PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
UNIVERSITY AMAR TELIDJI OF LAGHOUAT

FACULTY OF SCIENCES
DEPARTMENT OF COMPUTER SCIENCE

On Caches Management in NDN using Data Mining techniques

Thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science (LMD)

Abdelkader Tayeb HEROUALA

Jury members

Mr. Mohamed Bachir YAGOUBI	Professor	UAT Laghouat	President
Mr. Younes GUELLOUMA	MCA	UAT Laghouat	Examiner
Mr. Amine KHALDI	MCA	UKM Ouargla	Examiner
Mr. Akram Zine Eddine BOUKHAMLA	MCA	UKM Ouargla	Examiner
Mr. Benameur ZIANI	MCA	UAT Laghouat	Advisor
Mr. Chaker Abdelaziz KERRACHE	MCA	UAT Laghouat	Co-Advisor

16 June 2022

I would like to dedicate this thesis to my loving parents, sisters and my friends.

Acknowledgements

Many thanks to Allah, the Most Exalted, the Most Merciful, for his help in the realization of this work, and for providing me with the people who have played an important role in the success of this thesis.

Thanks to my Advisors, Mr. Ziani Benameur and Mr. Kerrache Chaker Abdelaziz, for everything they offered to ensure the success of this thesis through their advice, their good supervision and their patience.

I thank Professor Lagraa Nasredine and Professor Tahari Abdou el Karim, the people behind the scenes from whom I learned a lot and who due to their efforts made possible the success of this thesis.

I thank the members of the jury of this thesis for their great work to correct all the errors and distortions of this thesis to make it better.

I would like to thank each of Khamloul Fakhredin, Mohamed Aliane, Mohamed Fadla, and Adel Allal for the wonderful moments we spent together during this period and for the many things I learned from them within and outside of this work.

A special thanks to my mother, father and sisters who always supported me in the most difficult moments and who are the main reason that pushed me to the success of this thesis.

Finally, I thank all the professors who have been teaching me at the University Amar Telidji Laghouat as well as at other previous levels, which are also the reason behind my ability to reach this advanced level.

Abstract

Named Data Network (NDN) is one of the most promising proposals as a future Internet architecture, which addresses the current Internet problems. NDN moves from the current host-centric TCP/IP architecture to a content-centric architecture by replacing IP addresses with content names and supporting network caching.

NDN caching make content distribution becomes more efficient as it reduces delays and network strain. Various caching strategies have been proposed to cache only important data to increase network performance. Most of the proposed strategies have considered the frequency of the requested data as a very important factor for data caching. Although this factor gives good results, the frequency of data names can cause many problems, because the number of content names searched is huge and varies rapidly. Similarly, considering caching based on the popularity of the requested data may not be an adequate option in all cases, especially in some limited networks, where most of the data consumed by users is not useful for the owners of these networks. In this case, these communities will not benefit from the use of NDNs because most of the cached data will not be useful to reduce the pressure on their servers or to reduce the access time to their data

The CaDaCa strategy is proposed as a solution to identify data of importance in the network and to avoid calculating the popularity of data names. This solution aims to analyze the data categories by classifying the data names using a machine learning technique. Then, based on the data classification, two new caching strategies are proposed. Both strategies performed well against the compared strategies. Similarly, the problem of limited networks was addressed by proposing the RADC strategy. This strategy is based on resource allocation by classifying the data according to a machine learning model using the Naïve Bayes method. Then, based on the frequency of requests in each category, an adaptive resource allocation strategy based on the Lagrangian utility function was proposed. From the obtained results, we can say that this strategy can be a powerful tool for administrators to manage the trade-off between efficiency and user satisfaction.

Keywords: Information centric networks, Named data networks, Caching in NDN, Placement strategies, Replacement strategies, Data categorization, Resources allocation in NDN, Machine learning.

Résumé

Les Réseau de données nommé (NDN) est l'une des propositions les plus prometteuses en tant que future architecture de l'Internet. Le NDN passe de l'architecture actuelle TCP/IP centrée sur l'hôte à une architecture centrée sur le contenu en remplaçant les adresses IP par des noms de contenu et en prenant en charge la mise en cache du réseau.

En exploitant la mise en cache dans l'NDN, la diffusion du contenu devient plus efficace car elle réduit le délai et la pression sur le réseau. Diverses stratégies de mise en cache ont été proposées dans le but de mettre en cache uniquement les données importantes afin d'augmenter les performances du réseau. La plupart des stratégies proposées ont pris en compte la fréquence des données demandées comme un facteur très important pour la mise en cache des données. Bien que ce facteur donne de bons résultats, la fréquence des noms de données peut poser de nombreux problèmes, car le nombre des noms de contenu recherchés est énorme et varie rapidement. De même, prendre en compte la mise en cache sur la base de la popularité des données demandées peut ne pas être une option adéquate dans tous les cas, en particulier dans certains réseaux limités, où la plupart des données consommées par les utilisateurs ne sont pas utiles pour les propriétaires de ces réseaux. Dans ce cas, ces communautés ne bénéficieront pas de l'utilisation de l'NDN car la plupart des données mises en cache ne seront pas utiles pour réduire la pression sur leurs propres serveurs ni pour réduire le temps d'accès à leurs propres données.

La stratégie CADACA est proposée comme solution pour identifier les données d'importance dans le réseau et pour éviter de calculer la popularité des noms de données. Cette solution vise a analysé les catégories de données en classifiant les noms de données à l'aide d'une technique d'apprentissage automatique. Ensuite, sur la base de la classification des données, deux nouvelles stratégies de mise en cache ont été proposées. Les deux stratégies ont donné de bons résultats par rapport aux autres stratégies utilisées pour la comparaison. De même, le problème des réseaux limités a été abordé en proposant la stratégie RADC. Cette stratégie est basée sur l'allocation des ressources en classant les données selon un modèle d'apprentissage

automatique utilisant la méthode Naïve Bayes. Ensuite, en fonction de la fréquence des demandes dans chaque catégorie, une stratégie d'allocation adaptative des ressources basée sur la méthode de Lagrange a été proposée. D'après les résultats obtenus, nous pouvons dire que cette stratégie peut être un outil puissant pour les administrateurs afin de gérer le compromis entre l'efficacité et la satisfaction des utilisateurs.

Mots clés: Réseaux orienté contenu, Réseau de données nommé, La mise en cache dans NDN, Strategies de placement dans NDN, Strategies de remplacement dans NDN, Apprentissage automatique.

ملخص

يشهد الإنترنت اليوم عددًا كبيشهد الإنترنت اليوم عددًا كبيرا من المستخدمين ونموًا هائلاً للمحتوى من حيث التخزين والاسترجاع والتوزيع. وقد أدى ذلك إلى تغيير كبير في استخدام الإنترنت، حيث تحول استخدام شبكات الكمبيوتر من مشاركة الموارد المادية إلى مشاركة المحتوى و الوصول إليه. ونتيجة لذلك أصبحت تواجه الإنترنت الحالية العديد من مشكلات التكيف من حيث التوجيه والتنقل والأمن البيانات. NDN (شبكة البيانات المسماة) هي واحدة من المقترحات الواعدة باعتبارها بنية مستقبلية للإنترنت التي يمكنها معالجة المشكلات الحالية. ينتقل NDN من بنية TCP/IP الحالية التي تتمحور حول المضيف إلى بنية تتمحور حول المحتوى عن طريق استبدال عناوين IP بأسماء المحتوى وبدعم التخزين المؤقت في الشبكة. الاستفادة من التخزين المؤقت في NDN يجعل تسليم المحتوى أكثر كفاءة من خلال تقليل التأخير وإنقاص الضغط على الشبكة. تتمثل إستراتيجية التخزين المؤقت المقترحة في هذه البنية في تخزين نسخة من كل محتوى مطلوب مؤقتًا في طريق العودة إلى المستهلك. على الرغم من أهمية هذه الاستراتيجية، إلا أن بساطتها تأدي إلى إهدار موارد الذاكرة التي تبقى محدودة مقارنة بالكم الهائل من البيانات المستهلكة. لذلك، تم اقتراح العديد من استراتيجيات ذاكرة التخزين المؤقت بهدف الاحتفاظ بالبيانات المهمة فقط من أجل زيادة أداء هذه البنية. معظم الاستراتيجيات المقترحة تعتبر تكرار البيانات المطلوبة عاملاً مهمًا جدًا لتخزين البيانات مؤقتًا. على الرغم من أن هذا العامل يعطي نتائج جيدة، إلا أن دراسة تكرار أسماء البيانات في الممارسة العملية لا يزال يمثل العديد من التحديات، حيث أن عدد الأسماء المحتويات المطلوبة تعد ضخمة و متغيرة بسرعة. من ناحية أخرى، اتخاذ قرار التخزين المؤقت بناءً على شعبية البيانات المطلوبة كعامل أساسي ليس خيارًا جيدًا في جميع الحالات، خاصة في بعض الشبكات المقيدة، حيث تكون معظم البيانات المستهلكة من طرف المستخدمين لتحم أي فائدة لأصحاب هذه الشبكات. في هذه الحالة، لن تستفيد هذه الأماكن من استخدام شبكات NDN لأن معظم البيانات المخزنة مؤقتًا لن تكون لها أهمية من حيث تقليل الضغط على خوادم هذه الأماكن و لن تقلل الوقت المستغرق للحصول على بياناتها الخاصة. حاولنا في هذه الرسالة معالجة هذه المشاكل والتحديات وإيجاد الحلول لها من خلال اقتراح إستراتيجيات جديدة تساعد في حلها. تم اقتراح إستراتيجية CaDaCa كحل لمعرفت البيانات ذات الأهمية في الشبكة و لتجب حساب شعبية أسماء البيانات التي تعد صعبة عن طريق تحليل فئات البيانات من خلال تصنيف أسماء البيانات باستخدام تقنية التعلم الآلي. بعد ذلك، بناءً على تصنيف البيانات، تم اقتراح استراتيجيتين جديدتين، تعتمد الأولى على تكرار فئات البيانات المستهلكة لاتخاذ قرار وضع البيانات. أما الاستراتيجية الثانية فتعتمد على قرار استبدال البيانات بدراسة تكرار أسماء البيانات. أداء كلتا الاستراتيجيتين من خلال النتائج المعطات يعتبر جيد مقارنة بالاستراتيجيات الأخرى المستخدمة للمقارنة. أيضا، تمت معالجة مشكلة الشبكات المحدودة من خلال اقتراح استراتيجية RADC . تعتمد هذه الإستراتيجية على تخصيص الموارد عن طريق تصنيف البيانات بناءً على نموذج التعلم الآلي باستخدام طريقة Naïve Bayes . بعد ذلك، بناءً على تكرار الطلبات في كل فئة، تم اقتراح إستراتيجية تخصيص موارد تكيفية بناءً على وظيفة الأداة المساعدة لاجرائج. من خلال النتائج التي تم الحصول عليها، يمكننا القول أن هذه الإستراتيجية يمكن أن تكون أداة قوية لإدارة المفاضلة بين الكفاءة و رضا المستخدم في مثل هذه الأماكن. لكفاءة و رضا المستخدم في مثل هذه الأماكن.

Table of contents

List of figures

List of tables

Nomenclature

Acronyms / Abbreviations

BH Browser History

CaDaCa Categorized Data for Caching

CAPC CAche based on Popularity and Class

CCN Content-Centric Networking

CCP Cache Content Popularity

CDN Content Distribution Networks

CnS Cache and Split

CPCCS Compound Popular Content Caching Strategy

CPT Content Popularity Table

CR Core Routers

CS Content Store

DONA Data-Oriented Network Architecture

EHCP Efficient Hybrid Content Placement

ER Edge Routers

FIA Future Internet Architecture

FIB Forwarding Information Base

FIFO First In First Out

HT History Table

ICN Information Centric Network

IMU Immature Used

IP Internet Protocol

LCD Leave Copy Down

LCE Leave Copy Everywhere

LdC Limited domain Cache strategy

LFU Least Frequently Used

LPC Least Popular Content

LR Logistic Regression

LRU Least Recently Used

MAPC Multi-Attribute Probability Caching algorithm

MPC Most-Popular content Caching

NB Naïve Bayes

NDN Named Data Networking

NetInf Network of Information

NPA Name Popularity Algorithm

NRS Name Resolution System

NUR Node Utilization Ratio

OCPCP Optimal Cache Placement based on Content Popularity

OPC Optimal Popular Content

P-CBC Popularity-weighted Content Based Centrality

P2P Peer to Peer

PCSD Proportional Cache Size Division

PDPU Push Down popular, Push Up less-popular

PIT Pending Interest Table

PSRIP Internet Publish/Subscribe Routing Paradigm

QoS Quality of Service

RADC Resource Allocation based Data Classification

RRT Request Record Table

SDN Software Defined Networking

SVM Support Vector Machine

TCP Transmission Control Protocol

TF-IDF Term-Frequency Inverse Document Frequency

TOPSIS Technique for Order Preference by Similarity to Ideal Solution

UATL University Amar Telidji of Laghouat

URL Uniform Resource Locator

VoD Video on Demand

XIA eXpressive Internet Architecture

Chapter 1

Introduction

1.1 Motivation

The internet was originally developed in the context of military experiments in the 60s and 70s with the objective of accessing a particular target that shares a predefined resource based on an IP address [1]. The main idea on which the Internet was founded is the telephone, because at that time, the telephone was the only example of successful and efficient communication on a world scale [2].

Today the Internet has experienced a huge progress due to several factors, which include the performance, availability and the ability to easily purchase mobile equipment (e.g., smart phones and tablets), which has resulted in more than 4.02 billion users of the Internet today [3]. The progress in digital coding can be considered as another factor because it made possible to transform text and the most complex objects such as videos and images into bit strings. For example, according to statistics, YouTube recorded 5 billion videos watched in a single day [4]. As well as NETFLIX counts 100 million hours of videos are viewed in one day [5]. In addition, the evolution of the web has allowed users to easily discover, consume and create content, so exabytes of new content are produced and distributed every year. For example, Google has already indexed one trillion web pages [6]. Social networks are also among the factors that increased the Internet's ability by facilitating social relationships in the digital world. For example, Facebook has about 2.912 billion monthly active users in January 2022 [7].

The enormous growth in the number of Internet users and the amount of data consumed will not be limited to this point but will still increase. According to Cisco's annual report [8],

the percentage of people who will be able to connect to the Internet will be 66% of the total number of the world's population in 2023. In addition, the number of devices connected to the IP network will be three times the current number (29.3 billion devices in 2023). Despite the great success achieved by the current network, the developments it faces today present several challenges, including security, mobility, scalability and quality of service [9, 10]. Today, IP is widely used in data distribution, while its origin comes from point-to-point communication. In addition, it is known that the IP address design requires that a host set an IP address to be identified during the TCP connection, which can be a problem with mobile devices. This is because every time these hosts move from one location to another, they need to be re-identified to connect, which can be a huge burden on the current network, especially when taking into account the huge increase in mobile device users in the future. In addition, the characteristics of IP datagrams allow only the endpoints of the connection to be specified (source and destination IP addresses). So, several similar objects are sent over the Internet, which results in additional delay as each time a user sends a request, it must go to the server. Moreover, the fact of reaching the server each time to retrieve the data puts a huge demand on these servers and creates uncontrollable congestion.

Although there are many solutions to address the current needs of the Internet, such as mobileIP [11], Content Distribution Networks (CDNs) [12], Peer to Peer (P2P) [13], etc. These solutions are only temporary mechanisms to address the current problems [14], as they are still based on the original host-to-host communication model [15, 16]. Today's Internet needs a new architecture that fits the current situation of the Internet, not just temporary solutions that will face the same problems. The Internet should now focus more on information and content rather than remaining in the host-to-host communication concept.

To overcome the challenges of the current Internet architecture, many researches and projects have been proposed under the name of Information Centric Network (ICN) as future Internet architecture [1]. Some of the most famous of such projects are the TRIAD architecture [17], Internet Publish/Subscribe Routing Paradigm (PSRIP) [18], Network Information (NetInf), the Data-Oriented Network Architecture (DONA) [19], eXpressive Internet Architecture (XIA) [20] and MobilityFirst [21]. Similarly, the U.S. research community has also initiated a number of ICN-based projects, including Content-Centric Networking (CCN) [22] and Named Data Networking (NDN) [23]. The common feature of these architectures is that they focus on the content rather than the location of the information, by substituting the

IP address with the name of the content as a fundamental solution to the current problems of the Internet.

