it's record from Content Table, then save the new content record.

- The content ranking will be depended on stale time or similarity.

Step2 : Use Fuzzy Multiple Attribute Decision Making (MADM), Simple Additive Weighting (SAW) Method.

Phase Three:

Step1 : Implementing, validating, and verifying the design of the alternative caching strategy which has two mechanisms; CRM and CUM. The strategy will be tested using Mini-NDN and ndnSIM. The alternative caching strategy will be validated and verified by executing experiment with various multimedia applications sequence activities. This additionally includes the discussion on how the experiments are designed, and results are verified and validated.

Step2 : Evaluate the performance of the proposed strategy using cache management schemes and compare with the existing strategies in simulated network environment. The evaluate process includes:

- Evaluates the performance of an alternative caching strategy and its comparison with (FIFO, LFU, LRU, and CCP) through simulations.
- The new strategy performance is analysed in terms of throughput, reduce server load, cache hit ratio, bandwidth and low latency metrics and the details of the experiment setup are presented. The output from this step is a set of data for performing evaluation.

The MADM methods to be included are those which can be understood easily,

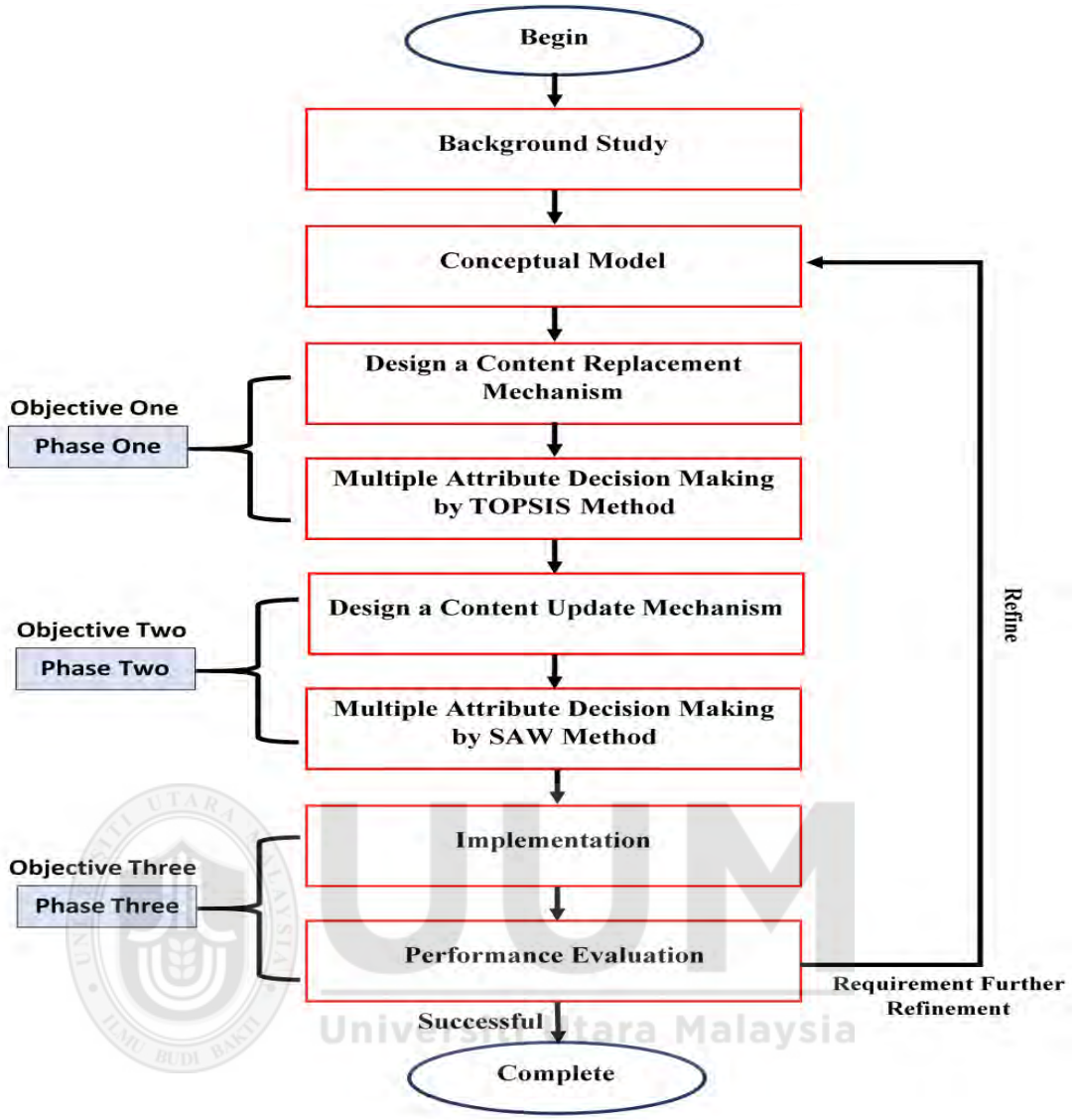


Figure 3.1: Research Steps

and applied easily to large size problems existing in the real world. Note that the presentation will be limited to only the essential ideas. The adopted methods are Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) and Simple Additive Weighting method (SAW). Subsequently in chapter Two, the characteristics unique to each these methods are described in terms of certain properties which include: its basic principle, the logic of the method, step-by-step procedure, advantages and disadvantages, requirements and reference.

All the steps highlighted in the research framework are considered in various chapters

of this study. The background of study is discussed elaborately in Chapter One and Chapter Two. The details of the conceptual model is given in this chapter (Chapter Three). The alternative caching strategies mentioned will be discussed in terms of design and thoroughly elaborated upon in Chapters Four and Five respectively. The implementation, validation and verification of the alternative caching strategy will also be addressed in Chapters Four , by respect to the two mechanisms being considered (CRM and CUM) with using (TOPSIS and SAW). The discussion of this research will be included the metrics used in performance evaluation of the new strategy and the details of the experiment setup which considers the evaluation results by comparing to other caching strategies will be discussed in Chapter Five.3.5

3.3 The Proposed Conceptual Model for the Alternative Cache Strategy

In this section, the overall description of the research concepts will be identified. The conceptual model improves our understanding of the subject being modelled. This is because the conceptual model produces a concise description of the alternative caching strategy behavior. Specifically, it produces the cache's priority, and popularity. The popularity is the number of same data packets accessed to the node, while the priority is the number of the same content name updated in content store. The popularity and priority, together with eviction directly yield the content store performance, as the cache's hit rate shows the ability of the caching strategy to manage the amount of redundant content copies in terms of searching and how often is the data found. Additionally, priority, popularity and eviction give a rich picture of the content store behavior that can be used.

The Alternative Caching Strategy has two mechanisms; Content Replacement Mechanism (CRM) which is explained in in Section 3.2.1, and Content Update Mechanism (CUM), discussed in Section 3.2.2. Note that ACS needs the information

of one table which is Popularity and Priority Table (PPT). These table is discussed subsequently.

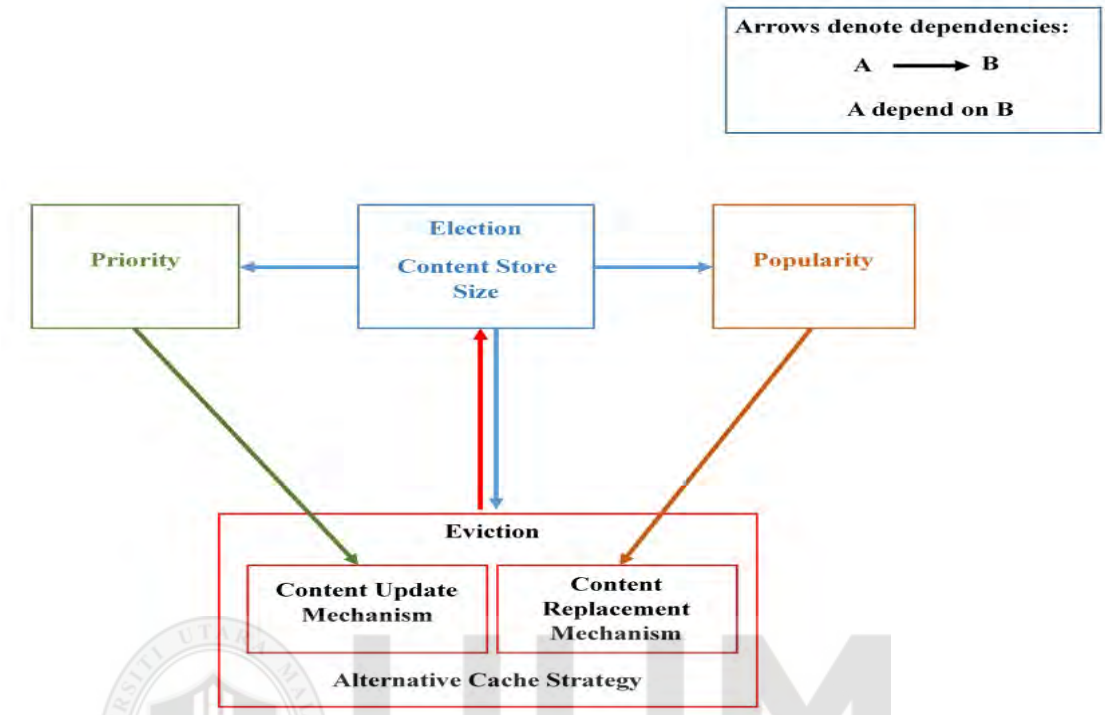


Figure 3.2: Conceptual Model for the Alternative Cache Strategy

In an alternative caching strategy, each NDN router maintains a table called Popularity and Priority Table (PPT) as show in 3.1 containing information concerning the popularity and priority of a content. This table contains the Content Name (*CN*), a counter called Content Miss Popularity (*CMP*), Request Time (R_T), Arrival Time (A_T), Cost Time (C_T), and a counter called Content Hit Popularity (*CHP*). When a request for a new content arrives, the information of request will store in the PPT. This PPT table saves *CN*, *CMP* and R_T. When data content is received, the PPT stores A_T, and C_T by using this equation 3.1, The *CHP* is saved in this table after one cycle for router.

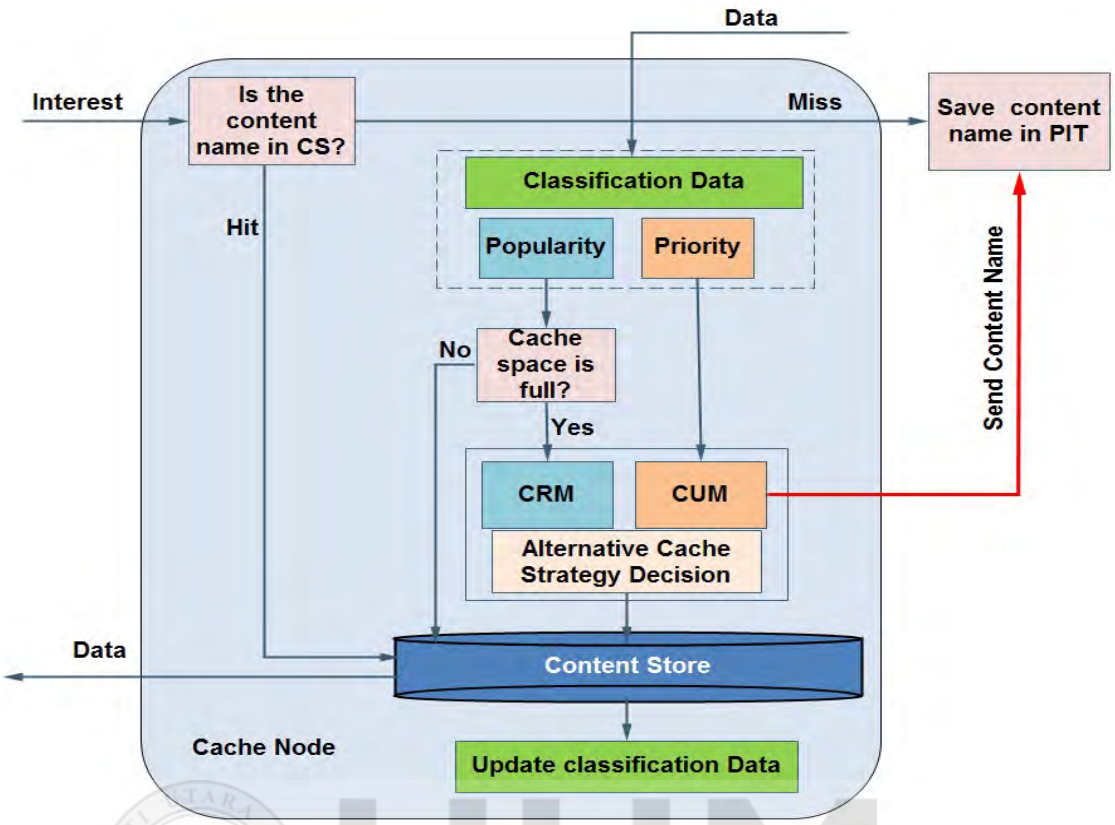


Figure 3.3: Conceptual Model for a Single Cache Node in Named Data Networking

$$Cost\,Time\,(C_T) \quad = \quad Arrival\,Time\,(A_T) - Request\,Time\,(R_T) \qquad (3.1)$$

And other attributes its taken from content router about each content as Size of Content (S_C), Freshness Period of content (F_C) , and Remaining Time (RE_T) It refers to the remaining actual time to expire a content.

$$Remaining\,Time(RE_T) \quad = \quad Freshness\,Period\,of\,content \quad - \quad (Current\,Time \quad - \quad Arrival\,Time)$$

When the interest packet arriving via a router NDN initiates a preliminary examination of the content store (CS) for data matching, and the data content is not found in the

Table 3.1: *The Popularity and Priority Table for Data Contents*

Content Name	Content Miss Popularity	Request Time	Arrival Time	Cost Time	Content Hit Popularity	Size of Content	Freshness Period of content	the remaining time
/youtube/media	5	10:20:40	10:21:00	00:00:60	10	5	1:20:00	0:30:00
/uum/news	6	09:40:00	09:40:50	00:00:50	12	2	0:05:50	0:0:40

CS, the attributes of data content will be registered in PPT. The alternative caching strategy will store in PPT the CN, CMP, and RT for the same content. CMP field is a counter for in case of a request for more than once, this content will be considered previous popularity.

When a CS receives a data content which hasn't been stored, it needs to check period time for the contents in the content router. If it isn't expired, the alternative caching strategy will use the content replacement mechanism. Otherwise, the alternative caching strategy will use the content update mechanism.

The alternative caching strategy needs to calculate the difference between the threshold value of the cache size and the number of cached contents to get the remaining size of cache space. If the rest of the space is less than a data packet, the replacement will happen. Otherwise, save data content in CS.

3.3.1 Conceptual Model for the Content Replacement Mechanism (CRM).

This is a mechanism that is based on popularity. As aforementioned, CRM require PPT . Whenever the router receives a new data packet that its name is not in PPT, this mechanism will perform the following:

1) The router calculates the difference between the threshold value of the cache size and the number of cached contents to obtain the remaining size of the cache space,

2) If a given packet is larger than the rest of the space, then this mechanism will accomplish the following:

i. Find which content name has lowest popularity in PPT,

ii. Evict the election content which has the lowest popularity from content store,

iii. Delete the record of the lowest popularity from PPT,

iv. Go to (a)

Otherwise:

a) Save the new data packet in content store,

b) Save the information of the new data packet attributes in PPT

The general concept is as depicted in Figure 3.4.

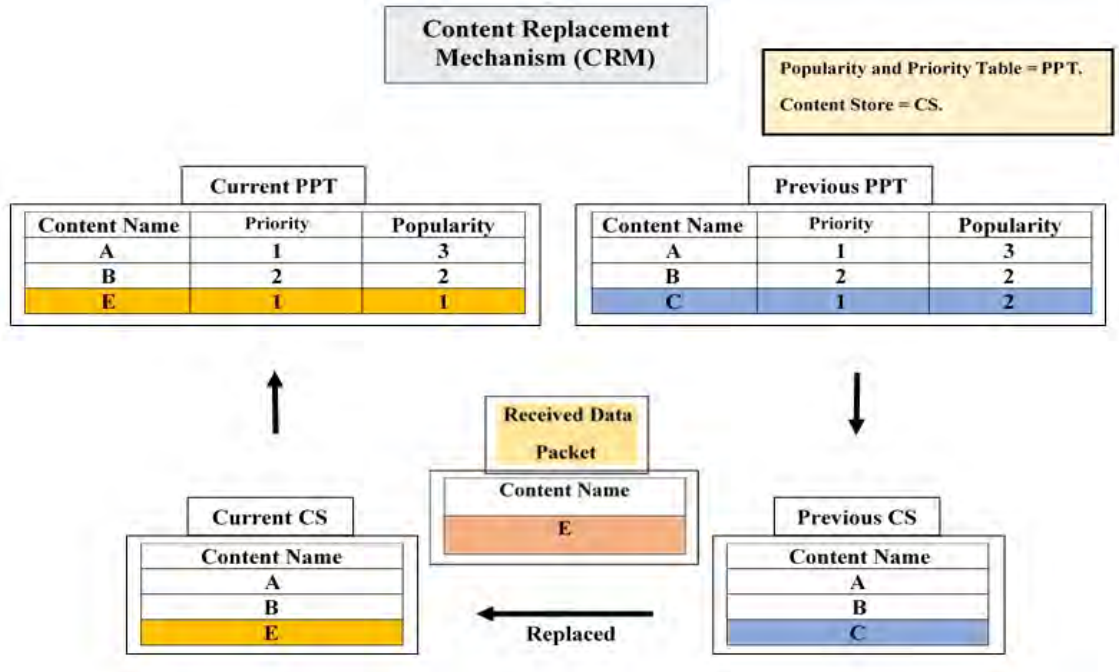


Figure 3.4: Content Replacement Mechanism

3.3.2 Conceptual Model for the Content Update Mechanism (CUM)

CUM is a mechanism using the concept of priority, that makes the router have the recent data. This mechanism needs to send content name to PIT depend on PPT. When a data content expired date for a data packet which has been stored, it must be deleted and update. The concept is depicted in Figure 3.5

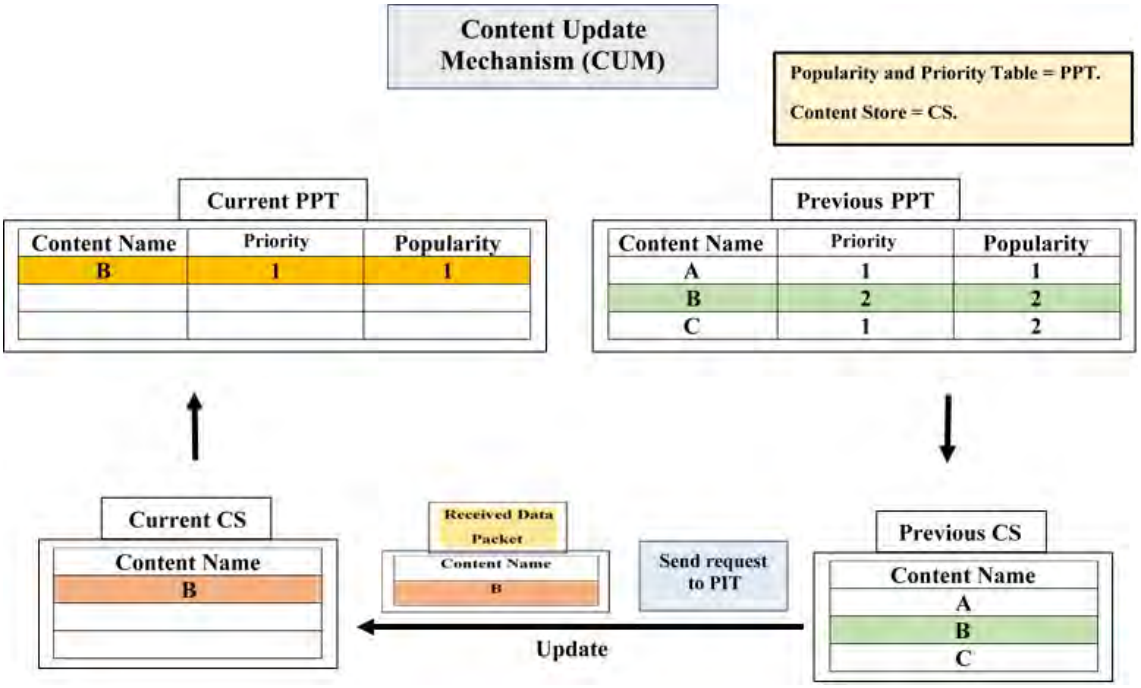


Figure 3.5: Content Update Mechanism

3.4 Multiple Attribute Decision Making (MADM)

Recall that the Alternative Caching Strategy has two mechanisms, which are Content Replacement Mechanism (CRM) and Content Update Mechanism (CUM). According to [114, 115, 120, 119], TOPSIS technique has several high value attributes considered as high ranked alternative, and the first mechanism, named CRM, is based on TOPSIS method. Thus, when the memory is full then one of the cached contents is replaced with a new arrival content which will be nearer to the users. Considering SAW method on the other hand, the total values of an alternative is calculated and the one with the highest attribute value is selected. According to this definition, SAW is the suitable method for the second mechanism, known as CUM. This is because after the arrival of the new contents, that is the update, this new content is saved which will be nearer to the users. Hence, SAW is the most appropriate technique to be followed.

In these proposed mechanisms, the contents are selected based on their popularity

values and prioritisation values. The adopted TOPSIS and SAW methods are presented in Chapter Four with more detailed explanation.

3.5 Verification and Validation

Verification and validation are defined as a process of verifying the accuracy and internal consistency of data, and confirming that it embodies the real-world entities suitable for its expected goal or intended number of objectives [126]. According to the modelling and simulation community [127], validation is the procedure to find out the level to which a prototype, simulation, or combination of prototype and simulation and their allied data precisely represent the real world from the perspective of its expected use(s). Verification on the other hand is the procedure to find out that a prototype, simulation, or combination of prototype and simulation and their allied data precisely represent the developer's conceptual model and its explanation. Thus, it is a step for ensuring the accurate assumptions of the model implementations.

This includes evaluation of the simulation program to guarantee that the conceptual model is a true reflection of the operational model [128]. Figure 3.6 shows the main steps for validation and verification of this research. Bearing in mind that the network model will be created in the two simulators: Mini-NDN and ndnSim.

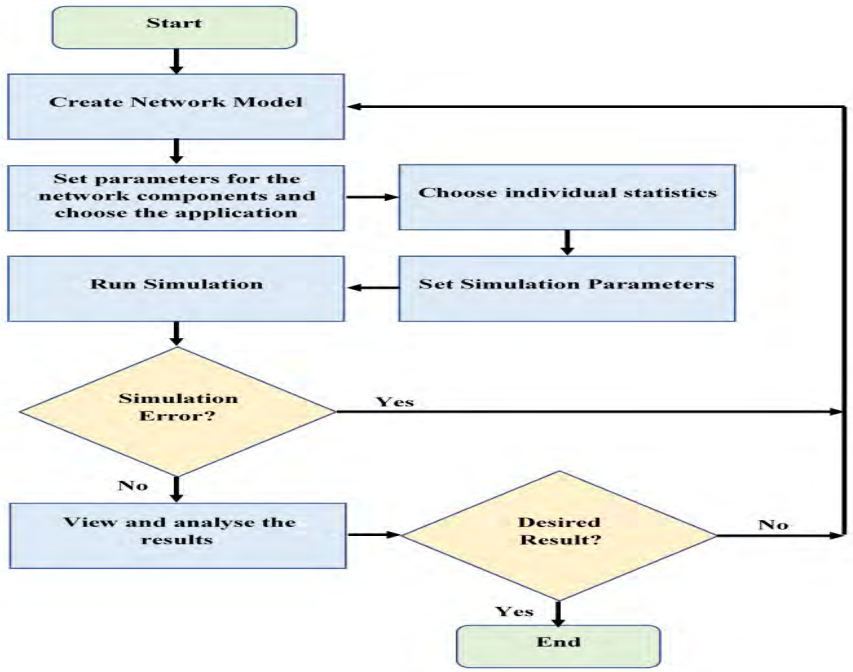


Figure 3.6: Main Steps for Validation and Verification [129]

Verification is the process done to ensure that:

i. The model is programmed appropriately

ii. The procedures have been applied correctly

iii. The model conversion from conceptualization to the program design is free from errors, oversights, or bugs.

iv. A high level statistical certainty is realized by testing more cases.

Model verification is a determination that the model is transformed with sufficient consistent accuracy from one phase to the next phase [130]. Furthermore, model verification evaluates the accurate transformation from pseudo code to actual

implementation, that is, the executable program code. In this research, design mechanisms were transformed to C++ code, since NS-3 simulator requires C++ as its programming language, and all mechanisms were verified to ensure the code is bug free. Additionally, the C++ code in NS-3 is capable of assisting the researcher highlight and indicate possible syntax errors during the compilation of the code.

They are validated by comparing with the existing standards and further accredited by evaluating with the contemporary mechanisms. This evaluation results not only evaluates the performance progress but also consequently validates the proposed mechanism.

The concept of validation is the process of ensuring a simulation model and its related facts are an exact illustration of the actual world, from the perception of the designer, in order to obtain the desired results with the employed mechanisms [131].

There are different ways that the validation and accreditation of a developed conceptual model can be done. Table 3.2 below briefly illustrates the various ways a model can be validated.

In line with Table 3.2 this research follows two methods of validation which are

• Validation by Evaluation with/to other Models, and

• Validation using Constraint Changeability – Sensitivity Analysis.

Before implementing the validation phase, the developed mechanisms implemented in caching are firstly designed using the analytical model based on fuzzy MADM theory.

Table 3.2: *Model Validation Approaches*

No	Validation Method	Description
1	Validation by Evaluation with/to other Models	This approach endorses simulation results through a comparison with the cognate results of the valid model.
2	Validation by Appearance	The accuracy or correctness of the model is validated from taking expert opinion.
3	Validation from Previous Data	This approach uses historical data to build and test the model
4	Validation using Constraint Changeability – Sensitivity Analysis	This approach involves varying the model's input parameters in order to determine the effect of the parameter variations on the model's output with respect to the real system. However, sensitive parameters should be made adequately accurate prior to their substitution into the model.
5	Validation using Prediction	In this approach, the model forecasts the system's behavior. Afterwards, the exact system behavior is compared to the model's forecast, to determine if the forecast is accurate to the exact behavior. The data adopted in the system may be extracted from an operational system or obtained from experiments conducted on the system, such as field tests

Then the analytical formulation of equations is transformed into the programmable code.

The first level of validation is implicitly done by the theory adopted by the consistency range acceptable for the mechanism implementation. The final result of the proposed mechanism is compared with the parallel contemporary mechanisms and evaluated for validating the proposed model. The results obtained for many cases to rigorously examine the validation of the proposed mechanism 3.6. The detailed numerical analysis of the validation, evaluation and final accreditation of the results are presented in Chapter Four and Five for respective mechanisms. To check for the sensitivity of the constraint changeability, the simulation is run randomly for numerous decision points.

3.6 Methods used in Evaluation of Communication Network Performance

The essential issue in network performance evaluation is the choice of suitable measurement approach, representative measurement settings, and a correct

implementation technique, as mentioned in[132]. It has been established in exiting literature that performance evaluation of a communication network can be conducted by three different approaches, which are, experimentation, mathematical/analytical modelling, and simulation [133, 134, 135, 136]. Figure 3.7 demonstrates these three approaches for evaluating the performance of communication network.

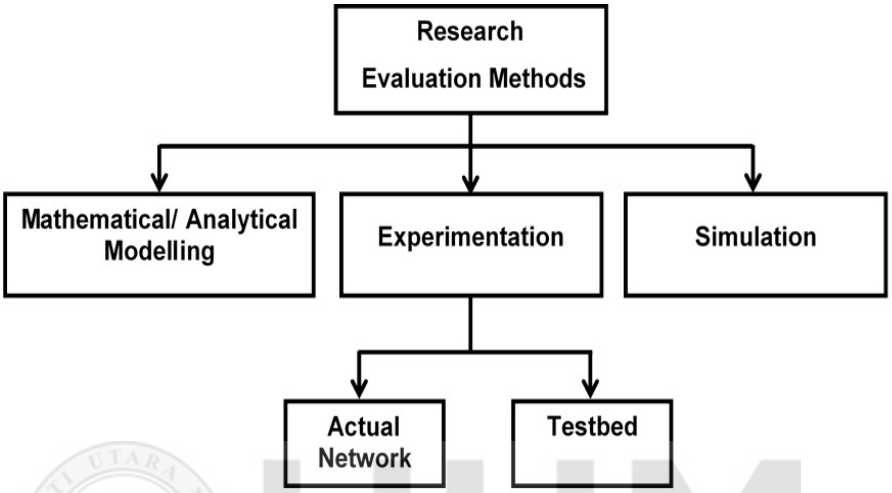


Figure 3.7: Research Evaluation Methods [136, 137]

Firstly, the mathematical/analytical modelling method involves a set of equations that represent the procedure of a communication network system under study, using mathematical symbols. The method includes designing, determining and validating the analytical model of the system to investigate and explain the problem [112, 134, 137]. Analytical models are adapted to use when the observed facts are measurable. Which can be further investigated using computer programming. Also, translates the equations to an executable code and further can be transferred into the graph. Users have a liberty to vary the conditions in input parameters of the code to obtain desired results. According to Kadhum [135], modeling has several advantage such as low cost, less time required and easy in trade-off evaluation. This may lead to an improved declaration of the system before the real implementation starts.

The second method for evaluating network performance is experimentation which computes prototype of the communication system [138]. Using this approach, prototypes of a communication network system can be designed and tested in a particular network environment. Measurement of network traffic provides a way of understanding that it is either working properly or not, in a local-area or wide-area network [139].

The third method for evaluating network performance is a simulation, which is an abstraction or estimation of the real world [127]. Simulation is an experimentation of an object that includes devices, tools, systems, or subsystems in a realistic form [140]. In this way, the researchers are allowed to carry out analyses on the procedures of the targeted network model without disturbing the actual network. In addition, simulation is an exercise of developing and analyzing a scheme of theoretical communication network system using network simulation software [141].

3.6.1 Evaluation Approach Consideration

Evaluation techniques are quite vital as they aid the measurement of system performance. However, since there are various evaluation techniques, the process of selecting the most adequate technique and tool is also of paramount importance [142]. Table 3.3 outlines the strength and weaknesses of performance evaluation techniques as stated in [143].

Measurement or Experimentation, Analytical Modeling and Simulation, are the three techniques adopted for performance evaluations, with two of these techniques (analytical modeling and simulation) being categorised as performance modeling approaches. For each of these techniques, there are certain criteria that need to be investigated, in order to help the researcher make an informed decision of the

Table 3.3: *Comparison of Techniques for Performance Evaluation*

Criteria	Performance Modeling		Performance experimentation/Measurement
	Analytical Modeling	Simulation	
Time Requirement	Small	Medium	Varies
Rate of Accuracy	Low	Moderate	Varies
Tool	Formal Method	Computer Programs	Instrumentation
Evaluation of Trade-off	Easy	Moderate	Difficult
Cost Implication	Small	Moderate	High

most suitable technique to be adopted. These criteria includes the time requirement, rate of accuracy, tool utilized, evaluation of trade-off, and cost implication (see Table 3.3). In the table, the list is arranged such that the most important criteria is at the top and then trails off to the least important criteria at the end of the table. As aforementioned, the techniques are broadly classified into two streams, that is, performance modeling (which includes analytical modeling and simulation) and performance measurement/experimentation.