Named Data Networking is one of the most promising candidates for future Internet architecture [24, 25]. This project is funded by the U.S. National Science Foundation under the Future Internet Architecture program [23]. The intuition behind NDN is that users are generally interested in the content itself without having to care about the physical location of the data. Therefore, NDN is characterized by accessing content by its name rather than its address, which is more appropriate to the current Internet usage practices. Content names in NDN are structured hierarchically like URLs. For example, a video of a computer science lecture produced by the University Amar Telidji of Laghouat (UATL) might take the form /uatl/cs/network/videos/ndn.mpg. Another key feature of NDN is in-network caching, where content is decoupled from its physical location and dispersed geographically in order to be closer to the users. All communications are performed using interest and data packets where the names of the data (or content) replace the addresses of the hosts. To retrieve data, the consumer first send an Interest packet with the name of the desired data. When a router (or node) receives the Interest packet, it checks the Pending Interest Table (PIT). If there is an old request with the same name, it adds the interface of the new request and waits to receive the requested data. Otherwise, the interest will be added to the PIT and then it will be forwarded to the next node according the Forwarding Information Base (FIB). Along the routing path, the router that has a copy of the requested data will respond with a data packet that is forwarded in the reverse path of the interest packet to the consumer. In addition, when the router receives the data packet, it caches a copy of the data in the Content Store (CS) table to satisfy future interests. Obviously, caching in the NDN can significantly reduce data traffic since data with the same names of interest can be satisfied at the nearest nodes.

Since the cache size is still very limited compared to the amount of data consumed by users, and the data is not of equal importance, cache management in NDN is a very important topic given its advantages in reducing network bandwidth, congestion, and server workload [26, 27].

1.2 Problem statement

Caching in NDN is a hot topic that attracts the attention of many researchers and companies. Today, NDN outlines many different caching strategies with the same objective: to identify

the most useful content among all requested content that would be suitable for caching to generate maximum subsequent responses and thus optimize the overall network performance. Strategies that consider the popularity of user-requested content as an important factor in the data caching decision are among the most successful strategies. Despite their importance, the popularity based strategies still suffers from several limitations and challenges. A common limitation of these methods is the calculation of content popularity based on the full name of the data. Due to the length and the unlimited number of content names, each node in the network will calculate the content popularity with taking into account a huge number of content names. Moreover, the rapid variation in content popularity at the node level requires recalculating its popularity each time, which leads to an imbalance in the decision to predict the important data for preservation and adds a significant load on the available processing resources at each node. Consequently, this can introduce undesirable delays in the quality of service. In addition, user requests typically share common hierarchies where many content names are associated with the same topic of interest (e.g., sports, games...etc). Such information is not leveraged by current solutions, which can have a significant impact on predicting future requests accurately and provide better results in caching strategies.

Apart from these considerations, it is not always useful to suggest data caching strategies from the perspective of user satisfaction (the data popularity calculation). Specifically, in some workplaces such as universities or enterprises where the most data consumption that satisfies the users has no relevance regarding the direction of these networks. Therefore, places suffering from this problem will have less benefit from using NDN. For example, the study that was conducted on the data consumed from the University of Amar Telidji of Laghouat during one week in 2006, showed that only 4% of the consumed data belong to the university compared to the overall consumption. In this case, with using the current NDN strategies, the university-related data will take more delay to be consumed compared to the data that are not necessary for the university network. In addition, the load on the university's server will not decrease significantly and when the server is down, it can be expected that the majority of the cached data will not match its server. Therefore, finding better and more intelligent caching strategies is essential to address these challenges.

1.3 Thesis objectives

The overall goal of the current thesis is to study caching in NDN as an important factor in the development of network efficiency by offering solutions to some of the challenges still faced by current caching strategies. To this end, this thesis will attempt to achieve the following objectives.

As a first objective, we will address the investigation of the basic characteristics and structures of NDN, with consideration of the most important caching strategies in NDN, and their classification according to the most significant features that distinguish each of these strategies. The goal is to gain a deeper understanding of the NDN architecture, as well as the characteristics and challenges that caching strategies still face.

As a second objective, we aim to analyze the popularity of the consumed data categories and how could be used in caching strategies, since we believe that the data categories provide more general and accurate insight regarding the consumers' interests, which can be a better solution for predicting future traffic and improving data placement and replacement decisions in NDN. Therefore, we see that the best way to achieve this is to study the categories of content names, where names in NDN are considered as a rich source of knowledge due to their structural hierarchy and readability. Moreover, names in NDNs have almost the same structural format as URLs, which allows us to adapt and exploit URL classification techniques in this domain. Specifically, the use of machine learning techniques that are making great progress in URL classification and have been successfully used to improve the performance of systems in various research domains.

As a final objective of this thesis, we will attempt to find a solution to limited networks that suffer from the trade-off problem between user satisfaction and resource efficiency by combining an efficient model that classifies the consumed data and a function that maximizes the utility of both types of these data.

1.4 Thesis organization

This thesis is organized in the following chapter bellow:

Chapter Two, provides a brief overview of the NDN architecture, including its important features and structures.

Chapter Three, a review of the existing literature on cache management strategies is done to classify and extract the most important characteristics of these strategies.

Chapter Four, presents the basic concepts of data mining and focuses mainly on the techniques used to solve the addressed problems. In this chapter, we present two techniques, where the first one is related to automatic data classification by the Multinomial Naïve Bayes method and the second one is related to the Lagrange optimization method. The purpose of this chapter is to give an overview of the techniques used and to introduce the concepts necessary for the next chapters of this work.

Chapter Five, try to explore the role of data categorization in enhancing the cache mechanisms in NDN. We present a new caching strategy called CaDaCa (Categorized Data for Caching), where popular requests are categorized using a machine learning technique enabling in-depth knowledge about users' behavior, which can be valuable in creating a more powerful caching mechanism. The performance of the proposed approach is evaluated using a real-world data-set extracted from browsing history and compared to other well-known strategies in the literature.

Chapter Six, addresses the fundamental problem of the trade-off between resource efficiency and user satisfaction in the limited environments of Named Data Networks. The proposed strategy is named RADC (Resource Allocation based Data Classification), which aims at managing the trade-off by controlling the systems fairness index. To this end, a machine learning technique based on Multinomial Naive Bayes is used to classify the received contents. Then, an adaptive resource allocation strategy based on the Lagrange utility function is proposed. To cache the received content, adequate content placement, and replacement mechanism is applied.

Chapter Seven, concludes the study by giving a summary of the research, contributions, and explore suggestions for future research direction.

Chapter 2

NDN Overview

2.1 Introduction

The Internet was originally conceived for the interconnection of a few remote hosts, where each characterized by an IP address. This conception is due to several factors, of which the telephone was the only example of successful communication on a world-scale, alongside the availability and cost of the devices, who were initially very expensive. With the rapid growth of the Internet, its main use has experienced an important evolution where the interest of the users has passed from the search of a specific address to obtain information to a direct interest in the information itself. This new scenario motivated the development of the ICN approach [15, 28, 29], which aims to natively support the change in Internet usage in term "what" data they want rather than "where" it is located. The main idea of this approach is to consider the name of content as the central element of the network, by replacing IP addresses with names that identify the content directly. The principle for obtaining content in this concept level is that a requester expresses interest in the content by specifying its name. The network then routes the request to the best source with a copy and returns the content to the requester in the reverse path [30].

This chapter presents the limit of the current internet. After that motivations and basic concepts about the ICN architecture with the most known projects under this architecture are presented. Afterward, the chapter is focuses on the characteristics and basic structure of the Named Data Networking architecture [31] which is the main topic of this thesis. The reason for choosing this architecture because it covers all the naming, routing and security trends of the ICN approach and forms the basis of our contributions.

2.2 The limits of the current Internet

Even though it has become an indispensable form of communication in our daily lives, the current Internet faces many challenges in its ability to satisfy the increased needs in terms of content distribution, mobility and security.

2.2.1 Content distribution problem

With the emergence of mobile devices and the high debit of internet access, it has experienced rapid development in the distribution of large amounts of content to consumers [32]. According to Cisco's annual report [8], the percentage of people who will be able to connect to the Internet will reach 66% of the total number of the world's population in 2023. In addition, the number of devices connected to the IP network will be three times the current number, with 29.3 billion devices in 2023. To meet the increasing demand for content, solutions such as Content Delivery Networks [12] and Peer to Peer networks [13] have been proposed. These solutions have contributed to the improvement of content distribution based on caching in the network [33]. However, they remain partial solution because it is fixed to a specific application or provider and cannot operate over IP. In addition, these solutions have some limitations [34].

These limitations include the difficulty of choosing the best P2P node to distribute content, which results in expensive traffic between providers [35]. Also, CDNs just handle specific types of traffic depending on the application, as well as CDN caches are located on specific servers and only keep the data of the owning content provider [17].

2.2.2 Mobility problem

The issue of mobility is also part of the problem with the current Internet. Indeed, IP addresses play at the same time the role of locator and identifier [36]. In the case where a host changes its location, a change in the IP address will also imply and they need to be re-identified to connect. This will result in the interruption of connectivity. Several proposals have been made to overcome this problem. However, they do not provide radical solutions. Some of these proposals rely on the use of another IP address that requires a major change in the naming approach (e.g., MobileIP) [37].

2.2.3 The security problem

The purpose of the Internet, in the beginning was to ensure the interconnection of a couple of trusted remote hosts. But with the current development, the Internet has also experienced many security problems and attacks [38]. Despite the proposed solutions that aim to secure specific protocols, their composition does not necessarily guarantee a secured system [39]. Similarly, security solutions under the IP network, link the security of a content to the reliability of the host that stores it [29]. Indeed, a user who stores content that has been previously retrieved from the original source cannot be sure that this content has not been modified by malware. In addition, users interested in the same content can only get it directly from the original source by establishing a secure connection, which adds a huge load to the servers and the network.

2.3 Motivations and basic concepts of ICN architecture

The major development in the use of the Internet has had several problems among which the previously mentioned ones in the previous subsection. In this context, the idea of developing a new architecture of Information-centric networks is proposed as a radical solution that fits with the current internet development. This approach focuses on the distribution of content independently of the hosts who store it. It considers the named content as a central element of the network and natively supports caching in the network. As a result, a copy of a requested content can be retrieved from the nearest router in the network rather than from its original source. Such properties allow for more efficient content distribution. Moreover, since the content names play the role of the identifier and are independent of the content locators, mobility is no longer a problem. Finally, to address security issues, the ICN proposes a content-oriented model, based essentially on the integration of cryptographic mechanisms on the content and an adequate naming system [40].

2.3.1 Basic concepts of ICN and proposed architectures

The ICN architecture has a set of concepts that are common to all projects that were proposed under this name. The basic concepts of this approach are naming, routing, caching and security.

Naming

Names in ICN are critical since they represent the fundamental core for all network operations. These names are used in all aspects of the network, from content identification and caching to content routing and distribution.

Two types of naming were proposed [41]. The first one proposes hierarchical and comprehensible names [42]. These names have a readable hierarchical structure similar to URLs', which repeat information related to the nature of the content itself and allow consumers to use them directly. The second type of naming approach proposes self-certifying and flat names [40, 19]. In this approach, cryptographic mechanisms are included in the names to allow the assurance and integrity of the contents.

Caching

The main purpose of caching is to allow the temporary placement of content in a particular space for subsequent utilization.

In ICN, each node in the network has a cache that significantly improves the network performance by allowing retrieval of a copy of the requested content from a geographically closer node to the requester [43]. In ICN we can find two types of caching approaches: on-path and off-path caching. On-path caching allows content to be retrieved from nodes located only on the transmission path, while off-path caching allows any available copy in the network to be used [28].

Routing and Forwarding

The ICN approach is primarily name-based for routing consumer interests and content in two phases: (1) routing the request toward a copy of the requested content and (2) routing the requested content toward its consumer. We can find two main approaches for routing packets in ICN.

The first approach uses a Name Resolution System (NRS) that keeps track of the connections between names and locations of content in the network [44]. Here, a consumer's request goes through three phases until the requested content arrives. The first phase corresponds to the routing of the content request to the NRS that translates the name of the request to the location of the nodes that maintain a copy of the same requested content. The second phase

corresponds to the request routing to the indicated node. The third phase involves routing the requested content to the consumer.

For the second routing approach, no name resolution is performed since the consumer's requests in this case are directly routed to one or more content sources using just the name. Once the node receives the request and possesses a copy of the requested content, that content is routed directly to the consumer via the reverse path [44].

Routing using NRS has the advantage of providing the locators of all available copies of the content on the path or off the path. In contrast, direct name-based routing has the advantage of eliminating the name resolution step which reduces overall latency and simplifies routing operations [45]. In addition, using hierarchical names for routing enables route aggregation which also improves the scalability of routing (e.g., NDN uses this technique).

Security

The ICN design enables a requested content to be satisfied by any node in the network owning that copy. Consequently, security in ICN is linked to the content and not to a specific host or storage location. Thus, ICN proposes a content-oriented security model that integrates the security aspects into the content itself [46–48].

Thus, through what we have discussed about the features of the ICN and the current Internet, we can summarize the most important differences between them in Table 2.1.

Table 2.1 Comparison between Internet based IP and ICN

Design	Internet based IP	Inernet based ICN
Naming	Related to the location of the host	Related content which is not related to its location
Caching	In dedicated servers	At any node in the network
Routing	From host to host using IP addresses	Between a requester and any node in the network that has a copy of the content.
Security	Ensuring the security of communication channels between hosts	Securing the content itself

2.3.2 The most famous architectures proposed under the name of ICN

Several efforts have been deployed to propose a data-centric model as a radical solution to the current Internet problem. These proposals have subsequently been commonly referred to as ICN architectures since they all share major common features and differ in minor design choices as previously discussed. For this reason, we are not going to discuss extensively all

the architectures since it is not the main focus of this thesis, but we will mainly concentrate on the NDN as our work will be based on the notions and terminologies of this architecture.

The notion of networking content objects was first proposed in 2000 with an architecture called TRIAD [17]. TRIAD introduced the concept of identifying endpoints by unique names and defined a content layer that provides routing and caching of content [49]. Another architecture called the Data-Oriented Network Architecture (DONA) [19], has also been proposed. This architecture is based on the publish-subscribe paradigm, where content owners publish content objects and recipients can then subscribe to any published content that interests them. In particular, content producers and enabled caching nodes can serve data by registering against resolution managers. The difference between DONA and TRIAD is that DONA uses flat, self-certified names to identify content rather than hierarchical names. Inspired by the DONA routing paradigm, Internet Publish-Subscribe was proposed next. In PSIRP, content objects are named by Rendez-vous identifiers [18]. The Network Information (NetInf) architecture is also another information-centric network architecture [50]. In NetInf, data sources publish the content they have by registering name/locator bindings with a name resolution service. In this case, the consumer sends its request to the NRS, and the NRS responds with a set of locators where the data can be found. Similarly, an architecture called eXpressive Internet Architecture (XIA) has been developed [20]. This architecture makes a self-certification for the identification of content objects, hosts and services, then an expressive Internet protocol is designed to support the communication between the three aforementioned components. MobilityFirst [21] is also an information-centric architecture that aims to address the need for mobility support. This Architecture uses self-certified names and makes use of a name resolution and global routing service to locate and route content. Finally, an architecture called Content-Centric Networking has also been proposed [22], one of its realizations being NDN. Since the NDN architecture [23] is arguably the most popular and promising ICN implementation for a future Internet architecture, in the next section we focus on the details of this approach and it is characteristic.

2.4 Named Data Networking : NDN

Named Data Networking is a promising architecture that was proposed by Van Jacobson in 2010 [42]. This architecture was initially proposed in 2006 also by Jacobson under the name

CCN [51]. Afterward, this project was funded by U.S. National Science Foundation under the Future Internet Architecture Program and it is officially known as NDN.

2.4.1 Naming in NDN

Names in NDN have a hierarchical structure that is readable, meaningful and provides information about the content itself. These names are similar in structure to URIs, where a name in NDN is constructed by several components separated by the delimiter "/". Overall, a name in NDN is composed of three main components. The first part of the names provides global routing information. The second part contains the name of the data with its type and the last part gives information about the version and the segment number. The notion of segmentation is used when the data is large since it allows the data to be split into several parts. Each of these parts is identified by a segment number and can be retrieved individually. For example, a segment of a video conference in computer science produced by the University Amar Telidji of Laghouat can take the form /uatl/cs/network/videos/ndn.mpg/v1/1. In this example, /uatl/cs/network/videos/ represents the routable part. /ndn.mpg/ is the part that represents the name and type of the requested data and /v1/1 indicates that it is the first version of the first segment of the video conference.

Retrieving the requested content from its name without having seen the name or the data before, remains an open challenge for this architecture. For this, two solutions are currently proposed by this architecture. First either by using a deterministic algorithm that allows the producer and the consumer to arrive at the same name based on the data available to both of them. The second solution is that consumers can retrieve data based on partial names (Prefix name). For example, the consumer requests the video /uatl/cs/network/videos/ndn.mpg and receives in return a data packet named /uatl/cs/network/videos/ndn.mpg/v1/1. So, afterward the consumer can specify the following segments and request them.

2.4.2 Packet types in NDN

Communications in NDN are based on two types of packets that are consecutively called Interest and Data [52, 31]. The Interest packet represents a request for content by the consumer and the Data packet represents the response to this request with the desired content (Fig . 2.1). The Interest packet consists of the "Request Name" which represents the name of the requested content and a "Selectors" that contains information about the retrieved content,

such as the digest of the producer's public key [42]. It also contains a random value called "Nonce" that has the role of identifying duplicate interests.

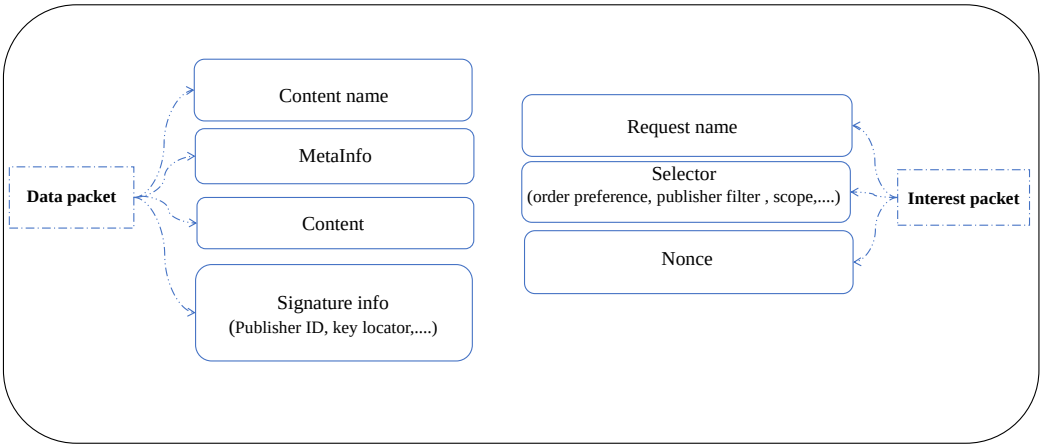


Fig. 2.1 Packets in the NDN Architecture

The Data package consists of a "Content name" which represents the same name of the requested interest. It also contains a field called "MetaInfo" which includes information about the content, such as a key when the contained packet matches with a public key. In addition, this packet contains a "Content" field that is reserved for the content itself. The Data packet also contains a field named " Signature info" which carries the information about the identity of the " Publisher " or the locator of the public key " KeyLocator " necessary for the verification.

2.4.3 NDN structures

To ensure adequately the processing of the received packets, NDN routers rely on three important data structures (Figures 2.2): the Content Store, the Pending Interest Table and the Forwarding Information Base.

The Content Store (CS), represent the structure responsible for caching the data received in a router. The CS hold the names and the content corresponding to those names. Placement and replacement policies are also integrated into this structure for updating the cached data.

The Pending Interests Table (PIT), maintains the list of Interests previously forwarded, but not yet answered. This information is needed to deliver data packets to their users. The

PIT table contains the names of the requests received with the interfaces through which they have been requested. Each interest requested in this structure has a life span to increase the use of the PIT.

The Forwarding Information Base (FIB), is a table that is used as a routing table in an IP router. The FIB records the prefixes of the content names and the destination interfaces for that content.

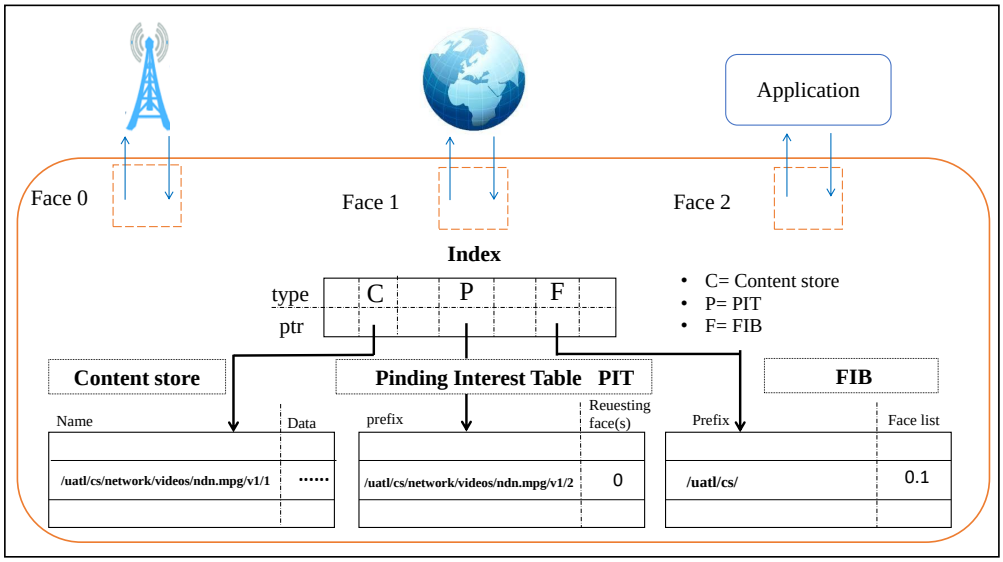


Fig. 2.2 Structures of an NDN Node

2.4.4 Forwarding and routing in NDN

Routing in NDN is somewhat similar to current IP routing, where instead of announcing IP prefixes we announce name prefixes for the data that the router is ready to serve. Each router builds its FIB based on the routing advertisements collected via a routing protocol.

When an Interest is received the router simply matches the longest prefix in the FIB to the name that exists in the the Interest packet. For example, /uatl/cs/network/videos/ndn.mpg may match both /uatl/cs and /uatl/cs/network in the FIB, where the last name is corresponding to the longer prefix.

NDN also can support multipath routing where an Interest can be sent using multiple interfaces without worrying about loops since the first data that return will satisfy the requested Interest and will be cached locally and later copies will be discarded.

2.4.5 Caching in NDN

NDN natively supports on-path caching, where when a router receives requested content, it keeps a copy in its CS before sending it in the reverse path to the consumer. This mechanism allows future interests in the same content to be satisfied by the closest router in the network. As a result, several benefits can be achieved, such as reduced delay and server load. Caching in NDN nodes has two main aspects: placement and replacement of content [53]. Placement is the strategy that decides where and what content should be stored on the network. While replacement strategies, consist of replacing the newly received content with the old content [17]. These two aspects enable routers to use the available space appropriately and to keep only the data that is relevant for improving the performance of the network. In the next chapter, we will discuss these techniques in more detail.

2.4.6 Security in NDN

The security concept in NDN is integrated directly into the data, where each data element is signed with its name. The signature determines the origin of the data by associating the publisher's information with it. This gives the consumer more confidence. In addition, the NDN naming mechanism completely removes the information about the person requesting the data. As a result, routers have no way of knowing the origin of requests, which provides more confidentiality and privacy than the IP packets which indicate the destination address.

2.4.7 Communication model

As is shown in Figure 2.3, when an Interest packet reaches a router, if the Content Store has a data copy with the same name as the Interest packet, the router creates a data packet and sends it through the interested interface. If there is no match between the CS entries and the received interest name, the PIT is consulted. If it has a matched entry of the same name as the Interest packet, the interface of the received packet is added to the list of interfaces with the same entry. If the CS and PIT do not provide any information, the FIB is checked. If the received interest name has a matched name prefix(es) in FIB, the packet is forwarded to the interfaces indicated in the FIB with regard to the longest prefix. In this case, a new entry is also added to the PIT containing the name of the Interest packet, the demanded face, the nonce and a lifetime to this entry will also add [54]. If no matching name is found in the FIB table, the Interest packet is discarded.

In the reception of the Data packet, as it is shown in Figure 2.4, a lookup function of its name is made in the PIT. If a matched request is found, the data packet is sent to all the requesting interfaces at the same time a copy of this data is kept in the CS. Then, the entry that matches to the name of the received data is discarded from the PIT.

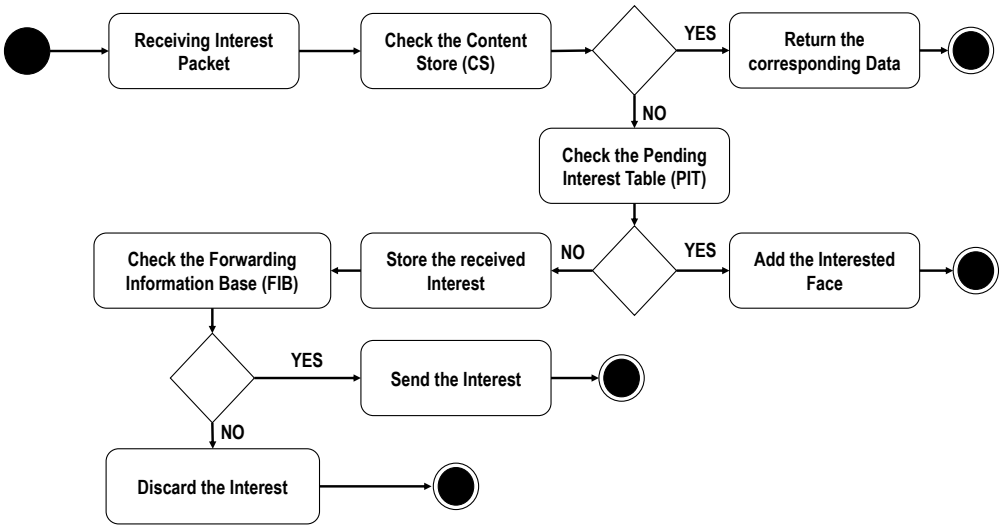


Fig. 2.3 Activity diagram of NDN node when receiving Interest Packet

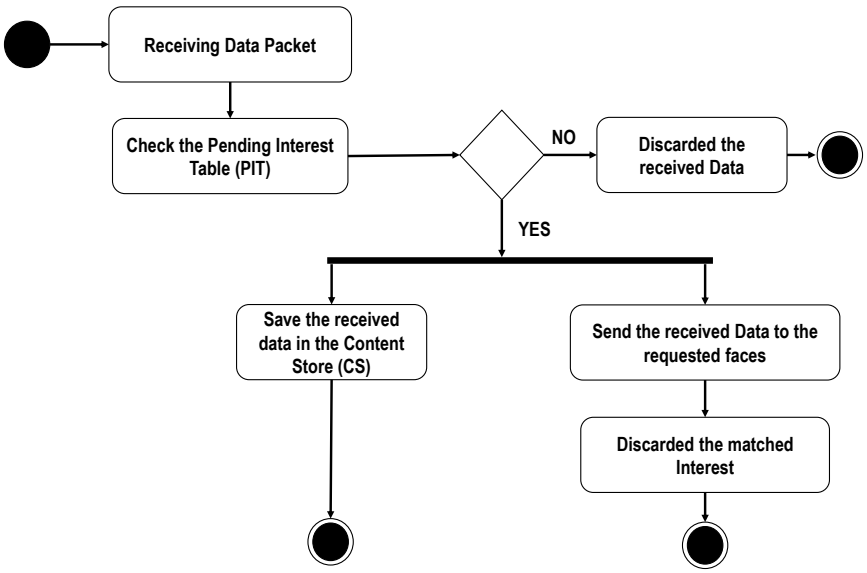


Fig. 2.4 Activity diagram of NDN node when receiving Data Packet

2.5 Conclusion

The new concepts proposed by the ICN architecture allow it to be a promising candidate for future architecture. The solutions proposed by this architecture are mainly based on the content naming as the primary entity.

In this chapter, the limitations of the current Internet regarding mobility, routing and security have been presented. Given the problems of the current Internet, we have shown how the ICN can overcome these limitations by supporting native caching in the network and naming that allows the separation of the locator. We also introduced the basic concepts of content-oriented networks in terms of naming, routing, caching and security. After that, we discussed in brief the mechanism of the most popular architectures known under the name ICN. At the end of these points, we started discussing the NDN architecture which represents the core of this thesis. We have described all the essential points and features in this architecture such as the type of packets and structures used. We also addressed the naming, routing, caching and security mechanisms of the architecture. Finally, we finished with a communication model in this architecture.

Chapter 3

Cache Management in NDN

3.1 Introduction

In NDN, the caching functionality is integrated directly at each node in the network, so when a consumer generate an interest in content, each node maintains a copy of that content in the content store. This feature allows future content requests to be satisfied from the closest nodes in the network without passing through the servers each time. Caching in the NDN is of great importance in improving the overall performance of the network. One of the benefits of using caching is the reduction of delay and load on servers. In reality the cache sizes in the network are still very limited to maintain all the requested data in the network. This means that the effectiveness of NDN depends mainly on caching strategies that manage the distribution of content along the networks and make the content available to consumers. Recently, NDN architecture has known several caching strategies to improve the efficiency of caching performance in the network. These strategies are mainly designed to address two major questions. The first one aims to answer the question of which content should be cached among all the received contents and the second one focuses on which content will be replaced when the CS becomes full for achieving better network performance. So, the proposed strategies are different in terms of principle, decision making and objective. To this end, this chapter aims to provide a comprehensive review on existing caching strategies. In this study, the caching strategies will be explained and are classified to facilitate the understanding on how the caching strategies work. In this chapter existing caching strategies will be compared and summarized to clarify their goal, decision making, differences, advantages and disadvantages. The existing caching strategies are simulated with different tools and

settings. For this purpose, the simulation tools and the factors that impact the performance during the simulation will be presented. At the end of this study, research issues related to network caching are discussed.

The remaining chapter is organized as follows. Section 3.2 presents the benefits of caching in the NDN. Section 3.3, presents a review and classification of existing caching strategies. In Section 3.4, the evaluation tools and factors that can influence the performance of strategies during simulation are presented. Section 3.5 mentions some research issues related to caching in NDN.

3.2 Benefits of Network Caching in NDN

Introducing the concept of cache in NDN is extremely beneficial as it allows for the possibility of reusing content, which in turn offers many advantages to the network, including:

- **Data recovery with a minimum cost:** Caching minimizes the use of communication links, especially expensive transit links, and reduces a significant percentage of the total traffic approaching the origin server [55]
- **Reduction in Latency:** NDN caching allows consumers to obtain their desired content from the closest node in the network instead of going to the source location, which in turn leads to an efficient latency reduction and thus improves the overall quality of service (QoS) perceived by the users.
- **Manipulation of heavy loads within the servers:** During periods of high load on servers, especially during peak periods when there is a large amount of popular content, caches can act as an intermediary for these servers, helping them deal with this pressure very efficiently.
- **Efficient Re-transmissions of the loosed packet:** In-network caching ensures greater resiliency against packet loss by providing fast re-transmission from the nearest intermediate node where there is a cached copy of the content to be recovered.
- **Content availability:** In-network caching significantly improves content availability, which significantly reduces the risk of a denial-of-service (DoS) attack.

- **Support for intermittent connection:** Caching also plays a crucial role in building the resiliency inherent in unstable connectivity and allows mobile nodes to act as a network medium for areas that are not covered by the network.

3.3 Cache decision in NDN

Due to the limited cache size to store all the requests from the consumers, the decision-making policy concerning the received data is one of the most important factors that increase the efficiency of the network. The cache decision policies in NDN are consists of two main parts which represent consecutively placement and replacement policies.

- **The placement policy (also known as cache insertion)**, decides what contents should be stored on the network.
- **The replacement policy (also known as cache eviction)**, refers to the process of removing old content to make space for new content, when the cache becomes full.

In the literature, there is a considerable amount of related work based on these two caching policies. In the next subsections we will examine these works.

3.3.1 Placement policies

Typically, an NDN node needs to make a placement decision when it receives a new Data packet, to determine whether the packet will be stored in the CS. To cache the contents near the end-user, the most straightforward placement decision strategy is Leave Copy Everywhere (LCE) [23, 56], where each NDN node caches every incoming Data packet that is not already in the CS. Despite its simplicity, the LCE strategy leads to excessive content redundancy which increases cache miss probability as the cache size is limited. To overcome this inefficiency, many placement strategies have been proposed, and we can classify them into the following approaches.