Although in performance measurement, real hardware and software settings are utilized, it still may not produce exact or accurate results. Likewise in its procedure, some parameters used, which may include the configuration of the system, measurement time, and workload type, may be specifically defined for the experiment, however, they may fail to give an actual representation of variables' range available on the internet. Therefore, the accuracy of performance measurement technique varies, which implies that high accuracy is not guaranteed as the technique could also be of low accuracy. Realistically, performing experiments in a real network is very interesting and more representative than other approaches, it clearly has its own constraints. The following lists the major shortcomings to the employment of this measurement technique, specifically for communication network system performance

evaluation, as discussed in [128, 132, 137, 138]:

• Test bed development and maintenance requires a lot of resources, money, and time. For this reason, very few researchers, usually having financial backing by big telecommunication companies, can afford this evaluation technique.

• Scaling of actual network in a test bed is not possible. This is due to the fact that setting up the multi-hop topology requires a large physical area, in addition to the scaling involving hundreds of nodes.

• Difficulty in the reconfiguration, replication and sharing of test bed experimentation using a prototype due to its inflexibility.

• Initial testing of newly developed network algorithms or protocols in the actual network has high cost implications, in addition to dire consequences, with the ability to disrupt the entire network.

Performance measurement is done when the system of research interest is available as an actual system with real-time parameters. The measurement can be in the form of the test bed, on-chip performance monitoring, software monitoring, off-chip hardware monitoring, among other forms. Performance modeling, as an evaluation technique, is done either by simulating the actual system or adopting an analytical model. Analytical modeling is the mathematical formulation of the theory or mechanism which are based on operational laws, queuing networks, process algebra, stochastic networks, etc.[142, 144]. On considering performance evaluation using analytical modeling, it is observed that in terms of accuracy, this technique has the lowest accuracy. Whereas when adopting performance modeling technique using simulation,

it is observed that the approach can incorporate a more detailed system description, in terms of various scenarios, than an analytical model. Moreover, less number of assumptions are required when using simulation in comparison to analytical modeling. In addition, remodelling of the simulation model can be initiated multiple times until the most realistic implementation is achieved. Thus making the simulation scenarios closer to reality.

Therefore, after examining closely the different evaluation technique approaches, this research chose the performance modeling technique to accomplish the research objectives stated in Chapter One. Performance modeling is the normalized technique, on taking into consideration all the important criteria, from time to cost for the researcher [145].

3.6.2 Analytical Modeling

Analytical modeling is a description of the system in terms of mathematical concepts, which includes the logical model based on some theory to agree with the results of repeatable experiments. An analytical model defines a set of equations with mathematical notations to represent an actual system [146], that is, it is the abstraction of the actual model formulated using mathematical symbolism. Analytical models are adapted to use when the observed facts are measurable and can be further investigated using computer programming. The approach translates the equations to executable code whose results can be presented graphically. Users have the liberty to vary the conditions in the input parameters of the code in order to obtain desired results. According to Jain [143], there are quite of number of advantages attributed to modeling, and these include lesser cost implications, shorter time requirement and easier evaluation of trade-off.

Therefore, this research adopts analytical modeling as a formal method of verification and validation of the mechanisms. A step is taken further with the actual replica of the model being done through simulation for performance evaluation. The verification of the analytical model is initiated in both mechanisms and transformed into the computer programming codes.

3.6.3 Network Simulators

The most popular network simulator available is NS3 [147]. It is a discrete-event network simulator implementing simulation whilst modelling in C++. In its function as a library, NS3 is linked dynamically or statically to a main program in C++. This defines the topology of the simulation in details and initiates the simulator. Furthermore, simulation using NS3 is adopted with the aim of achieving high extensibility and scalability in the evaluation of the NDN cache, which has close similarities to multimedia services [148, 149].In addition, the packet used for NS3 is an open source, implementing the NDN protocol stack for Network Simulation Version3 (NS3) with location address, http://www.nsnam.org/ns-3-22/.

From definition, the concept of simulation follows the concept of imitating reality or process, by providing an understanding of scenarios through imagination and creativity, using simulator software. In terms of estimating performance in reality of a real network, simulations are used for observing the network's behavior under various scenarios, in order to identify the preferable of them. Simulation modelling is frequently used in NDN networks. However, due to their complexity in nature, NDN techniques are modelled in a real world with respect to a number of assumptions, with the most preferred performing the "what if" technique. For this reason, the most suitable means to test or evaluate its performance is via the use of simulation modelling, wherein easy design, analysis and evaluation of software system models

are conducted. The simulation can also change and modify parts of the system for rectification of mistakes, if any, or re-evaluate the system according to requirements of the user.

Following the researches already existent in literature, this research uses NS3 (Mini-NDN and ndnSim) simulators to evaluate the performance of the NDN cache replacement strategies in order to find out the preferable cache replacement strategy for congestion improvement and ensuring high cache efficiency [149, 150, 151, 152, 153, 154], Therefore, their features are discussed in the following subsections.

3.6.4 Features of Mini-NDN

A NDN project team sponsored by NSF developed Mini-NDN, which is a set of utilities that generates synthetic social network traces. Mini-NDN is a lightweight networking emulation tool under NS3 enabling experimentation, research and testing on the NDN platform. Similar to the Mini-CCNx which is a fork of Mininet, Mini-NDN conducts its implementation using the NDN libraries, NDN Forwarding Daemon (NFD), Named-data Link State Routing Protocol (NLSR) and other tools which were outcomes of the conducted NDN project in emulating an NDN network via a single system. Mini-NDN is an open and free software licensed under the General Public License (GPL) 3.0 license to all users and developers. Any caching strategy can be implemented in Mini-NDN, but in this study the cache replacement strategies that are evaluated are [152, 153], First-In-First-Out (FIFO), Least Frequently Used (LFU), Least Recently Used (LRU), and Cache Content Popularity (CCP). While, the placement caching strategies that is evaluated are Leave Copy Everywhere (LCE).

3.6.5 Features of ndnSim

The design of ndnSIM follows the viewpoint of network simulations in NS3, which devises maximum abstraction for all modelled components. Comparable to IPv4 and IPv6 stacks already in existence, the design of ndnSIM is to serve as an independent protocol stack suitable for installation on a network node already simulated. Additional to the core protocol stack, ndnSIM consists of an inclusion of certain helper classes and basic traffic generator applications for simplification of the production of simulation scenarios, and tools needed for gathering simulation statistics for the purpose of measurement [152]. A sample case of a simulation scenario is the helper which installs on nodes, the NDN stack and applications as show in figure 3.8.

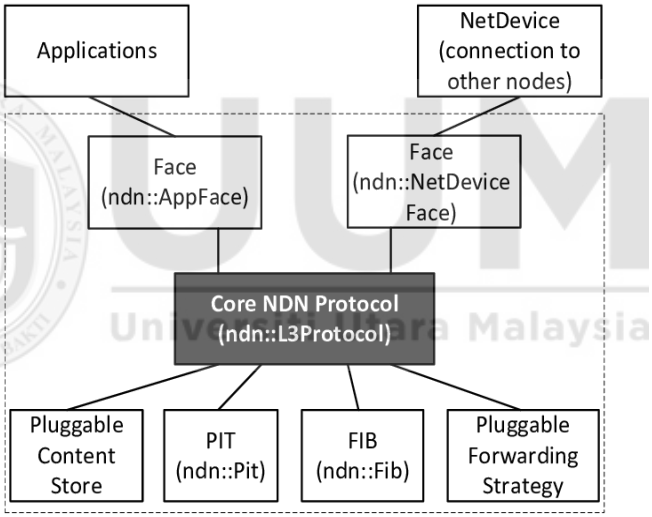


Figure 3.8: Block Diagram of ndnSIM Components [152]

Every individual component, excluding the core ndn::L3Protocol has a number of alternative implementation that can be chosen arbitrarily by the simulation scenario using helper classes. Details can be accessed from ndnSIM online documentation via http://ndnsim.net/helpers.html. For instance, ndnSIM currently provides implementations for Content Store abstraction with First-In-First-Out (FIFO), Least-Recently Used (LRU), and random replacement strategy for cached

data.

3.6.6 Performance Metrics

According to Al-Momani and Ghazali [138, 141],the selection of the performance evaluation metric is a key step and an important part of all performance evaluations. Performance metrics are adopted to evaluate the performance of the cache replacement strategies for NDN, and this evaluation is needed when researchers want to make comparison between some alternative designs in order to identify the best design [138].Following the studies already existent in literature [91, 155], the selected metrics for performance measurement in this study depend on the Table 3.3 to choose the best performance metric and discussed as follows.

A- Cache Hit Ratio: Hit ratio means the portion of content requests satisfied by caches implemented within the network. Cache Hit ratio returns the amount in an average of available content hit (found) when requests are sent. Several studies in database and networking domain have emphasized on the performance of the network through better cache hit ratio.

B- Throughput: Throughput is defined as a measurement of the total quantity of information units being processed per a specific time by a system, measured in bytes per second (bps). However, when considering throughput in the area of storage systems, it refers to the quantity of incoming data, either read from media, or written to the storage medium, and then sent back to the system initiating the request. In terms of the applications of throughput, its areas of applications in organizations are vast, with its usage being adopted in different aspects of network and computer systems. Some criteria for measuring its system productivity include workload completion speed, response time to requests, and time duration from the initiation of a single interactive

Table 3.4: *Comparison of Performance Metrics*

No.	Replacement Strategies	Metrics						
		cache hit ratio	average number of hops	average response time	Aggregated network delay	Average server load	The average throughput	byte hit rate
1	**Popularity Prediction Caching (PPC)**	X				X	X	
2	**Least Frequent Recently Used (LFRU)**	X						
3	**Popularity Based Content Eviction (PBCE)**	X		X				
4	**Time Aware Least Recently Used (TLRU)**	X						
5	**Age-based Cooperative Caching (ABC)**	X			X			
6	**Cache Content Popularity (CCP)**	X				X	X	
7	**Information-maximizing Caching (IC)**				X			
8	**Frequency/ recency based strategies**	X	X			X	X	X
9	**Recency based strategies**	X	X	X	X	X	X	X
10	**Frequency based strategies**	X	X	X	X	X	X	X
Total		9	3	3	4	5	5	3

request by the user till when the user receives the corresponding response.

C- Server Load

Server Load (SL) is defined as the traffic quantity carried by the server. Its expression is in number format, representing the number of processes standing in chain for seeking processor.

D- Average Number of Hops

The average number of hops requires obtaining a measurement of the reduced value for the number of hops traversed in satisfying a request, in comparison to the number of hops required for the retrieval of content from the server. It is worth noting that the direct measurement of the latency refers to the hop count, as all the paths possess matching features or characteristics.

Impact Model by Performance Metrics

Performance metrics are used to examine the performance of the alternative cache strategy impact model for NDN. The following metrics are selected for performance measurement:

1. Hit Rate,

2. Server Load,

3. Throughput, and

4. Average Number of Hops.

Arrows are used to denote dependencies. For instance, $(A \rightarrow B)$, this signifies that B is dependent on A. Whereas the sign denoting increase and decrease are $(+)$ and $(-)$ respectively..

Impact model describes the congestion and occupancy of cache, how these could

be decreased, and when to use CRM and CUM by performance metrics at the node properly, thus leading to cache efficiency. An illustration of Impact Model by Performance Metrics is detailed. in Figure 3.9.

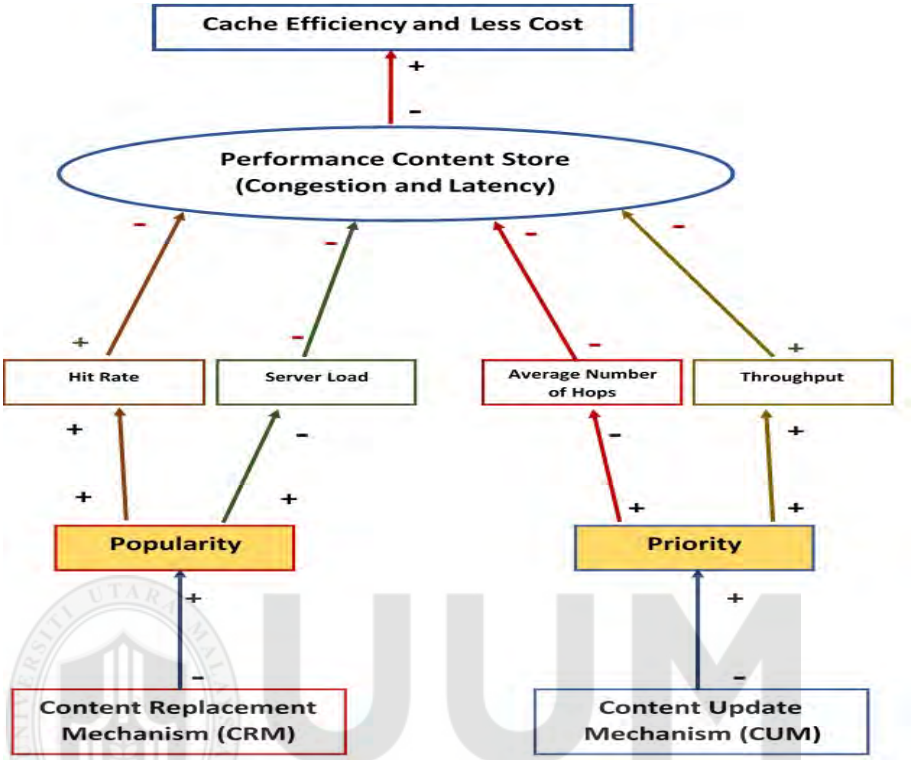


Figure 3.9: Impact Model by Performance Metrics

3.6.7 Experiment Setup

It is helpful to divide the simulation performance evaluation into steps. For instance, Hassan and Jain [156] have decomposed the simulation task into eight steps, as shown in Figure 3.10. Their work has two stages, that is, pre-software stage and software stage, which was likewise adopted in this research [112, 156, 128].

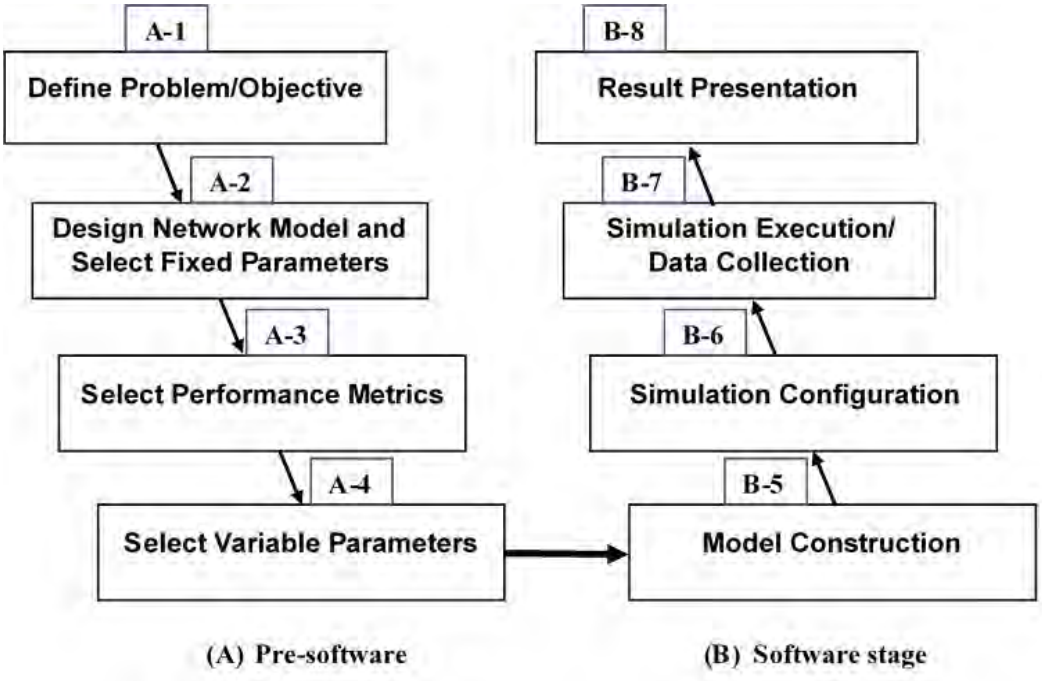


Figure 3.10: Simulation Steps [156]

A- Pre-software stage

This stage consists of the following four steps:

1. Research objectives are explained accurately;

2. Network model is created and fixed parameters are chosen. In this step network topology is drawn on a piece of paper and suitable network parameters are chosen to reflect an acceptable and real scenario;

3. Performance metrics are selected for the evaluation of proposed strategy;

4. Variable parameters are selected as shown in Table 6.1.

Generally, in NDN cache management, the purpose of performance evaluation is to examine the effect of particular variables on the chosen performance metrics.

B- Software stage

This stage consists of the following four stages:

1. Network topology designed in step A-2 is implemented in a simulator;

2. Simulator is configured in such a way that appropriate performance metrics are selected in step A-3;

3. Simulator is executed and when the simulation is finished, the performance metrics data are collected as explained in Subsection 3.6.6.

4. The collected data in step B-3 are presented in a meaningful form, and the presented results are then interpreted.

3.6.8 Simulation Scenario

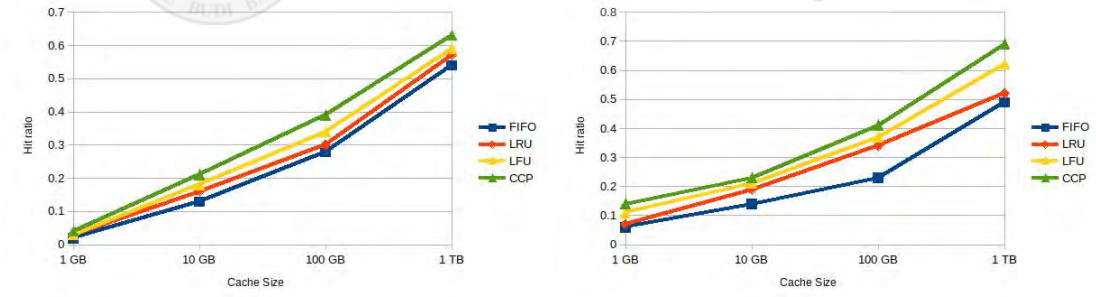
The proposed work has been simulated in the ndnSIM 2.7. In our simulations, ndnSIM 2.7 is used for evaluating four caching strategies (i.e., FIFO, LRU, LFU, CCP) with LCE cache placement policy as in [91, 157] for all NDN nodes. In ndnSIM 2.7, for modeling content requests, the Zipf distribution function [157, 158] is used. Zipf generalized form Maldelbrot-Zipf (MZipf) [159, 160, 161]. In the ndnSIM 2.7, the researcher has used three ISP-level topologies, i.e., Abilene, Tree, and Grid [162, 163] for evaluating these strategies (see Table 6.1). In our simulations, the Zipf probability distribution is used as popularity model with the α parameter varies between 0.65,

0.7 and 0.85; the cache size (which specifies the available space in every node for temporally storing content objects) ranges from 100 to 100,000 elements (1GB to 1TB); and the catalog (which represents the total number of contents in a network) is 10^6 elements.

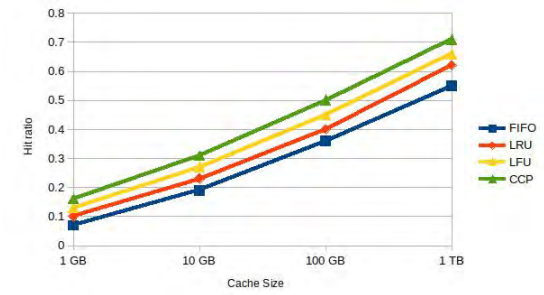
The scenario was simulated for 10 number of runs and the average values were taken for cache hit rate, as shown in Figures 3.11, 3.12, and 3.13.

Table 3.5: *Simulation Scenario*

Parameter	Description
Cache Size	$100 - 100,000$ elements
Catalog Size	10^6 elements
α Parameter	0.65, 0.7, 0.85
Topology	Abilene, Tree, Grid
NDN Project Topology	NDN Project
Simulator	ndnSIM 2.7
No. of Simulation	10 runs
Average Bit Rate	100 Mbps
Simulation Time	One day

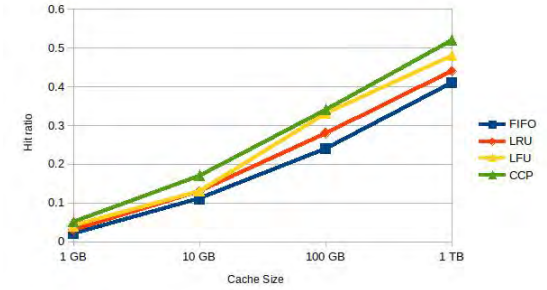


(a) *(b)*

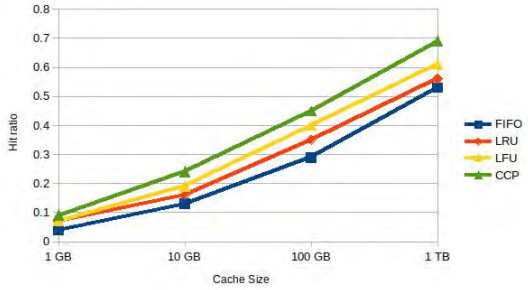


(c)

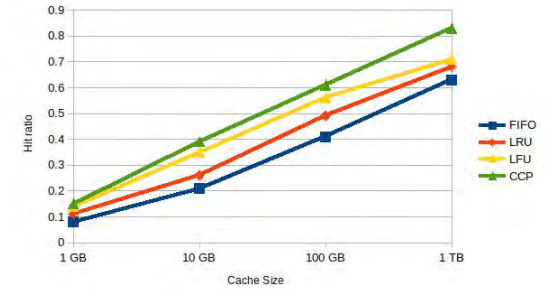
Figure 3.11: Cache Hit on Abilene Topology



(a)



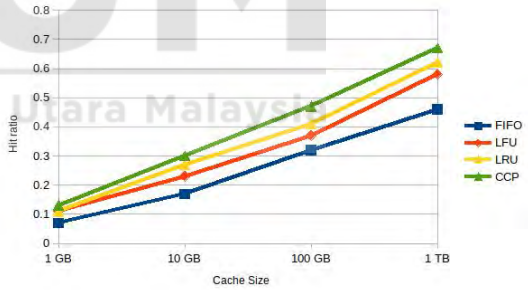
(b)



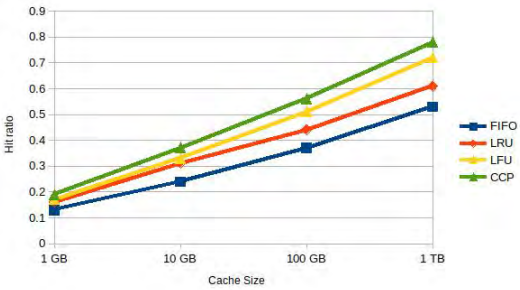
(c)

Figure 3.12: Cache Hit on Tree Topology

(a)



(b)



(c)

Figure 3.13: Cache Hit on Grid (5X5) Topology

Table 3.6: *Hit Rate on Different Topologies, Popularity Model α and Cache Size*

Topology	Popularity model α			Cache Size				Cache Hit			
	0.65	0.7	0.85	100(1GB)	1000(10GB)	10,000(100GB)	100,000(1TB)	FIFO	LRU	LFU	CCP
Abilene	X			X				0.02	0.03	0.03	0.04
	X				X			0.13	0.16	0.18	0.21
	X					X		0.28	0.3	0.34	0.39
	X						X	0.54	0.57	0.59	0.63
		X		X				0.06	0.07	0.11	0.14
		X			X			0.14	0.19	0.21	0.23
		X				X		0.23	0.34	0.37	0.41
		X					X	0.49	0.52	0.62	0.69
			X	X				0.07	0.1	0.13	0.16
			X		X			0.19	0.23	0.27	0.31
			X			X		0.36	0.40	0.45	0.50
			X				X	0.55	0.62	0.66	0.71
Tree	X			X				0.02	0.03	0.04	0.05
	X				X			0.11	0.13	0.13	0.17
	X					X		0.24	0.28	0.33	0.34
	X						X	0.41	0.44	0.48	0.52
		X		X				0.04	0.07	0.07	0.09
		X			X			0.13	0.16	0.19	0.24
		X				X		0.29	0.35	0.40	0.45
		X					X	0.53	0.56	0.61	0.69
			X	X				0.08	0.11	0.14	0.15
			X		X			0.21	0.26	0.35	0.39
			X			X		0.41	0.49	0.56	0.61
			X				X	0.63	0.68	0.71	0.83
5x5 Grid	X			X				0.03	0.05	0.05	0.07
	X				X			0.11	0.15	0.15	0.19
	X					X		0.2	0.26	0.29	0.34
	X						X	0.37	0.42	0.47	0.52
		X		X				0.07	0.11	0.11	0.13
		X			X			0.17	0.23	0.27	0.30
		X				X		0.32	0.37	0.41	0.47
		X					X	0.46	0.58	0.62	0.67
			X	X				0.13	0.16	0.17	0.19
			X		X			0.24	0.31	0.33	0.37
			X			X		0.37	0.44	0.51	0.56
			X				X	0.53	0.61	0.72	0.78

3.7 Summary

This chapter comprehensively explains the approach to guarantee the achievement of the research objectives. The research focuses on designing Content Update Mechanism (CUM) and Content Replacement Mechanism (CRM) that leads to optimal performance of the alternative caching strategy for NDN.

This chapter started by mentioning the research steps in achieving the research objectives. These steps include categorizing and sorting out a research problem, research questions, and objectives which have both academic and practical significance. Also, the steps for obtaining adequate knowledge of the current situation are discussed. This leads to the design of a reference model as well as proposing a conceptual model, and methods adopted in designing the proposed alternative caching strategy which is MADM. In addition, this chapter discusses how to the evaluate the design of alternative caching strategy. It also presents the validation of the simulator and describes which simulation steps are followed for the performance evaluation.

In the evaluation of the new strategy, this study opted for the simulation approach, which has been employed by many other researchers in this area. Thus, the evaluation metrics that have been selected are throughput, server load, cache hit ratio, bandwidth and low latency metrics. Similarly, the topologies of the NDN research approaches that have been selected are Grid, Abilene, and Tree topologies.

The next chapter will discuss comprehensively the empirical analysis of the alternative caching strategy, by giving a thorough discussion to understand its strengths and weaknesses. In addition, the chapter will discuss this new strategy based on its effective design.

CHAPTER FOUR
CONTENT REPLACEMENT MECHANISM

This chapter builds upon the rigorous literature review in Chapter Two which details the gap between this study, and the research methodology established as a guideline to achieve the first objective of this research in Chapter Three. Thus, a Content Replacement Mechanism (CRM) is a proposed mechanism for caching multimedia applications in NDN. This proposed mechanism focuses on content replacement to determine how to evict content.

The key sections in this chapter include Section 4.1 explains the system model for the content replacement mechanism based on a fuzzy approach for MADM, such that the popularity and priority values are counted for the content store to be recommended for replacement. The pseudo-code of the fuzzy TOPSIS method for content replacement is summarized in Section 4.2. While Section 4.3 is to explain the programming of this mechanism. The verification and validation of CRM are presented in Section 4.4 which is summarized in Section 4.5.

4.1 System Model

This section discusses the fuzzy Multiple Attribute Decision Making (MADM) methods, which are adopted for the resolution of the problems associated with fuzzy data. In other words, it is used for the measurement of various attribute values in different units [115, 119].

The system model defines the environment in the context of NDN, and MADM counts the popularity and priority values for each content in the content store to be recommended for replacement. The system model also outlines all the contents

informing the content store environment to select the content to be replaced.

4.1.1 Fuzzy TOPSIS Method for Content Replacement

The fuzzy MADM method is used to resolve the problems associated with fuzzy data. In other words, it measures the values of the multiple attributes in various units. The first part of our proposed strategy in this study, which is the Content Replacement Mechanism (CRM), is based on a method called TOPSIS which was developed by researchers Yoon and Hwang in 1981. These researchers developed the method to possess the property of simultaneously considering the distance to the ideal solution. Its choice of alternative is to pick the alternative having minimum distance (shortest difference) from the positive ideal solution and consequently being in maximum distance (farthest difference) position from the negative ideal solution. The usage of the fuzzy TOPSIS technique in solving multiple attributes decision-making problems when considering fuzzy environments was proposed by Chen [115]. The aim was to efficiently handle uncertainty in the evaluation matrices based on alternative evaluations and judgments.

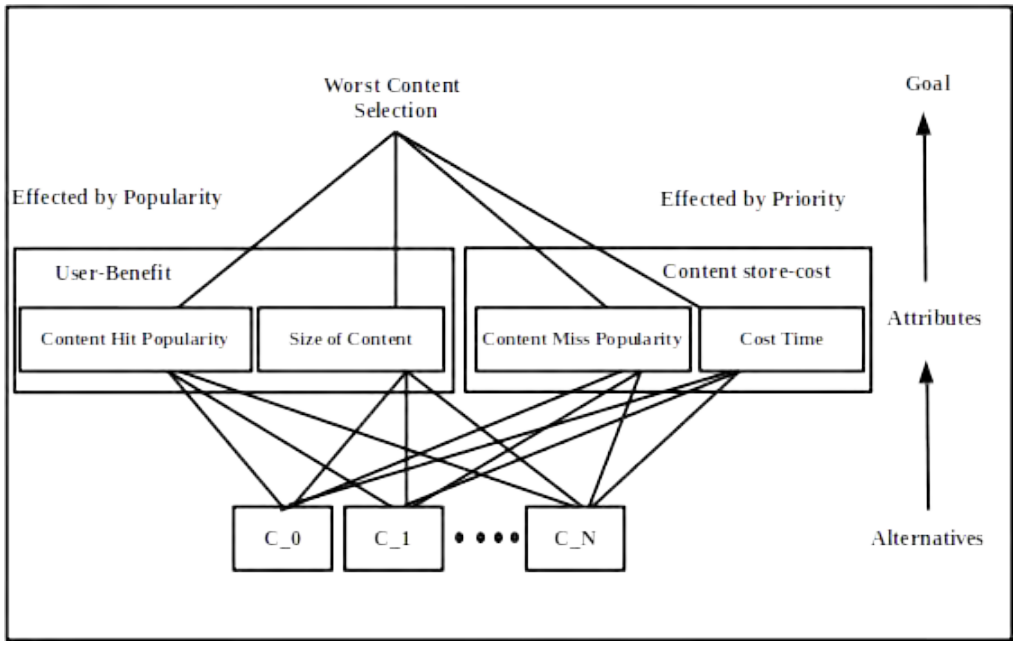


Figure 4.1: The Content Replacement Mechanism Process Model

The fuzzy TOPSIS technique is founded on two major features known as options and attributes. The options feature includes a list of alternatives that will be put into consideration when making a decision, while the attributes include parameters that will be considered for selection when making the best or optimum decision [114, 115, 118]. This technique transforms the user-benefit attributes and content store-cost attributes into two categories which are benefit and cost, as shown in Figure 4.1. The chosen attributes of popularity (benefit) are content hit popularity and content size, while the attributes of priority (cost) are content miss popularity and cost time.

The main focus of the technique is the application of the ability of multiple rule bases in decision processes, with the aim of obtaining an improvement in the transparency level of each attribute. In this approach, there is a division of the attributes into two categories, which are the benefit rule and the cost rule, as shown in Figure 4.2. Therefore, by using the division at the beginning of the TOPSIS analysis, the performance of both attributes can be tracked.

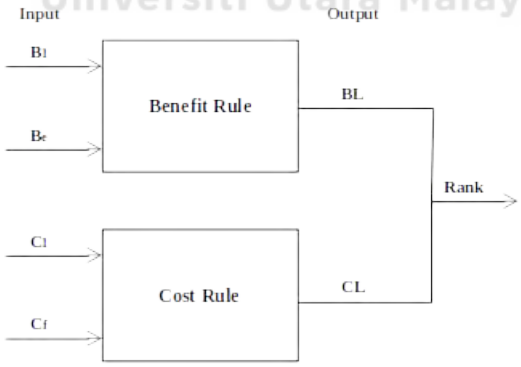


Figure 4.2: Fuzzy Model Using Multiple Rule Bases

The main procedure for implementing the fuzzy TOPSIS MADM approach by selecting the best alternative is described as follows:

Step 1: Define a decision matrix with every content being independently evaluated

and categorised into one of two attributes categories (either benefit attribute or cost attribute).

The decision matrices are denoted by D_k^B , D_k^C and the weight matrices are given as W_k^B,W_k^C , for $k = 1,...,K$. e is defined as the benefit attribute number and f is the cost attribute number, as defined in Equation 4.1 below.

$$D_k^B = \begin{array}{c} \\ B_1 \\ B_2 \\ .. \\ B_e \end{array} \begin{array}{c} \begin{array}{cccc} A_1 & A_2 & ... & A_m \end{array} \\ \begin{bmatrix} x_{11,k} & x_{12,k} & ... & x_{1m,k} \\ x_{21,k} & x_{22,k} & ... & x_{2m,k} \\ ... & ... & ... & ... \\ x_{e1,k} & x_{e2,k} & ... & x_{em,k} \end{bmatrix} \end{array} \text{ and}$$

$$D_k^C = \begin{array}{c} \\ C_1 \\ C_2 \\ .. \\ C_f \end{array} \begin{array}{c} \begin{array}{cccc} A_1 & A_2 & ... & A_m \end{array} \\ \begin{bmatrix} y_{11,k} & y_{12,k} & ... & y_{1m,k} \\ y_{21,k} & y_{22,k} & ... & y_{2m,k} \\ ... & ... & ... & ... \\ y_{f1,k} & y_{f2,k} & ... & y_{fm,k} \end{bmatrix} \end{array}$$

$$W_k^B = \begin{array}{c} \begin{array}{cccc} B_1 & B_2 & ... & B_e \end{array} \\ \begin{bmatrix} g_{1,k} & g_{2,k} & ... & g_{e,k} \end{bmatrix} \end{array} \quad \text{and} \quad W_k^C = \begin{array}{c} \begin{array}{cccc} C_1 & C_2 & ... & C_e \end{array} \\ \begin{bmatrix} h_{1,k} & h_{2,k} & ... & h_{e,k} \end{bmatrix} \end{array} \tag{4.1}$$

The variables $x_{ij,k}$, and $y_{ij,k}$ in the decision matrices represent the alternatives' rating$A_j (j = 1,...,m)$ with respect to the benefit attribute B_i $(i = 1,...,e)$ and cost attribute C_i $(i = 1,...,f)$ respectively. On the other hand, the variables $g_{i,k}$and $h_{i,k}$

in the weight matrices represent the weights of benefit attribute $B_{i,k}$ $(i = 1, ..., e)$ and the weights of cost attribute C_i $(i = 1, ..., f)$ respectively. All four variables are fuzzy sets evaluated based on the k^{th} decision, where$k = 1, ..., K$.

Step 2: Assign weight importance to each attribute with respect to the objective. The variables $W_k^B$and W_k^C in the weight matrices represent the weights of benefit attributes $B_i(i = 1, ..., e)$ and the weights of cost attributes $C_i(i = 1, ..., f)$ respectively, which is calculated as follows..

$$T_{ij,k}^B = \frac{x_{ij,k}}{\sum_{i=1}^m x_{ij,k}^2}, \quad i = 1, 2, ..., m, and \ j = 1, 2, ..., n \tag{4.2}$$

$$W_k^B = \frac{\sum_{j=1}^n T_{ij,k}}{n} \tag{4.3}$$

$$T_{ij,k}^C = \frac{y_{ij}}{\sum_{i=1}^m y_{ij}^2}, \quad i = 1, 2, ..., m, and \ j = 1, 2, ..., n \tag{4.4}$$

$$W_k^C = \frac{\sum_{j=1}^n T_{ij}}{n} \tag{4.5}$$

with n representing the numerical value for the comparable attributes.

Step 3: Define the normalized and weighted decision matrices, with their individual fuzzy rating and criterion weight being linguistic terms defined as triangular fuzzy numbers. Aj(j = 1; :::;m), which represents the ratings of alternatives, is defined using

triangular fuzzy numbers x = (axi j;k;bxi j;k;cxi j;k) and y = (ayi j;k;byi j;k;cyi j;k), while the importance of benefit attribute Bi;k (i = 1; :::;e) and the weights of cost attribute Ci (i = 1; :::; f), are represented by gi;k = (ag i;k;bg i;k;cgi ;k) and hi;k = (ah i;k;bh i;k;chi ;k) respectively, for k = 1; :::;K. The normalised fuzzy decision matrices Rk and the weighted normalised fuzzy decision matrices Vk are calculated using the expressions defined in Equation

Define the normalized and weighted decision matrices, with their individual fuzzy rating and criterion weight being linguistic terms defined as triangular fuzzy numbers. $A_j(j = 1,...,m)$, which represents the ratings of alternatives, is defined using triangular fuzzy numbers $x = (a^x_{ij,k}, b^x_{ij,k}, c^x_{ij,k})$ and $y = (a^y_{ij,k}, b^y_{ij,k}, c^y_{ij,k})$, while the importance of benefit attribute $B_{i,k}$ $(i = 1,...,e)$ and the weights of cost attribute C_i $(i = 1,...,f)$,are represented by $g_{i,k} = (a^g_{i,k}, b^g_{i,k}, c^g_{i,k})$ and $h_{i,k} = (a^h_{i,k}, b^h_{i,k}, c^h_{i,k})$ for $k = 1,...,K$. The normalised fuzzy decision matrices R_k and the weighted normalised fuzzy decision matrices V_k are calculated using the expressions defined in Equation 4.6.

$$R_k \quad = \quad \left[r_{ij,k}\right]_{(e+f)\times m}, \tag{4.6}$$

where

$$r_{ij,k} = \begin{cases} r^B_{ij,k} = \left(\frac{a^x_{ij,k}}{c^*_{i,k}}, \frac{b^x_{ij,k}}{c^*_{i,k}}, \frac{c^x_{ij,k}}{c^*_{i,k}}\right) & , \; for \; B_i \; \in \; B \\ r^C_{ij,k} = \left(\frac{a^{y^*}_{i,k}}{c^*_{ij,k}}, \frac{a^{y^*}_{i,k}}{b^*_{ij,k}}, \frac{a^{y^*}_{i,k}}{a^*_{ij,k}}\right) & , \; for \; C_i \; \in \; C \end{cases}$$

$c^{x^*}_{i,k} = max_j c^x_{ij,k}, \; (i = 1,...,e) \; , \; (j = 1,...,m)$

$a_{i,k}^{y*} = min_j a_{ij,k}^{y}$, $(i = 1, ..., f)$, $(j = 1, ..., m)$

and B represent the set of benefit attribute, while C represents the set of cost attribute.

$V_k = \left[v_{ij,k}\right]_{(e+f)\times m}$,

where

$$v_{ij,k} = \begin{cases} v_{ij,k}^{B} = r_{ij,k}(.)\, g_{i,k} & , \; for B_i \in B \\ v_{ij,k}^{C} = r_{ij,k}(.)\, h_{i,k} & , \; for C_i \in C \end{cases}$$

and

$v_{ij,k} = (a_{ij,k}^{v}, b_{ij,k}^{v}, c_{ij,k}^{v})$ are fuzzy sets, for $k = 1, ..., K$.

Step 4: Compute each alternative's Fuzzy Positive Ideal Solution (FPIS) and Fuzzy Negative Ideal Solution (FNIS). The FPIS solution is given as $A^{+} = (v_{1,k}^{+}, v_{2,k}^{+}, ..., v_{(e+f),k}^{+})$ and the FNIS solution is given as $A^{-} = (v_{1,k}^{-}, v_{2,k}^{-}, ..., v_{(e+f),k}^{-})$, with $v_{i,k}^{+} = (1\ 1\ 1)$ and $v_{i,k}^{-} = (0\ 0\ 0)$ being fuzzy sets for $k = 1, ..., K$.

Step 5: Compute the distance (calculating the difference) between each alternative to the FPIS solution and also to the FNIS solution.

The distance for the benefit attributes of each alternativeA_j from A_k^{+} , which is also the difference between each A_j from A_k^{+},is $\triangle_{j,k}^{B+}$, computed as defined in Equation 4.7.

$$\triangle_{j,k}^{B+} = \sum_{i=1}^{e} \triangle_{k}^{B}(v_{ij,k}^{B}, v_{i,k}^{+}), \tag{4.7}$$

where

$$\triangle_{k}^{B}(v_{ij,k}^{B}, v_{i,k}^{+}) = \sqrt{1/3[(a_{ij,k}^{v,B}-1)^2+(b_{ij,k}^{v,B}-1)^2+(c_{ij,k}^{v,B}-1)^2]} \tag{4.8}$$

for $j = 1, ..., m$, and $B_i \in B$, and $k = 1, ..., K$.

On the other hand, the distance for the benefit attributes of each alternative A_j from A_k^- , which is also the difference between each A_j and A_k^- , is $\triangle_{j,k}^{B-}$, computed as defined in Equation 4.9

$$\triangle_{j,k}^{B-} = \sum_{i=1}^{e} \triangle_{k}^{B}(v_{ij,k}^{B}, v_{i,k}^{-}), \tag{4.9}$$

where

$$\triangle_{k}^{B}(v_{ij,k}^{B}, v_{i,k}^{-}) = \sqrt{1/3[(a_{ij,k}^{v,B}-1)^2+(b_{ij,k}^{v,B}-1)^2+(c_{ij,k}^{v,B}-1)^2]} \tag{4.10}$$

for $j = 1, ..., m$, and $B_i \in B$, and $k = 1, ..., K$.

In the same vein, the distance for the cost attributes of each alternative A_j from A_k^+, which is also the difference between A_j from A_k^+, is $\triangle_{j,k}^{C+}$, computed as defined in Equation 4.11

$$\triangle_{j,k}^{C+} = \sum_{i=1}^{f} \triangle_k^C(v_{ij,k}^C, v_{i,k}^+), \tag{4.11}$$

whereas

$$\triangle_k^C(v_{ij,k}^C, v_{i,k}^+) = \sqrt{1/3[(a_{ij,k}^{v,C} - 1)^2 + (b_{ij,k}^{v,C} - 1)^2 + (c_{ij,k}^{v,C} - 1)^2]} \tag{4.12}$$

for $j = 1, ..., m$, and $C_i \in C$, and $k = 1, ..., K$.

Likewise, the distance for the benefit attributes of each alternative A_j from A_k^- , which is also the difference between A_j from A_k^-, is $\triangle_{j,k}^{C-}$, computed as defined in Equation 4.13

$$\triangle_{j,k}^{C-} = \sum_{i=1}^{f} \triangle_k^C(v_{ij,k}^C, v_{i,k}^-), \tag{4.13}$$

whereas

$$\triangle_k^C(v_{ij,k}^C, v_{i,k}^-) = \sqrt{1/3[(a_{ij,k}^{v,C}-1)^2+(b_{ij,k}^{v,C}-1)^2+(c_{ij,k}^{v,C}-1)^2]} \quad (4.14)$$

for $j = 1,...,m$, and $C_i \in C$, and $k = 1,...,K$.

Step 6: Compute the Closeness Coefficients (CC) for the benefit and cost attributes, denoted as $CC_{j,k}^B$ and $CC_{j,k}^C$ respectively. The CC values are calculated as given below

$$CC_{j,k}^B = \frac{\triangle_{j,k}^{B-}}{\triangle_{j,k}^{B+}+\triangle_{j,k}^{B-}} \quad (4.15)$$

,

$$CC_{j,k}^C = \frac{\triangle_{j,k}^{C-}}{\triangle_{j,k}^{C+}+\triangle_{j,k}^{C-}} \quad (4.16)$$

For $j = 1,...,m$ and $k = 1,...,K$.

Step 7: Obtain the value of the Influenced Closeness Coefficient (ICC). This value is computed through the application of the degree of influence of each decision, and thereafter the Normalised ICC (NICC) is computed from a division of the NICC value by the maximum value of NICC.

To expatiate on the process involved in this step, let θ_k denote the degree of influence of decision k; $k = 1, ..., K$ lying within the interval (0;1) with 0 being un-influential level and 10 being very influential level. Similarly, let σ_k denote the normalized degree of influence of the kth decision,k; $k = 1, ..., K$, computed using Equation 4.17

$$\sigma_k = \theta_k / \sum_{l=1}^{K} \theta_l \tag{4.17}$$

Equations 4.18 and 4.19 below then evaluates the influence closeness coefficients for each decision k along the benefit and cost attribute, which are $ICC^B_{j,k}$ and $ICC^C_{j,k}$; k respectively.

$$ICC^B_{j,k} = \sigma_k \times CC^B_{j,k} \tag{4.18}$$

$$ICC^C_{j,k} = \sigma_k \times CC^C_{j,k} \tag{4.19}$$

For $j = 1, ..., m$ and $k = 1, ..., K$.

The coefficients are then normalised so that their values vary between 0 to 1. Equations 4.20 and 4.21 evaluates the NICC values for the benefit attributes and cost attributes as

related to the kth decision, which is denoted as $NICC^B_{j,k}$ and $NICC^C_{j,k}$; k respectively.

$$NICC^B_{j,k} = ICC^B_{j,k}/max_j ICC^B_{j,k} \tag{4.20}$$

$$NICC^C_{j,k} = ICC^C_{j,k}/max_j ICC^C_{j,k} \tag{4.21}$$

For $j = 1, ..., m$ and $k = 1, ..., K$.

Step 8: Calculate each alternative's final score, with the final scorer_j for each alternative j defined as

$$r_j = \frac{\sum_{k=1}^{k} \left(NICC^B_{j,k} + NICC^C_{j,k} \right)}{2K} \tag{4.22}$$

Step 9: Rank alternatives based on the final calculated score value. Note that if a lower final value is obtained, then it implies a better performance of the alternative. For this reason, the ranking order of all the alternatives is investigated such that the conclusion drawn follows that the lowest values of r_j implies better alternatives of j.

4.1.2 Content Replacement Mechanism (CRM)

The content replacement mechanism (CRM) decides which documents stay and which documents will be replaced. It also affects which future requests will be cache hits. The CRM must decide which content to evict based only on the information it has collected about previously used contents. In other words, CRM must use past information to predict the future.

In order to determine if replacement will be initiated using CRM, the content router first needs to calculate the difference between the number of cached contents and the cache size threshold value. This calculation is done when a content router receives a new data packet that is yet to be stored, with the aim of obtaining the remaining cache size space. If the data packet is greater than the rest of the space, then replacement will be initiated, and if otherwise, replacement will not be initiated. Specifically on the basis of CRM, the content with less popular (less benefit) and low priority (low cost) will be replaced depending on TOPSIS procedure, with a simultaneous deletion of its PPT record. Then the newly arriving content will be cached in the content store and at the same time, set up its own PPT record as shown in Figure 4.3.

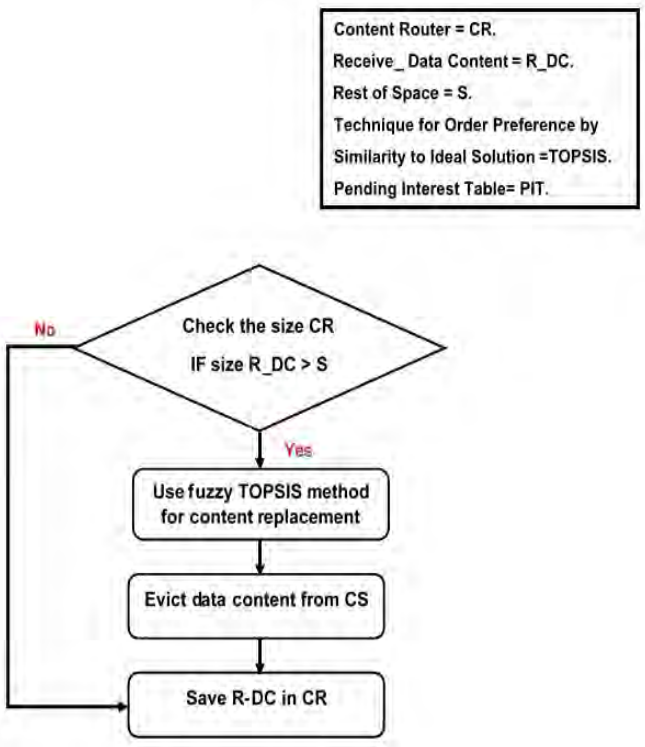


Figure 4.3: The Content Replacement Mechanism (CRM) Model

4.2 The Pseudo-Code of Content Replacement Mechanism

The pseudo-code of the fuzzy TOPSIS method for content replacement is summarized in Algorithm 4.1.

Algorithm 4.1 Fuzzy TOPSIS Method for Content Replacement

1: $Input : i : alternatives, j : attributes^{Benefit}, e : attributes^{Cost}, X_{i,j}^{Benefit}, Y_{i,e}^{Cost}$
2: $Output :$
3: $FUZZY\ TOPSIS(alternatives, attributes);$
4: $Data\ packets - Selection.removeRange();$

In the same vein, the pseudo-code of the TOPSIS method for the decision-making strategy is detailed in Algorithm 4.2. After the definition of the key criteria and alternatives related to content replacement have been given, then TOPSIS formulates the decision making matrices.

Algorithm 4.2 Fuzzy TOPSIS Decision List Formulation for Content Replacement

1: **Input:** Decision Making Matrices $X_k^{Benefit} = (x_{ij})_{m*n}$, $Y_k^{Cost} = (y_{ij})_{m*n}$
2: **Output:** list $L = (L1, j); j \in 1, m$
3: Let x, y be a fuzzy triangular number
4: $X_{ij,k}^{Benefit} = (a_{ij,k}^x, b_{ij,k}^x, c_{ij,k}^x)$
5:$Y_{ij,k}^{Cost} = (a_{ij,k}^y, b_{ij,k}^y, c_{ij,k}^y)$
6: $Q = m$
7: **while** $Q \neq 0$ **do**
8: **if** Categories = Benefit **then**
9: $\forall$s $\in$ {1,n}; $c_{i,k}^{x^*} \leftarrow max_j c_{ij,k}^x$
10: $\forall$s $\in$ {1,n}; $r_{ij,k}^B \leftarrow (\frac{a_{ij,k}^x}{c_{i,k}^*}, \frac{b_{ij,k}^x}{c_{i,k}^*}, \frac{c_{ij,k}^x}{c_{i,k}^*})_{i,s}$
11 $\forall$s $\in$ {1,n};$w_k^B \leftarrow x_{ij,k}/sum(x_{ij,k}^2)$
12 $\forall$s $\in$ {1,n};$v_{ij,k}^B = r_{ij,k}(.)\, g_{i,k}$
13: $\forall$j $\in$ {1,n}; $\triangle_{j,k}^{B+} \leftarrow [sum(\triangle_k^B(v_{ij,k}^B, v_{i,k}^+))]$
14: $\forall$j $\in$ {1,n}; $\triangle_{j,k}^{B-} \leftarrow [sum(\triangle_k^B(v_{ij,k}^B, v_{i,k}^-))]$
15: $\forall$j $\in$ {1,n};$\triangle_k^B(v_{ij,k}^B, v_{i,k}^-) \leftarrow sqrt\left(1/3\left[sum((a_{ij}^{v,B}-0)^2 + (b_{ij}^{v,B}-0)^2 + (c_{ij}^{v,B}-0)^2)\right]\right)$
16: $\forall$j $\in$ {1,n};$CC_{j,k}^B \leftarrow \triangle_{j,k}^{B-} / \left(\triangle_{j,k}^{B+} + \triangle_{j,k}^{B-}\right)$
17: $\forall$j $\in$ {1,n};$\sigma_k \leftarrow \theta_k / sum(\theta_l)$
18: $\forall$j $\in$ {1,n};$ICC_{j,k}^B \leftarrow \sigma_k \times CC_{j,k}^B$
19: $\forall$j $\in$ {1,n};$NICC_{j,k}^B \leftarrow ICC_{j,k}^B / max_j ICC_{j,k}^B$
20: **endif**
21: **if** Categories = Cost **then**
22: $\forall$s $\in$ {1,n}; $a_{i,k}^{y^*} \leftarrow min_j a_{ij,k}^y$,
23: $\forall$s $\in$ {1,n}; $r_{ij,k}^C \leftarrow (\frac{a_{i,k}^{y^*}}{c_{ij,k}^*}, \frac{a_{i,k}^{y^*}}{b_{ij,k}^*}, \frac{a_{i,k}^{y^*}}{a_{ij,k}^*})_{i,s}$
24 $\forall$s $\in$ {1,n};$w_k^C \leftarrow y_{ij,k}/sum(y_{ij,k}^2)$
25 $\forall$s $\in$ {1,n};$v_{ij,k}^C = r_{ij,k}(.)\, h_{i,k}$
26: $\forall$j $\in$ {1,n}; $\triangle_{j,k}^{C+} \leftarrow [sum(\triangle_k^C(v_{ij,k}^C, v_{i,k}^+))]$
27: $\forall$j $\in$ {1,n}; $\triangle_{j,k}^{C-} \leftarrow [sum(\triangle_k^C(v_{ij,k}^C, v_{i,k}^-))]$
28: $\forall$j $\in$ {1,n};$\triangle_k^C(v_{ij,k}^C, v_{i,k}^-) \leftarrow sqrt\left(1/3\left[sum((a_{ij}^{v,C}-0)^2 + (b_{ij}^{v,C}-0)^2 + (c_{ij}^{v,C}-0)^2)\right]\right)$
29: $\forall$j $\in$ {1,n};$CC_{j,k}^C \leftarrow \triangle_{j,k}^{C-} / \left(\triangle_{j,k}^{C+} + \triangle_{j,k}^{C-}\right)$
30: $\forall$j $\in$ {1,n};$\sigma_k \leftarrow \theta_k / sum(\theta_l)$
31: $\forall$j $\in$ {1,n};$ICC_{j,k}^C \leftarrow \sigma_k \times CC_{j,k}^C$
32: $\forall$j $\in$ {1,n};$NICC_{j,k}^C \leftarrow ICC_{j,k}^C / max_j ICC_{j,k}^C$
33: **endif**
34: $\forall$j $\in$ {1,m};$\forall$k $\in$ {1,K};$r_j \leftarrow sum\left(NICC_{j,k}^B + NICC_{j,k}^C\right)/2K$
35: $\forall$j $\in$ {1,m}; $L_{1,j} \leftarrow sort(r_j)$
36: **end while**

4.3 The Programming of Content Replacement Mechanism

The programming of the mechanism is a very important step in the designing of the algorithm. In this step, the algorithm in Algorithm 4.1 is converted to coding usingubuntu 18.4 and ndnSIM 2.7. Four criteria are used in this coding, which are

HIT, SIZE, MISS, and TIME as shown in Figure 4.4a. Whereas in Figure 4.4b, all the rules to build the programming as defining the content and decision matrices are followed, the initial weight vector is given, then the decision matrices are built, and the transition continues in that manner.

(a)

(b)

Figure 4.4: The Code of CRM

4.4 CRM Verification and Validation

The primary concern in this phase is to ensure that the proposed CRM has been programmed correctly inside the ndnSIM, and that it has been implemented properly on the computer system. Verification was conducted following the techniques mentioned in Chapter Three. To confirm the verification, a snapshot of the CRM implementation and ndnSIM 2.7 codes are presented in Figures 4.5 and 4.6 respectively.

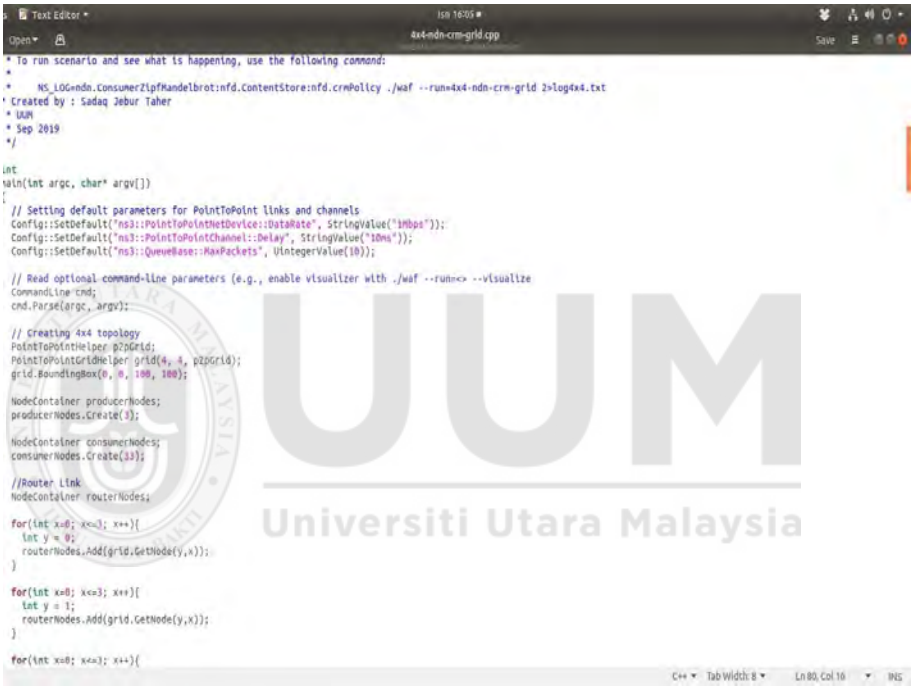


Figure 4.5: The Main Code for CRM

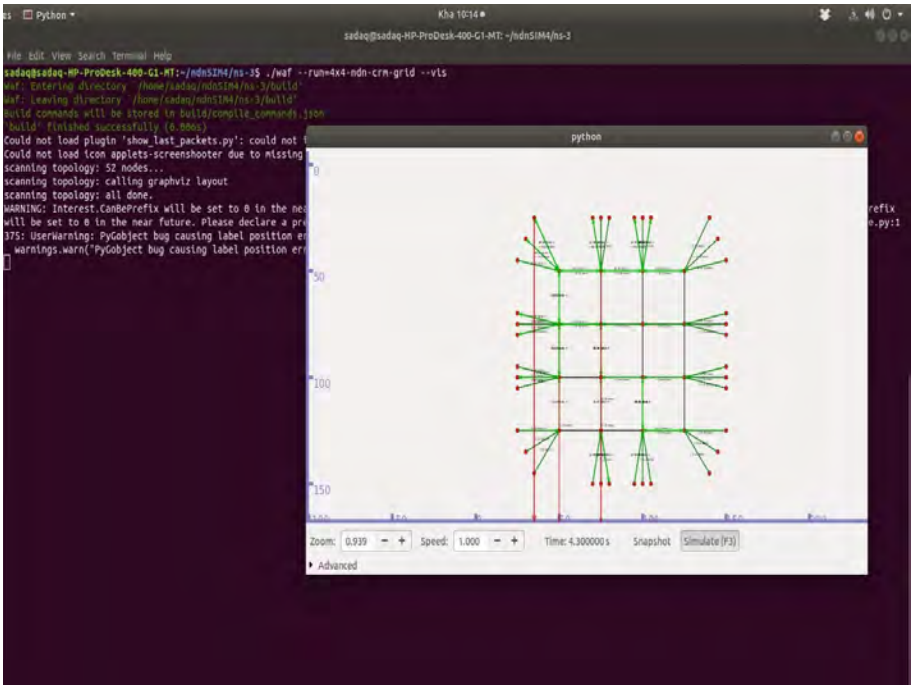


Figure 4.6: CRM Implementation

The images serve as confirmation that

• CRM is programmed correctly, and

• CRM implementation does not contain any errors or bugs.

Next, the CRM was validated to ensure that it meets the intended requirements in terms of cache hit rate. For surety, the CRM was run on a 44 grid with 33 consumers randomly distributed across the network and three origin. The obtained results were compared with four popular strategies, which include First-In-First-Out (FIFO), Least Frequently Used (LFU), Least Recently Used (LRU), and Cache Content Popularity (CCP). The validation was done using ndnSIM and the value of the Zipf probability model parameter was set as q=5, α=0.85. The cache size (specifying each node's available space for the temporal storage of content objects) ranged from 100 to 2000

elements (1*GB* to 20*GB*); and the catalog (representing the total number of contents within the network) is 10^6elements. The simulation was run for 10 times on topology and the average value of cache hit rate was obtained for FIFO, LFU, LRU, CCP and CRM. The obtained resultant graph is presented in Figure 4.7. It is observed that the cache hit rate needed for getting the contents from the router dramatically increases.

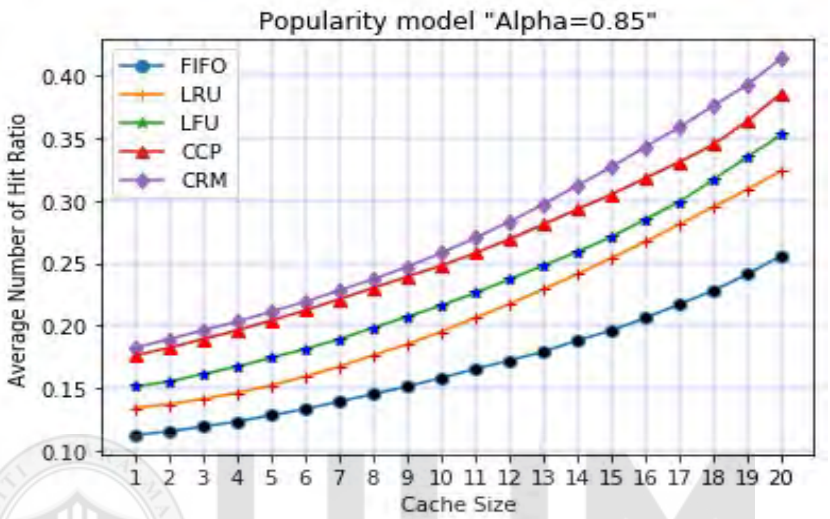


Figure 4.7: Cache Hit Ratio on Different Cache Sizes

4.5 Summary

This chapter introduced a caching replacement mechanism (CRM) for named data networking under ICN. The chapter presented the CRM as part of an alternative caching strategy. CRM depended on the less popular and low-priority content being replaced first. It is observed from the results that the cache hit rate needed for getting the contents from the router dramatically increases.

CHAPTER FIVE
CONTENT UPDATE MECHANISM

This chapter is based on the literature review in Chapter Two and the research methodology established as a guideline to achieve the objectives of this research in Chapter Three. Thus, a Content Update Mechanism (CUM) is proposed to determine when to update the content. Then, an Alternative Caching Strategy (ACS) is proposed in this Chapter as a whole strategy that contains CUM in addition to the Content Replacement Mechanism (CRM) Which is explained in the previous Chapter.