Probabilistic based approaches

These placement strategies are relying mainly on simple probabilistic models to alleviate the redundancy of content and give more accurate performance compared to LCE. In the Prob(p) strategy [57], each node estimates the probability of caching a given content by randomly

generating a number n between 0 and 1. Referring to a preset threshold of probability p, the Data packet is saved in the CS if $n \leqslant p$, otherwise, the content is forwarded downstream toward the consumer without being cached. The LCE strategy can be considered as a special case of Prob(p) with $p = 1$. In [58], a similar approach called LCProb strategy is proposed where the caching probability is defined by the number of hops between the node and the content producer. The ProbCache strategy, introduced in [59], estimates the caching probability of a given content considering the number of hops between the content producer and the consumer, as well as the number of hops remaining on the path to the consumer. In this strategy, nodes connected near the content source will be given priority to cache content over other nodes along the path.

Popularity based approach

Other approaches have been proposed where the caching decisions are mainly based on the content's popularity. The intuition is that some content, which is called popular, is requested and sent more often than others [60]. In the Most-Popular Content Caching (MPC) scheme [61], each node autonomously computes content request frequencies. When content reaches a popularity threshold, the node suggests its neighbors to cache this content, therefore, increasing the caching redundancy for popular contents in the network. A similar approach, called Optimal Cache Placement based on Content Popularity (OCPCP) is proposed in [62]. Using the Request Record Table (RRT), the OCPCP approach estimates the popularity of contents, and only popular ones will be cached.

Other techniques attempt to progressively cache popular content in the network to mitigate redundancy like the strategy Leave Copy Down (LCD) [63], which try to alleviate the load on the producer by gradually placing the frequently accessed contents towards the edges of the network. In WAVE [64], the proposed placement scheme aims at achieving an efficient distribution of contents as well as reducing the average cache management cost. In this strategy, content is cached in the form of related chunks based on the requirements of a consumer. Similarly, in [65], a progressive caching scheme is proposed to make efficient use of edge nodes and reduce the redundancy of caching contents in the same path.

Other different strategies try to combine content popularity with other information related to topology like the number of hops, node centrality, link congestion, etc. The Popularity-weighted Content Based Centrality (P-CBC) is suggested in [66], where a metric combining both the popularity and the centrality of contents is used for cache decisions. Similarly,

in [67], a placement strategy is suggested considering both the betweenness centrality of nodes and the popularity of contents. Thus, the most popular content will be cached in the most important nodes to optimize the cache redundancy and replacement rate. The EHCP (Efficient Hybrid Content Placement) [68] caches popular contents as close as possible to their producers (at central routers in the data delivery path). In contrast, the least requested contents are placed at edge nodes. In Compound Popular Content Caching Strategy (CPCCS) [69], based on the threshold value, CPCCS identifies in the first phase the optimal popular content (OPC) and least popular content (LPC). In a second phase, information on transmitted content, such as the number of hops, the path distance, and the location, are used for caching decisions at the intermediate nodes. A similar approach based on compound popularity is described in [70] where content popularity and node popularity are combined to improve the caching decision. The node popularity is estimated by the number of faces requesting popular content. Similarly, in [71], the authors try to find the optimal cache node for the object to be cached. To do so, they analyze the state of the nodes and the popularity of the content of each node based on an entropy weighting model and TOPSIS (Technique for Order Preference by Similarity to Ideal Solution). In [72], the Limited domain Cache strategy (LdC) is proposed to cache popular contents on a set of representative selected nodes called a custodian. Such nodes are identified using two factors: the cache hit rate and the hierarchical name of contents. In [73], the authors propose a distributed cache placement strategy 5G-based ICN, called Cache and Split (CnS). CnS moves popular content closer to consumers and keeps less popular content in the core of the network. A local decision is also made at the nodes to evict content based on access frequency and content popularity. Similarly, in [74], an edge network-level placement strategy for IoT applications named PDPU (Push Down popular, Push Up less-popular) is proposed with a collaborative mechanism to fetch content from the nearest neighbors. In [75], the authors propose a strategy that combines routing and resource allocation in the NDN network. This strategy uses a Q-learning algorithm based on the basic components of a Semi-Markov Decision Process called SMDP to prove the best routing and resource allocation path. In [76], the authors propose a joint routing strategy with a resource allocation algorithm for wireless environments, which they called dynamic routing and resource allocation. In [77], the authors propose a resource allocation strategy for a limited network environment. They formulate an optimization problem that attempts to maximize the caching of popular content to reduce latency. At the same time, they minimize the hit variance to achieve a balanced resource allocation among the network nodes. In [78],

the authors propose a strategy named Proportional Cache Size Division (PCSD) in which they attempt to divide the network topology into different tiers based on the demands of different interfaces and video applications. The goal is to minimize redundancy and cache popular content closer to the consumers. In [79], the authors studied the object allocation problem using game theory as a distributed many-to-one in ISP networks to improve cache utilization in NDN.

Classification based approach

Another category of placement strategies has been more interested in classifying data to allow a specific class or all classes to have a chance to get a space for caching their data. In [80], the authors propose a method for the dynamic allocation of cache space. They divide the content into five categories based on the packet characteristics, where each category has a different priority, to decide how many items should be cached in each category. In [81], the authors propose a caching strategy called CAche based on Popularity and Class (CAPC). The strategy cache a copy of the data one hop closer to the consumer for each request. In addition, when the CS of a node is full, the router checks the class of the received data and replaces it in the portion of the same class. Each class has a portion that changes periodically depending on the calculated popularity of the classes. The defined classes are consecutively Real-time VoD, User Generated Content, Web access and emails. In [82], the Multi-attribute Probability Caching Algorithm (MAPC) is proposed to divides the contents into two categories: non-shareable and shareable data. Private data such as emails are considered non-shareable and will be not cached. In contrast, the shareable contents are cached with a probability threshold taking into account both the node centrality and the distance to the consumer.

3.3.2 Replacement policies

NDN nodes are not able to cache the entire requested contents because of their limited size. The cache replacement decision is made once incoming content is decided to be cached, but the CS has reached its capacity. Then the node has to choose content that need to replace by the new content. LRU (Least Recently Used) and LFU (Least Frequently Used) are two popular cache replacement strategies in NDN. In LRU [83], a node performs a cache replacement by evicting cached content that has been in the least demand recently. This means that little-used cached content must be replaced with fresher and newer content. In LFU [83], when a cache replacement is required, the least popular content is evicted. To

alleviate the complexity of LFU, a Random-LFU strategy is suggested in [84]. The Random-LFU chooses at random two contents positions in the CS and evict the least frequently used content between them. The CCP (Cache Content Popularity) strategy [85] introduces a Content Popularity Table (CPT) to periodically track content popularity. The content with minimum popularity is replaced by incoming content. In [86], the k-Bloom filters are used to keep track on content requests statistics. Then, based on a sliding window scheme, contents popularities are updated to evict the content with minimum popularity. In [87], the authors try to improve the replacement decision by using an algorithm named NPA (Name Popularity Algorithm). The algorithm uses a History Table (HT), to count the popularity of the requested contents even after its eviction from the CS. In [88], the authors propose a replacement strategy named IMU (ImMature Used), on which they attempt to deal with the problem of evicting contents before it becomes popular. To deal with the problem the authors make a maturity classification for the cached content using the arrival time and the frequency of contents in a specific period to evict the least immature content from the CS. In [89, 90], the NC-SDN cache replacement strategy, using a combination of NDN and SDN (Software Defined Networking), is presented . The proposal relies on data popularity calculation using an SDN controller to replace cached data with the most popular content in each traversed node. In [91], authors propose PBCR (Price Based Cache Replacement) as a cache replacement policy for paid content. In PBCR, the most popular and expensive content cached and not replaced for a longer period. In [92], authors propose a replacement strategy based on a utility function. The proposed function is calculated based on the popularity of the file, content-download delay, and path-level congestion. Then if the utility value of the newly received content is higher than the least existing utility in the cache, the content will be cached instead of content with the least utility value.

3.3.3 Summary and comparison of existing caching strategies

As it is mentioned before, NDN knows many caching strategies which are mainly divided into two main parts (placement and replacement strategies). The Table 3.2 summarizes the most common caching strategies in the NDN field by comparing them in terms of the factor used to make the decision, objectives, advantages and disadvantages of each strategy. Reviewing the proposed strategies, we can find some that are mainly based on simple probabilistic models with the objective of reducing redundancy. Although these strategies are simple and effective in the computation time, the fact that the contents are not distinguished may lead to

the placement of unimportant contents for satisfying future requests [93]. To address this problem, most of the proposed caching strategies are based mainly on the calculation of the popularity of the requested content as an important factor for caching the received content. Although they are designed to improve network performance, popularity-based caching strategies have a common limitation, where the popularity of content is calculated based on the content name of each incoming request. However, the length of content names is not fixed and is theoretically unlimited. Also, with an ever-increasing number of users, the number of requests would be huge and the statistics need to be updated over time. The biggest limitation here is that the process of estimating the popularity of content based on names would be large. This can drain the resources of the nodes, especially for those with limited memory or processing power, which can result in a large drop in network performance [58]. There are other strategies where researchers have been more interested in classifying the data into many classes to give each class a space to cache its data. The main reason is that sharing space in caches for all types of data with giving priority to cache the most popular data will prevent the rest of the data from having a chance to be cached. We believe that this type of strategy is betters specially in the limited network environments. The main reason is that, in these strategies, many types of data can be cached. Yet, these strategies remain limited because they do not provide a robust classification model for the data and they are based only on simple classification models [80].

3.4 Evaluation tools ans factors that affect the performance of caching strategies

3.4.1 Evaluation tools

There are a variety of open-source code evaluation tools used to simulate caching strategies in various ICN architectures. Most of the available tools are oriented and designed within the CCN/NDN architecture because that it is mostly of interest to researchers and designers. The main point of difference between these tools is in the design space in terms of simplicity and realism [94]. These tools are available in multiple programming languages and for specific operating systems, as summarized in Table 3.1.

Table 3.1 Summary of Simulator Tools in NDN

Software	Operating system	Language
ndnSIM [95]	Linux	C, Python
Ns3-DCE CCNx [96]	Linux	C, Python
CCN-Lite [97]	Linux, Android, MacOS	C
CCNx [98]	Linux	C, Java
Mini-CCNx [99]	Linux	C, Python
CCN-Joker [100]	Linux, Android	Java
ccnSim [101]	Linux	C++
SocialCCNSim [102]	Linux	Python
Icarus [103]	Linux	C, Python

3.4.2 Factors that affect the performance of caching strategies during simulation

- **Topology of the network:** The performance of caching strategies is different when it is applied in different topologies [104].
- **Cache size:** It can be said that cache size is a very important factor that strongly influences the performance of network caching strategies. Using caches with small sizes leads to an increase in the number of caching operations (placement and replacement) and reduces the caching hit ratio. Similarly, the use of large cache sizes can result in a slow response time when content is fetched from large caches in the network [105].
- **Routing protocols:** The NDN has known several routing protocols designed with different mechanisms that can directly influence the performance of the proposed caching strategies [106]. For this, which routing protocol is more appropriate for the chosen caching strategy.
- **Content Population Size:** Reflects the number of distinct pieces of content for which a network could be requested. Content population size has a critical role in characterizing the performance of the cache [104].
- **Content size:** Content size also affects the performance of caching strategies during simulation. As an example, when the size of the content is very big, only a few contents will be cached in the CS, which may result in a lower cache hit ratio.

- **Popularity distribution:** Content popularity is an important factor that affects caching performance. For example, the LFU strategy gives good performance when the distribution of content popularity is not uniform, but it gives poor performance when the distribution of content popularity is uniform. The popularity distribution of contents has been introduced either by using the Zipf distribution [107] to generate a synthetic workload or by using real traffic traces [108, 109].

3.5 Open challenges in caching research

Caching in NDN has great importance for improving network performance. Even though the proposed strategies offer good performance, there are still many challenges and open issues for research on caching in NDN. Most of the proposed strategies are generally divided into placement and replacement strategies, there are some hybrid strategies but they remain limited. An appropriate and standard algorithm that will be suitable for both placement and replacement remains an open area of research. Finding new methods or techniques that can reduce the resources consumed during the process of searching for content popularity also remains an open topic in this field. Finding the false popularity content and eliminating them remains a challenge. The proposal of content compression mechanisms in the content store in a way that will not add extra overhead to the process is also an open issue. Privacy and security issues of contents must also be addressed during the caching process.

3.6 Conclusion

In-network caching is among the most important factors that characterize the NDN architecture by providing the ability to obtain data from the nearest node in the network instead of going to the source each time. The cache size in the network remains very limited compared to the huge amount of data created and consumed each time, which requires good cache management to efficiently handle the cached data. Several caching strategies have been proposed in this regard as solutions to achieve the highest possible efficiency in the network by taking advantage of specific characteristics from the network or from consumed content. This chapter presents the benefits of the use of caches in NDN. Then, the most known strategies related to cache management are presented and classified to make clear their differentiation for decision making and to clarify their advantages and drawbacks. In

addition, the most commonly known tools for evaluating caching strategies in NDN have been presented. Similarly, the most important factors related to NDN that can affect the performance of caching strategies during simulation have also been discussed. Finally, the main research trends and challenges of content caching strategies have been discussed.

Table 3.2 Summary and Comparison of Existing Caching Strategies

Strategy	Placement	Replacement	Decision to cache based	Objective	Advantages	Inconvenient
LCE [23]	*		No	cache all requested contents in the returning path	Simplicity , Quick content distribution	High I/O consumption, No content distinction
Prob(p) [57]	*		Fixed parameter	Reduce redundancy	Simplicity	No content distinction
LCProb [58]	*		Number of hop	Reduce redundancy, place more content close to consumers	Simplicity	No content distinction
ProbCache [59]	*		Path caching capability using the number of hops	Place more content close to the content source, reduce redundancy and load on the servers	Distributes content efficiently along the path	No content distinction
OCPCP [61]	*		Popularity	Cache popular contents	Contents distinction	Calculate popular contents by their names
MPC [62]	*		Popularity	Cache the popular contents on the local nodes an over their neighbour's	Contents distinction	Threshold definition, Calculate popular contents by their names
LCD [63]	*		Popularity	Progressively cache content on the network	Reduce redundancy	take more time to get contents
WAVE [64]	*		Popularity	Progressively cache content over the path	Reduce redundancy	Require more time to access popular content, Calculate popular contents by their names
Noor et al. [65]	*		Popularity	Progressively cache content over the path	Reduce redundancy, Place more content close to the content source	Calculate popular contents by their names
P-CBC [66]	*		Popularity + Centrality of nodes	Place popular content in nodes with high centrality	Reduce redundancy	Calculate popular contents by their names
BEP [67]	*		Popularity + Centrality of nodes	Place popular content in nodes with high centrality	Reduce redundancy	Calculate popular contents by their names
YIQI GUI et al. [71]	*		Popularity + Centrality of nodes	Place popular content in nodes with high centrality	Reduce redundancy	Calculate popular contents by their names
EHCP [68]	*		Popularity + Centrality of nodes	Place popular content in the core nodes and less popular content in the edge nodes	Reduce redundancy, Content variation	Calculate popular contents by their names

Table 3.3 Summary and Comparison of Existing Caching Strategies

Strategy	Placement	Replacement	Decision to cache based	Objective	Advantages	Inconvenient
CPCCS [69]	*		Popularity + Centrality of the nodes + number of hops + threshold values	Place popular content in the core nodes and less popular content in the edge nodes	Reduce redundancy, Content variation	Calculate popular contents by their names
YIQI GUI et al. [70]	*		Popularity + entropy weighting model and TOPSIS	cache popular on the optimal location in the path	Reduce redundancy, Efficient content distribution	Calculate popular contents by their names
LdC [72]	*		Cache hit + Hierarchical name of contents	place popular contents in the most important nodes in the path	Reduce redundancy, Efficient content distribution	Calculate popular contents by their names
CnS [73]	*		Popularity	Place popular content closer to consumers and keeps less popular content in the core routers	Reduce redundancy	Calculate popular contents by their names
PDPU [74]	*		Popularity	Place popular content closer to consumers and keeps less popular content in the core routers	Reduce redundancy	Calculate popular contents by their names
SMDP [75]	*		Popularity + Semi-markov decision process	Place the requested contents in the best forwarding path	Reduce the delay	Calculate popular contents by their names
Li et al. [76]	*		Popularity + Game theory	Improve the resource allocation in the network	Reduce the delay, Efficient content distribution	Calculate popular contents by their names
Yuan et al. [77]	*		Popularity	Cache popular content, achieve a balanced resource allocation	Reduce the delay	Calculate popular contents by their names
PCSD [78]	*		Popularity + Network level information	Place popular content close to the consumers	Reduce redundancy	Calculate popular contents by their names
Ehsanpour et al. [79]	*		Popularity + Game theory	Improve cache utilization in ISP networks	Efficient content distribution	Calculate popular contents by their names
Huo et al. [80]	*		Popularity + Name categorization	Cache contents from popular category	Content diversity	Simple classification model
CAPIC [81]	*		Content class	progressively cache the content	Reduce redundancy, Content diversity	Simple classification model
MAPC [82]	*		Name categorization + Centrality of nodes + Number of hops	place popular content in the core router	Reduce redundancy, Content diversity	Simple classification model

Table 3.4 Summary and Comparison of Existing Caching Strategies

Strategy	Placement	Replacement	Decision to cache based	Objective	Advantages	Inconvenient
LRU [83]		*	Recency	Replace least used content	Simplicity	No content distinction
LFU [83]		*	Popularity	Replace the less frequent used contents	Simplicity	Calculate popular contents by their names
Random-LFU [84]		*	Popularity	Randomly replace the less frequently used contents	Simplicity, time efficient	Calculate popular contents by their names
CCP [85]		*	Popularity	Replace the less popular contents	Track the popularity of contents even when it is evicted, simplicity	Calculate popular contents by their names
Dai et al. [86]		*	Popularity + K-Bloom filter	Keep track on content requests statistic, Replace less frequent contents	Time efficiency of popularity calculation	Calculate popular contents by their names
NPA [87]		*	Popularity	Replace the less popular contents	Track the popularity of contents even when it is evicted, simplicity	Calculate popular contents by their names
IMU [88]		*	Popularity + Arrival time	Deal with the problem of evicting contents before it becomes popular	Track efficiently popular contents	Calculate popular contents by their names
Anwar et al. [89]		*	Popularity	Replace the less popular contents using SDN	Simplicity	Calculate popular contents by their names
Anwar et al. [90]		*	Popularity	Replace the less popular contents using SDN	Simplicity	Calculate popular contents by their names
PBCR [91]		*	Popularity + Prize of contents	Replace the less popular and inexpensive contents	Uses new metrics in replacing contents	Calculate popular contents by their names
Badov et al. [92]		*	Popularity + Content-download delay + Path congestion	Reduce the delay	Uses new metrics in replacing contents	Calculate popular contents by their names

Chapter 4

Fundamental notions about the techniques involved with our proposed caching strategies

4.1 Introduction

Nowadays data mining has a wide scope of application in almost all industries where data is generated, that is why data mining is considered one of the most promising interdisciplinary developments in information technology. Because of their importance, several data mining techniques have been developed for knowledge discovery. Another technique that has had a profound impact on improving performance in many areas such as transportation, energy and especially critical resource allocation is the constrained optimization (CO) methods [110]. Therefore, the use of data mining techniques and CO methods for caching in NDN can improve the overall network performance.