The key sections in this chapter include Section 5.1 explains the system model for the content update mechanism based on a fuzzy approach for MADM, such that the priority and popularity values are counted for the content store to be recommended for updating. The pseudo-code of the fuzzy SAW method for content update is summarized in Section 5.1.1. While Section 5.2 is to explain the programming of this mechanism. The verification and validation of CUM are presented in Section 5.3.

An Alternative Caching Strategy (ACS) is proposed based on Content Replacement Mechanism (CRM) in addition to CUM to get an efficient cache, as shown in Section 5.4. The pseudo-code of ACS is explained in Section 5.5 while the verification and validation of ACS are presented in Section 5.6. The results of this ACS strategy will be explained in the next chapter. The whole Chapter is summarized in Section 5.7.

5.1 System Model

This section discusses the fuzzy Multiple Attribute Decision Making (MADM) method, which is adopted for the resolution of the problems associated with fuzzy data. In other words, it is used for the measurement of various attribute values in

different units[115, 119].

The system model defines the environment in the context of NDN, and MADM counts the popularity and priority values for each content in the content store to be recommended forupdate. The system model also outlines all the contents informing the content store environment to select the content to be updated.

5.1.1 Fuzzy SAW Method for Content Update

The SAW method has been categorised among the most widely adopted approaches when resolving special decision analysis problems. The technique follows a process of assigning weights of relative importance to every attribute, and afterwards, a total score is calculated for the alternatives through the multiplication of the scaled value given to the attribute's alternative and the importance weight value assigned for each attribute. This concept of eventually assigning crisp scores to the alternatives makes it a more realistic and practical technique [114, 115, 120, 119, 164].

This technique transforms the user-benefit attributes and content store-cost attributes into two categories are benefit and cost, as shown in Figure 5.1. The chosen attributes of popularity (benefit) are content hit popularity and content size, while the attributes of priority (cost) are content miss popularity, freshness period of content, and cost time.

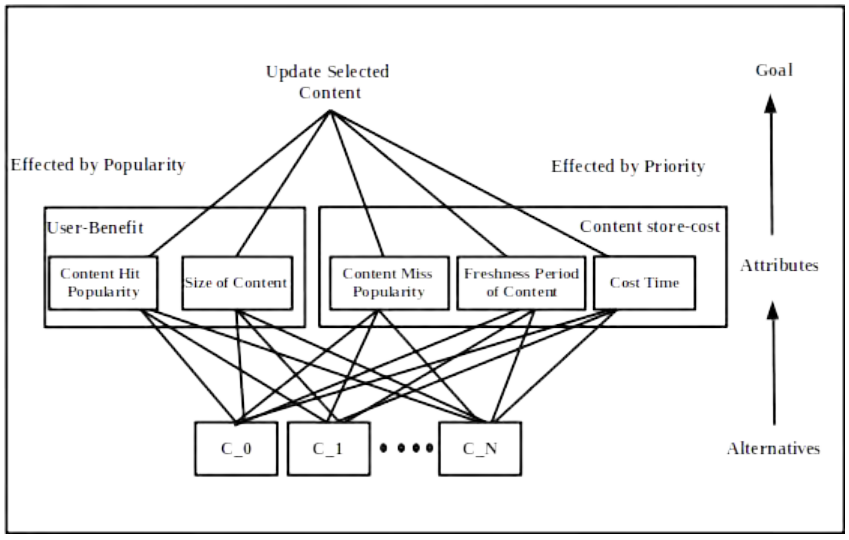


Figure 5.1: Content Update Mechanism Process Model

A fuzzy update mechanism is made up of a one-rule base wherein the inputs are simultaneously attended to, and $\{C1,...,Cn\}$ denotes the set of inputs. In this case, the multiple inputs consisting of benefit and cost attributes, are joined together in the alternative rule base having multiple inputs and a single output. The main procedure for implementing Fuzzy SAW method in selecting the best alternative is described as follows.

Assuming that the DM makes an assignment of a set of weights, $W = (W_1,...,W_n)$, corresponding to the attributes, $X_j, j = 1,...,n$. Then A_i (the performance of the alternative), is calculated as:

$$S_i = \left(\left(\sum_{j=1}^{M} w_i (\cdot)\, p_{ij} \right) / \sum_{j=1}^{M} w_j \right)$$

where p_{ij} is i^{th} alternative's rating under the $j^{th}\}$ attribute with a scale that is numerically comparable. This form defined above is the most basic form in the Multiple Attribute Utility Theory (MAUT), where A^* defined as

$$A^* = \{A_i \mid max_i S_i\}$$

is then selected, with A^* being the most preferred alternative.

Taking a step further, a model of fuzzy simple additive weighting using TFNs is adopted

Step 1: Define a decision matrix with every content being independently evaluated and categorised into one of two attribute categories (either benefit attribute or cost attribute).

The decision matrices are denoted by D_k^B , D_k^C and the weight matrices are given as W_k^B,W_k^C , for $k = 1, ..., K$. e is defined as the benefit attribute number and f is the cost attribute number, as defined in Equation 5.1 below.

$$D_k^B = \begin{array}{c} \\ B_1 \\ B_2 \\ .. \\ B_e \end{array} \begin{array}{c} \begin{array}{cccc} A_1 & A_2 & ... & A_m \end{array} \\ \begin{bmatrix} x_{11,k} & x_{12,k} & ... & x_{1m,k} \\ x_{21,k} & x_{22,k} & ... & x_{2m,k} \\ ... & ... & ... & ... \\ x_{e1,k} & x_{e2,k} & ... & x_{em,k} \end{bmatrix} \end{array},$$

$$D_k^C = \begin{array}{c} \\ C_1 \\ C_2 \\ .. \\ C_f \end{array} \begin{array}{c} \begin{array}{cccc} A_1 & A_2 & ... & A_m \end{array} \\ \begin{bmatrix} y_{11,k} & y_{12,k} & \cdots & y_{1m,k} \\ y_{21,k} & y_{22,k} & \cdots & y_{2m,k} \\ ... & ... & ... & ... \\ y_{f1,k} & y_{f2,k} & \cdots & y_{fm,k} \end{bmatrix} \end{array},$$

$$W_k^B = \begin{matrix} B_1 & B_2 & ... & B_e \end{matrix} \begin{bmatrix} g_{1,k} & g_{2,k} & \cdots & g_{e,k} \end{bmatrix} \quad \text{and} \quad W_k^C = \begin{matrix} C_1 & C_2 & ... & C_e \end{matrix} \begin{bmatrix} h_{1,k} & h_{2,k} & ... & h_{e,k} \end{bmatrix} \tag{5.1}$$

The variables $x_{ij,k}$, and $y_{ij,k}$ in the decision matrices represent the alternatives' rating $A_j(j = 1,...,m)$ with respect to the benefit attributeB_i $(i = 1,...,e)$ and cost attribute C_i $(i = 1,...,f)$ respectively. On the other hand, the variables $g_{i,k}$and $h_{i,k}$ in the weight matrices represent the weights of benefit attribute $B_{i,k}$ $(i = 1,...,e)$ and the weights of cost attribute C_i $(i = 1,...,f)$ respectively. These four variables are fuzzy sets evaluated based on the k^{th} decision, where $k = 1,...,K$.

Step 2: Assign weight importance to each attribute with respect to the objective. The variables $W_k^B$and W_k^C in the weight matrices represent the weights of benefit attributes $B_i(i = 1,...,e)$ and the weights of cost attributes $C_i(i = 1,...,f)$, which is calculated as follows.

$$T_{ij,k}^B = \frac{x_{ij,k}}{\sum_{i=1}^m x_{ij,k}^2} i = 1,2,...,m, \text{ and } j = 1,2,...,n \tag{5.2}$$

$$W_k^B = \frac{\sum_{j=1}^n T_{ij,k}}{n} \tag{5.3}$$

$$T_{ij,k}^C = \frac{y_{ij}}{\sum_{i=1}^m y_{ij}^2} i = 1,2,...,m, \text{ and } j = 1,2,...,n \tag{5.4}$$

$$W_k^C = \frac{\sum_{j=1}^{n} T_{ij}}{n} \tag{5.5}$$

with n representing the numerical value for the comparable attributes.

Step 3: Define the normalized and weighted decision matrices, with their individual fuzzy rating and criterion weight being linguistic terms defined as triangular fuzzy numbers. $A_j (j = 1, ..., m)$ which represents the ratings of alternatives, is defined using triangular fuzzy numbers $x = (a_{ij,k}^x, b_{ij,k}^x, c_{ij,k}^x)$ and $y = (a_{ij,k}^y, b_{ij,k}^y, c_{ij,k}^y)$, while the importance of benefit attribute $B_{i,k}$ $(i = 1, ..., e)$ and the weights of cost attribute C_i $(i = 1, ..., f)$, are represented by$g_{i,k} = (a_{i,k}^g, b_{i,k}^g, c_{i,k}^g)$ and $h_{i,k} = (a_{i,k}^h, b_{i,k}^h, c_{i,k}^h)$ respectively, for $k = 1, ..., K$.

The assigning of the attribute types could follow such that the the benefit type is categorised for a greater value, while the cost type is categorised for a smaller value. Equations 5.6 and 5.7 define the expressions for the computation of the value for the benefit type and cost type respectively.

$$D_{ij,k}^B = \left(\frac{a_{ij,k}^x}{c_{i,k}^*}, \frac{b_{ij,k}^x}{c_{i,k}^*}, \frac{c_{ij,k}^x}{c_{i,k}^*} \right), \tag{5.6}$$

$$D_{ij,k}^C = \left(\frac{a_{i,k}^{y^*}}{c_{ij,k}^*}, \frac{a_{i,k}^{y^*}}{b_{ij,k}^*}, \frac{a_{i,k}^{y^*}}{a_{ij,k}^*} \right) \tag{5.7}$$

with $c_{i,k}^{x^*} = max_j c_{ij,k}^{x}$, $(i = 1, ..., e)$, $(j = 1, ..., m)$, and

$a_{i,k}^{y^*} = min_j a_{ij,k}^{y}$, $(i = 1, ..., f)$, $(j = 1, ..., m)$

Step 4: Collate the performance ratings for each alternative (with respect to all the attributes) using the following equations:

$$\mathrm{S_i^B} = \left(\left(\sum_{\mathrm{j=1}}^{\mathrm{M}} \mathrm{w_i^B} \left(\cdot\right) \mathrm{D_{ij}^B} \right) / \sum_{\mathrm{j=1}}^{\mathrm{M}} \mathrm{w_j^B} \right) \tag{5.8}$$

$$\mathrm{S_i^C} = \left(\left(\sum_{j=1}^{M} w_i^C \left(\cdot\right) D_{ij}^C \right) / \sum_{j=1}^{M} w_j^C \right) \tag{5.9}$$

$$S_i \quad = \quad S_i^B + S_i^C \tag{5.10}$$

Step 5: Rank alternatives based on the final calculated score value, where the highest final value implies that alternative is performing better. For this reason, the ranking order for all the alternatives under consideration is investigated such that the highest values of S_i implies a better alternative of i.

The goal of the Fuzzy SAW method is to update the good content having high popular and high priority content at a content store after deleting it, and then send a request to PIT. The pseudo-code of the fuzzy SAW method for the content update is summarized

in Algorithm 5.1.

Algorithm 5.1 Fuzzy SAW Method for Content Update

1: $Input : i : alternatives, j : attributes^{Benefit}, e : attributes^{Cost}, X_{i,j}^{Benefit}, Y_{i,e}^{Cost}$
2: $Output :$
3: $FSAW(alternatives, attributes)$;
4: $Data\ packets - Selection.updateRange()$;
5: $Send\ request\ to\ PIT$

In the same vein, the pseudo-code of the SAW decision-making strategy is detailed in this Algorithm. After the definition of the main attributes and alternatives related to the content update have been given, then SAW formulates the decision making matrix.

Algorithm 5.2 Fuzzy SAW Decision List Formulation for Content Update

1: **Input:** Decision Making Matrices $X_k^{Benefit} = (x_{ij})_{m*n}$, $Y_k^{Cost} = (y_{ij})_{m*n}$
2: **Output:** list $L = (L1, j); j \in 1, m$
3: Let x, y be a fuzzy triangular number
4: $X_{ij,k}^{Benefit} = (a_{ij,k}^x, b_{ij,k}^x, c_{ij,k}^x)$
5:$Y_{ij,k}^{Cost} = (a_{ij,k}^y, b_{ij,k}^y, c_{ij,k}^y)$
6: $Q = m$
7: **while** $Q \neq 0$ **do**
8: **if** Categories = Benefit **then**
9: $\forall s \in \{1,n\}; c_{i,k}^{x^*} \leftarrow max_j c_{ij,k}^x$
10: $\forall s \in \{1,n\}; r_{ij,k}^B \leftarrow (\frac{a_{ij,k}^x}{c_{i,k}^*}, \frac{b_{ij,k}^x}{c_{i,k}^*}, \frac{c_{ij,k}^x}{c_{i,k}^*})_{i,s}$
11 $\forall s \in \{1,n\}; w_k^B \leftarrow x_{ij,k}/sum(x_{ij,k}^2)$
12 $\forall s \in \{1,n\}; v_{ij,k}^B = r_{ij,k}(.)\, g_{i,k}$
20: **endif**
21: **if** Categories = Cost **then**
22: $\forall s \in \{1,n\}; a_{i,k}^{y^*} \leftarrow min_j a_{ij,k}^y,$
23: $\forall s \in \{1,n\}; r_{ij,k}^C \leftarrow (\frac{a_{i,k}^{y^*}}{c_{ij,k}^*}, \frac{a_{i,k}^{y^*}}{b_{ij,k}^*}, \frac{a_{i,k}^{y^*}}{a_{ij,k}^*})_{i,s}$
24 $\forall s \in \{1,n\}; w_k^C \leftarrow y_{ij,k}/sum(y_{ij,k}^2)$
25 $\forall s \in \{1,n\}; v_{ij,k}^C = r_{ij,k}(.)\, h_{i,k}$
33: **endif**
34: $\forall j \in \{1,m\}; \forall k \in \{1,K\}; r_j \leftarrow sum\left(v_{ij,k}^B + v_{ij,k}^C\right)$
35: $\forall j \in \{1,m\}; L_{1,j} \leftarrow sort(r_j)$
36: **end while**

5.1.2 Content Update Mechanism (CUM)

The rationale behind this mechanism is the prediction of future events using information from their past behaviour. The content update mechanism (CUM) decides which document will update after removal from the content store. The CUM must

decide which content to update based only on the information it has collected about previously used contents. In other words, the CUM must use past information to predict the future. Therefore, future requests for this content will increase hit cache and smaller N hops, leading to a smaller latency for retrieving data and bandwidth reduction.

When a data content attains the expired date for a data packet which has been stored, it must be deleted and update. The updated content is initiated if the rank of the data content is greater than that of the data contents currently in the content store. This update will happen by sending a request to PIT to fetch the updated content. On the basis of CUM, the content with more popular (more benefit) and high priority (low cost) will be updated (depending on the fuzzy SAW procedure) and deleted from the content store and PPT simultaneously. Thus, it will create free space in the content router. Then the newly arriving content will be cached in the content store and its own PPT record will be set up simultaneously. Figure 5.2 gives the visual details.

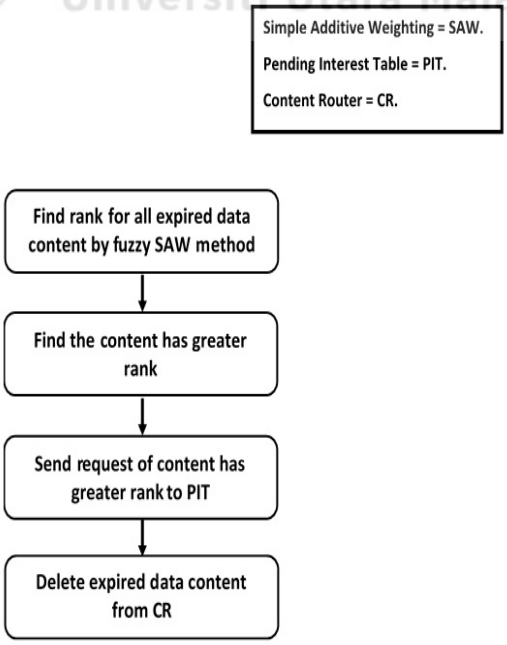
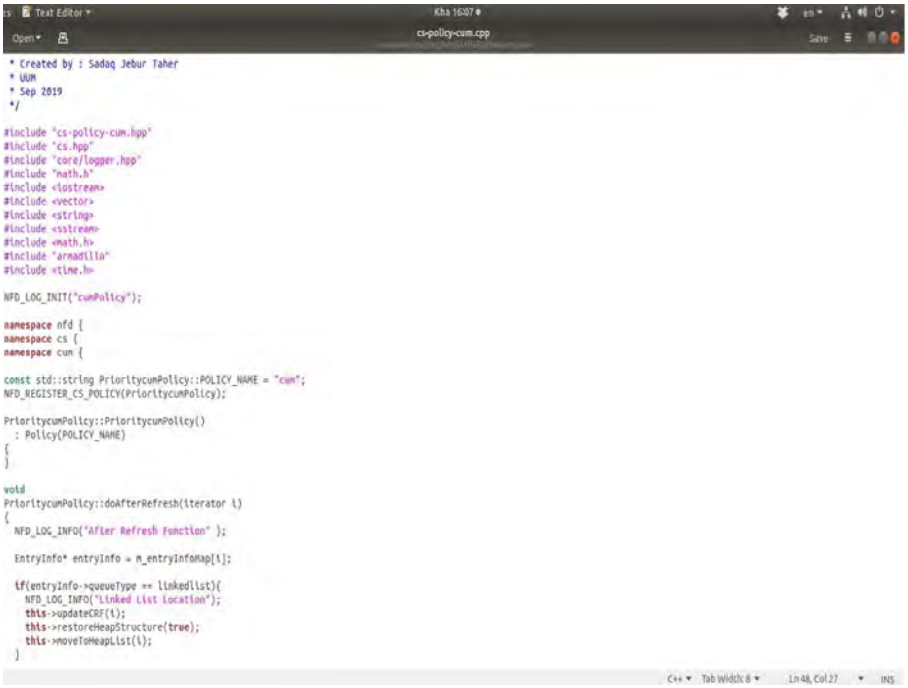


Figure 5.2: The Content Update Mechanism (CUM) Model

5.2 The Programming of CUM

In this programming step, which is quite essential in the algorithm design phase, Algorithm 4.1 is converted to coding using ubuntu 18.4 and ndnSIM 2.7. Five criteria are implemented in the coding, which are hit, size, miss, freshness and time (see Figure 5.3a). Also, all the rules to build the programming as defining the content and decision matrices are followed, giving the initial weight vector, and then building the decision matrices, followed by other steps of the transition (see Figure 5.3b).



(a)

(b)

Figure 5.3: The Code of CUM

5.3 CUM Verification and Validation

This phase aims to ensure that the proposed CUM has been programmed without error inside the ndnSIM, and implemented precisely on the computer system. The

techniques mentioned in Chapter Three were used for the verification process. To confirm the verification, a snapshot of the CUM implementation and ndnSIM 2.7 code are presented in Figure 5.4 and Figure5.5 respectively,

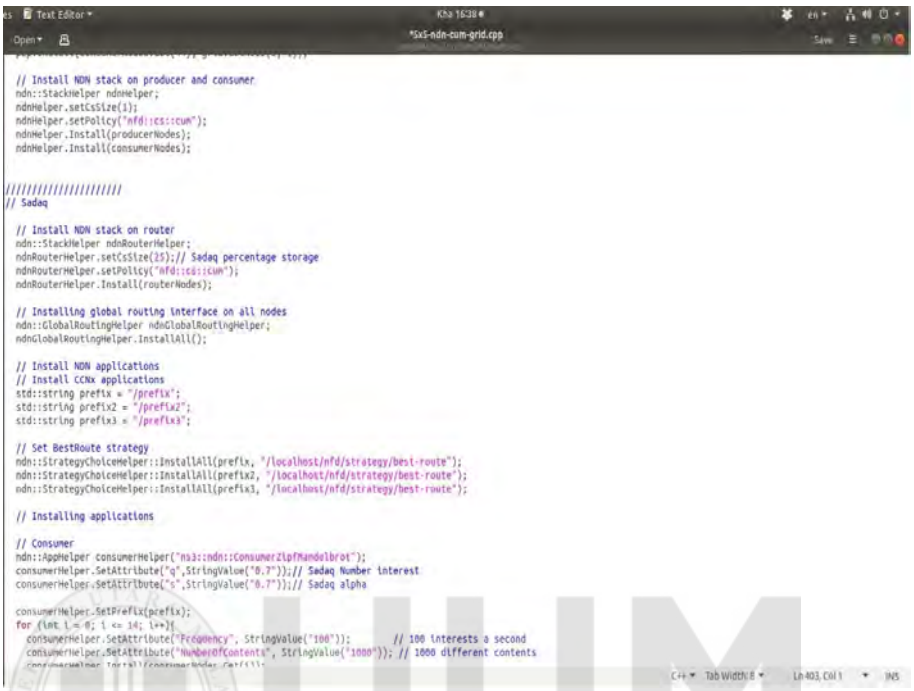


Figure 5.4: The Main Code for CUM

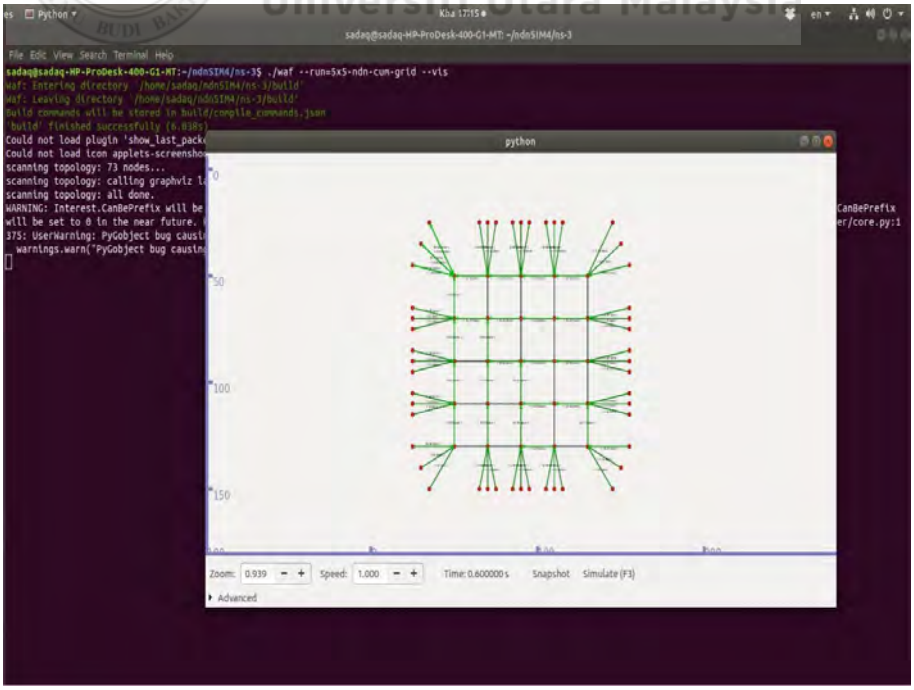


Figure 5.5: CUM Implementation

It is evident that the CUM has been programmed correctly, and the implementation of CUM does not contain any errors or bugs.

The validation phase is initiated next, as the CUM was validated to ensure that it meets the intended requirements in terms of average number of hops. For surety, the CUM was run on a 55 grid with 45 randomly distributed consumers across the network and three origins. The obtained results were compared with four popular strategies; FIFO, LRU, LFU, and CCP. This validation was conducted using ndnSIM and the value of the Zipf probability model parameter was set to $q = 0.7$ $\alpha = 0.7$. The cache size(specifying each node's available space for the temporal storage of content objects) ranged from 100 to 2000 elements ($1GB$ to $20GB$); and the catalog (representing the total number of contents within the network) equalled 10^6 elements. The simulation was run for 10 times on topology and the average number of hops was obtained for FIFO, LRU, LFU, CCP and CUM. The resultant graph is presented in Figure 5.6. The graph shows a significant decline in the value of the average number of hops required to get the contents from the router.

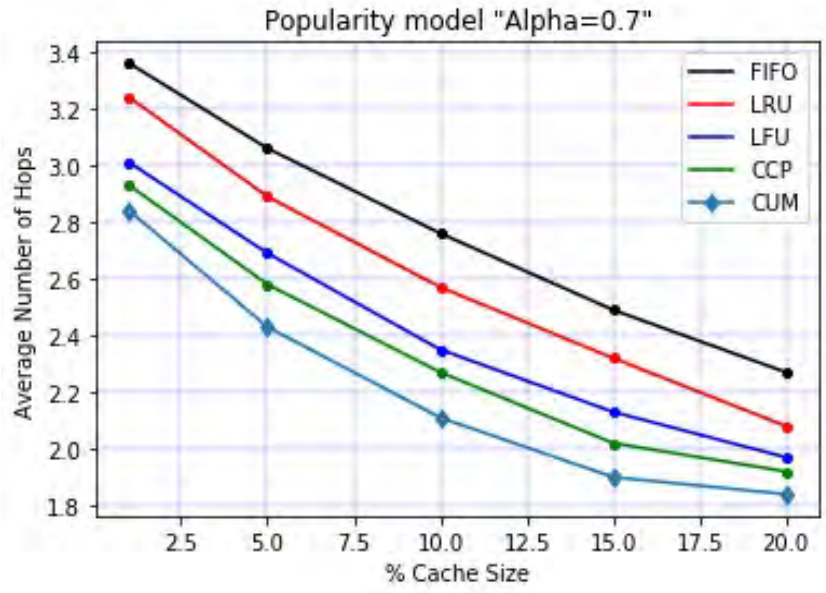


Figure 5.6: Average Number of Hops on Different Cache Sizes

5.4 The Alternative Caching Strategy

The objective of the alternative caching strategy is to get an efficient cache which contains each stored content that is useful by the consumers. The strategy also ensures that cached data is not evicted without satisfying any request or update, the consumers are be provided with valid content, and out-of-date contents are avoided. This helps to avoid congestion and thus, there is space in the cache. The strategy contains two mechanisms as shown in Figure 5.7, These mechanisms are the Content Replacement Mechanism (CRM) which is implemented based on fuzzy TOPSIS, and the Content Update Mechanism (CUM) which is based on fuzzy SAW.

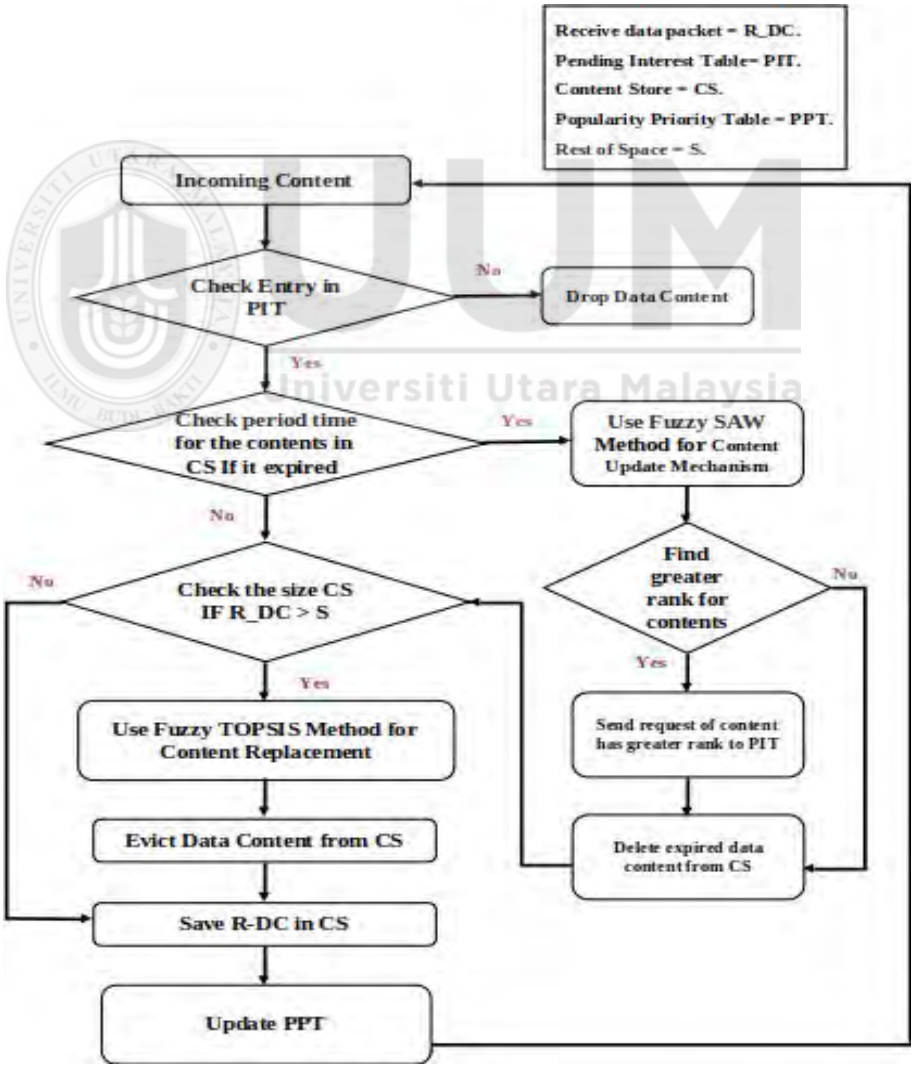


Figure 5.7: The Alternative Caching Strategy Model

5.5 The pseudo-code of the Alternative Caching Strategy

The pseudo-code of the ACS strategy is detailed in Algorithm 5.3 The algorithm follows after defining the main attributes and alternatives that depend on CRM and CUM (popularity and priority respectively), relating to content replacement and content update. This stage is similar to the previous section detailing design of mechanisms in this chapter.

Algorithm 5.3 The Pseudo-Code of the Alternative Caching Strategy

Incoming Content:
Calculate remaining time for all content name in PPT table
Find the content names having Remaining Time = 0
Send requests of content names to **PIT** that has the **greatest** rank depending on the output list for **Fuzzy SAW**
Remove all contents from **PPT** table without the Content Name and Request Time, where the Request Time = Current time
Remove a content from content store
Remove a content from **PPT** table (has **less** rank)
Calculate the size of the free content store (cs) and mark as cz
Calculate the size of the incoming content (ic) and mark as iz
while cz < iz
Remove a content from content store and all the values related to it from **PPT** table depending on output list for **Fuzzy TOPSIS**
end
Save ic to cs
Find the data item corresponding to the name ic in **PPT** table
Add arrival time , cost time , content size , freshness period of content, and remaining time
Incoming Request :
Incoming request mark as ir
if ir in cs
Find the data item corresponding to the name ir in **PPT** table
Increase counter content hit popularity in **PPT** table
else
Add content name, content miss popularity and request time to **PPT** table
endif

5.6 ACS Verification and Validation

ACS verification was conducted following the techniques mentioned in Chapter Three. Note that the objective is to ensure that the proposed ACS has been programmed

correctly in ndnSIM 2.7, and also implemented without mistakes on the computer system. A confirmation of the verification is shown in the snapshot of the ACS implementation, and ndnSIM 2.7 codes presented in Figure 5.8 and Figure 5.9 respectively.

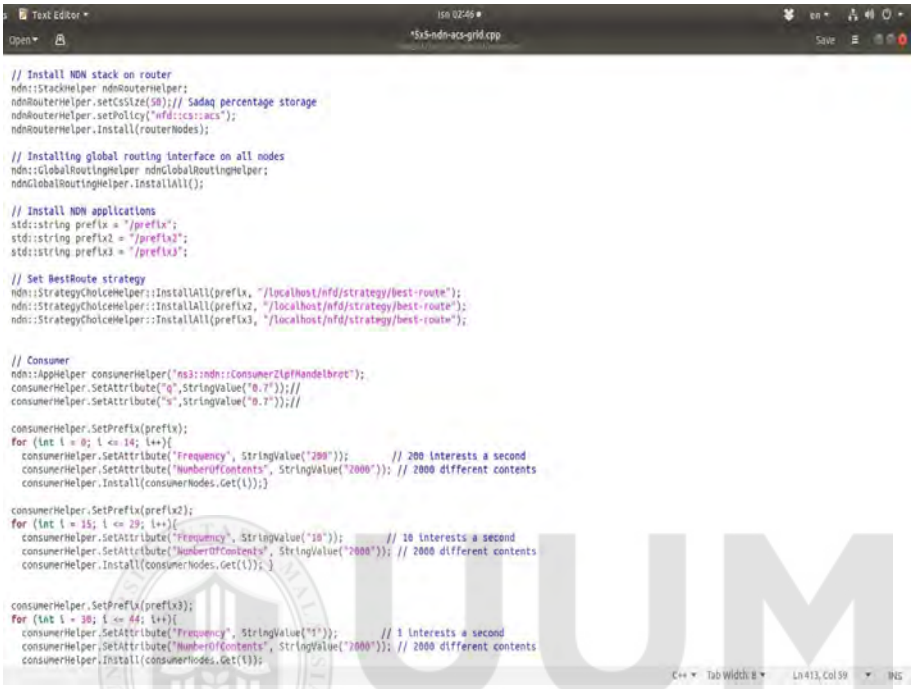


Figure 5.8: The Main Code for ACS

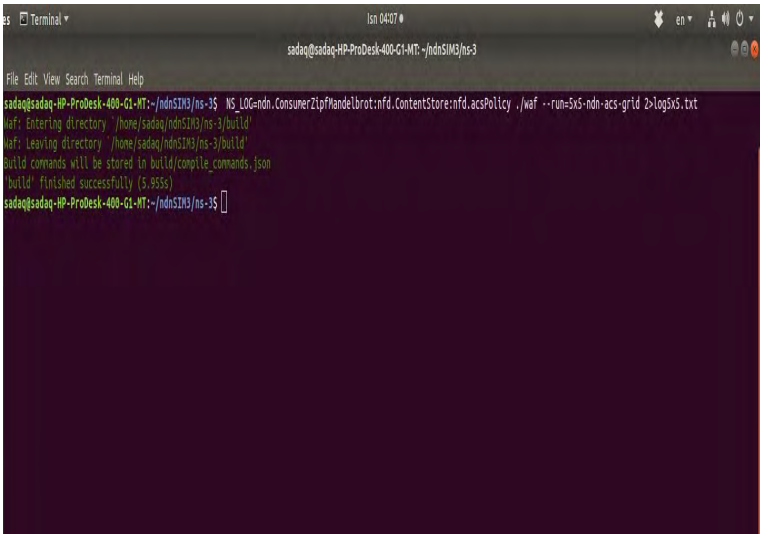


Figure 5.9: ACS Implementation

The following conclusions are drawn from the images presented:

• ACS is programmed correctly, and

• ACS implementation does not contain any errors or bugs.

Then the ACS was validated to ensure that it meets the intended requirements in terms of the average number of hops and cache hit rate. For surety, the ACS was run on a 5x5 grid, with 45 consumers randomly distributed across the network on three origins.

The obtained results were compared with three popular strategies, which were the LRU, LFU, and CCP. The validation was done using ndnSIM and the value of the Zipf probability model parameter was selected as $q = 0.85$ $\alpha = 0.85$. A range from 100 to 2000 elements ($1GB$ to $20GB$) was used for the cache size (specifying each node's available space for the temporal storage of content objects), and the catalog (representing the total number of contents within the network) contained 10^6 elements. The simulation was run for 10 times on topology and the average value of cache hit rate was obtained for LRU, LFU, CCP and ACS. The obtained graph is presented in Figures 5.10 5.11. The results show a significant decline in the value of the average number of hops required to get the contents from the router. Thus, network congestion leads to high latency and low throughput.

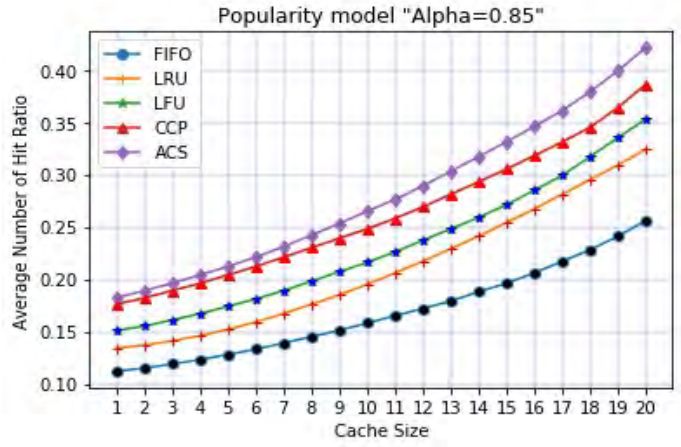


Figure 5.10: Cache Hit Ratio on Different Cache Sizes

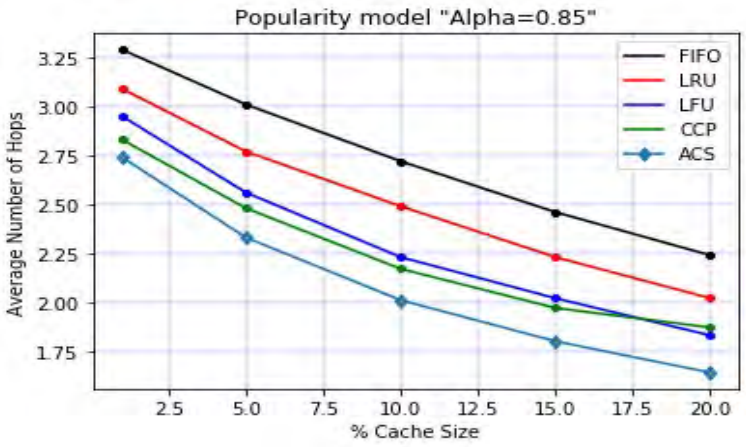


Figure 5.11: Average Number of Hops on Different Cache Sizes

5.7 Summary

This chapter introduced a Content Update Mechanism (CUM) for named data networking under ICN. The chapter presented the CUM as part of an Alternative Caching Strategy (ACS). CUM depended on the highest of both priority and popularity. It is observed from the results of CUM that a significant decline in the average number of hops required to get the contents from the router. While the ACS can increase cache hit rate, and minimize the average number of hops. The next Chapter will explain the whole results of the ACS.

CHAPTER SIX
SIMULATION RESULTS

The previous chapter introduced the alternative caching strategy consisting of two mechanisms, CRM (for content replacement) and CUM (for content update), in NDN. In addition, the previous chapter also discussed two access methods, TOPSIS and SAW. The TOPSIS method is utilized for content eviction, while SAW is implemented for update content caching. This chapter details the next phase of this research by presenting the performance evaluation of the alternative caching strategy, as well as its comparison with conventional caching strategies which include FIFO, LRU, LFU, and CCP. The chapter starts with introducing the simulation environments for ICN caching in Section 6.1, while the performance evaluation results of the alternative caching strategy in comparison with the selected conventional strategies is discussed in Section 6.2. The general discussion on simulation results related to the technical factors is outlined in Section 6.3, with a summary of the chapter given in Section 6.4.

6.1 Simulation Environments

The NDN concept has become very popular since its introduction in 2010 [14] due to its novel feature of being a clean slate approach to a new Internet architecture. Thus, it has garnered much attention as a research topic with quite a number of new proposals presented in renowned conferences and published in reputable journals. Few of these proposals deal with routing schemes [165], congestion control [166, 167], and content namespace [168], with a larger number focusing on the in-network caching strategies, as caching is one of the major and critical issues in the NDN architecture.

With respect to caching strategies, usually researchers evaluate their design policies according to their simulation scenarios and choose their configuration parameters at

their own choice, which make it unfeasible to compare the difference among caching strategies. The primary goal of this thesis is concerned with the evaluation of the effect of caching on future architectures. In order to achieve this, an investigation of the factors that will affect the employment of caching, and also understanding how these factors relate to the actual quantitative impact of caching is conducted.

With respect to caching strategies, researchers usually evaluate their design policies according to their simulation scenarios and also define their configuration parameters at their discretion. This unique design and definition makes it infeasible to compare the difference among caching strategies. Thus, the focus of researchers in introducing adequate evaluation approaches for caching strategies. Various studies have designed quite a number of simulation scenarios, proposed for the evaluation of NDN caching strategies. Specifically, this research considers four main simulation parameters that provides support to the overall network flow as documented in [[169], which includes, network topology, user request distribution, cache size, and catalog size. In order to quantify the effects of caching, an investigation on the adequate definition of the metrics will be conducted.

6.1.1 Network Topology

Various studies as regards the subject matter of internet topology have made findings showing that its topology structure follows a hierarchical domain structure, having stub and transit domains, with the stub nodes having attachments from the end user nodes [170]. The individual domain is made up of interconnected nodes sharing routing information, with the node-to-node connections staying within the specified set of nodes in the domain. Whereas, for transport domains, there is an effective interconnection with the stub domains such that there is no need for direct routing between the sub domains, as the connection between sub domains and transit domains

is usually via a gateway node.

Considering NDN research approaches in particular, there arise the need for verification and validation on the universal widely used network topologies, with the literature leading researchers in NDN using several topologies to test the efficiency of each strategy. The popular topologies are GEANT (Pan-European and Educational network) [171], WIDE (Japanese Educational network), TISCALI (Italian telecommunications company), and GARR (Italian academic and scientific community). These topologies were also tested on the study by [90, 172]. In other research, Rossi et al., [169, 173] using a total of 4 topologies namely: Level 3 topology, TIGER, DTelekom and Abilene, while the recent study by Bernadini in [174, 175] used the TIGER, GEANT, Abilene, DTelekom and TREE topologies.

This study will be subjecting the newly developed strategy in this thesis to three distinct and carefully selected topologies for implementation and simulation comparisons. The representation of the three selected topologies which includes the Grid, Abilene network, and Tree network topologies, are presented in Figure 6.1.

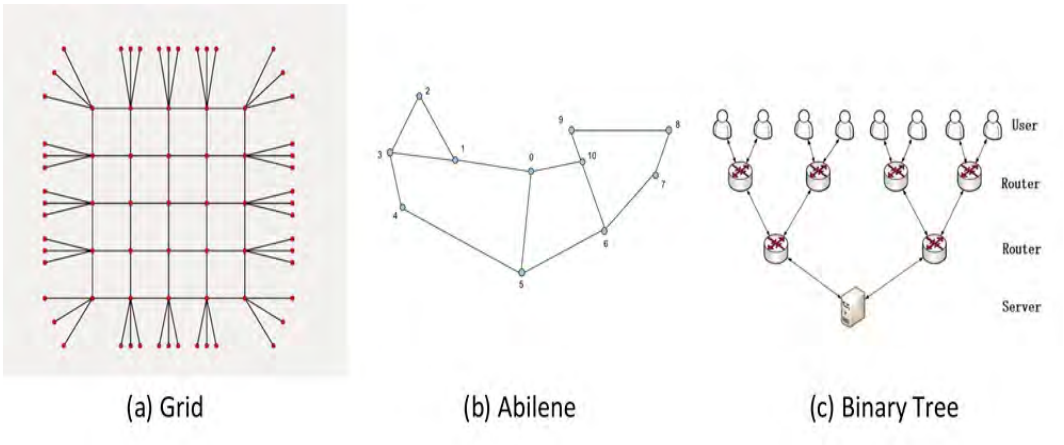


Figure 6.1: ISP-Level Network Topologies [174]

6.1.1.1 Grid Topology

The topology is shown in Figure 6.1(a). which is a topology of a twenty five-grid structure. The number of users and servers in this topology is just 45 and 3 respectively, with the users being able to reach the servers through multiple links [176].

6.1.1.2 Abilene Network Topology

This topology is sometimes referred to Internet2 network topology. The sole objective of designing the Abilene topology is to transfer data of large sizes across nodes. The development of this topology was suitable for comparing and testing simulation parameters, recording cache hits and observing how efficient the cache managements are. The heterogeneous nature of Abilene recognizes eleven (11) station sites as nodes (see Figure 6.1(b)).

6.1.1.3 Tree Network Topology

The binary tree structure is among the commonly and widely used network set-ups. Figure 6.1 (c) depicts tree topology, with a simple head-node (parent-child relationship). In this topology, there is just one server with multiple users, with the users having just a single link with which they can reach the server [177].

6.1.2 Request Arrival distribution

Looking into studies about user internet traces that were initially conducted from the 90s, majority of the work done with respect to developing a model for user internet requests have assumed that the requests by users are independent and follow a Zipflike distribution [178, 179]. A group of researchers argue that an item's individual popularity is proportional to a $\frac{1}{i}$ frequency, where This suggests that the α $0 < \alpha <= 1$. Note that a large percentage of the total requests is made up of top few popular items concurring with previous trace results which show that the top 1 percent of

the documents account for a range of twenty to thirty-five percent of all requests. This collaborative study by the researcher Breslau acknowledged that the assumption that requests are independent is obviously an oversimplification, as this simplification ignores the impact of spatial and temporal locality [180, 181].

For example, if we look at the rise in a viral video, aspects of these localities come into play. Specifically in the social media environment, it is observed that in a small period of time, people can influence those spatially close to and around them. The increase of population of people becoming influenced by and interconnected to each other, implies that more weight should be given to these localities. To this end, a model where user requests are stochastically generated for the population of a simulated trace was presented by Garcia et al [182] with the introduction of an additional locality factor using a Poisson process.

Conventionally, when considering traditional networks, the arrival of content is assumed to follow a Poisson distribution. Although, it has been observed that it is more practical to assume a modelling using Heavy Tail Distribution for content arrival, as the contents arriving in modern day traffic at any node have higher correlation. Likewise, it has also been observed in the literature that a better approximation of the arrival of request at any router can be achieved using Mandelbrot-Zipf (MZipf) distribution [179, 183], with the probability of accessing an object at rank r be defined mathematically as follows.

$$p(r) = \frac{H}{(r+q)^{\alpha}}$$

where, $H = \sum_{c=1}^{N} \frac{1}{(c+q)^{\alpha}}$

α is the skewness factor controlling the curve's slope, and $q \geq 0$ denotes the plateau factor which decides the curve's flatness [181]. In the system model introduced in this research,it is considered that the arrival of requests follow a MZipf distribution.

6.1.3 Cache Size

The cache size specifies the available space in every node for the temporal storage of content objects. The cache size at each router, relative to the total number of possible objects, will affect the overall performance gains. It is either denoted with absolute value or a ratio with respect to the catalog size (20 elements or 0.02 of the catalog size) [184]. In existing studies, various cache sizes have be utilized, such as Ran et al. [91] using cache size 20 to 400Kbits, Zhang et al.[157] using cache size 100 to 500, Dron et al.[94] using cache size 1MB to 10MB, Bilal et al. [95] using cache size 100 to 500, among many others. With consideration of the element values alongside the cache sizes, Sung et al. [185] used 2,500 elements each of size 2GB with cache space set to 1TB, while Wang et al. [186] simulated scenarios using cache capacity of 50Mb and 200Mb with the content size of 100Mb and 300Mb respectively. Bernardini et al. [184] evaluated their experiments with cache size ranging between 1 and 10^3elements, corresponding to 10^{-6} to 10^{-3} as cache size ratio, while Sun et al. [49] used cache size 1 GB to 1TB.

6.1.4 Catalog Size

The catalog represents the total number of contents in a network. In existing literature, different values have been used for the catalog size as seen in studies by Chai et al. [22] with a value of 10^3. while Rossi et al. [169] considered a YouTube-like catalog, which includes 10^8objects with an average of 10MB chunk. Bilal and Kang [96] used a catalog size of 10^4, with Sun et al. [49] taking consideration of a PPTV catalog, which includes 10^9objects. Although, some researchers have not given detail about

the catalog size, but in certain studies [148, 172], the authors have mentioned that for all catalog, they generate 10^5 requests.

6.1.5 Simulation Setup

This study conducted its simulations in ndnSIM 2.7, which is used for evaluating four caching strategies (FIFO, LRU, LFU, CCP), with LCE cache placement policy, which is in line with [91, 157]for all NDN nodes. For modeling of content requests in ndnSIM 2.7, the Zipf distribution function [157, 158] is used in its Zipf generalized form; Maldelbrot-Zipf (MZipf) [159, 160, 161]. Specifically, the simulations in the ndnSIM 2.7 for this study has used three ISP-level topologies as earlier mentioned. These topologies include Abilene, Tree, and Grid topologies [162, 163] for the evaluation of these strategies (see Table 6.1).

parameter being defined to vary between 0.65, 0.7 and 0.85. A range from 100 to 100,000 elements, that is 1GB to 1TB, was used for the cache size, which gives the information of the vacant space in each node that is available to temporarily store content objects, and the catalog representing the total value of all the contents in a network was chosen as 10^6 elements.

Table 6.1: *Simulation Scenario*

Parameter	Description
Cache Size	$100 - 100,000$ elements
Catalog Size	10^6 elements
α Parameter	0.65, 0.7, 0.85
Topology	Abilene, Tree, Grid5x5
Placement Policy	LCE
NDN Project Topology	NDN Project
Simulator	ndnSIM 2.7
No. of Simulation	10 runs
Average Bit Rate	100 Mbps
Simulation Time	One day

6.2 Simulation Results

In this section we present our simulated work in line with the following objectives: Cache Hit Ratio, Throughput, Server Load, Bandwidth and Average Number of Hops.

A- Cache Hit Ratio: Generally, hit ratio refers to the percentage of content requests that are satisfied by caches being implemented within the network. Thus, cache hit ratio returns the amount in the value of an average of available content hit (found) when requests are sent. It has been affirmed by several studies in the field of networking and database that better cache hit ratio implies better performance of the network. This metric display the desirable ability of the caching strategy to moderate the number of content copies that are redundant (in terms of searching), and how often the data is found. The major purpose of the cache has been to gather together recently used data in a small area (this area is faster than the slow and large primary storage area), to be accessed in the future, hence resulting in quicker access times for the system.

Basically, The *CacheHitRatio* depicts the ratio of the number of cache hits to the number of misses. Its value varies depending on the nature of the cache, and this value has been observed to be from 60% to greater than 99%. The equation to calculate cache hit ratio is given below:

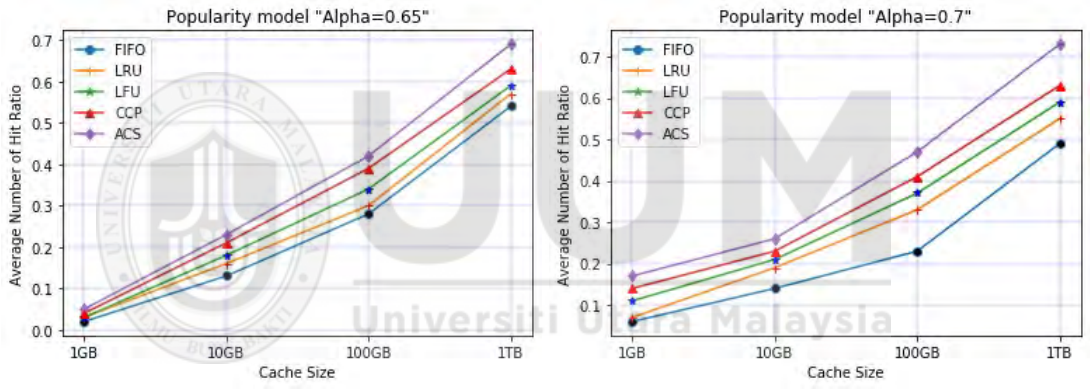
$$CacheHitRatio(CHR) = \frac{C_H}{C_H + C_M} \tag{6.1}$$

C_H is the number of cache hits and C_M is the number of cache misses. The conversion of CHR to a relative performance metric involves obtaining an estimate of the relative cost of a cache hit and a cache miss. Mathematically, CHR is fundamentally

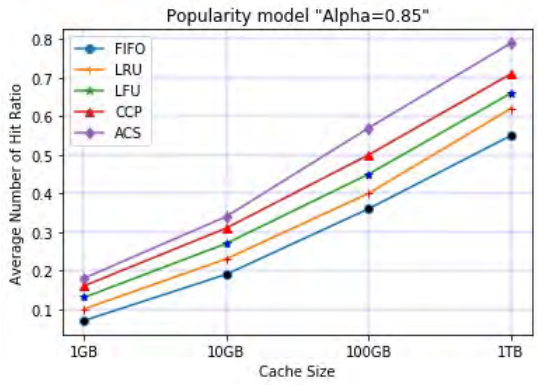
logarithmic, and this indicates that the closer CHR gets to 100%, its gains becomes exponentially greater.

1- Cache Hit Ratio with Abilene Topology

The ratio of cache hit with respect to the Abilene topology is shown in Figure 6.2. In low popularity scenario (when α = 0.65), the ratio of cache hit is low on cache sizes 100 to 100,000 (1GB to 1TB) elements as shown in Figure 6.2(a). Whereas an increase of the probability parameter a from 0.65 to 0.7 (Figure 6.2(b)) and 0.85 (Figure 6.2(c)), with an increase of the cache size, showed a higher hit ratio being achieved.



(a) *(b)*



(c)

Figure 6.2: Cache Hit on Abilene Topology

Table 6.2: *Hit Ratio on Different Popularity Model α and Cache Size by Abeline Topology*

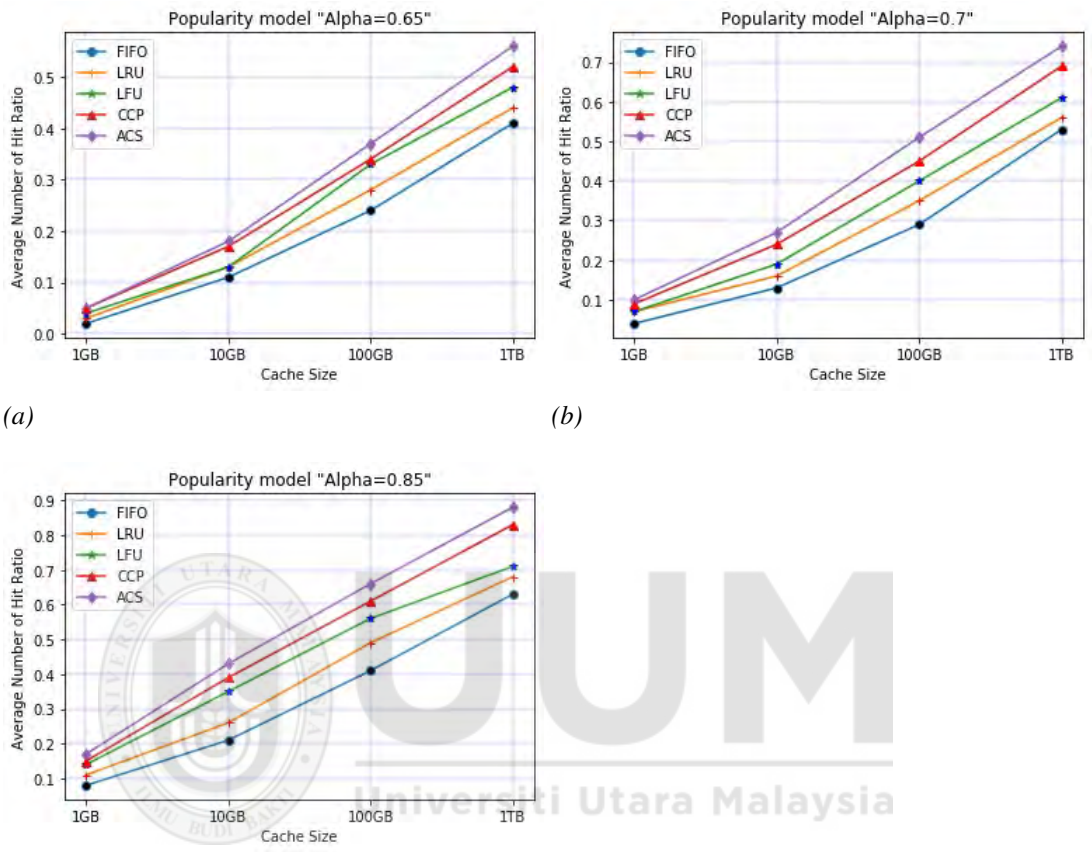
Topology	Popularity model α			Cache Size				Cache Hit Ratio				
	0.65	0.7	0.85	1 GB	10 GB	100 GB	1 TB	FIFO	LRU	LFU	CCP	ACS
	X			X				0.02	0.03	0.03	0.04	0.05
	X				X			0.13	0.16	0.18	0.21	0.23
	X					X		0.28	0.3	0.34	0.39	0.42
	X						X	0.54	0.57	0.59	0.63	0.69
Abilene		X		X				0.06	0.07	0.11	0.14	0.17
		X			X			0.14	0.19	0.21	0.23	0.26
		X				X		0.23	0.34	0.37	0.41	0.47
		X					X	0.49	0.52	0.62	0.69	0.73
			X	X				0.07	0.10	0.13	0.16	0.18
			X		X			0.19	0.23	0.27	0.31	0.34
			X			X		0.36	0.40	0.45	0.50	0.57
			X				X	0.55	0.62	0.66	0.71	0.79

In addition to the results in Figure 6.2, the productivity of the diverse cache strategies for defined scenarios having varying cache sizes with respect to the determined cache hit proportion is detailed in Table 6.2. The ACS, over the CCP, LFU, LRU, and FIFO for all various cache sizes obviously shows essentially the best outcomes. Bearing in mind that the cache hit proportion is generally adopted to give a decent understanding of the caching performance of the network, when the data packets are distributed in a consistent way which is dependent on the dimensionality of the caches. Therefore, the small difference is a sign in the cache hit ratio.

2- Cache Hit Ratio with Tree Topology

The ratio of cache hit with respect to Tree topology is shown in Figure 6.3. When α = 0.65, which depicts a low popularity scenario, the cache hit ratio is low on cache sizes 100 to 100,000 (1GB to 1TB) elements (Figure 6.3(a)). Whereas, a higher cache

hit ratio is observed when the probability parameter a is increased from 0.65 (Figure 6.3(a)) to 0.7 (Figure 6.3(b)) and 0.85 (Figure 6.3(c)), with a simultaneous increase in the cache sizes.



(a) *(b)*

(c)

Figure 6.3: Cache Hit on Tree Topology

Figure 6.3 in addition to the information given in Table 6.3 shows the productivity of the diverse cache strategies for given scenario with different cache sizes based on the determined cache hit proportion. Similar to the observation of the cache hit ratio with the Abilene topology above, the ACS, over the CCP, LFU, LRU, and FIFO for all various cache sizes obviously shows essentially the best outcomes. Noting that the adoption of the cache hit proportion is generally aimed at giving a holistic understanding of the network's caching performance when the data packets are distributed consistently and dependent on the dimensionality of the caches. This

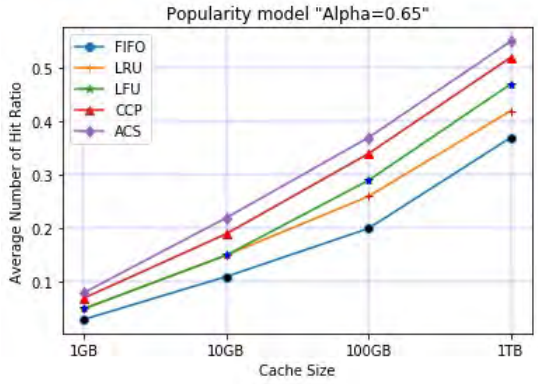
also implies that the small difference is a sign in the cache hit ratio.

Table 6.3: *Hit Ratio on Different Popularity Model* ∝ *and Cache Size by Tree Topology*

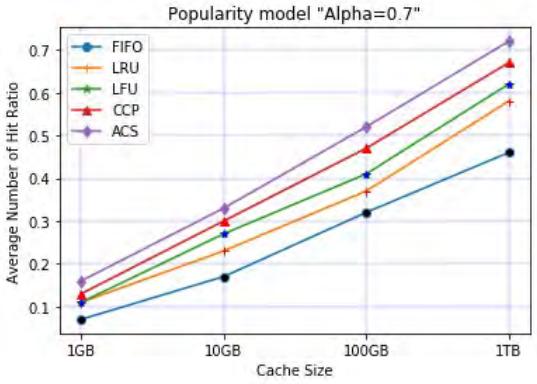
Topology	Popularity model ∝			Cache Size				Cache Hit Ratio				
	0.65	0.7	0.85	1 GB	10 GB	100 GB	1 TB	FIFO	LRU	LFU	CCP	ACS
Tree	X			X				0.02	0.03	0.04	0.05	0.05
	X				X			0.11	0.13	0.13	0.17	0.18
	X					X		0.24	0.28	0.33	0.34	0.37
	X						X	0.41	0.44	0.48	0.52	0.56
		X		X				0.04	0.07	0.07	0.09	0.10
		X			X			0.13	0.16	0.19	0.24	0.27
		X				X		0.29	0.35	0.40	0.45	0.51
		X					X	0.53	0.56	0.61	0.69	0.74
			X	X				0.08	0.11	0.14	0.15	0.17
			X		X			0.21	0.26	0.35	0.39	0.43
			X			X		0.41	0.49	0.56	0.61	0.66
			X				X	0.63	0.68	0.71	0.83	0.88

3- Cache Hit Ratio with Grid Topology

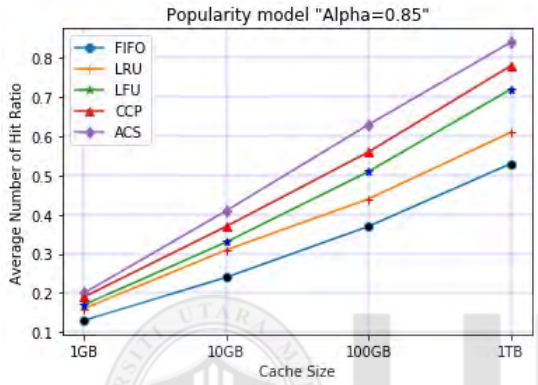
The ratio of cache hit with respect to Grid topology is shown in Figure 6.4. Similar to the results obtained for the cache hit ratio of Abilene and Tree topologies, a higher cache hit ratio will be achieved when the probability parameter a is increased from a low popularity scenario (α = 0.65 with cache size of 100 to 100,000, that is 1GB to 1TB, elements) in Figure 6.4(a) to a higher value of 0.7 (Figure 6.4(b)) and 0.85 (Figure 6.4(c)), with an increase of the cache size as well.