This chapter aims at presenting the basic notions concerning the techniques used in the next chapters, in particular the technique used for the text classification along with the constrained optimization method used to address the tackled problems. This presentation is not only necessary to show the interest of the chosen techniques, but also to understand the rest of our work. In this chapter, we present a quick and clear overview of the techniques used. First, we will briefly present the technique and concepts related to the Multinomial Naïve Bayes method for text classification. Then, we present the Lagrange method, which will be used in the future chapter for resource allocation in the network.

4.2 From data to knowledge

Data mining techniques especially text classification and document categorization approaches are generally decomposed into four main phases: Feature Extraction, Dimension Reduction, Classifier Selection and Evaluation as shown in Figure 4.1.

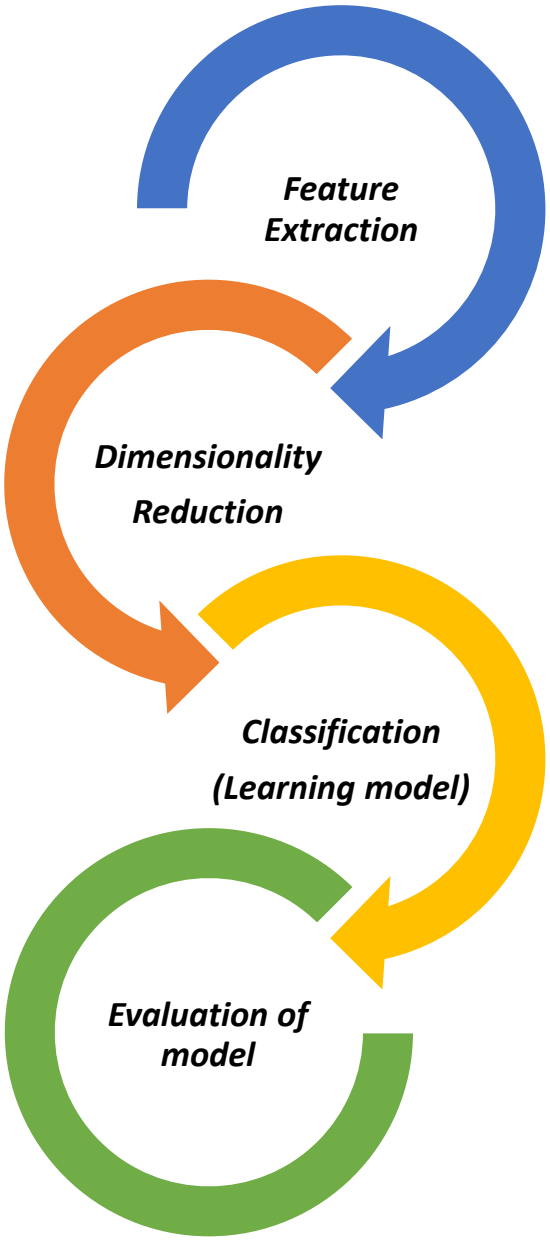


Fig. 4.1 Principal phases for text classification

The feature extraction phase: the texts and documents are generally unstructured data sets. So the goal of this phase is to convert the unstructured text sequences into a structured

feature space when using mathematical modeling in a classifier. The data must first be cleaned to eliminate unnecessary characters and words that may disturb the classification model and add unnecessary computation time (e.g. stop words such as "and, or, the..."). Once the data is cleaned, formal feature extraction methods can be applied. Among the common feature extraction techniques used is the $n-gram$ technique [111] which represent a set of n words that appear "in the same order" in a set of texts to extract more features that could give more accuracy in classifying the text. It is very common to use both uni-grams and bi-grams or sometimes tri-grams because the selection of a single word is not always sufficient to understand the classes of the data. Thus, two words can give more correct features and can give another meaning that helps to understand the class of data. An example bi-gram of the link "$www.univ-lagh.dz$" after removing the stop words "www", "." and "$-$", is: $\{univ\,lagh, lagh\,dz\}$.

Frequency and Inverse Document Frequency ($TF-IDF$) [112] is also one of the powerful techniques that can define the weights of the words more precisely and select the more appropriate features for classification. This technique calculates the importance of the word at the name level and also in the whole training dataset; the $TF-IDF$ can be defined as follows:

$$TF-IDF = TF * IDF \tag{4.1}$$

where TF represent the occurrence of vocabulary in the whole requested name, and IDF represent the inverse of the frequency in the whole document. The IDF is important since it is shown that the terms that have a high frequency in the whole document are not important in the determination of the classes. IDF is calculated as follows:

$$IDF = \log \frac{|D|}{|\{Dataset : t \in Dataset|\} + \alpha} \tag{4.2}$$

where $|\{Dataset : t \in Dataset|\}$ represent the number of the requested names where the term t appears, $|D|$ represent all the requested names or links in the training dataset. The parameter α represent the used smooth parameter, because sometimes we may receive requests that contain new words that have not yet been seen in the training dataset.

Dimensionality reduction phase: this phase is optional since there are some techniques that allow to overcome this phase like Word2Vec [113] and Bert [114]. This phase is mainly used because the textual datasets are often contain many unique words, where each word may represent a very important feature in the used classification model. Therefore the data

pre-processing step can be complex in terms of time and memory. The objective of this step is to reduce the time and memory complexity of their applications. Using dimensionality reduction for pre-processing could be more efficient than developing cost-effective classifiers [115]. One of the most known technique for dimensionality reduction in text classification is the Compressed Sparse Row (CSR) [116]. The sparse matrix is known as a matrix that has a maximum of elements equal to zeros. In text classification the sparse matrix is always produced because a name or a document does not use all the existing vocabulary. In CSR, the matrix will be compressed into three tables that represent respectively the non-zero values, the extents of the rows, and the indices of the columns.

Classification techniques phase: this phase represents the most important phase in text classification, where it consists mainly in selecting the best classifier. The data mining area has known many text classification techniques such as the Naïve Bayes classifier (NBC), Logistic Regression (LR) [117], Support Vector Machine (SVM) [118], Neural networks [119], etc. Among these models, the Multinomial Naïve Bayes Classifier (NBC) has been proved that generally gives good results in the classification of URLs compared some other techniques like SVM and LR[120], which makes it a good candidate for solving the next problems. Moreover, the Naïve Bayes classifier is computationally inexpensive and also requires a very small amount of memory [121], which makes it an effective solution for dynamic environments such as NDN. More detail about the used model is in the subsection 4.3.

The evaluation phase: represents the last phase of text classification, which includes the metrics and techniques used to evaluate the performance of a model. There are many methods to evaluate supervised techniques. The calculation of Accuracy is the most common evaluation method, there are other methods such as Recall, Precision and F-Measure [122].

4.3 Multinomial Naïve Bayes Classifier

As mentioned above, when the features are extracted with their weights, the received data can be classified using one of the chosen classifier methods. The Naive Bayes classification method is one of these methods, which represents a supervised learning algorithm based on Bayes' theorem, defined as follows:

$$P(c \mid d) = \frac{P(c)P(d \mid c)}{P(N)} \tag{4.3}$$

where c, d represent respectively the class and the received data, $P(c \mid d)$ is the probability of the class c given the received data, $P(d \mid c)$ is the probability of the data appearing in a sample belonging to class c. $p(c)$ and $p(d)$ represent respectively the probability of c and d in the sample. Since the received data is composed of several terms t we can define it as follows:

$$P(c \mid t_1, \ldots, t_n) = \frac{P(c)P(t_1, \ldots, t_n \mid c)}{P(t_1, \ldots, t_n)} \tag{4.4}$$

The Naïve Bays assume that the conditional independence of feature probabilities are independent and can be defined as:

$$P(c \mid t_1, \ldots, t_n) = \frac{P(c)\prod_{i=1}^{n} P(t_i \mid c)}{P(t_1, \ldots, t_n)} \tag{4.5}$$

also, it can be defined in the equation below:

$$P(c \mid t_1, \ldots, t_n) \propto P(c)\prod_{i=1}^{n} P(t_i \mid c) \tag{4.6}$$

$P(t_1, \ldots, t_n)$ is dropped because it is a constant value. Finally, the Naïve Bays can assign the data to the class (c_{map}) that has the highest probability such as:

$$c_{map} = \operatorname{argmax} P(\hat{c} \mid d) = \operatorname*{argmax}_{c} P(c)\prod_{i=1}^{n} P(t_i \mid c) \tag{4.7}$$

the different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of $P(t_i \mid c)$. In this work, we use the Multinomial distribution because it gives a good results compared to other distribution like Gaussian and Bernoulli in the text classification [123]. The Multinomial distribution captures the term frequencies in documents and cares about multiple features that are frequently occurred [124], the distribution can be formulated as:

$$P(c \mid t_i) = \frac{count(t_i, c) + \alpha}{count(t, c) + \alpha n} \tag{4.8}$$

where $count(t_i, c)$ is the number of times a feature or term appears in a sample of class c in the training dataset, $count(t, c)$ is the total count of all features for class c. α represent the smoothing parameter and n is a constant value that represents the number of terms that exist in the model.

4.3.1 Illustrative example

let's assume a dataset that contains multiple web links, as shown in the example of Table 4.1, where we are looking to create a model that can automatically estimate the category of new incoming links.

Table 4.1 Data-set example

	LinkID	**words in the Link**	**Category**
Training set	1	$A/B/A$	Cat_1
	2	$A/A/C$	Cat_1
	3	A/D	Cat_1
	4	$E/F/A$	Cat_2
Test set	5	$A/A/A/E/F$	?

First of all, we remove the delimiter "/" because in this case, its consideration will have no importance for the classification of the tested link and it will only add extra computation time. Second, in order to classify the test link, the multinomial parameters that we need are the priors $\hat{P}(Cat_1) = 3/4$, $\hat{P}(Cat_2) = 1/4$, and the conditional probabilities are:

$$\begin{aligned}
\hat{P}(\text{ A } \mid Cat_1) &= (5+1)/(8+6) = 6/14 = 3/7 \\
\hat{P}(\text{ E } \mid Cat_1) = \hat{P}(\text{ F } \mid Cat_1) &= (0+1)/(8+6) = 1/14 \\
\hat{P}(\text{ A } \mid Cat_2) &= (1+1)/(3+6) = 2/9 \\
\hat{P}(\text{ E } \mid Cat_2) = \hat{P}(\text{F} \mid Cat_2) &= (1+1)/(3+6) = 2/9
\end{aligned}$$

The denominators are equal to $(8+6)$ and $(3+6)$ in this example because the lengths of $text_{Cat_1}$ and $text_{Cat_2}$ are respectively 8 and 3, and also because the constant n in equation 4.8 is equal to 6 as the vocabulary is composed in this case of six terms. After the calculation we obtain the following results :

$$\begin{aligned}
\hat{P}(Cat_1 \mid link_5) &\propto 3/4 \cdot (3/7)^3 \cdot 1/14 \cdot 1/14 \approx 0.0003. \\
\hat{P}(Cat_2 \mid link_5) &\propto 1/4 \cdot (2/9)^3 \cdot 2/9 \cdot 2/9 \approx 0.0001
\end{aligned}$$

Thus, the classifier will classify the tested link into category one ($Category = Cat_1$). This classification decision was made because the three occurrences of the category Cat_1 in $link_5$ outweigh the two occurrences of the category Cat_2 E and F.

4.4 Lagrange multiplier method

Optimization methods represent an important concept in many fields such as applied sciences, health and also in economics. Today, optimization is at the heart of artificial intelligence and many studies try to solve their problems by using different optimization methods. One of the most common problems in this field is to find the maximum or minimum of a function, especially when maximizing or minimizing a function under some fixed conditions or constraints. Among the powerful proposed method to solve this class of problems is the Lagrange multiplier method which is proposed by the mathematician Joseph-Louis Lagrange $(1736-1813)$. The Lagrange method represents a simple solution that does not require complex calculations and gives optimal real results. To find the optimal solution using the Lagrange method, it is enough to determine the objective function with the variables of the constraint than knowing the derivatives of the derivable functions.

4.4.1 Mathematical modeling using the Lagrange method

The Lagrange multiplier method is a well-known technique for solving constrained optimization problems in which the optimal points $X^* = (x*, y*, ...)$ in a multidimensional space that locally optimizes the multi-variable objective function $f(x, y, ...)$ with subject to the constraint of another multi-variable function equals a constant $g(x, y, ...) = c$. Hence, to solve this type of problem, the following steps should be followed (figure 4.2): :

- First, we will introduce a new variable λ, and we define a new function L as follows:

$$L(x, y, \ldots, \lambda) = f(x, y, \ldots) - \lambda(g(x, y, \ldots) - c) \tag{4.9}$$

  where, L is called the "Lagrangian" function, and the additional variable λ is called the "Lagrange multiplier".

- Secondly, we calculate the gradient of L where it is equal to the zero vector to find the critical point of L.

$$\nabla L(x, y, \ldots, \lambda) = \mathbf{0} \leftarrow \textit{Zero vector} \tag{4.10}$$

  To verify the extremum points (maximum or minimum), we have to compute the Hessian matrix [125], which corresponds to the secondary derivatives of the Lagrangian expression.

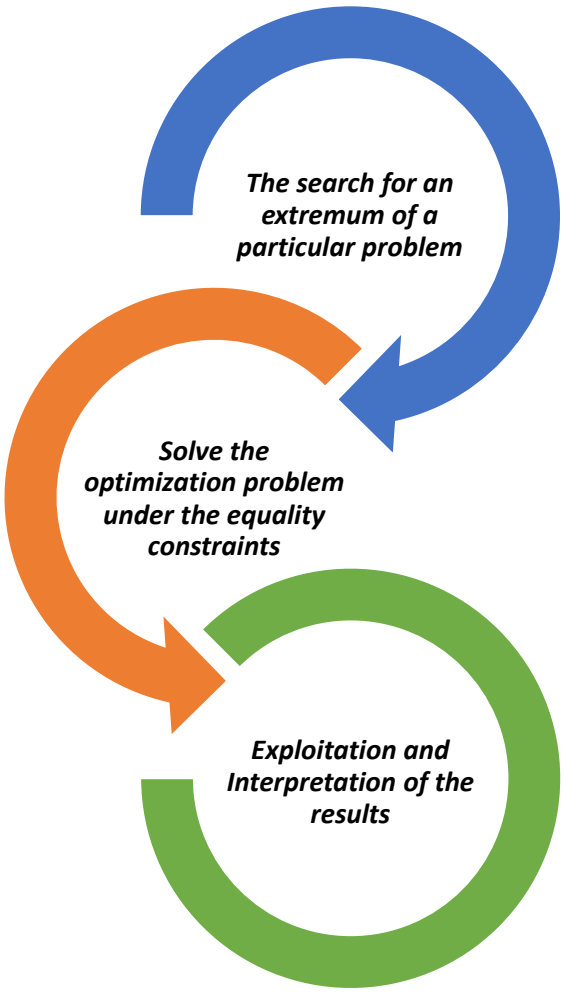


Fig. 4.2 Process of Lagrange method

- Finally, the obtained results $(x_0, y_0, \ldots, \lambda_0)$ can be directly exploited as optimal solutions to the considered problem.