(a)



(b)



(c)

Figure 6.4: Cache Hit on Grid 5x5 Topology

Table 6.4: *Hit Ratio on Different Popularity Model* α *and Cache Size by Grid Topology*

Topology	Popularity model α			Cache Size				Cache Hit Ratio				
	0.65	0.7	0.85	1 GB	10 GB	100 GB	1 TB	FIFO	LRU	LFU	CCP	ACS
5x5 Grid	X			X				0.03	0.05	0.05	0.07	0.08
	X				X			0.11	0.15	0.15	0.19	0.22
	X					X		0.2	0.26	0.29	0.34	0.37
	X						X	0.37	0.42	0.47	0.52	0.55
		X		X				0.07	0.11	0.11	0.13	0.16
		X			X			0.17	0.23	0.27	0.30	0.33
		X				X		0.32	0.37	0.41	0.47	0.52
		X					X	0.46	0.58	0.62	0.67	0.72
			X	X				0.13	0.16	0.17	0.19	0.20
			X		X			0.24	0.31	0.33	0.37	0.41
			X			X		0.37	0.44	0.51	0.56	0.63
			X				X	0.53	0.61	0.72	0.78	0.84

The productivity of the different cache strategies for given scenarios, defined with ranging values of cache sizes with respect to the determined cache hit proportion is similarly detailed in Table 6.4. The results shown in Figure 6.4 and Table 6.4 interprets that the ACS obviously shows essentially the best outcomes in comparison to the conventional CCP, LFU, LRU, and FIFO for all various cache sizes. Not forgetting that the small difference is a sign in the cache hit ratio. This is due to the fact that the computation of the cache hit proportion, when the data packets are distributed in a consistent way dependent on the dimensionality of the caches, is adopted in order to obtain a clearer understanding of the caching performance of the network.

B- Server Load

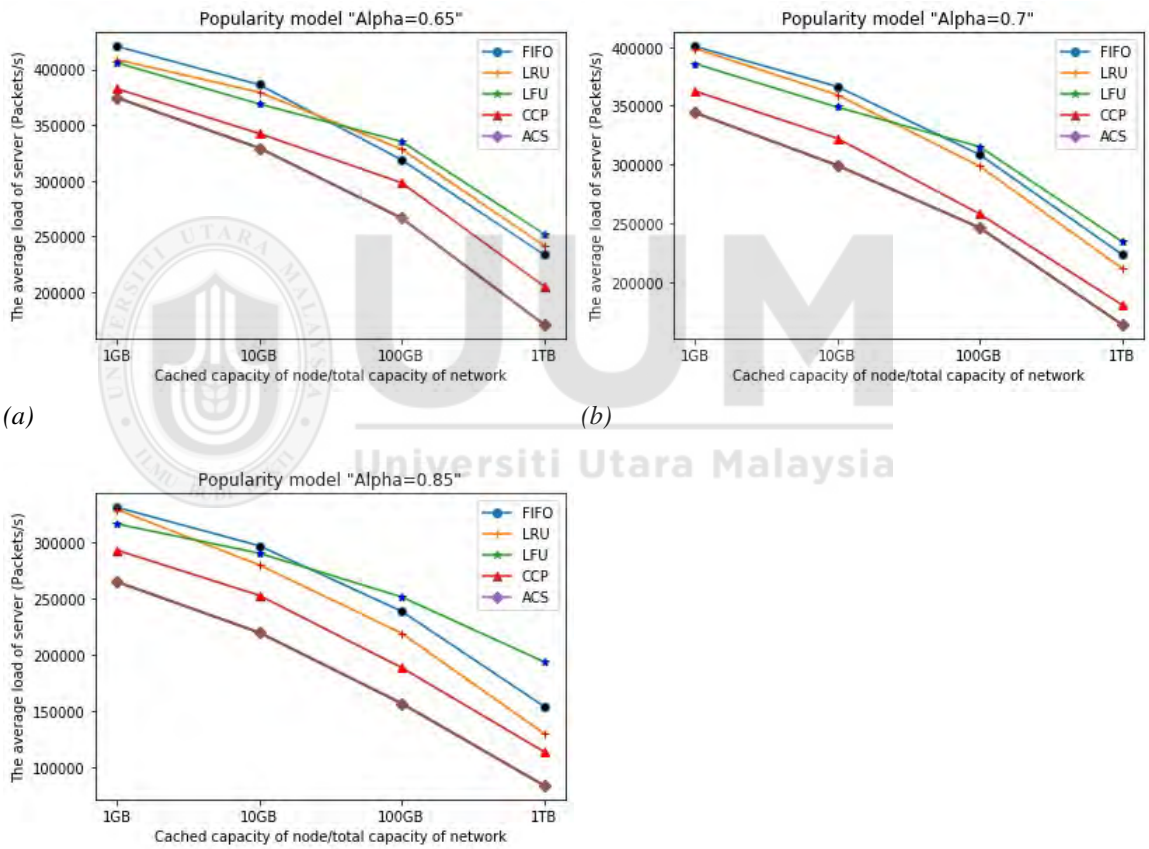
Server load (SL) is defined as the traffic quantity carried by the server. It could likewise be referred to as a count of the packets received during a measurement period, represented in seconds. Its expression is in number format, and it represents the number of processes standing in chain for seeking processor. The relation follows that the smaller the server loads, the better the network performance, although all processes vary one to another. The server load is computed using the following equation:

$$SL = \frac{NumCachedReq}{R} \tag{6.2}$$

In Equation 6.2, *NumCachedReq* denotes the number of requests that are served from some in-network cache, while the variable R denotes all requests. For low-priority processes, in the appearance of a new server request, prompt handling and immediate response can be initiated without experiencing any delay while the lower priority processes wait.

1- Average Server Load with Abilene Topology

The average server load in comparison to the cache capability is shown in Figure 6.5. In low popularity scenario (when α =0.65), the average server load decreases on cache sizes 100 to 100,000 (1GB to 1TB) elements as shown Figure 6.5(a). Whereas, an increase of the probability parameter a from 0.65 to 0.7 (Figure 6.5(b)) and 0.85 (Figure 6.5(c)), with an increase of cache size, showed a decreased load being achieved.



(a) *(b)*

(c)

Figure 6.5: Average Server Load on Abilene Topology

Table 6.5: *Average Server Load on Different Popularity Model α and Cache Size by Abilene Topology*

Topology	Popularity model α			Cache Size				Average Server Load				
	0.65	0.7	0.85	1 GB	10 GB	100 GB	1 TB	FIFO	LRU	LFU	CCP	ACS
Abilene	X			X				420512	408641	405781	382563	374464
	X				X			386143	379302	368820	342307	329349
	X					X		318733	328369	335187	298192	266432
	X						X	233667	241956	252253	205460	171124
		X		X				400520	398660	385784	362514	344162
		X			X			366340	359352	348871	322320	299389
		X				X		308550	298702	315229	258100	246404
		X					X	223647	211945	234657	180837	164170
			X	X				330574	328662	315826	292534	264411
			X		X			296457	279380	289840	252373	219327
			X			X		238244	218737	250951	188161	156414
			X				X	153696	129702	193401	113491	83468

In addition to the results in Figure 6.5, the productivity of the diverse cache strategies for defined scenarios having varying cache sizes with respect to the determined average server load is detailed in Table 6.5. The ACS, over the CCP, LFU, LRU, and FIFO for all various cache sizes obviously shows the best outcomes essentially. Bearing in mind that the average server load is generally adopted to give a decent understanding of the caching performance of the network, when the data packets are distributed in a consistent way dependent on the dimensionality of the caches. Therefore, the small difference is a sign in the average server load.

2- Average Server Load with Tree Topology

The average server load compared with the capability of the cache is shown in Figure 6.6. When α = 0.65, which depicts a low popularity scenario, the average server load is decreased on cache sizes 100 to 100,000 (1GB to 1TB) elements (Figure 6.6(a)).

Whereas, the expected decreased load is achieved when the probability parameter a is increased from 0.65 (Figure 6.6(a)) to 0.7 (Figure 6.6(b)) and 0.85 (Figure 6.6(c)), with a simultaneous increase in the cache sizes.

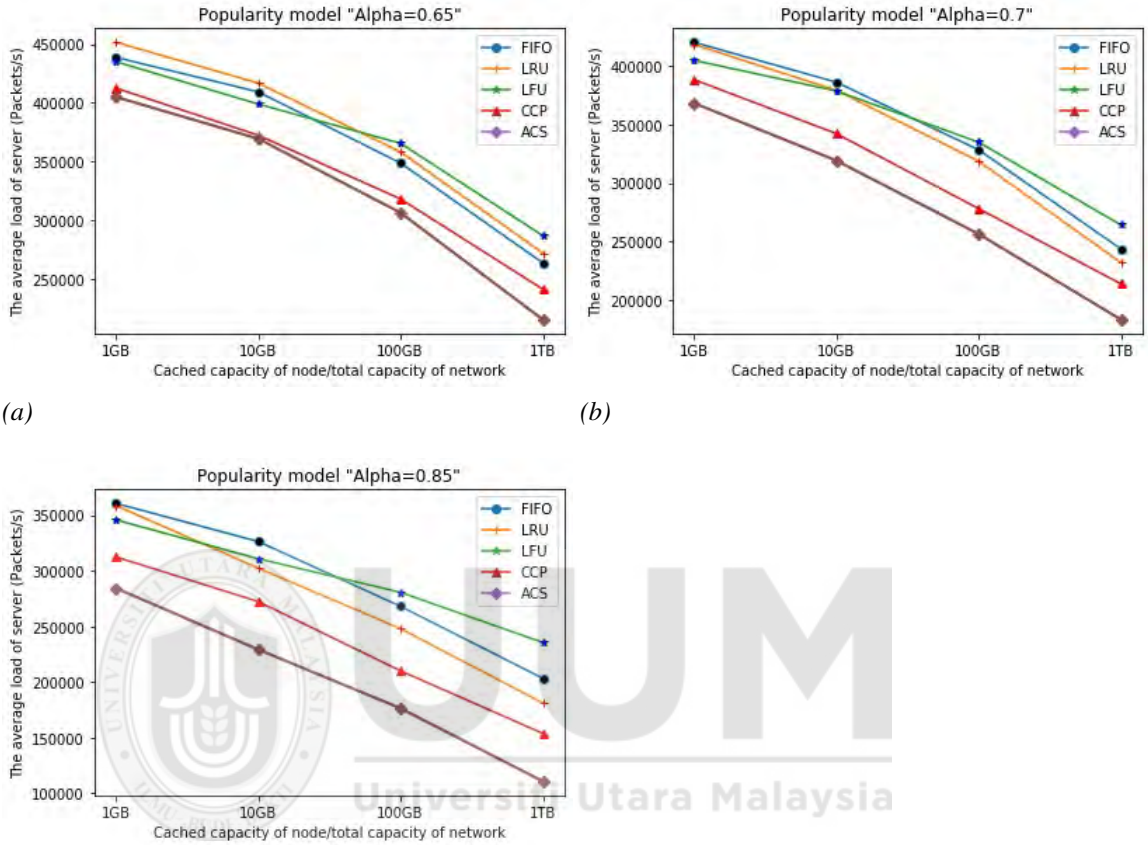


Figure 6.6: Average Server Load on Tree Topology

Figure 6.6 in addition to the information given in Table 6.6 shows the productivity of the diverse cache strategies for given scenario with different cache sizes based on the determined average server load. Similar to the observation of the average server load with the Abilene topology above, the ACS, over the CCP, LFU, LRU, and FIFO for all various cache sizes obviously shows the best outcomes essentially. Noting that the adoption of the average server load is generally aimed at giving a holistic understanding of the network's caching performance when the data packets are distributed consistently and dependent on the dimensionality of the caches. This

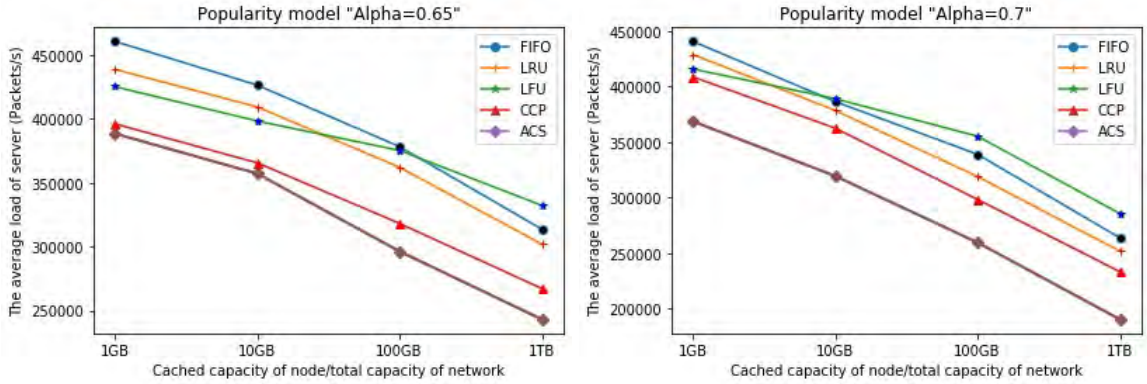
also implies that the small difference is a sign in the average server load.

Table 6.6: *Average Server Load on Different Popularity Model α and Cache Size by Tree Topology*

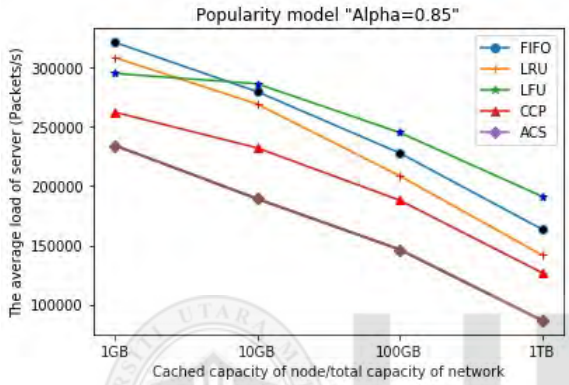
Topology	Popularity model α			Cache Size				Average Server Load				
	0.65	0.7	0.85	1 GB	10 GB	100 GB	1 TB	FIFO	LRU	LFU	CCP	ACS
	X			X				438663	451558	435023	412551	404996
	X				X			409314	416753	398899	372371	369382
	X					X		348708	358147	365454	318100	306403
	X						X	263647	271960	286841	241249	215441
Tree		X		X				420509	418601	405012	388563	368439
		X			X			386332	379398	378811	342380	319304
		X				X		328487	318746	335014	278129	256479
		X					X	243622	231977	264181	213795	183206
			X	X				360581	358626	345762	312520	284400
			X		X			326307	302396	310840	272369	229300
			X			X		268069	248044	280634	210108	176465
			X				X	203391	181138	235532	153861	110806

3- Average Server Load with Grid Topology

The average server load in comparison to the cache capability is shown in Figure 6.7. Similar to the results obtained for the average server load in the Abilene and Tree topologies, a decreased load is achieved when the probability parameter a is increased from a low popularity scenario ($\alpha = 0.65$ with cache sizes 100 to 100,000, that is 1GB to 1TB elements) in Figure 6.7(a) to a higher value of 0.7 (Figure 6.7(b)) and 0.85 (Figure 6.7(c)), with an increase of the cache size as well.



(a) *(b)*



(c)

Figure 6.7: Average Server Load on Grid Topology

Table 6.7: *Average Server Load on Different Popularity Model α and Cache Size by Grid Topology*

Topology	Popularity model α			Cache Size				Average Server Load				
	0.65	0.7	0.85	1 GB	10 GB	100 GB	1 TB	FIFO	LRU	LFU	CCP	ACS
5x5 Grid	X			X				460562	438655	425138	396043	388423
	X				X			426331	409308	398254	365563	357104
	X					X		378201	362056	375241	318147	296325
	X						X	313626	301948	332206	267231	243117
		X		X				440533	428681	415622	408544	368459
		X			X			386248	378302	388890	362380	319307
		X				X		338792	318793	355086	298162	259412
		X					X	263605	251642	285479	233084	190211
			X	X				321519	308612	295201	262536	234439
			X		X			279823	269341	286540	232327	189301
			X			X		228054	208752	245357	188159	146497
			X				X	163692	141909	191427	126933	86602

The productivity of the different cache strategies for given scenarios, defined with ranging values of cache sizes with respect to the determined average server load is similarly detailed in Table 6.7. The results shown in Figure 6.7 and Table 6.7 interprets that the ACS obviously shows the best outcomes essentially in comparison to the conventional CCP, LFU, LRU, and FIFO for all various cache sizes. Not forgetting that the small difference is a sign in the average server load. This is due to the fact that the computation of the average server load, when the data packets are distributed in a consistent way dependent on the dimensionality of the caches, is adopted in order to obtain a clearer understanding of the caching performance of the network.

C- Throughput

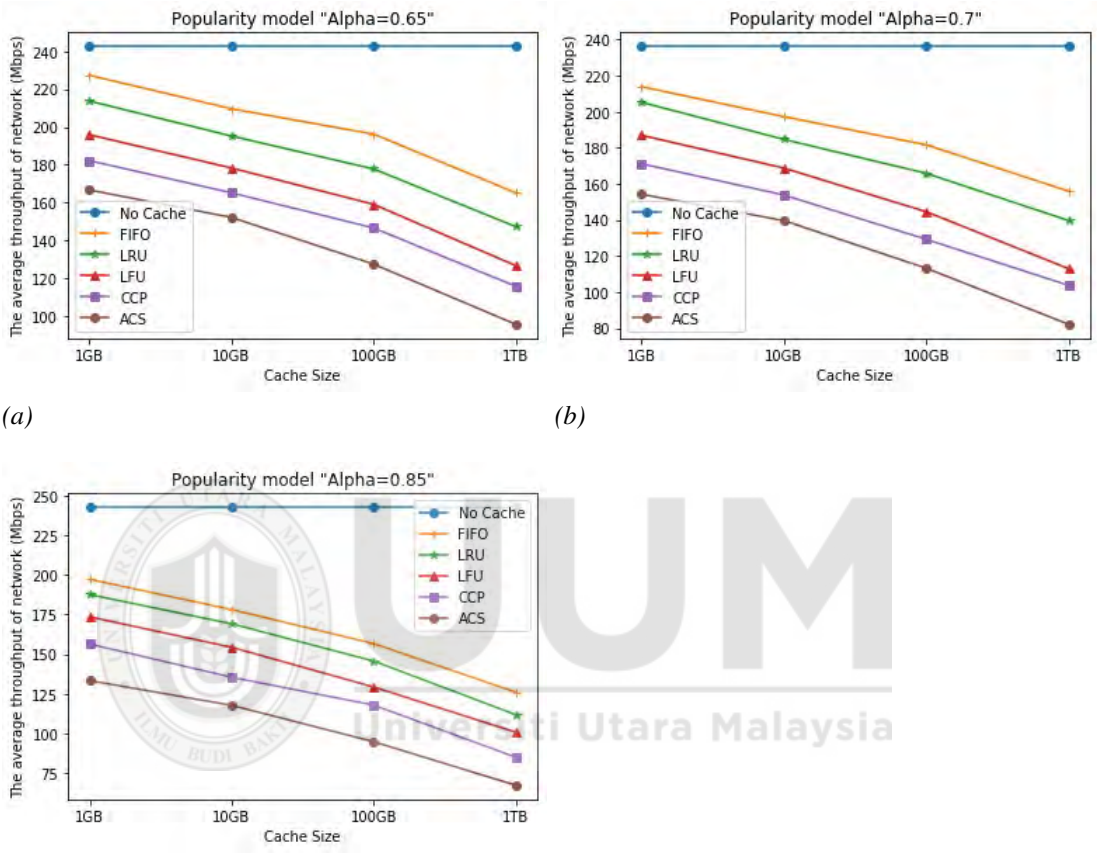
Throughput is the maximum rate for the processing of a certain concept or simply put, the maximum rate of production. In communication networks context, throughput or network throughput refers to the successful message delivery rate over a communication channel. The data these messages belong to can pass through a certain network node, or may be delivered over a link. The units for measuring throughput is usually bits per second (bit/s or bps), bps), and also in data packets per second (p/s or pps) for certain cases. Throughput is calculated using the following equation:

$$Throughput = \frac{Size\,of\,information}{Time} \tag{6.3}$$

1- Average Throughput with Abilene Topology

The average throughput in comparison to the cache capability is shown in Figure 6.8. In low popularity scenario (when $\alpha = 0.65$), the average throughput decreases on cache

sizes 100 to 100,000 (1GB to 1TB) elements as shown in Figure 6.8(a). Whereas, an increase of the probability parameter a from 0.65 to 0.7 (Figure 6.8(b)) and 0.85 (Figure 6.8(c)), with an increase of cache size, showed a decreased average throughput being achieved.



(a) *(b)*

(c)

Figure 6.8: AverageThroughput on Abilene Topology

In addition to the results in Figure 6.8, the productivity of the diverse cache strategies for defined scenarios having varying cache sizes with respect to the determined average throughput is detailed in Table 6.8. The ACS, over the CCP, LFU, LRU, FIFO and No Cache for all various cache sizes obviously shows the best outcomes essentially. Bearing in mind that the average throughput is generally adopted to give a decent understanding of the caching performance of the network, when the data packets are distributed in a consistent way dependent on the dimensionality of the caches.

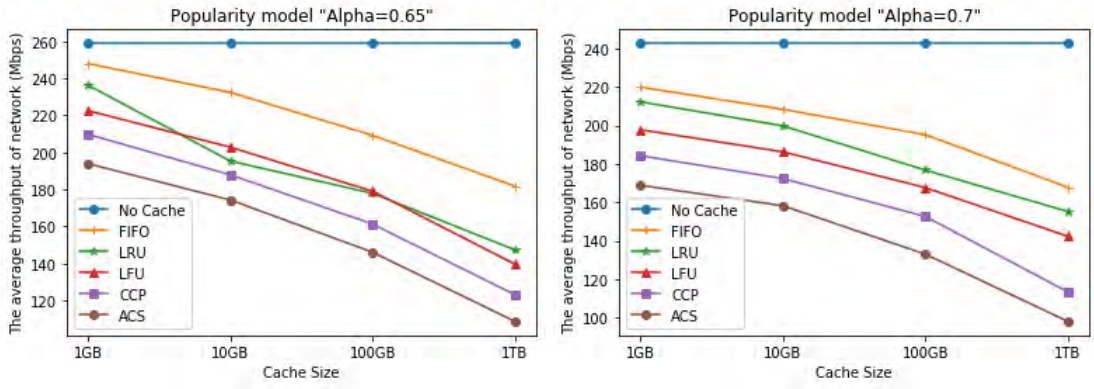
Therefore, the small difference is a sign in the average throughput.

Table 6.8: *Average Throughput on Different Popularity Model α and Cache Size by Abilene Topology*

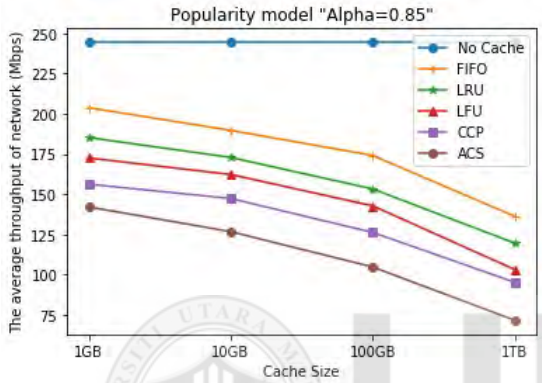
Topology	Popularity model α			Cache Size				Average Throughput					
	0.65	0.7	0.85	1 GB	10 GB	100 GB	1 TB	No Cache	FIFO	LRU	LFU	CCP	ACS
	X			X				242.6	227.3	213.7	195.9	182.2	166.7
	X				X			242.6	209.6	195.2	178.3	165.2	152.2
	X					X		242.6	196.2	177.8	159.1	146.6	127.5
	X						X	242.6	165.2	147.3	126.7	115.8	95.7
Abilene		X		X				236.2	213.8	205.2	186.9	171.1	154.2
		X			X			236.2	197.2	184.6	168.7	153.8	139.6
		X				X		236.2	181.6	165.8	144.5	129.2	113.2
		X					X	236.2	155.9	139.6	112.9	103.7	82.1
			X	X				242.6	197.2	187.6	173.5	156.4	133.3
			X		X			242.6	178.1	169.1	154.3	135.4	117.7
			X			X		242.6	156.6	145.6	129.2	117.8	94.7
			X				X	242.6	125.9	111.7	100.7	84.9	67.2

2- Average Throughput with Tree Topology

The average throughput compared with the capability of the cache is shown in Figure 6.9. When $\alpha = 0.65$, which depicts a low popularity scenario, the average throughput is decreased on cache sizes 100 to 100,000 (1GB to 1TB) elements (Figure 6.9(a)). Whereas, the expected decreased average throughput is achieved when the probability parameter a is increased from 0.65 ((Figure 6.9(a))) to 0.7 (Figure 6.9(b)) and 0.85 (Figure 6.9(c)) with a simultaneous increase in the cache sizes.



(a) *(b)*



(c)

Figure 6.9: Average Throughput on Tree Topology

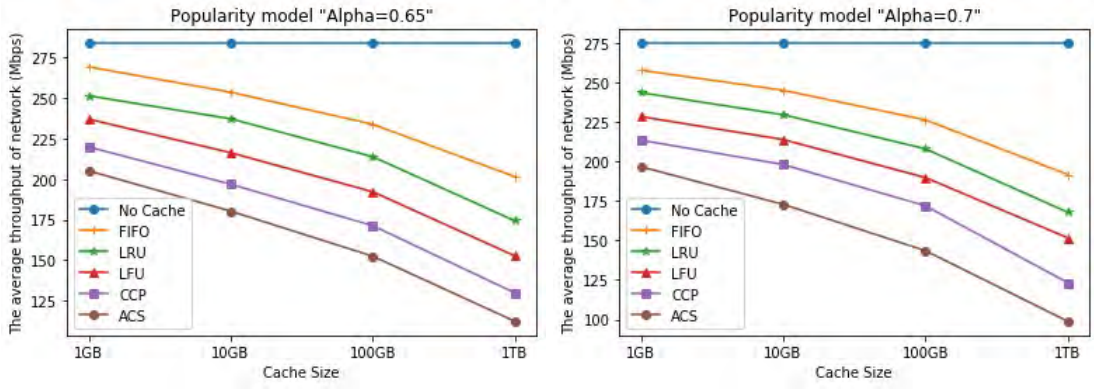
Table 6.9: *Average Throughput on Different Popularity Model α and Cache Size by Tree Topology*

Topology	Popularity model α			Cache Size				Average Throughput					
	0.65	0.7	0.85	1 GB	10 GB	100 GB	1 TB	No Cache	FIFO	LRU	LFU	CCP	ACS
Tree	X			X				258.8	247.8	236.1	222.4	209.6	193.8
	X				X			258.8	232.3	195.2	202.7	187.8	174.2
	X					X		258.8	209.1	177.8	178.9	161.2	146.0
	X						X	258.8	181.7	147.3	139.6	123.1	108.7
		X		X				242.6	219.8	212.2	197.7	184.2	168.8
		X			X			242.6	208.3	199.7	186.2	172.3	158.2
		X				X		242.6	195.1	176.7	167.6	152.6	133.0
		X					X	242.6	167.8	155.2	142.3	113.4	98.1
			X	X				244.3	203.7	185.2	172.5	156.2	142.1
			X		X			244.3	189.6	172.8	162.2	147.3	126.7
			X			X		244.3	174.0	153.2	142.7	126.2	104.7
			X				X	244.3	136.1	119.5	103.1	95.2	71.6

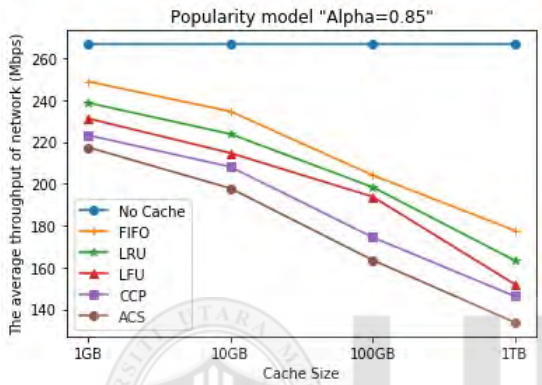
Figure 6.9 in addition to the information given in Table 6.9 shows the productivity of the diverse cache strategies for given scenario with different cache sizes based on the determined average throughput. Similar to the observation of the average throughput with the Abilene topology above, the ACS, over the CCP, LFU, LRU, FIFO and No Cache for all various cache size obviously shows the best outcomes essentially. Noting that the adoption of the average throughput is generally aimed at giving a holistic understanding of the network's caching performance when the data packets are distributed consistently and dependent on the dimensionality of the caches. This also implies that the small difference is a sign in the average throughput.

3- Average Throughput with Grid Topology

The average throughput in comparison to the cache capability is shown in Figure 6.10. Similar to the results obtained for the average throughput in Abilene and Tree topologies, a decreased average throughput is achieved when the probability parameter a is increased from a low popularity scenario (α = 0.65 with cache sizes 100 to 100,000, that is 1GB to 1TB elements) in Figure 6.10(a) to a higher value of 0.7 (Figure 6.10(b)) and 0.85 (Figure 6.10(c)), with an increase of the cache size as well. The productivity of the different cache strategies for given scenarios, defined with ranging values of cache sizes with respect to the determined average throughput is similarly detailed in Table 6.10. The results shown in Figure 6.10 and Table 6.10 interprets that the ACS obviously shows the best outcomes essentially in comparison to the conventional CCP, LFU, LRU, FIFO and No Cache for all various cache sizes.



(a) *(b)*



(c)

Figure 6.10: Average Throughput on Grid Topology

Table 6.10: *Average Throughput on Different Popularity Model α and Cache Size by Grid Topology*

Topology	Popularity model α			Cache Size				Average Hops Number					
	0.65	0.7	0.85	1 GB	10 GB	100 GB	1 TB	No Cache	FIFO	LRU	LFU	CCP	ACS
Grid	X			X				283.6	269.1	251.3	236.9	219.8	205.1
	X				X			283.6	253.5	237.1	216.2	196.8	180.2
	X					X		283.6	233.7	213.7	192.1	171.2	152.4
	X						X	283.6	201.4	174.2	152.6	129.8	112.3
		X		X				274.8	257.8	243.6	228.4	213.5	196.7
		X			X			274.8	245.1	229.6	213.9	198.1	172.9
		X				X		274.8	226.3	208.0	189.8	171.9	143.5
		X					X	274.8	191.7	167.8	151.6	123.2	98.8
			X	X				266.7	248.9	238.7	231.2	223.2	217.4
			X		X			266.7	234.6	223.8	214.7	208.1	197.8
			X			X		266.7	204.1	198.4	193.8	174.4	163.5
			X				X	266.7	177.6	163.3	151.9	146.2	133.7

Not forgetting that the small difference is a sign in the average throughput. This is due to the fact that the computation of the average throughput, when the data packets are distributed in a consistent way dependent on the dimensionality of the caches, is adopted in order to obtain a clearer understanding of the caching performance of the network.