4.5 Conclusion

This chapter has illustrated the universal characteristics and the practical interest of the exposed techniques, which motivates their use both for modeling and for solving the studied problems. At first time this chapter introduces the basic notions of data mining, focusing mainly on the technique of the text classification by the Naive Bayes method. Then, the Lagrange method which represents one of the most known constrained optimization techniques has been presented. These two methods are proved that require little computation time and

memory, which motivate their application for solving the addressed problems in the next chapters.

Chapter 5

Data categorization as a new caching strategy in the NDN

5.1 Introduction

Indisputably, the Internet is experiencing a communication explosion and a significant shift in usage. Originally conceived to enable communication between hosts, the Internet is becoming more and more used for content distribution and retrieval. This is mainly due to the ever-growing use of content-oriented applications such as social media networking and Video on Demand (VoD) services [126, 127]. Consequently, Internet users increasingly expect instant, high-quality connectivity so those very demanding applications (e.g video related applications) become fast enough. This raises the concern about how to efficiently store and distribute huge content, particularly when many users request the same content at the same time. Here, it is worth noting that today's Internet architecture does not serve this purpose efficiently and has become one of the key bottlenecks that limit performance and scalability. In the current Internet architecture, a user has to specify a host fixed location to retrieve the content. Thus, the requested contents will repeatedly be transmitted from the hosts (servers) to the users which incurs a waste of bandwidth and increases the delay of content delivery. To overcome this inefficiency, content-oriented networking technologies have been proposed as an attempt to accommodate content distribution within the current Internet infrastructure [128].

Named Data Networking is an example of such a technology which become an active and challenging field of research, as it is considered one of the most prominent candidates for

future Internet architecture [129, 25]. The intuition behind NDN is that users are typically interested in the content itself without concern for the physical location of the data. Therefore, NDN has a distinctive feature in which fetching for content, from an initially unknown location, is done through the name of the content rather than its address, which is more appropriate to users' practices. Similar to URLs, content names in NDN are hierarchically structured. For instance, a video of a lecture in computer science produced by the university Amar Telidji of Laghouat might take the form /uatl/cs/network/videos/ndn.mpg.

As compared to legacy networks, a key feature in NDN is the in-network caching where the contents are decoupled from their physical locations [23, 130]. Thus, redundant copies of the same content are geographically dispersed with the aim to be closer to users. This ensures a scalable and effective approach to managing contents as it reduces network bandwidth and congestion. It in turn alleviates the workload of the contents producers as well as the workload in each node in the network.

So, in NDN architecture nodes (routers) have a twofold responsibility as they perform data forwarding and they can also cache the requested content independently of the original producer. Receiving a data request, an NDN node responds with a locally-cached copy, which has been opportunistically captured as the data previously flowed through [131]. Thus, a subsequent request for the same content may be satisfied from a closer NDN node instead of reaching host servers (actual producers of the content). Obviously, this significantly benefits network users by reducing the transfer of duplicate data, wherever possible, which improves accessibility and searchability of contents [132, 133].

Though the significance of content caching in the network is undoubtedly large, challenges are also complex. Caching capacity in routers is relatively small and cannot hold all the contents requested by users. Therefore, cache space management becomes critical to address and maintain [134, 135]. The main issue that NDN-caching still facing is how each router takes a decision to cache a requested content to produce maximum subsequent responses and achieve consequently optimal performance. It can be inferred that the network performance heavily relies on robust caching mechanisms as the success of content delivery will be affected by cache hit and miss rates [136, 137].

Caching management in NDN nodes concerns two important policies that are placement and replacement of contents [138]. The placement policy (known also as cache insertion) decides where and what contents should be stored on the network. When the cache becomes

full, the replacement policy (known also as cache eviction) refers to the process of removing old content to make space for new content [139].

A naive solution to NDN-caching will save requested contents at each node of the network. Such a strategy incurs high contents redundancy and it is, therefore, impractical. Therefore, several NDN-caching strategies have been proposed in the literature. Often, the proposed solutions rely on the rational intuition that it is more useful to cache the most popular contents, where estimating the popularity of content ranges from simple statistical counts to probabilistic models.

A common limit in existing approaches is that the popularity of content is calculated based on the user request (the full name of the content). However, the lengths of content names are not fixed and are theoretically unbounded. Thus, in each network-node, the popularity of a content is computed considering a potentially huge number of contents names especially since new ones are occurring every time. This may add a burden on the computing resources available at each node. Consequently, the calculation of contents popularity might be significant and can cause serious service-delays. Moreover, users' requests typically share common hierarchies (e.g. prefixes) as there are much content names associated with one domain of interest (e.g. sport, games,...etc) [140]. Existing solutions do not capitalize such information and cannot therefore perform a prediction on previously unused requests.

To overcome this inefficiency, we propose a novel approach named CaDaCa (Categorized Data for Caching) to enhance cache management in NDN. We exploit the semantic value of the hierarchical scheme of the contents names to automatically decompose the users' requests into a set of a domain of interest or categories. For instance, through the name "/playstation/games/demons-souls", it is easy to observe that the requested content belongs to the "games" category by looking at the two words "playstation" and "games". Thus, users' requests are described in a smart expressive way and the popularity of content is expressed as the popularity of its category.

Therefore, we consider the NDN-caching issues as a supervised learning task and we demonstrate how the Bayes classifier, a powerful and proven machine learning technique, can be used to solve them. The proposed approach allows computations for popularities to be made efficiently as the number of categories is far much smaller than the number of content names. On the other hand, this allows to handling previously unused requests by simply predicting their categories. Moreover, the contents categorization provides the

network operator with an overview of traffic usage with in-depth knowledge about users' behavioral information which is the keystone in security issues in NDN.

The remainder of this chapter is organized as follows. In Section 5.2, we motivate our proposed solution. Then we give a detail of our proposal in Section 5.3. In Section 5.4, we study the complexity of our strategy. In Section 5.5, we evaluate the performance of our proposal and we analyze the obtained results. Finally, we finish our chapter with a conclusion in Section 5.6 .

5.2 Motivation

Table 5.1 Percentage of requests by category in the DMOZ dataset

Category	**Number of requests**	**Pourcentage (%)**
Adult	35325	2.26
Art	253840	16.24
Business	240177	15.37
Computers	117962	7.55
Games	56477	3.61
Health	60097	3.85
Home	28269	1.81
Kids	46182	2.95
News	8989	0.58
Recreation	106586	6.82
Reference	58247	3.73
Science	110286	7.06
Shopping	95270	6.10
Society	243943	15.61
Sport	101328	6.48

To motivate our proposal, let us consider the example shown in Table 5.1, which shows categorized web pages using a real-world trace from the Open Directory Project (also called DMOZ) [141]. DMOZ is the most widely distributed database of Web content categorized by humans [141]. It is used for research purposes in various areas, particularly with popular search engines [142]. Table 5.1 illustrates web page topic categorization from URLs on a large collection of 1.5 million web pages which are categorized into 15 categories. As expected, the number of categories is by far much smaller than the number of users requests. In practice, many requests are typically associated with a domain of interest (category) such

as sport, games,...etc. This implies that using a name-based process to maintain statistics on users' requests would be computationally expensive, especially with a relatively large number of requests. To achieve effective cache management, we believe that the vast amount of users' requests have to be well described and organized. To this end, machine learning provides a powerful tool to automatically categorize contents names. Thus, computations for contents popularity are maintained efficiently by consolidating, in a small number of categories, the contents that have similar topics together, and the popularity of content is expressed as the popularity of its category. One more benefit is that users requests categorization provides automatic users behavior modeling making our approach adaptive. Thus, new requests can be handled by accurately predicting their categories.

5.3 An overview of CaDaCa approach

5.3.1 System model

As in autonomous systems, we divide the network into areas, where each area consists of a coordinator node, a set of Edge Routers (ER) and a set of Core Routers (CR). This provides a global view of the complete area to the coordinator node. For instance, Figure 5.1 illustrates a network with 2 coordinators, 8 ER and 4 CR.

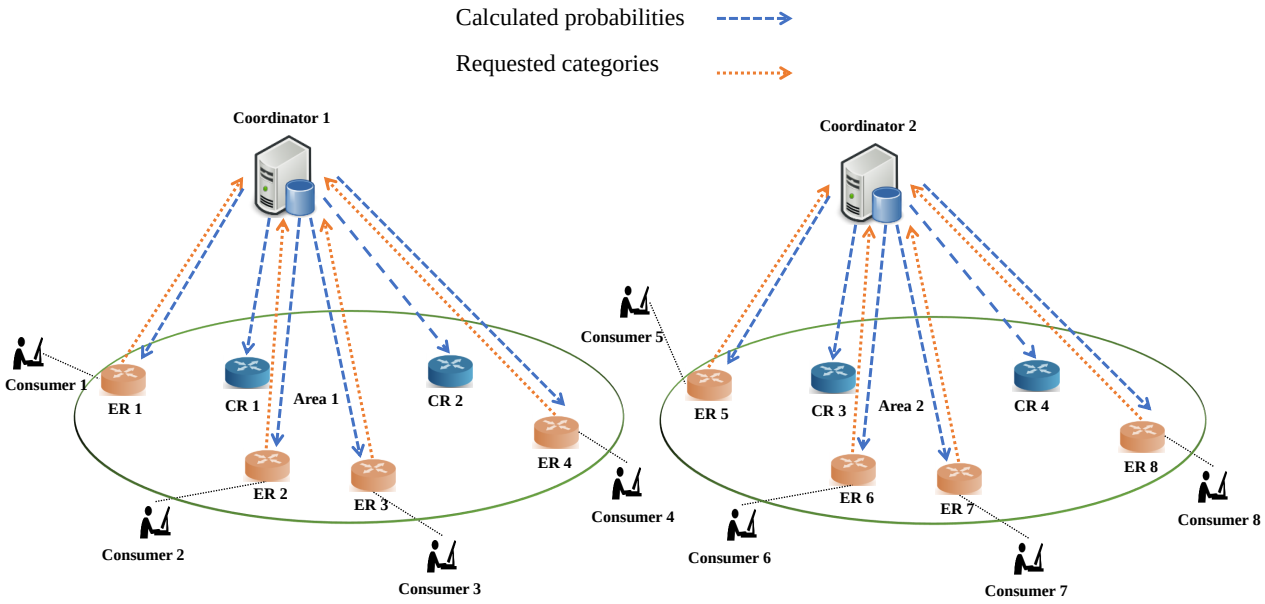


Fig. 5.1 Our system model

We assume further that the content producer specifies the category of the content by reserving 1 byte in the content field. Therefore, our model will use just the name of the content to identify its corresponding category. With this way, the caching decisions are made on the returning path based on the categories popularity. Finally, ERs continuously communicate the contents category's to the coordinator node. Such information is used to maintain the category's popularity.

5.3.2 Content placement and replacement in CaDaCa

As mentioned above, the principle of CaDaCa is to focus on contents categories instead of contents names. Thus, in each area, the coordinator node estimates the probability of consuming a given category during the period $[t-1,t]$ as follows:

$$Pr(c_i) = \frac{Cardinality(c_i \in H([t-1,t]))}{Cardinality(C \in H([t-1,t]))}, \quad c_i \in C = \{c_1, c_2, .., c_n\}, \tag{5.1}$$

where $H([t-1,t])$ represents the history of categories in the period $[t-1,t]$. C and $Pr(c_i)$ denote respectively the set of categories and the probability assigned to the category c_i. To ensure that the most requested categories will be cached, each node in the area will store content belonging to a category with probability $Pr(c_i)$. On the other hand, nodes cannot cache a copy with $1 - Pr(c_i)$ (Fig. 5.5). Given a requested content l, the placement decision in CaDaCa can be formulated as:

$$Placement(l) = \begin{cases} 1, & if \;\; rand(n) \leqslant Pr(c_l) \\ 0, & else \end{cases}, \quad c_l \in C \;\; and \;\; n \in [0,1], \tag{5.2}$$

where c_l and C denote respectively the category of content l and set of all categories. The function $rand(n)$ randomly generates a number n between 0 and 1. $Pr(c_l)$ is the computed probability corresponding to the content category l, and it also represents a threshold for placing the received content. With this mechanism, the data packet is cached in the CS if $Placement(l) = 1$, otherwise the content is forwarded downstream to the consumer without being cached.

On the other hand, to ensure an effective replacement strategy, CaDaCa divides the CS into n partitions, each one corresponding to a given category. The size of a partition is proportional to the probability of the corresponding category as illustrated in the workflow of

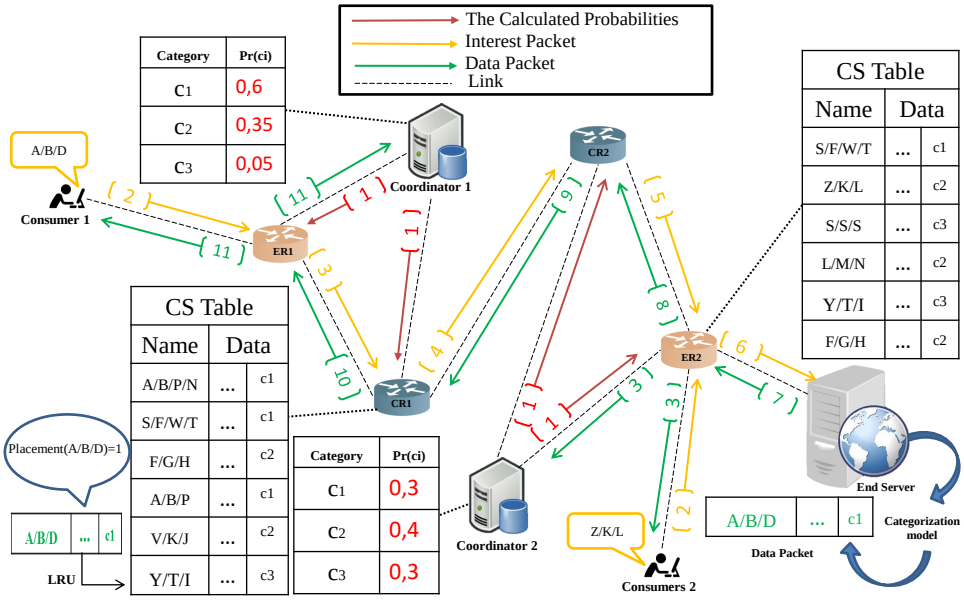


Fig. 5.2 Example of content placement in CaDaCa

Figure 5.4. More formally, the size of a partition c_i is estimated as:

$$Partition(c_i) = [Pr(c_i) * Sizemax], \quad c_i \in C = \{c_1, c_2, .., c_n\}, \tag{5.3}$$

5.3.3 Illustrative example

The purpose of this subsection is to illustrate how the CaDaCa approach works. We consider a simplified scenario in Figure 5.2 and Figure 5.3 to illustrate the placement and replacement mechanism. Assume that the estimated probabilities for the given categories on the coordinators 1 and 2 are consecutively $\{c1 = 0,6; c2 = 0,35; c3 = 0,05\}$ and $\{c1 = 0,3; c2 = 0,4; c3 = 0,3\}$. At time t1, coordinators send the calculated probabilities to the nodes of the same area. The received probabilities are then used according to the chosen strategy. Figure 5.2 uses the received probabilities to make a placement decision on the future received content using equation (5.2). Figure 5.3 uses the received probabilities to partition the CS to replace the content using equation (5.3). For both scenarios Consumer 1 and Consumer 2 request content A/B/D and Z/K/L at time t2. Consumer 2 is immediately satisfied by ER2 at time t3 because a local copy of the requested content is cached in ER2. Consumer 1 is satisfied by the end server because there is no cached copy of the requested content on the path.