D- Average Number of Hops

The average number of hops *AV* measures the reduction of the number of hops traversed for the satisfaction of a request in comparison to the number of hops that are required for content retrieval from the server. It is calculated as

$$AV = \frac{\sum_{i=1}^{N} \frac{\sum_{r=1}^{R} \frac{h_{ir}}{H_{ir}}}{R}}{N} \tag{6.4}$$

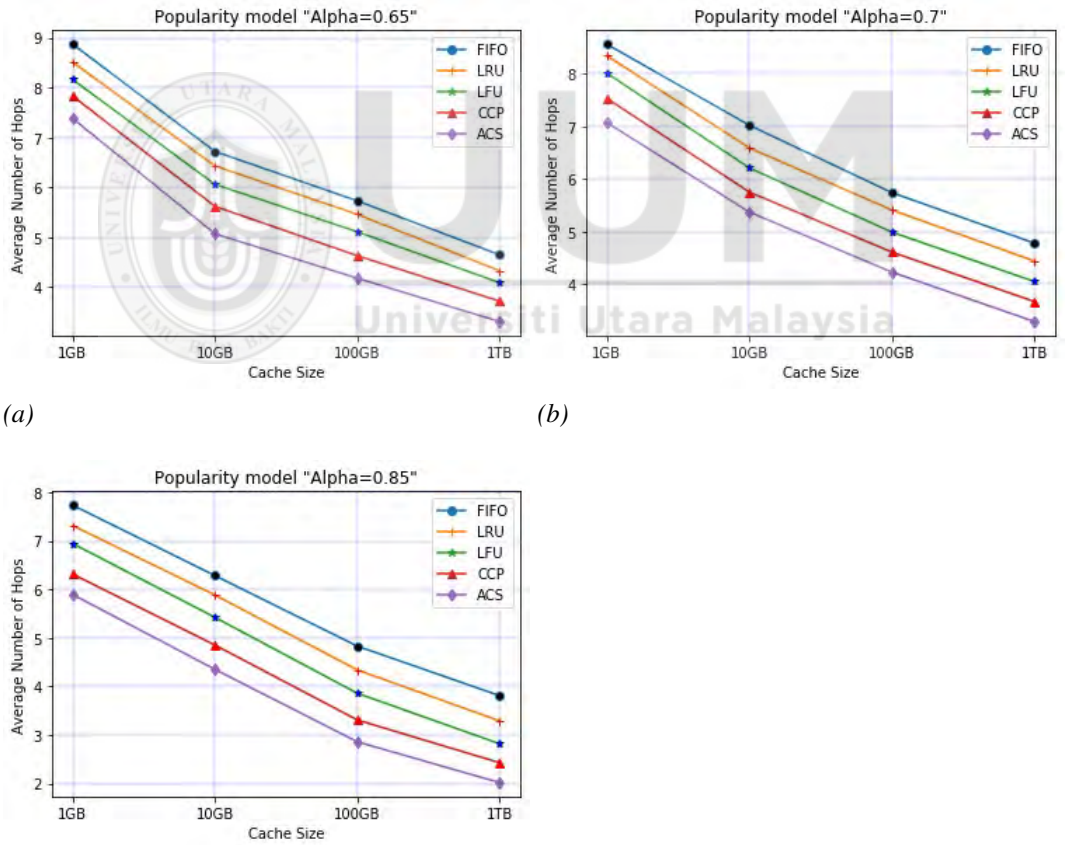
Where N denotes the number of consumers, and Rrepresents the number of requests created per consumer, h_{ir} is the number of hops that satisfies r from i to the cache , and H_{ir} is the number of hops from i to the producer.

1- Average Hop Number with Abilene Topology

The average hop number in comparison to the cache capability is shown in Figure 6.11. In low popularity scenario (when α = 0.65), the average hop number decreases on cache sizes 100 to 100,000 (1GB to 1TB) elements as shown in Figure 6.11(a). Whereas, an increase of the probability parameter a from 0.65 to 0.7 (Figure 6.11(b)) and 0.85 (Figure 6.11(c)), with an increase of cache size, showed a drop in the average

hop number.

In addition to the results in Figure 6.11, the productivity of the diverse cache strategies for defined scenarios having varying cache sizes with respect to the determined average hop number is detailed in Table 6.11. The ACS, over the CCP, LFU, LRU, and FIFO for all various cache sizes obviously shows the best outcomes essentially. Bearing in mind that the average hop number is generally adopted to give a decent understanding of the caching performance of the network, when the data packets are distributed in a consistent way dependent on the dimensionality of the caches. Therefore, the small difference is a sign in the average hop number.



(a) *(b)*

(c)

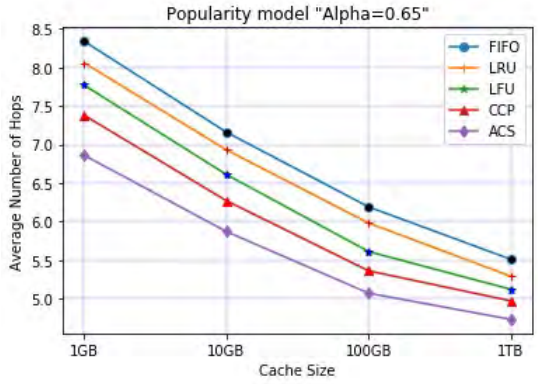
Figure 6.11: Average Hops Number on Abilene Topology

Table 6.11: *Average Hops Number on Different Popularity Model α and Cache Size by Abilene Topology*

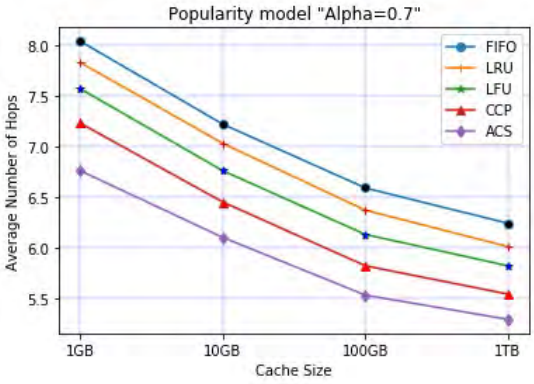
Topology	Popularity model α			Cache Size				Average Hops Number				
	0.65	0.7	0.85	1 GB	10 GB	100 GB	1 TB	FIFO	LRU	LFU	CCP	ACS
	X			X				8.34	8.06	7.77	7.38	6.86
	X				X			7.16	6.93	6.61	6.27	5.87
	X					X		6.19	5.98	5.61	5.36	5.07
	X						X	5.51	5.29	5.12	4.97	4.73
Abilene		X		X				8.04	7.83	7.57	7.23	6.76
		X			X			7.22	7.03	6.76	6.45	6.10
		X				X		6.59	6.37	6.13	5.82	5.53
		X					X	6.24	6.01	5.82	5.54	5.29
			X	X				7.41	7.12	6.87	6.63	6.32
			X		X			6.76	6.47	6.26	6.01	5.71
			X			X		6.32	6.07	5.83	5.62	5.33
			X				X	6.04	5.81	5.57	5.34	5.09

2- Average Hop Number with Tree Topology

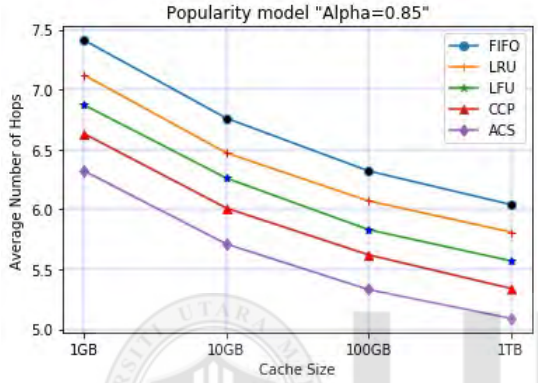
The average hop number compared with the capability of the cache is shown in Figure 6.12. When α = 0.65, which depicts a low popularity scenario, the average hop number is decreased on cache sizes 100 to 100,000 (1GB to 1TB) elements (Figure 6.12(a)). Whereas, the expected drop in the average hop number is achieved when the probability parameter a is increased from 0.65 (Figure 6.12(a)) to 0.7 (Figure 6.12(b)) and 0.85 (Figure 6.12(c)) with a simultaneous increase in the cache sizes.



(a)



(b)



(c)

Figure 6.12: Average Hops Number on Tree Topology

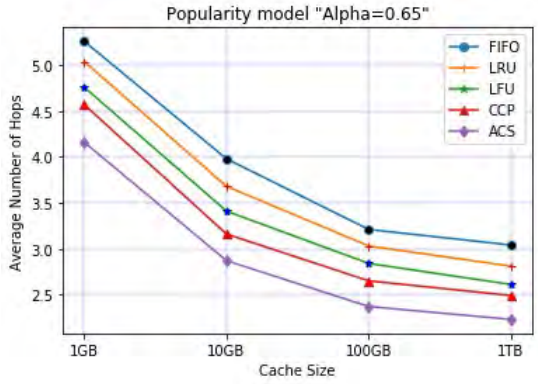
Table 6.12: *Average Hops Number on Different Popularity Model α and Cache Size by Tree Topology*

Topology	Popularity model α			Cache Size				Average Hops Number				
	0.65	0.7	0.85	1 GB	10 GB	100 GB	1 TB	FIFO	LRU	LFU	CCP	ACS
Abilene	X			X				8.87	8.51	8.16	7.83	7.38
	X				X			6.72	6.43	6.06	5.62	5.07
	X					X		4.65	5.46	5.11	4.63	4.18
	X						X	2.55	4.33	4.09	3.72	3.31
		X		X				8.56	8.34	8.01	7.53	7.07
		X			X			7.02	6.59	6.21	5.75	5.37
		X				X		5.74	5.41	4.99	4.61	4.23
		X					X	4.78	4.44	4.05	3.67	3.29
			X	X				7.73	7.31	6.94	6.31	5.89
			X		X			6.28	5.88	5.42	4.85	4.35
			X			X		4.83	4.34	3.86	3.31	2.86
			X				X	3.81	3.29	2.82	2.43	2.02

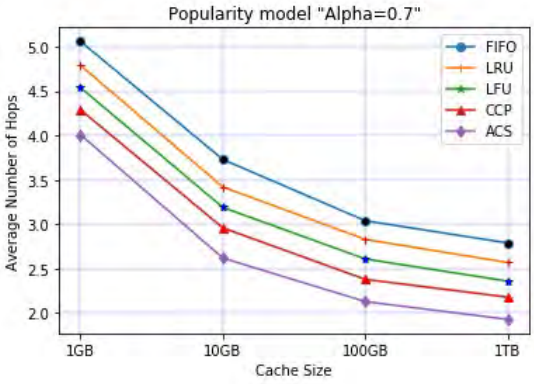
Figure 6.12 in addition to the information given in Table 6.12 shows the productivity of the diverse cache strategies for given scenario with different cache sizes based on the determined average hop number. Similar to the observation of the average hop number with the Abilene topology above, the ACS, over the CCP, LFU, LRU, and FIFO for all various cache size obviously shows the best outcomes essentially. Noting that the adoption of the average hop number is generally aimed at giving a holistic understanding of the network's caching performance when the data packets are distributed consistently and dependent on the dimensionality of the caches. This also implies that the small difference is a sign in the average hop number.

3- Average Hop Number with Grid Topology

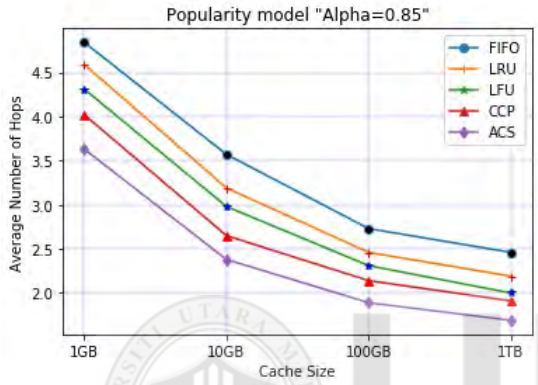
The average hop number in comparison to the cache capability is shown in Figure 6.13. Similar to the results obtained for the average hop number in Abilene and Tree topologies, a drop in the average hop number is achieved when the probability parameter a is increased from a low popularity scenario (α = 0.65 with cache sizes 100 to 100,000, that is 1GB to 1TB elements) in Figure 6.13(a) to a higher value of 0.7 (Figure 6.13(b)) and 0.85 (Figure 6.13(c)), with an increase of the cache size as well. The productivity of the different cache strategies for given scenarios, defined with ranging values of cache sizes with respect to the determined average hop number is similarly detailed in Table 6.13. The results shown in Figure 6.13 and Table 6.13 interprets that the ACS obviously shows the best outcomes essentially in comparison to the conventional CCP, LFU, LRU, and FIFO for all various cache sizes. Not forgetting that the small difference is a sign in the average hop number. This is due to the fact that the computation of the average hop number, when the data packets are distributed in a consistent way dependent on the dimensionality of the caches, is adopted in order to obtain a clearer understanding of the caching performance of the network.



(a)



(b)



(c)

Figure 6.13: Average Hops Number on Grid Topology

Table 6.13: *Average Hops Numberon Different Popularity Model α and Cache Size by Grid Topology*

Topology	Popularity model α			Cache Size				Average Hops Number				
	0.65	0.7	0.85	1 GB	10 GB	100 GB	1 TB	FIFO	LRU	LFU	CCP	ACS
	X			X				5.26	5.04	4.76	4.57	4.16
	X				X			3.98	3.68	3.41	3.16	2.87
	X					X		3.21	3.03	2.84	2.65	2.37
	X						X	3.04	2.81	2.61	2.49	2.23
5x5 Grid		X		X				5.06	4.79	4.54	4.29	4.01
		X			X			3.73	3.42	3.19	2.96	2.62
		X				X		3.04	2.83	2.61	2.38	2.13
		X					X	2.79	2.57	2.36	2.18	1.93
			X	X				4.84	4.59	4.31	4.02	3.63
			X		X			3.57	3.19	2.98	2.65	2.38
			X			X		2.73	2.46	2.31	2.14	1.89
			X				X	2.46	2.19	2.00	1.91	1.69

6.3 Discussion

Throughout this chapter, various parameters which include the popularity model variation, metrics measurement, cache size, and topological impact, were used for the assessment of the performance of ACS. As discussed earlier, ACS gives a solution to the 'what to cache' problem with the relation of popularity, and 'what to update' with the relation of priority. Since the current Dynamic Random Access Memory (DRAM) space available is 10GB (which is also the maximum vacant space), it is expected that after a period of time, the memory fills up and the eviction of content is initiated, so that there is room to accommodate the contents that are arriving newly. Thus, ACS will give impressively high performance with respect to the cache hit ratio, while decreasing server load, throughput, and average number of hops with size 1GB to 1TB concurrently.

Therefore, ACS has shown to solve many problems in caching in current NDN architecture, which includes the maintenance of content data consistency (during the occurrence of updating), increasing request satisfaction, and reduction of content data item redundancy in different caches. This will translate to making an efficient and effective cache mechanism in NDN which will lead to a bright future of NDN in current internet access scenarios.

In consideration of ACS comparison with FIFO, LRU, LFU, and CCP strategy, the results showed that ACS is considered as the optimal technique for offering reliable cache, within the distribution of full caching by LCE strategy. Since each cache overflows its buffer opportunistic that is dependent on requests from users within its regional community, there is no collaborative work of data to just about every cache. Whereas with regards to a cache miss with the local cache, the area associated with data replacement is based on items' association with the network path that results in a

cache query similar to the one requested by clients. Meanwhile, it should be noted also that from almost requests, all peer are cached before forwarding the request towards the original resource.

It can be stated that the alternative cache strategy is a certain caching protocol that is found to be easily adapted to the information-centric connection scheme, and can benefit from real-time caching mechanism. In this process, the end-users can increase the publication in addition to retrieval associated with notifications about posting items of the cached data, which includes multimedia elements and other VoD items. Likewise, ACS confronts to the fact that staging the items of the clients in sequential order helps initiate an effective deployment that sometimes can be obliged to help serve these kinds of requests. However, a new user can reap the benefits of similar likes and dislikes issued by neighbouring end users and the cached items serve the purpose of increasing the performance to retrieve the necessary information/data.

The process of replying requests involves a certain amount of overhead that may be unavoidable. Also, the cache of the node is restricted in such a way that the end-users have no idea whether the cached items have reached the destination or not, based on the identical period at every single cache. This means that even in cases where there exist two nodes included clients subscribed for the same filter and have been cached in the past, the same items associated with the filter's reference send no certainty that will request only at least one node acquired by the same client. From this, it can be concluded that every single item "lives" with a cache is not the same and is dependent upon the nearby workload of the node where ACS can effectively deal with. This is why it offers higher cache hit ratio as compared to the other conventional techniques.

6.4 Summary

This chapter discussed the performance evaluation of ACS in comparison with FIFO, LRU, LFU and CCP caching strategies designed for NDN. The proposed strategy was implemented in the simulator for the evaluation of cache hit rate, server load, throughput, and average number of hops in ndnSIM 2.7, for measuring the utilization of memory. The obtained results showed that ACS achieved better performance than FIFO, LRU, LFU and CCP in the cache hit rate, server load, throughput, and average number of hops. Consequently, the ACS proved that it could be the accurate choice for considering caching strategy in the future Internet.

CHAPTER SEVEN
CONCLUSION

The main goal of this thesis is to design an Alternative Cache Strategy (ACS) for Named Data Networking (NDN). During the course of the research, certain challenges for cross-comparison of the caching contributions for NDN were discovered. With reference to existing literature, a handful number of simulation environments was studied and parameters ranging from popularity models to topologies were identified. Based on the draft of Internet Engineering Task Force (IETF) [15], simulation parameters were carefully selected, and the performance of ACS strategy was extensively evaluated through simulation in the ndnSIM 2.7 simulator. As discussed in Chapter Five, the validation of ACS was tested on three ISP-level topologies (Abilene, Tree, and Grid), with catalog size chosen as 10^6 elements and cache size ranging form 100 to 100,000 (1GB to 1TB) elements. For popularity, three values of a were selected for the Zipf model (0.65, 0.7, and 0.85). In general, this chapter documents the importance of ACS for the future Internet with its possible implementation (Section 6.1), the research contributions of this thesis (Section 6.2), the limitations of the research (Section 6.3), and suggestions for future studies (Section 6.4).

7.1 Research Summary

The existing Internet architecture follows source-driven approach, where the source needs to establish a connection with the receiver before sending any data. This approach reduces the ability of full resource utilization that are available all the way from the publisher (receiver) to the subscriber (user). To overcome this problem, Van Jacobson [15] proposed an idea in which content request by customers is done without knowledge of the providing host and the path is defined by the publisher to the subscriber. In other words, the approach follows a receiver-driven approach, while

a reverse path is followed by the data. Then the system takes the responsibility of mapping the requested data and its location. This approach is called Information-Centric Networking (ICN).

As presented in Chapter One, the NDN network is classified into a multi-stage caching system where the requests are generated from geographically dispersed nodes and the caches are physically located across the network. Now, if the requested content is available somewhere cached by a network node, the request is satisfied locally. This leads to the problem of effective caching strategies when data packet comes to the node, and this node does not have space to save it. Thus leading to the question of which previous data packet will be evicted from a node and on what basis is the eviction justified. Instances of such include, when the evicted file is to be requested again in the future when the evicted content is to have a high cost, when data packets are stored forever in the cache, or data cached may quickly becoming obsolete as they are transient and frequently updated by their producers. Hence, the main goal of this thesis, which is to introduce a new caching strategy for content replacement and content update to deal with the aforementioned problems.

Although many strategies have been designed for this reason, as discussed in Chapter Two, however, it was observed and concluded that the existing strategies could not handle the challenges caused by content replacement and update. For example, FIFO, which is the default-NDN strategy, has one of its disadvantages to be static analysis which deals with its inability and difficulty to achieve the extraction of information. While LRU, LFU and CCP have the shortcoming of having no update policy to accommodate the newly arrived contents and thus there is a replacement of the popular contents with the newly arriving ones, which reduce the cache hit rate and in turn increase content retrieval delay. These limitations of the conventional approaches

motivated and prompted this research work to design a new caching strategy for NDN called an Alternative Cache Strategy (ACS).

To achieve the objectives of this research, the research steps were followed as a guideline and framework, and network modelling and simulation process was adopted as proposed in [143, 187] and discussed in Chapter Three. Moreover, two mechanisms of ACS which are the Content Replacement Mechanism (CRM), which is used for popular content replacement, and the Content Update Mechanism (CUM), designed for a content update, were introduced in Chapter Four. In that same Chapter Four, the designed and implemented mechanisms undergo the verification and validation process to attain accreditation. The findings from the verification and validation showed that CRM increases cache hit rate and CUM decreases average hop number. Furthermore, the simulation results showed that ACS performs impressively in terms of cache hit ratio, and simultaneously decrease server load, throughput, and average hop number, with cache size 1GB to 1TB, as presented extensively in Chapter Five. In achieving the performance evaluation of ACS strategy, the simulation results confidently emphasize that the main objectives of this study have been fully accomplished.

7.2 Research Contributions

This research's vital contribution is the design of an alternative cache strategy (ACS) for the improvement of the performance of the future Internet. Moreover, two mechanisms (CRM and CUM) were proposed for achieving better results for ACS.

The specific contributions consist of the following:

1. The development of an alternative cache strategy for the future Internet that can

serve as a benchmark for any intended improvement of NDN caching.

2. The design of a new NDN caching strategy for multimedia applications that leads to the following applicable contributions:

a) Efficiently increasing cache hit rate of the requested content items, which is more effective for data requests of frequently accessed objects.

b) Reduce path redundancy during content placement, which results in a better memory management.

c) Minimize searching overhead during content eviction and increase hop decrement that lead to reduced latency and stretch in content delivery.

3. The adaptation of caching was made by developing two new mechanisms for content placement and eviction to improve caching gain and for simplification of content access.

4. Technique for Order Preference by Similarity to Ideal Solution (TOPSIS), which is one of the Multiple Attribute Decision Making (MADM) approaches (based on Fuzzy Set Theory), was adapted for replacement of content to decide which content to evict from the content store.

5. Content update based on another MADM approach called Simple Additive Weighting (SAW) was introduced, where the updated content decides which document will update after removal from the content store.

6. The verification and validation of the proposed strategy were achieved by implementing it in ndnSIM 2.7 simulator and comparing the obtained results with other caching strategies, that is, FIFO, LRU, LFU, and CCP.

A summary of the research contributions is demonstrated in Figure 7.1. In the first phase, the CRM mechanism was modeled following the fuzzy TOPSIS approach for content replacement. In the next phase, using fuzzy SAW method, the CUM mechanism was modeled to update the contents and send content name to PIT. Finally, the ACS was simulated, and the results were compared with those of FIFO, LRU, LFU, and CCP in the ndnSIM 2.7 simulator with respect to cache hit ratio, server load, throughput, and average hop number with cache size 1GB to 1TB.

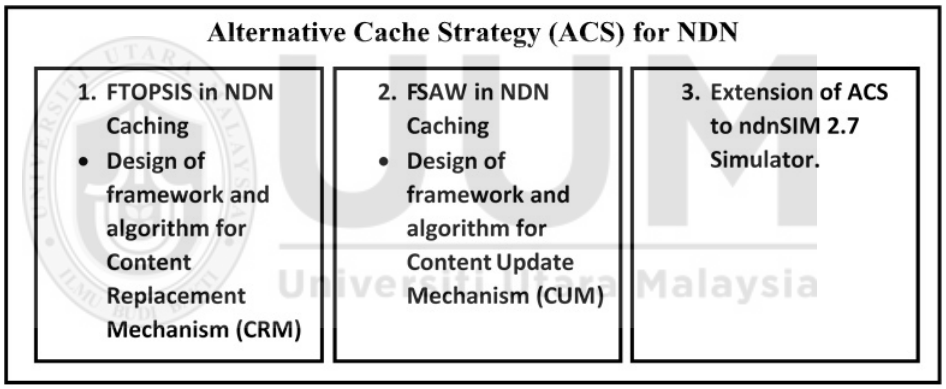


Figure 7.1: Research Contributions

7.3 Research Limitations

A number of limitations were identified in this work through ICN management, which is the idea of the future internet. NDN research appears as one of the many ICN-related concepts still under research. The result showed that a modification of the data caches with respect to the replacement techniques can improve the system's efficiency in a more significantly, which can be interpreted to being the access time in average, resulting from the distribution of the node. Specifically, certain limitations identified in this study which must be highlighted include the application of certain levels of caches

with every level possessing its own features, and also, the associativity and execution length being the major parameters that vary and it is necessary for define differing constraint to both options. Likewise, the performance of some studies happen to be founded on the association to desktop systems which has the shortcoming of not being optimized for embedded processors' characteristics.

Furthermore, the proposed strategy performs well in the infrastructure-based (fixed) wireless networks, however, it will be difficult to specify the router with maximum outgoing interfaces if applied on every ad hoc network. Besides, if two cached contents, for example, A and B, have the same value, then it is very tough to decide which content should be evicted or updated to accommodate new incoming content. For this process, the decision is left on ACS, either to evict A or B. Although, due to the same popularity value and access time, this eviction process will not affect the overall result.

7.4 Future Works

The suggestions for future works with reference to the objectives considered in this thesis includes that future studies can take into consideration additional cache sizes and also consider the invalidation of the coherence protocol which may have an effect on the object of node replacement. An in-depth look into caching in the current NDN architecture have identified quite a number of open issues, such as the maintenance of synchronization with different caching routers concerning the server, reduction of content data item redundancy in different caches, and cache space optimization for achieving high capacity storage of content data item. Also, cache pollution attack particularly targets the popular contents that are stored in the cache and tries to fill the cache with unpopular data. This forces the users to get their requested content from the original source rather than the nearest location. All these can be taken into

consideration for future work with the aim of making an effective and efficient cache mechanism in NDN. This should lead to a bright future of NDN in current internet access scenarios.

Moreover, the performance of ACS in this study was compared with FIFO, LRU, and LFU (the default NDN strategy), and CCP (cache), with better results being obtained from ACS than FIFO, LRU, and LFU [91, 157].However, it would likewise be worthy to examine its performance against other caching strategies in different network domains such as 5G network and cloud infrastructure. Furthermore, the overall simulation results are obtained on Abilene, Tree, and Grid topologies. Nevertheless, it would also be useful to test the performance of ACS against other metrics.

REFERENCES

[1] I. U. Din, S. Hassan, and A. Habbal, "Redundancy elimination in the future Internet," in *Proceedings of the International Conference on Computing, Mathematics and Statistics (iCMS 2015)*. Singapore: Springer, 2017, pp. 67–73.

[2] B. Fidler and A. L. Russell, "Infrastructure and maintenance at the defense communications agency: Recasting computer networks in histories of technology," *Technology and Culture*, vol. 59, no. 4, 2018.

[3] Y. Li, J. Wang, and R. Han, "Pb-ncc: A popularity-based caching strategy with number-of-copies control in information-centric networks," *Applied Sciences*, vol. 12, no. 2, p. 653, 2022.

[4] V. Cisco, "Cisco Visual Networking Index: Forecast and Methodology 2016–2021(2017)," 2017.

[5] R. S. Abbas, S. A. Nor, and A. Ahmad, "A review of cache placement algorithms for video on demand in named data networking," *Journal of Advanced Research in Dynamical and Control Systems*, pp. 1322–1332, 2018.

[6] V. Jain, M. Singh, and A. Bharti, "Product recommendation platform based on natural language processing," in *Data Analytics and Management*. Springer, 2021, pp. 627–635.

[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*. Berlin, Germany: ACM, Nov. 2011, pp. 549–558.

[8] M. Wahlisch, T. C. Schmidt, and M. Vahlenkamp, "Backscatter from the data plane–threats to stability and security in Information-Centric Network infrastructure," *Computer Networks*, vol. 57, no. 16, pp. 3192–3206, 2013.

[9] S. Wang, J. Bi, J. Wu, and A. V. Vasilakos, "Cphr: In-network caching for information-centric networking with partitioning and hash-routing," *IEEE/ACM transactions on networking*, vol. 24, no. 5, pp. 2742–2755, 2015.

[10] C. Fang, H. Yao, Z. Wang, W. Wu, X. Jin, and F. R. Yu, "A survey of mobile information-centric networking: Research issues and challenges," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2353–2371, 2018.

[11] B. Nour, H. Ibn-Khedher, H. Moungla, H. Afifi, F. Li, K. Sharif, H. Khelifi, and M. Guizani, "Internet of things mobility over information-centric/named-data networking," *IEEE Internet Computing*, vol. 24, no. 1, pp. 14–24, 2019.

[12] Content Delivery Networks CDN market (Online), Available: https://www.marketsandmarkets.com/Market-Reports/content-delivery-networks-cdn-market-657.html.

[13] K. Bhushan and B. Gupta, "A novel approach to defend multimedia flash crowd in cloud environment," *Multimedia Tools and Applications*, pp. 1–31, 2017.

[14] 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," *Relatorio Tecnico NDN-0001, Xerox Palo Alto Research Center-PARC*, 2010.

[15] 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*. Rome, Italy: ACM, Dec. 2009, pp. 1–12.

[16] A. Ghodsi, S. Shenker, T. Koponen, A. Singla, B. Raghavan, and J. Wilcox, "Information-Centric Networking: Seeing the forest for the trees," in *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*. ACM, 2011, p. 1.

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

[18] M. Amadeo, C. Campolo, J. Quevedo, D. Corujo, A. Molinaro, A. Iera, R. L. Aguiar, and A. V. Vasilakos, "Information-Centric Networking for the Internet of things: Challenges and opportunities," *IEEE Network*, vol. 30, no. 2, pp. 92–100, 2016.

[19] H. Dai, Y. Wang, H. Wu, J. Lu, and B. Liu, "Towards line-speed and accurate on-line popularity monitoring on NDN routers," in *Quality of Service (IWQoS), 2014 IEEE 22nd International Symposium of*. IEEE, 2014, pp. 178–187.

[20] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, E. Uzun, B. Zhang, G. Tsudik, D. Krioukov, D. Massey *et al.*, "Named Data Networking (NDN) project 2010-2011 progress summary," *PARC, http://www. nameddata. net/ndn-ar2011. html, Tech. Rep*, 2011.

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

[22] W. K. Chai, D. He, I. Psaras, and G. Pavlou, "Cache "less for more" in Information Centric Networks (extended version)," *Computer Communications*, vol. 36, no. 7, pp. 758–770, 2013.

[23] Y. Ren, J. Li, S. Shi, L. Li, G. Wang, and B. Zhang, "Congestion control in named data networking–A survey," *Computer Communications*, vol. 86, pp. 1–11, 2016.

[24] A. Arcia-Moret, I. Psaras, and J. Crowcroft, "Bringing Information Centric Networking to challenged environments: An overview of the second workshop

on future Internet architecture for developing regions," *arXiv preprint arXiv:1801.01824*, 2018.

[25] M. Amadeo, A. Molinaro, C. Campolo, M. Sifalakis, and C. Tschudin, "Transport layer design for named data wireless networking," in *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on*. Toronto, ON, Canada: IEEE, Apr. 2014, pp. 464–469.

[26] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of Information-Centric Networking," *IEEE Communications Magazine*, vol. 50, no. 7, 2012.

[27] H. Hu, J. Bi, T. Feng, S. Wang, P. Lin, and Y. Wang, "A survey on new architecture design of Internet," in *2011 International Conference on Computational and Information Sciences*. IEEE, 2011, pp. 729–732.

[28] J. Pan, S. Paul, and R. Jain, "A survey of the research on future Internet architectures," *IEEE Communications Magazine*, vol. 49, no. 7, 2011.