In Figure 5.2, the content requested by consumer 1 is not placed in ER2 because the placement function calculated for the A/B/D content is equal to 0. When the requested content is forwarded to CR1 at time t9, the router caches a local copy because the category of A/B/D is likely to be cached in this router. The A/B/D content caching will automatically replace the least recently used data (Y/T/I).

In Figure 5.3, when the requested A/B/D content follows the return path, a local copy will be inserted into the partition with the same category as A/B/D at the moment where the CS is not full (CR1), or replace the least used data inside the partition of the same category (ER2).

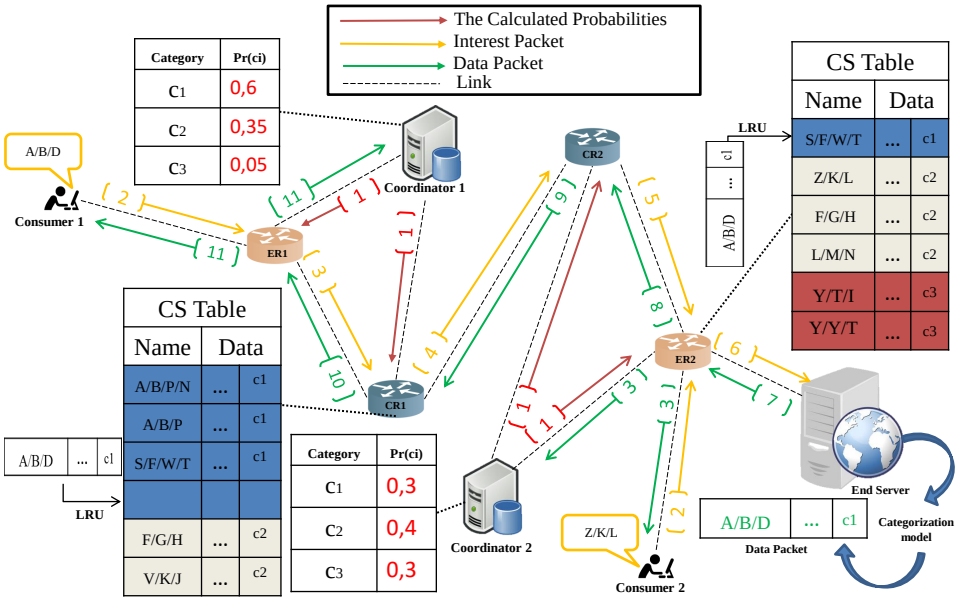


Fig. 5.3 Example of content replacement in CaDaCa

5.4 Complexity study

In this section, we quantitatively estimate the computation time of an NDN router by analyzing the packet processing flow under the two proposed caching strategies, based on the methodology used in [143]. We additionally calculate the computation time at the coordinator side and at the final producer side.

5.4.1 Packet flow with CaDaCa replacement strategy

In this subsection, we describe the processing flow of Interest/Data packets used in our replacement strategy for both hit and miss cases as shown in Figure 5.4. The flow of our strategy differs from the naive cache mechanism [143] by two additional blocks B8, B10 and by a different replacement mechanism of the data presented by block B11. B11 use the same naive algorithm (LRU or LFU) to evict only the data of the same category as described in subsection 5.3. If the data comes from the coordinator, B10 uses the received probabilities and puts them to updates the categories partitions in the CS. Otherwise, B8 checks the category of the received data packet. After B11 selects and evicts a victim from the cache partition of the same category. We group the blocks so that the average CPU cycles spent to process a pair of interest and data packets are calculated with two parameters, the cache hit rate and the cache insertion rate, as below:

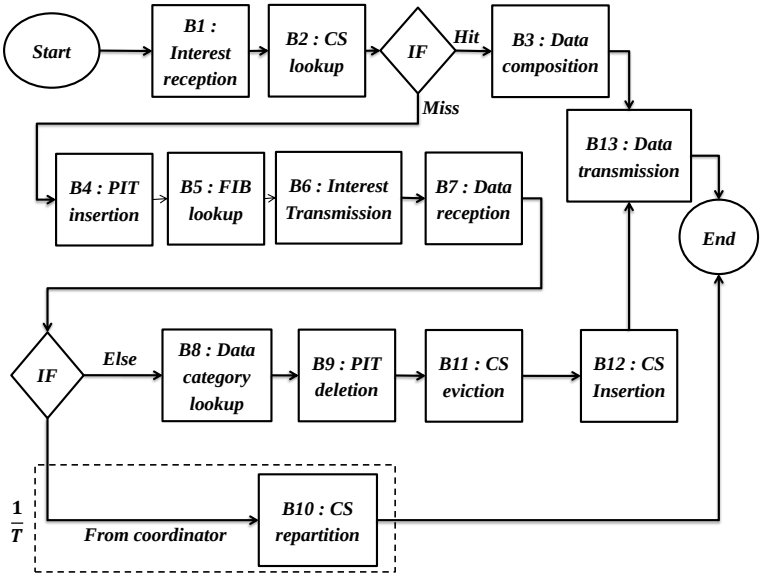


Fig. 5.4 Workflow of CaDaCa replacement strategy

G1 is the set of blocks executed always for each Interest packet: B1, B2 and B13.

G2 is the block executed at a cache hit: B3.

G3 is the set of blocks executed at a cache miss except for the blocks for cache eviction (i.e., B11, B12): B4, B5, B6, B7, B8 and B9.

G4 is the block executed at the reception of the data packet from the coordinator: B10.

The average CPU cycles spent for processing a pair of an Interest and Data packet $C^{\text{Avg}}_{\text{replacement}}$ are defined by the Eq. (5.4), where Pr^{Hit} is the probability that Interest packets

hit the cache, $C_{b,i}$ is the CPU cycles of block B_i, and $C_{g,i}$ is the CPU cycles of group G_i. $1 - Pr^{Hit}$ is the probability that Interest packets miss the cache and $\frac{1}{T}$ represent the periodic time of receiving a data packet from the coordinator.

$$C^{\text{Avg}}_{\text{replacement}} = C_{g,1} + \Pr^{\text{Hit}} C_{g,2} + \left(1 - \Pr^{\text{Hit}}\right)\left(C_{g,3} + C_{b,11} + C_{b,12}\right) + \frac{1}{T} C_{g,4} \tag{5.4}$$

5.4.2 Packet flow with CaDaCa placement strategy

The packet processing flow of the placement strategy is shown in Figure 5.5. The flow in the case of a cache hit is the same as that with the cache replacement strategy (Figure 5.4). In the case of a cache miss, the flow with the proposed cache placement is different from those with cache replacement. First, block B14 is used to make the placement decision for the received data packet based on its category. Second, block B10 has a different role than the block in the replacement strategy, since in this case it is used only to update the received probabilities of each category to improve the placement decision making. Finally, blocks B11 and B12 are executed only when B14 decides to insert a data packet into the CS. Note that in this strategy, B11 uses a naive replacement strategy as in [143].

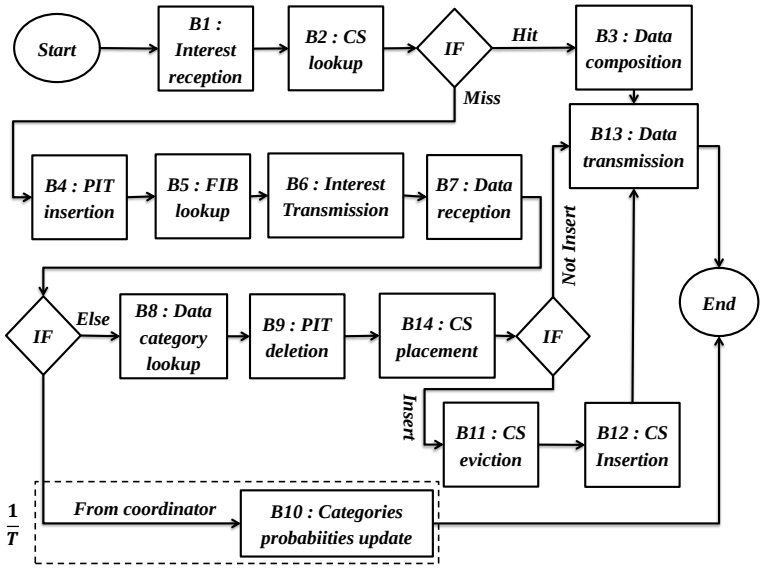


Fig. 5.5 Workflow of CaDaCa placement strategy

The average CPU cycles $C^{\text{Avg}}_{\text{placement}}$ are defined by Eq. (5.5), where $Pr^{Insertion}$ is defined as the probability that data packets are inserted into the cache.

$$C^{Avg}_{Evic} = C_{g,1} + Pr^{Hit} C_{g,2} + \left(1 - Pr^{Hit}\right)\left(C_{g,3} + C_{b,14}\right) + Pr^{\text{Insertion}}\left(C_{b,12} + C_{b,12}\right) + Pr^{\text{Coordinator}} C_{g,4} \tag{5.5}$$

5.4.3 Coordinator and producer flow

The packet processing flow on the coordinator side is shown in Figure 5.6. First, the B1 block receives and decodes the data packet received from the edge routers within its area. Then, block B2 has the role of calculating the probabilities of each category based on the previously received data. Finally, B3 sends the data packet to the routers in its area, which will be done periodically. The average CPU cycles $C^{Avg}_{\text{Coordinator}}$ are defined by Eq. (5.6).

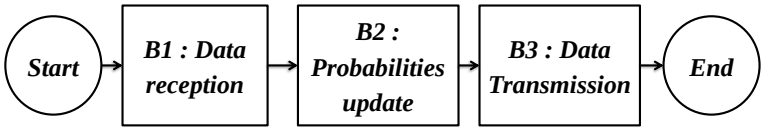


Fig. 5.6 Coordinator flow

$$C^{\text{Avg}}_{\text{Coordinator}} = C_{b,1} + C_{b,2} + \frac{1}{T} C_{b,3} \tag{5.6}$$

The packet processing flow with the final producer side is also illustrated in Figure 5.7, where, the average CPU cycles spent for processing the incoming interest and data packet $C^{\text{Avg}}_{\text{Final Producer}}$ are defined by Eq. (5.7).

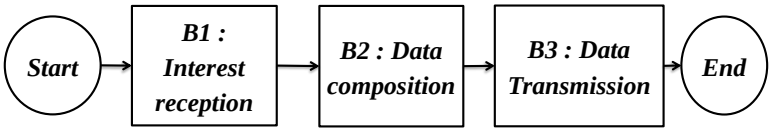


Fig. 5.7 Producer flow

$$C^{Avg}_{Producer} = C_{b,1} + C_{b,2} + C_{b,3} \tag{5.7}$$

5.5 Experimentation and Performance evaluation

To evaluate the performance of our proposal, we first describe the machine-learning model applied for the categorization of the data as well as the insights extracted from the used dataset. Afterward, simulation settings and results are then illustrated and discussed.

5.5.1 The classifier model

In our experimentation we implement a proved Web categorization model proposed in [120] to categorize and analyze the used dataset. In this model, the authors use the word-based multiple n-gram model [144] and the Multinomial Naïve Bayes classifier [145] under the Random Search pipeline [146] for hyper-parameter optimization that finds automatically best parameters. The selected parameters are n-gram range, word weight, and the smoothing parameter of the naïve Bayes classifier. The model was trained and tested based on DMOZ dataset [141]. The machine learning model used will categorize the requested interests online when it is deployed. The offline learning or training phase will only be performed once since the model can effectively predict the category of new content names. From our point of view, this model will have an acceptable effect on the network delay in terms of the results that can be obtained, also the number of categories used is small and limited, and the incoming names generally have a small vocabulary.

5.5.2 The used dataset for experimentation

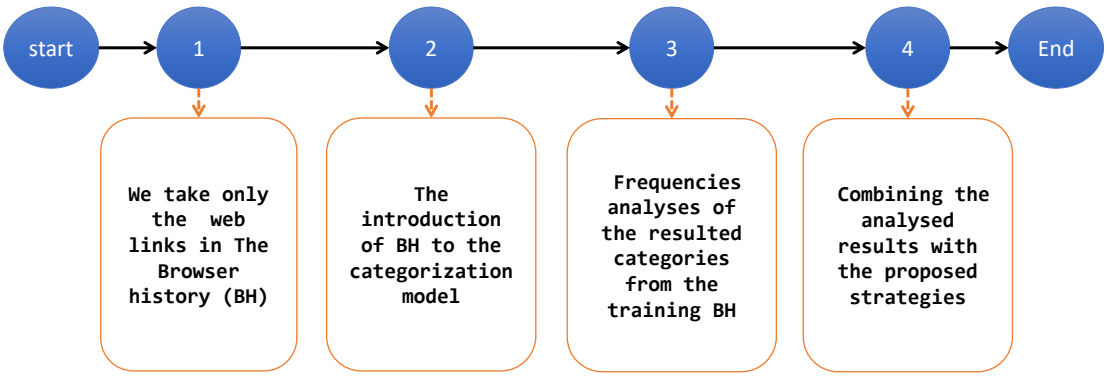


Fig. 5.8 Framework of the data categorization and knowledge extraction

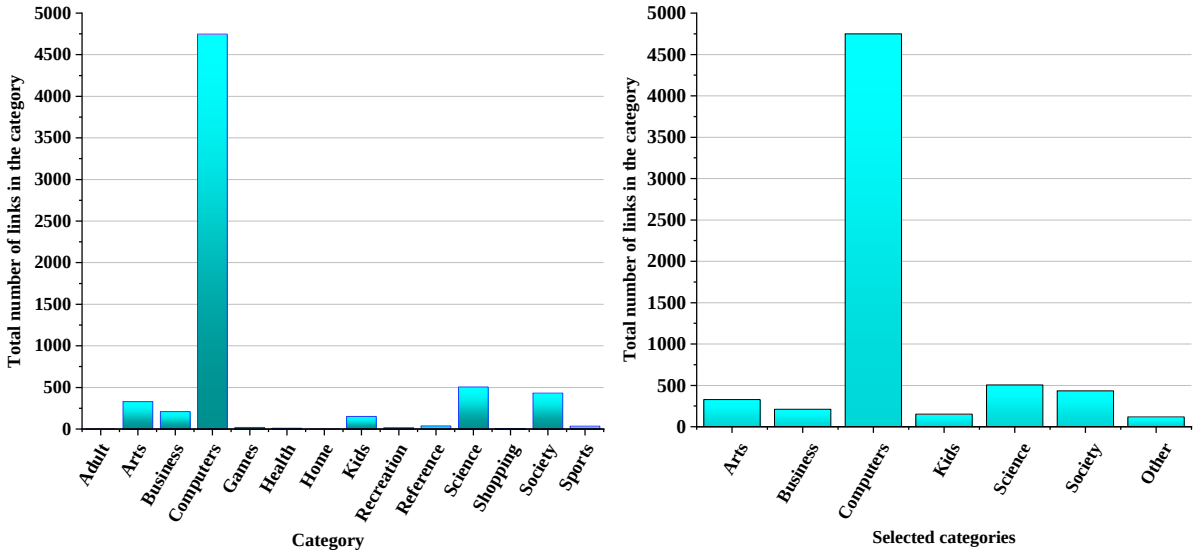


Fig. 5.9 The number of individual categories that exist in the training Browser History

Since NDN has not been deployed yet and there is no available real corpus for this project, we use a real dataset history extracted from a Mozilla Firefox browser where the links are adapted to the NDN request name format [147]. The dataset contains two files, one is used for training, and the other is for testing. The training file is analyzed and used as requested categories in the past, and the test file is used during the simulation, as shown in Figure 5.8.

Once the categorization and analysis of the training file are completed, we obtain fourteen categories (Fig. 5.9). In our case, we have selected seven categories among all the obtained categories, and the rest is gathered in a single category named **Other**. The selected categories are the following: Arts, Society, Business, Computers, Science, Kids and Other.

Table 5.2 The probability of occurrence of each category

Category	**Probability of appearing in the future %**
Computer	73.17%
Science	7.77%
Society	6.66%
Art	5.05%
Business	3.22%
Kids	2.33%
Other	1.8%

The frequencies of the selected categories are used as knowledge for the proposed placement and replacement strategy. From the results of Table 5.2 we can see that in the training dataset, consumers visit about 70% of the Web sites in the category "Computer" compared to the other consumed categories. The results showed in Table 5.2 that category-based analyses can provide a comprehensive view of data consumption and are much easier to process due to their limited number.

5.5.3 Performance evaluation

Simulation environment

The performance evaluation is done by implementing our proposed strategy using the ndnSIM simulator [148] which is an extension of NS-3. We compare the performance of our strategy with four different caching strategies.