[29] G. Tyson, N. Sastry, I. Rimac, R. Cuevas, and A. Mauthe, "A survey of mobility in Information-Centric Networks: Challenges and research directions," in *Proceedings of the 1st ACM workshop on Emerging Name-Oriented Mobile Networking Design-Architecture, Algorithms, and Applications*. ACM, 2012, pp. 1–6.

[30] M. F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and B. Mathieu, "A survey of naming and routing in Information-Centric Networks," *IEEE Communications Magazine*, vol. 50, no. 12, 2012.

[31] Z. Ahmad, M. Tahir *et al.*, "Named data networking (NDN), new approach to future Internet architecture design: A survey," *International Journal of Informatics and Communication Technology (IJ-ICT)*, vol. 2, no. 3, pp. 155–165, 2013.

[32] J. S. Khoury and C. T. Abdallah, "A survey of novel Internetwork (and naming) architectures," in *Internet Naming and Discovery*. Springer, 2013, pp. 13–33.

[33] G. Tyson, N. Sastry, R. Cuevas, I. Rimac, and A. Mauthe, "A survey of mobility in Information-Centric Networks," *Communications of the ACM*, vol. 56, no. 12, pp. 90–98, 2013.

[34] A. Raza, A. A. Ikram, A. Amin, and A. J. Ikram, "A review of low cost and power efficient development boards for IoT applications," in *Future Technologies Conference (FTC)*. San Francisco, CA, USA: IEEE, Dec. 2016, pp. 786–790.

[35] G. L. Stuber, *Principles of mobile communication*. Springer, 2017, vol. 3.

[36] Y.-F. Chen, Y. Huang, J. Rahe, and B. Wei, "Peer-to-peer distributed storage for Internet protocol television," Feb. 21 2017, uS Patent 9,578,288.

[37] S. Khoussi, D. Pesavento, L. Benmohamed, and A. Battou, "NDN-trace: A path tracing utility for Named Data Networking," in *Proceedings of the 4th ACM Conference on Information-Centric Networking*. Berlin, Germany: ACM, Sep. 2017, pp. 116–122.

[38] 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.

[39] R. N. K. Alubady, "An efficient pending interest table control management in named data network," Ph.D. dissertation, Universiti Utara Malaysia, 2017.

[40] A. M. M. Habbal and S. Hassan, "A model for congestion control of transmission control protocol in mobile wireless Ad Hoc networks," *Journal of Computer Science*, vol. 9, no. 3, pp. 335–342, 2013.

[41] G. Resende, P. Melo, H. Sousa, J. Messias, M. Vasconcelos, J. Almeida, and F. Benevenuto, "(mis) information dissemination in whatsapp: Gathering, analyzing and countermeasures," in *The World Wide Web Conference*, 2019, pp. 818–828.

[42] N. Statt, "Facebook stories somehow hits 500 million daily users," 2019.

[43] Ali, Aran. Here's What Happens Every Minute on the Internet in 2020.), Available: https://www.visualcapitalist.com/every-minute-internet-2020/.

[44] K. W. R. James F. Kurose, "Computer networking: A top-down approach," 2013.

[45] Z. Zhu and A. Afanasyev, "Let's chronoSync: Decentralized dataset state synchronization in named data networking," in *Network Protocols (ICNP), 2013 21st IEEE International Conference on*. Goettingen: IEEE, Oct. 2013, pp. 1–10.

[46] J. Burke, "Video streaming over named data networking," *E-LETTER*, 2013.

[47] Y. Yu, A. Afanasyev, D. Clark, V. Jacobson, L. Zhang *et al.*, "Schematizing trust in Named Data Networking," in *Proceedings of the 2nd International Conference on Information-Centric Networking*. San Francisco, California, USA: ACM, Sep. 2015, pp. 177–186.

[48] P. Gusev, Z. Wang, J. Burke, L. Zhang, T. Yoneda, R. Ohnishi, and E. Muramoto, "Real-time streaming data delivery over Named Data Networking," *IEICE Transactions on Communications*, vol. 99, no. 5, pp. 974–991, 2016.

[49] Y. Sun, S. K. Fayaz, Y. Guo, V. Sekar, Y. Jin, M. A. Kaafar, and S. Uhlig, "Trace-driven analysis of ICN caching algorithms on video-on-demand workloads," in *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*. Sydney, Australia: ACM, Dec. 2014, pp. 363–376.

[50] K. N. Lal and A. Kumar, "A popularity based content eviction scheme via betweenness-centrality caching approach for content-centric networking (CCN)," *Wireless Networks*, pp. 1–12, 2017.

[51] M. Conti, A. Gangwal, M. Hassan, C. Lal, and E. Losiouk, "The road ahead for networking: A survey on icn-ip coexistence solutions," *IEEE Communications Surveys Tutorials*, vol. 22, no. 3, pp. 2104–2129, 2020.

[52] Q. Fan, X. Li, J. Li, Q. He, K. Wang, and J. Wen, "Pa-cache: Evolving learning-based popularity-aware content caching in edge networks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1746–1757, 2021.

[53] P. Maniotis and N. Thomos, "Viewport-aware deep reinforcement learning approach for 360 backslashcirc video caching," *IEEE Transactions on Multimedia*, vol. 24, pp. 386–399, 2021.

[54] H. Xu, "Queuing modelling and performance analysis of content transfer in information centric networks," 2023.

[55] D. Kim, J. Bi, A. V. Vasilakos, and I. Yeom, "Security of cached content in NDN," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 12, pp. 2933–2944, 2017.

[56] S. Kumar and R. Tiwari, "Optimized content centric networking for future internet: dynamic popularity window based caching scheme," *Computer Networks*, vol. 179, p. 107434, 2020.

[57] N. Dutta, S. K. Patel, O. S. Faragallah, M. Baz, and A. N. Z. Rashed, "Caching scheme for information-centric networks with balanced content distribution," *International Journal of Communication Systems*, vol. 35, no. 7, p. e5104, 2022.

[58] S. H. Bouk, S. H. Ahmed, D. Kim, and H. Song, "Named-data-networking-based ITS for smart cities," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 105–111, 2017.

[59] S. Lee, I. Yeom, and D. Kim, "T-caching: Enhancing feasibility of in-network caching in icn," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 7, pp. 1486–1498, 2020.

[60] B. Feng, A. Tian, S. Yu, J. Li, H. Zhou, and H. Zhang, "Efficient cache consistency management for transient iot data in content-centric networking," *IEEE Internet of Things Journal*, vol. 9, no. 15, pp. 12 931–12 944, 2022.

[61] Named Data Networking project (Online), Available: http://www.named-data.net/.

[62] J. Quevedo, D. Corujo, and R. Aguiar, "Consumer driven information freshness approach for Content Centric Networking," in *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on*. Toronto, ON, Canada: IEEE, Apr. 2014, pp. 482–487.

[63] B. Feng, H. Zhou, H. Zhang, J. Jiang, and S. Yu, "A popularity-based cache consistency mechanism for Information-Centric Networking," in *Global Communications Conference (GLOBECOM), 2016 IEEE*. Washington, DC, USA: IEEE, Dec. 2016, pp. 1–6.

[64] M. Meddeb, A. Dhraief, A. Belghith, T. Monteil, K. Drira, and S. AlAhmadi, "Cache freshness in named data networking for the Internet of things," *The Computer Journal*, 2018.

[65] I. U. Din, S. Hassan, A. Almogren, F. Ayub, and M. Guizani, "Puc: Packet update caching for energy efficient iot-based information-centric networking," *Future Generation Computer Systems*, vol. 111, pp. 634–643, 2020.

[66] M. A. Naeem, R. Ullah, Y. Meng, R. Ali, and B. A. Lodhi, "Caching content on the network layer: a performance analysis of caching schemes in icn-based internet of things," *IEEE Internet of Things Journal*, vol. 9, no. 9, pp. 6477–6495, 2021.

[67] Z. Ming, M. Xu, and D. Wang, "Age-based cooperative caching in Information-Centric Networking," in *Computer Communication and Networks (ICCCN), 2014 23rd International Conference on*. Shanghai, China: IEEE, Aug. 2014, pp. 1–8.

[68] A. Gharaibeh, I. Hababeh, and M. Alshawaqfeh, "An efficient online cache replacement algorithm for 5g networks," *IEEE Access*, vol. 6, pp. 41 179–41 187, 2018.

[69] K. N. Lal and A. Kumar, "A popularity based content eviction scheme via betweenness-centrality caching approach for content-centric networking (ccn)," *Wireless Networks*, vol. 25, pp. 585–596, 2019.

[70] N. Lal, S. Kumar, G. Kadian, and V. K. Chaurasiya, "Caching methodologies in content centric networking (ccn): A survey," *Computer Science Review*, vol. 31, pp. 39–50, 2019.

[71] M. A. Hail, M. Amadeo, A. Molinaro, and S. Fischer, "Caching in named data networking for the wireless Internet of things," in *Recent Advances in Internet of Things (RIoT), 2015 International Conference on*. Singapore, Singapore: IEEE, Apr. 2015, pp. 1–6.

[72] Y. Zhang, X. Tan, and W. Li, "Ppc: Popularity prediction caching in icn," *IEEE Communications Letters*, vol. 22, no. 1, pp. 5–8, 2017.

[73] Q. N. Nguyen, J. Liu, Z. Pan, I. Benkacem, T. Tsuda, T. Taleb, S. Shimamoto, and T. Sato, "Ppcs: a progressive popularity-aware caching scheme for edge-based cache redundancy avoidance in information-centric networks," *Sensors*, vol. 19, no. 3, p. 694, 2019.

[74] S. Bommaraveni, T. X. Vu, S. Chatzinotas, and B. Ottersten, "Active content popularity learning and caching optimization with hit ratio guarantees," *IEEE Access*, vol. 8, pp. 151 350–151 359, 2020.

[75] S. Lee, I. Yeom, and D. Kim, "T-caching: Enhancing feasibility of in-network caching in icn," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 7, pp. 1486–1498, 2020.

[76] C. Bian, Z. Zhu, A. Afanasyev, E. Uzun, and L. Zhang, "Deploying key management on NDN testbed," *UCLA, Peking University and PARC, Tech. Rep*, 2013.

[77] R. K. Mok, X. Luo, E. W. Chan, and R. K. Chang, "QDASH: A QoE-aware DASH system," in *Proceedings of the 3rd Multimedia Systems Conference*. Chapel Hill, North Carolina: ACM, Feb. 2012, pp. 11–22.

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

[79] 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*. Paris, France: ACM, Sep. 2014, pp. 27–36.

[80] G. Zhang, Y. Li, and T. Lin, "Caching in Information Centric Networking: A survey," *Computer Networks*, vol. 57, no. 16, pp. 3128–3141, 2013.

[81] R. S. Abbas, S. A. Nor, and A. Ahmad, "A review of cache placement algorithms for video on demand in named data networking."

[82] M. A. Naeem, S. A. Nor, S. Hassan, and B.-S. Kim, "Compound popular content caching strategy in named data networking," *Electronics*, vol. 8, no. 7, p. 771, 2019.

[83] I. Ud Din, "Flexpop: A popularity-based caching strategy for multimedia applications in information-centric networking," Ph.D. dissertation, Universiti Utara Malaysia, 2016.

[84] J. Erman, A. Gerber, K. Ramadrishnan, S. Sen, and O. Spatscheck, "Over the top video: The gorilla in cellular networks," in *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*. Berlin, Germany: ACM, Nov. 2011, pp. 127–136.

[85] J. Li, H. Wu, B. Liu, J. Lu, Y. Wang, X. Wang, Y. Zhang, and L. Dong, "Popularity-driven coordinated caching in named data networking," in *Proceedings of the Eighth ACM/IEEE Symposium on Architectures for Networking and Communications Systems*. ACM, 2012, pp. 15–26.

[86] R. Margolies, A. Sridharan, V. Aggarwal, R. Jana, N. Shankaranarayanan, V. A. Vaishampayan, and G. Zussman, "Exploiting mobility in proportional fair cellular scheduling: Measurements and algorithms," *IEEE/ACM Transactions on Networking (TON)*, vol. 24, no. 1, pp. 355–367, 2016.

[87] G. Bianchi, A. Detti, A. Caponi, and N. Blefari Melazzi, "Check before storing: What is the performance price of content integrity verification in LRU caching?" *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 3, pp. 59–67, 2013.

[88] K. Arora and D. R. Ch, "Web cache page replacement by using LRU and LFU algorithms with hit ratio: A case unification," *(IJCSIT) International Journal of Computer Science and Information Technologies*, vol. 5, no. 3, pp. 3232–3235, 2014.

[89] M. Garetto, E. Leonardi, and V. Martina, "A unified approach to the performance analysis of caching systems," *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, vol. 1, no. 3, p. 12, 2016.

[90] L. Saino, I. Psaras, and G. Pavlou, "Icarus: A caching simulator for Information Centric Networking (ICN)," in *Proceedings of the 7th International ICST Conference on Simulation Tools and Techniques*. Lisbon, Portugal: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), Mar. 2014, pp. 66–75.

[91] J. Ran, N. Lv, D. Zhang, Y. Ma, and Z. Xie, "On performance of cache policies in Named Data Networking," in *International Conference on Advanced Computer Science and Electronics Information*, China, Jul. 2013, pp. 668–671.

[92] I. U. Din, S. Hassan, and A. Habbal, "A content eviction mechanism for Information-Centric Networking," *ARPN Journal of Engineering and Applied Sciences*, 2016.

[93] D. Saxena, V. Raychoudhury, N. Suri, C. Becker, and J. Cao, "Named data networking: A survey," *Computer Science Review*, vol. 19, pp. 15–55, 2016.

[94] W. Dron, A. Leung, M. Uddin, S. Wang, T. Abdelzaher, R. Govindan, and J. Hancock, "Information-maximizing caching in Ad Hoc networks with Named Data Networking," in *Network Science Workshop (NSW), 2013 IEEE 2nd*. IEEE, 2013, pp. 90–93.

[95] M. Bilal and S.-G. Kang, "A cache management scheme for efficient content eviction and replication in cache networks," *IEEE Access*, vol. 5, pp. 1692–1701, 2017.

[96] ——, "Time aware Least Recent Used (TLRU) cache management policy in ICN," in *Advanced Communication Technology (ICACT), 2014 16th International Conference on*. Pyeongchang, South Korea: IEEE, Aug. 2014, pp. 528–532.

[97] K. Shah, A. Mitra, and D. Matani, "An O (1) algorithm for implementing the LFU cache eviction scheme," *Technical report*, 2010.

[98] S. Podlipnig and L. Boszormenyi, "A survey of web cache replacement strategies," *ACM Computing Surveys (CSUR)*, vol. 35, no. 4, pp. 374–398, 2003.

[99] S. Romano and H. ElAarag, "A quantitative study of recency and frequency based web cache replacement strategies," in *Proceedings of the 11th communications and networking simulation symposium*. ACM, 2008, pp. 70–78.

[100] C. Forecast, "Cisco visual networking index: Forecast and methodology 2013-2018," *Cisco Public Information*, 2014.

[101] N. Megiddo and D. S. Modha, "ARC: A self-tuning, low overhead replacement cache," in *FAST*, vol. 3, no. 2003, 2003, pp. 115–130.

[102] ——, "Outperforming LRU with an adaptive replacement cache algorithm," *Computer*, vol. 37, no. 4, pp. 58–65, 2004.

[103] W. Lee, S. Park, B. Sung, and C. Park, "Improving Adaptive Replacement Cache (ARC) by reuse distance," in *9th USENIX Conference on File and Storage Technologies (FAST'11)*, 2011, pp. 1–2.

[104] A. Patil, M. Prakash, and A. Nimkar, "First-in not referenced first-out page replacement algorithm," in *Proceedings of the International Conference & Workshop on Emerging Trends in Technology*. Mumbai, Maharashtra, India: ACM, Feb. 2011, pp. 443–446.

[105] B. Jeannet, T. Jeron, and T. Le Gall, "Abstract Interpretation of FIFO channels," Ph.D. dissertation, INRIA, 2005.

[106] D. Grund and J. Reineke, "Precise and efficient FIFO-replacement analysis based on static phase detection," in *Real-Time Systems (ECRTS), 2010 22nd Euromicro Conference on*. Brussels: IEEE, Jul. 2010, pp. 155–164.

[107] M. S. Haque, J. Peddersen, and S. Parameswaran, "CIPARSim: Cache intersection property assisted rapid single-pass FIFO cache simulation technique," in *Proceedings of the International Conference on Computer-Aided Design*. San Jose, CA, USA: IEEE Press, Nov. 2011, pp. 126–133.

[108] Z. Feng, M. Xu, Y. Wang, and Q. Li, "An architecture for cache consistency support in Information Centric Networking," in *Global Communications Conference (GLOBECOM), 2013 IEEE*. Atlanta, GA, USA: IEEE, Dec. 2013, pp. 2126–2131.

[109] Z. Zhu, C. Bian, A. Afanasyev, V. Jacobson, and L. Zhang, "Chronos: Serverless multi-user chat over NDN," *University of California, Los Angeles, Tech. Rep. NDN-0008*, pp. 1–12, 2012.

[110] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in HTTP-based adaptive video streaming with festive," *IEEE/ACM Transactions on Networking (TON)*, vol. 22, no. 1, pp. 326–340, 2014.

[111] J. Dyer, "Multiple criteria decision analysis: State of the art surveys," *International Series in Operations Research & Management Science*, vol. 78, no. 4, pp. 265–292, 2005.

[112] I. Ud Din, "Flexpop: A popularity-based caching strategy for multimedia applications in Information Centric Networking," Ph.D. dissertation, Universiti Utara Malaysia, 2016.

[113] G.-H. Tzeng and J.-J. Huang, *Multiple attribute decision making: Methods and applications*. CRC press, 2011.

[114] K. P. Yoon and C.-L. Hwang, *Multiple attribute decision making: An introduction*. Sage publications, 1995, vol. 104.

[115] S.-J. Chen and C.-L. Hwang, "Fuzzy multiple attribute decision making methods," in *Fuzzy multiple attribute decision making*. Springer, 1992, pp. 289–486.

[116] L. A. Zadeh, "Fuzzy sets," *Information and control*, vol. 8, no. 3, pp. 338–353, 1965.

[117] I. Hwang, D. Kim, and Y.-B. Ko, "Analysis of NDN repository architectures," in *High Performance Computing & Simulation (HPCS), 2016 International Conference on*. Innsbruck, Austria: IEEE, Jul. 2016, pp. 985–988.

[118] A. M. B. Yaakob, "Multi criteria decision making methodology for fuzzy rule based systems and networks using topsis." Ph.D. dissertation, University of Portsmouth, 2017.

[119] C.-L. Hwang and K. Yoon, "Methods for multiple attribute decision making," in *Multiple attribute decision making*. Springer, 1981, pp. 58–191.

[120] P. C. Fishburn, "Letter to the editor—additive utilities with incomplete product sets: Application to priorities and assignments," *Operations Research*, vol. 15, no. 3, pp. 537–542, 1967.

[121] Y. Smaragdakis, "General adaptive replacement policies," in *Proceedings of the 4th international symposium on Memory management*. ACM, 2004, pp. 108–119.

[122] P. Scheuermann, J. Shim, and R. Vingralek, "A case for delay-conscious caching of Web documents," *Computer Networks and ISDN Systems*, vol. 29, no. 8-13, pp. 997–1005, 1997.

[123] S. Podlipnig and L. Böszörmenyi, "A survey of web cache replacement strategies," *ACM Computing Surveys (CSUR)*, vol. 35, no. 4, pp. 374–398, 2003.

[124] A. Vakali, "Proxy cache replacement algorithms: A history-based approach," *World Wide Web*, vol. 4, no. 4, pp. 277–297, 2001.

[125] M. Arlitt, L. Cherkasova, J. Dilley, R. Friedrich, and T. Jin, "Evaluating content management techniques for web proxy caches," *ACM SIGMETRICS Performance Evaluation Review*, vol. 27, no. 4, pp. 3–11, 2000.

[126] IEEE Standards Board, IEEE recommended practice for distributed interactive simulation 2013: Verification, validation, and accreditation, IEEE Standard 1278.4-1997, January 2010.(Online), Available: https://standards.ieee.org/findstds/standard/1730.1-2013.html.

[127] D. Thomas, A. Joiner, W. Lin, M. Lowry, and T. Pressburger, "The unique aspects of simulation verification and validation," in *Aerospace Conference, 2010 IEEE*. IEEE, 2010, pp. 1–7.

[128] A. Habbal, "TCP Sintok: Transmission control protocol with delay-based loss detection and contention avoidance mechanisms for mobile Ad Hoc networks," Ph.D. dissertation, Universiti Utara Malaysia, 2014.

[129] R. G. Sargent, "Validation and verification of simulation models," in *Proceedings of the 36th Conference on Winter Simulation*. Washington, D.C: Winter Simulation Conference, Dec. 2004, pp. 17–28.

[130] O. Balci, "Verification validation and accreditation of simulation models," in *Proceedings of the 29th conference on Winter simulation*, 1997, pp. 135–141.

[131] ——, "Validation, verification, and testing techniques throughout the life cycle of a simulation study," *Annals of operations research*, vol. 53, no. 1, pp. 121–173, 1994.

[132] M. M. Kadhum, "Fast congestion notification mechanism for next generation routers," Ph.D. dissertation, Universiti Utara Malaysia, 2010.

[133] S. Floyd and V. Paxson, "Difficulties in simulating the Internet," *IEEE/ACM Transactions on Networking (ToN)*, vol. 9, no. 4, pp. 392–403, 2001.

[134] S. Hassan, "Simulation-based performance evaluation of TCP-friendly protocols for supporting multimedia applications in the Internet," Ph.D. dissertation, University of Leeds (School of Computing), 2002.

[135] P. Barford and L. Landweber, "Bench-style network research in an Internet instance laboratory," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 3, pp. 21–26, 2003.

[136] A. Suki and C. M. Arif, "Slight-Delay Shaped Variable Bit Rate (SD-SVBR) technique for video transmission," Ph.D. dissertation, Universiti Utara Malaysia, 2011.

[137] S. Hassan and M. Kara, "Simulation-based performance comparison of TCP-friendly congestion control protocols," in *Proceedings of the 16th Annual UK Performance Engineering Workshop (UKPEW2000)*, 2000, pp. 199–210.

[138] G. Osman, "Scaleable and smooth TCP-friendly receiver-based layered multicast protocol," Ph.D. dissertation, Universiti Utara Malaysia, 2008.

[139] C. Williamson, "Internet traffic measurement," *IEEE Internet Computing*, vol. 5, no. 6, pp. 70–74, 2001.

[140] B. L. Ong, "A hybrid mechanism for SIP over IPv6 macromobility and micromobility management protocols," Ph.D. dissertation, Universiti Utara Malaysia, 2008.

[141] O. M. D. Al-Momani, "Dynamic redundancy forward error correction mechanism for the enhancement of Internet-based video streaming," Ph.D. dissertation, Universiti Utara Malaysia, 2010.

[142] P. N. D. Bukh, "The art of computer systems performance analysis, techniques for experimental design, measurement, simulation and modeling," 1992.

[143] R. Jain, *The Art Of Computer Systems Performance Analysis: Techniques For Experimental Measurement, Simulation, And Modeling*. john wiley & sons, 2008.

[144] L. K. John, "8.2 performance evaluation: Techniques, tools, and benchmarks," *The Computer Engineering Handbook*, vol. 8, p. 21, 2002.

[145] H. Al-Bahadili, *Simulation in Computer Network Design and Modeling: Use and Analysis: Use and Analysis*. IGI Global, 2012.

[146] J. Mo, "Performance modeling of communication networks with markov chains," *Synthesis Lectures on Data Management*, vol. 3, no. 1, pp. 1–90, 2010.

[147] A discrete-event network simulator - NS-3. (Online), Available: https://www.nsnam.org/.

[148] S. K. Fayazbakhsh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. Maggs, K. Ng, V. Sekar, and S. Shenker, "Less pain, most of the gain: Incrementally deployable ICN," in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4. ACM, 2013, pp. 147–158.

[149] B. Han, X. Wang, N. Choi, T. Kwon, and Y. Choi, "AMVS-NDN: Adaptive mobile video streaming and sharing in wireless named data networking," in *Computer Communications Workshops (INFOCOM WKSHPS), 2013 IEEE Conference on*. Turin, Italy: IEEE, Apr. 2013, pp. 375–380.

[150] A. Gawande, V. Lehman, L. Wang, J. Shi, B. Zhang, and A. Afanasyev, "Mini-NDN, Jun. 2016."

[151] Z. Wang and J. Meng, "NDN EBAMS node running in Mini-NDN."

[152] A. Afanasyev, I. Moiseenko, L. Zhang *et al.*, "ndnSIM: NDN simulator for NS-3," *University of California, Los Angeles, Tech. Rep*, vol. 4, 2012.

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

[154] S. Hassan, W. Elbreiki, M. Firdhous, and A. Monzer, "End-to-end networks vs named data network: A critical evaluation," *Jurnal Teknologi*, vol. 72, no. 5, pp. 71–76, 2015.

[155] F. Dobrian, A. Awan, D. Joseph, A. Ganjam, J. Zhan, V. Sekar, I. Stoica, and H. Zhang, "Understanding the impact of video quality on user engagement," *Communications of the ACM*, vol. 56, no. 3, pp. 91–99, 2013.

[156] M. Hassan and R. Jain, High performance TCP/IP networking. Prentice Hall, 2003.(Online), Available: http://www.4ward-project.eu/.

[157] Y. Zhang, X. Tan, and W. Li, "PPC: Popularity Prediction Caching in ICN," *IEEE Communications Letters*, vol. 22, no. 1, pp. 5–8, 2018.

[158] C. Fricker, P. Robert, J. Roberts, and N. Sbihi, "Impact of traffic mix on caching performance in a content-centric network," in *2012 Proceedings IEEE INFOCOM Workshops*. IEEE, 2012, pp. 310–315.

[159] L. A. Adamic and B. A. Huberman, "Zipf's law and the internet." *Glottometrics*, vol. 3, no. 1, pp. 143–150, 2002.

[160] K. Pentikousis, E. Davies, S. Spirou, G. Boggia, and P. Mahadevan, "Information-centric networking: Evaluation methodology draft-irtf-icnrg-evaluation-methodology-01," 2014.

[161] M. Hefeeda and O. Saleh, "Traffic modeling and proportional partial caching for peer-to-peer systems," *IEEE/ACM Transactions on networking*, vol. 16, no. 6, pp. 1447–1460, 2008.

[162] T.-A. Le, N. D. Thai, P. L. Vo *et al.*, "The performance of caching strategies in content centric networking," in *2017 international conference on information networking (ICOIN)*. IEEE, 2017, pp. 628–632.

[163] D. Saucez, L. A. Grieco, and C. Barakat, "AIMD and CCN: Past and novel acronyms working together in the future Internet," in *Proceedings of the 2012 ACM Workshop on Capacity Sharing*. ACM, 2012, pp. 21–26.

[164] C. W. Churchman and R. L. Ackoff, "An approximate measure of value," *Journal of the Operations Research Society of America*, vol. 2, no. 2, pp. 172–187, 1954.

[165] L. Wang, A. Hoque, C. Yi, A. Alyyan, and B. Zhang, "Ospfn: An ospf based routing protocol for named data networking," Technical Report NDN-0003, Tech. Rep., 2012.

[166] G. Carofiglio, M. Gallo, L. Muscariello, and M. Papali, "Multipath congestion control in content-centric networks," in *2013 IEEE conference on computer communications workshops (INFOCOM WKSHPS)*. IEEE, 2013, pp. 363–368.

[167] Y. Ren, J. Li, S. Shi, L. Li, G. Wang, and B. Zhang, "Congestion control in named data networking–a survey," *Computer Communications*, vol. 86, pp. 1–11, 2016.

[168] A. Ghodsi, T. Koponen, J. Rajahalme, P. Sarolahti, and S. Shenker, "Naming in content-oriented architectures," in *Proceedings of the ACM SIGCOMM Workshop on Information-Centric Networking*. ACM, 2011, pp. 1–6.

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

[170] S. Yang, J. Kurose, S. Heimlicher, and A. Venkataramani, "Measurement and modeling of user transitioning among networks," in *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 828–836.

[171] V. Capone and M. Usman, "The GEANT network: Addressing current and future needs of the HEP community," vol. 664, no. 5, p. 052005, 2015.

[172] L. Saino, I. Psaras, and G. Pavlou, "Hash-routing schemes for Information-Centric Networking," in *Proceedings of the 3rd ACM SIGCOMM Workshop on Information-Centric Networking*. ACM, 2013, pp. 27–32.

[173] D. Rossi and G. Rossini, "On sizing ccn content stores by exploiting topological information," in *2012 Proceedings IEEE INFOCOM Workshops*. IEEE, 2012, pp. 280–285.

[174] C. Bernardini, "Popularity-based caching strategies for content centric networking," Ph.D. dissertation, Universite de Lorraine, 2015.

[175] C. Bernardini, T. Silverston, and O. Festor, "MPC: Popularity-based caching strategy for content centric networks," in *Communications (ICC), 2013 IEEE International Conference on*. IEEE, 2013, pp. 3619–3623.

[176] A. Afanasyev, "Developing simple simulations with ndnsim," *NDN Tutorial–ACM ICN*, vol. 2016, 2016.

[177] M. Feng, H. Ma, Y. Hu, and M. Yu, "Research on the influence of different network topologies on cache performance for ndn," in *International Conference on Artificial Intelligence and Security*. Springer, 2020, pp. 365–375.

[178] S. Glassman, "A caching relay for the world wide web," *Computer Networks and ISDN systems*, vol. 27, no. 2, pp. 165–173, 1994.

[179] K. Pentikousis, B. Ohlman, E. Davies, S. Spirou, G. Boggia, and P. Mahadevan, "Information-centric networking: Evaluation methodology," *Internet Draft*, 2015.

[180] S. Jin and A. Bestavros, "Sources and characteristics of web temporal locality," in *Proceedings 8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (Cat. No. PR00728)*. IEEE, 2000, pp. 28–35.

[181] O. Saleh and M. Hefeeda, "Modeling and caching of peer-to-peer traffic," in *Proceedings of the 2006 IEEE international conference on network protocols*. IEEE, 2006, pp. 249–258.

[182] A. M. J. GarciaLunaAceves, "Understanding optimal caching and opportunistic caching at the edge of information-centric networks," in *Proceedings of the 1st ACM conference on information-centric networking*, 2014, pp. 47–56.

[183] M. Hefeeda and O. Saleh, "Traffic modeling and proportional partial caching for peer-to-peer systems," *IEEE/ACM Transactions on networking*, vol. 16, no. 6, pp. 1447–1460, 2008.

[184] C. Bernardini, "Strategies de cache basees sur la popularite pour content centric networking," Ph.D. dissertation, PhD Thesis, University of Lorraine, France, 2015.

[185] J. Sung, J.-K. K. Rhee, and S. Jung, "Lightweight caching strategy for wireless content delivery networks," *IEICE Communications Express*, vol. 3, no. 4, pp. 150–155, 2014.

[186] J. Wang, J. Ren, K. Lu, J. Wang, S. Liu, and C. Westphal, "An optimal cache management framework for information-centric networks with network coding," in *2014 IFIP networking conference*. IEEE, 2014, pp. 1–9.

[187] M. Guizani, A. Rayes, B. Khan, and A. Al-Fuqaha, *Network modeling and simulation: a practical perspective*. John Wiley & Sons, 2010.