We compare the CaDaCa placement strategy using the LRU replacement policy in the different benchmarks:

- **LCE**, leave a copy on each node in the returning path [129],
- **Prob(p)**, take a copy of data at each node with a fixed probability [57].

We compare the CaDaCa replacement strategy with:

- **LRU**, the least recently used content was replaced [93],
- **First In First Out (FIFO)**, replaces the first inserted element in the CS [93].

For the topology, we use a K-ary tree topology that is small enough to handle our analysis. It is defined by two parameters k and D (k represents the number of children and D represents the depth of the tree) [149–151]. In our work, we assume that the root node represents the server and that all requests are sent from the leaf nodes based on the test of dataset which is equally distributed among them. we also use the same parameters as in [149] keeping the same value of $k = 2$ with $D = 3$ instead of 4. The choice of the value of D is due to the limit of the requests of the dataset chosen to have results close to reality. Furthermore, the choice of D is made to find a balance between the size of the tree and the consumption time of the users' requests, since when the depth increases, the number of requests assigned to the leaf nodes decreases.

Table 5.3 Parameter setting for simulation

Parameter	**Default value**
Value of p in Prob(p)	0.5
Bandwidth	10 Mbps
Request frequency	100 Request/s
Link delay	10 ms
Range of node cache capacity	[10,100]

In the simulation, we selected the parameters shown in Table 5.3, where we set p from the strategy Prob(p) to $p = 0.5$. We chose the mean value for p because this strategy work like LCE when $p = 1$ and will not take any element in the CS when $p = 0$. The simulation time is the time required to consume all the requests in the dataset used for testing. For the evaluation, we use homogeneous cache size for the sake of simplicity. In addition, as in [150, 151], we set the cache size in the range of 10 to 100 content units. The rest of the parameter values shown in Table 5.3 are similar to the default values in ndnSIM.

We quantify the performance of the in-network caching strategies from the following four aspects:

– **Cache hit ratio**, measures the number of content requests served versus the number of requests received. The higher the cache hit ratio is, the faster to get data before the user request arrives at the data source and can be defined as:

$$CH_{ratio} = \frac{r}{|R|}, \tag{5.8}$$

where r represent the number of requests satisfied by in-network caches and $|R|$ is the total number of requests.

– **Hop reduction ratio**, which shows the change in the quality of data delivery perceived by the user and is defined as:

$$Hop_{ratio} = 1 - \frac{\sum_{r \in R} h_r}{\sum_{r \in R} H_r}, \tag{5.9}$$

where h_r and H_r are the hop counts from the requester of $r \in R$ to the node which serves the request and to the original content server respectively.

– **The server load ratio**, reflects the pressure on the server. The high server load may lead to insufficient processing ability which causes more network delay and packet loss. Therefore, it is also necessary to reduce the server load as much as possible. In this work, it is calculated the as:

$$Sever_{ratio} = \frac{C}{NC}, \tag{5.10}$$

where C represents the total received requests using the cache strategy and NC is the total received requests without using a caching mechanism.

– **Cache replacement ratio**, indicates the time and power consumption in I/O processes of cache storage, and was calculated as:

$$rep_{ratio} = \frac{\sum_{i \in N} repDat(i)}{\sum_{i \in N} recDat(i)}, \tag{5.11}$$

where $repDat(i)$ represents the number of content replacements on node i, $recDat(i)$ represents the number of the received content on the node i.

Results

Figure 5.10, shows the impact of cache size on network performance using the tested placement strategies (CaDaCa, LCE, Pob(p)). The CaDaCa strategy outperforms the tested strategies when the cache size is greater than 50 in terms of the cache hit ratio, hop reduction ratio, and server load ratio. LCE performs the worst of the tested strategies. Figure 5.11 shows the effect of the tested strategies on the replacement ratio in the network with a cache size of 100. Figure 5.11 shows also that Prob (p) achieved the lowest replacement ratio, followed by CaDaCa.

The results represented in Figure 5.12 show the impact of cache size on the cache hit ratio, hop reduction ratio and server load ratio when using the tested replacement strategies. We

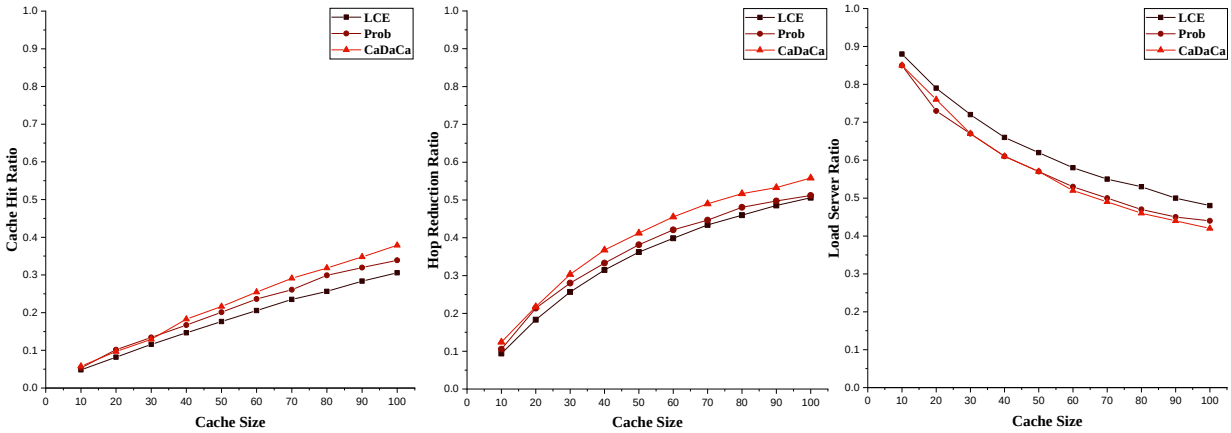


Fig. 5.10 Caching performances of CaDaCa placement with the tested strategies

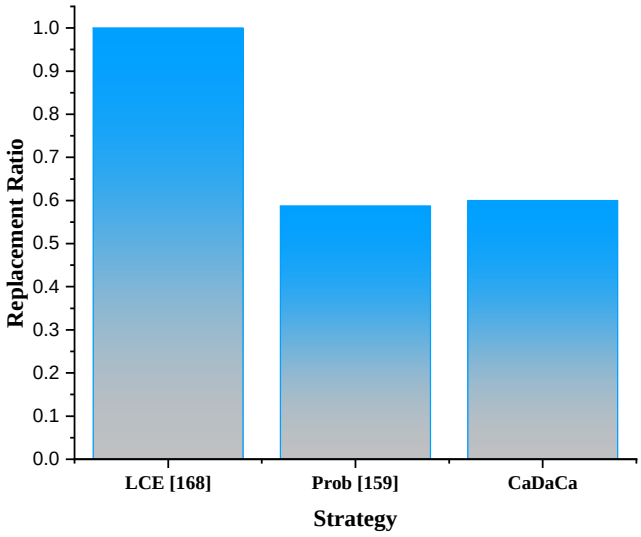


Fig. 5.11 Replacement ratio of the tested strategies

observe that CaDaCa with content replacement performs better as the cache size increases. In addition, CaDaCa combined with LRU (CaDaCa + LRU) performs better than CaDaCa with FIFO (CaDaCa + FIFO) when the cache size is larger than 40.

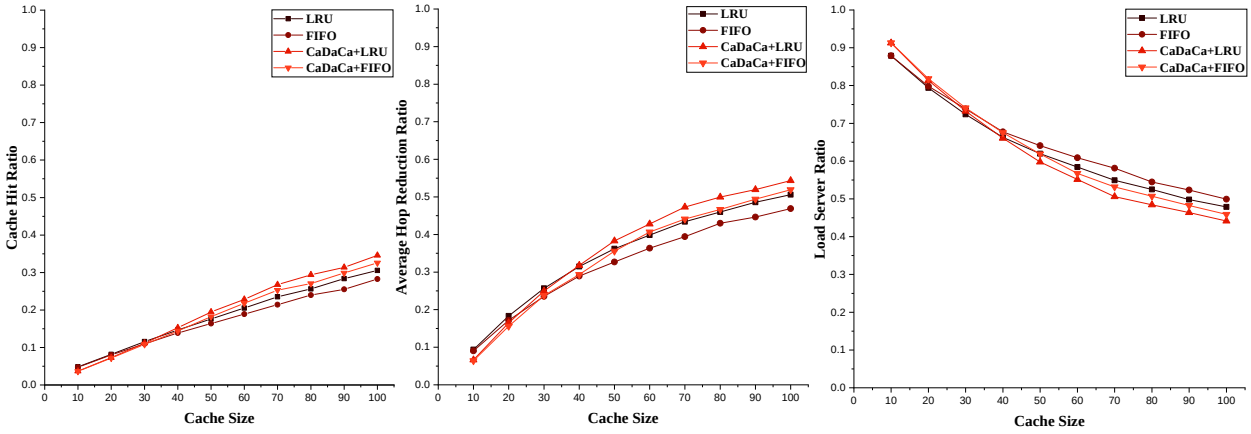


Fig. 5.12 Caching performances of CaDaCa replacement with the tested strategies

Discussion

The results show that the use of CaDaCa can improve the performance of the network and can place and replace content better than the tested strategies.

CaDaCa placement: Figure 5.10, shows that for $CS = 100$, CaDaCa with content placement achieved a 4% improvement in the caching hit ratio and a 5% hop reduction ratio compared to Prob(p) which is in second place according to the results. CaDaCa also reduced server load by more than 60% compared to the non-cache mechanism. With small cache sizes, our strategy works like Prob(p) since the replacement mechanism with this sizes can disrupt the desired percentage of data placement for each category. Figure 5.11 shows that CaDaCa reduces the cache eviction ratio by 40% compared to LCE and by 2% compared to the Prob(p) strategy. The reason for which CaDaCa achieves about a 60% of eviction ratio because content categories received during the simulation are approximately the same as the placement probabilities computed in Table 5.2. LCE gives a very high eviction ratio since it caches all the received content in the return path. Similarly, when the Prob(p) policy is set to $p = 0.5$, it caches about 50% of the received content. The eviction ratio in CaDaCa can be changed from period to period depending on the probabilities computed for each category unlike the tested strategy which has a fixed replacement ratio. For example, if the probabilities are uniform, our strategy will have a replacement ratio lower than Prob(p).

CaDaCa replacement: Figure 5.12 shows that CaDaCa with the replacement strategy gave us an advantage of about 4% in the cache hit ratio, 3% in hop reduction ratio, and 5% in server

load ratio, respectively compared to the LRU strategy. We can also see that CaDaCa+FIFO performs better than LRU and FIFO with high cache sizes. However, CaDaCa+FIFO gives less performance than CaDaCa+LRU, which is due to the fact that using LRU is more efficient than FIFO like the results show. With small cache sizes, CaDaCa+LRU and CaDaCa+FIFO do not give the expected results because the strategy with the small cache sizes cannot guarantee the desired partitions at the computed categories.

5.6 Conclusion

In this work, we have proposed CaDaCa, a new caching mechanism in NDN using data categorization. The proposed approach enables capturing the traffic trends, particularly requested content's themes and topics reflecting the users' habits. The target is to cache with a higher probability the content relative to its topic. This allows for better predicting future users' requests and enhances the placement/replacement decisions. Furthermore, the proposed approach can assist network operators in providing them with an in-depth overview of traffic usage. The proposed approach is evaluated on a real dataset describing users browsing history. Experimental results show that our strategy achieves better performance for both placement and replacement decisions compared to other approaches.

Chapter 6

Resource allocation in NDNs based on Naïve Bayes classifier and Lagrange method

6.1 Introduction

In recent years, new network architectures under the name of Information-Centric Networking (ICN) have attracted the attention of many academics and enterprises. ICN proposes new structures and concepts that can solve the traffic congestion problems caused by massive content distribution in the network [30]. The content is considered a very important factor in ICN , being the basis for content demand and reception. Among the ICN architectures, NDN is a promising active research project that aims to develop a candidate architecture for the future Internet [43]. NDN follows the same design as the IP system, the difference being that the names of the data replace the IP address. NDN data names are similar to the URL names structure, which better matches the current shift in intranet usage from site-specific to content-specific searches. Routing and data transmission also depends on the content names by implementing the longest prefix corresponding to the requested name prefix [23]. In-network caching is also an important key in NDN, where each router has a cache, and the received data is cached according to a specific caching strategy. The caching mechanism enables future requests to be satisfied by the nearest available routers instead of reaching the end producer of the data [130].

It is obvious that the in-network caching saves limited bandwidth resources and improves network performance. However, in practice, the size of the caches remains limited due to the huge amount of data circulating in the network. Therefore, the main challenge is how to use these limited caching resources for massive content while ensuring Quality of Service (QoS). In addition, the administration and allocation of caching resources in the network is also an important issue. These problems have been the subject of several researches in terms of caching strategies and algorithms. However, many of the proposed strategies focus on the large network environment and simply consider user satisfaction as a basic factor for caching important data. In practice, these methods are not always effective when dealing with a small limited network, such as a university network, a company network, etc. This type of network usually has a limited set of nodes, and its own data representing its direction, which is often stored in its own servers. In fact, the huge demand for content from social networks and video-on-demand has become a large part of the data consumed in everyday life [8, 56]. This poses a problem for this type of network since the owners of these networks will not be able to take advantage of the cached data using existing caching strategies. As an example, the analysis of data traffic at the University Amar Telidji Laghouat, shows that the consumed data belonging to the university represents only 4% of the total resource consumption. In this case, with the existing strategies in the NDN, the data related to this network has no advantage in terms of delay, satisfaction, and load on the servers. This looks to be contrary to the primary goals of NDN. Therefore, controlling the trade-off between maximum user satisfaction and resource efficiency is one of the most fundamental issues in a limited NDN environment. From the network administrator's perspective, it is very essential to use the network caches efficiently, and the benefits must be maximized. Similarly, from the users' perspective, it is more important that their Quality of Service (QoS) requirements are met adequately. Hence, the question is how to manage this trade-off?

In this chapter, we use a proven Bayesian classification-based machine learning method to classify the received data in such a network. Then, using the Lagrange method, we propose a utility function based on a preference parameter that allocates different partitions to the content classes in the routers. A new placement and replacement strategy are also defined for processing the data received by the routers. Finally, an exhaustive simulation of the results is presented to demonstrate the performance of the proposed model.

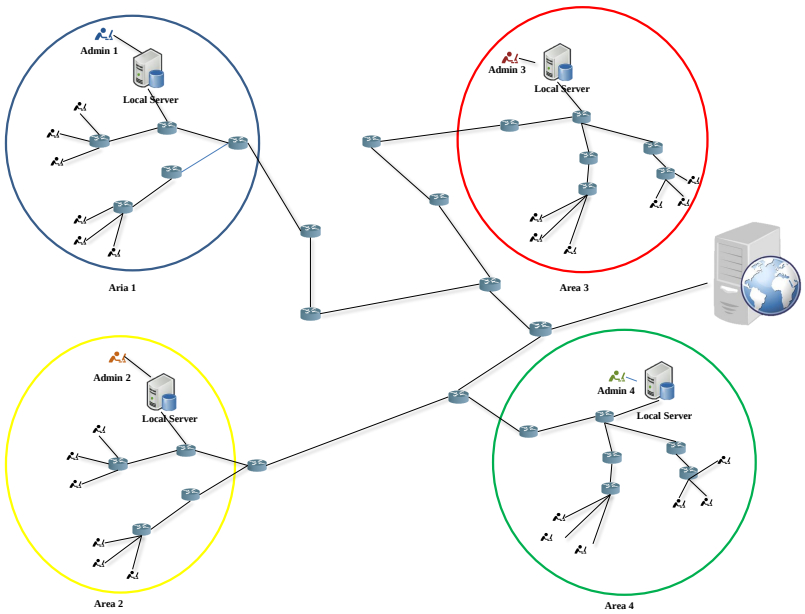


Fig. 6.1 Example of targeted networks.

This chapter is organized as follows. In Section 6.2, the proposed solution for resource allocation is described. Simulation results are then presented and discussed in Section 6.3. Finally, we conclude this study in Section 6.4.

6.2 RADC: Resource Allocation based Data Classification

6.2.1 System Model

The system model is much more focused on limited network environments, such as academic and industrial networks, etc. In these types of networks, each area consists of a network administrator, and a set of nodes and server(s). For instance, Figure 6.1 illustrates a network with 4 areas denoted $Area = \{Area_i | i \in [1.4]\}$.

Routers in the same network area sent periodically the sum of each consumed class to the administrator side. Similarly, the network administrator will calculate the allocated partitions for each class according to the utility function given in Subsection 6.2.3. The communication between administrator/routers of the same area can be done using the IP layer in the NDN hourglass [23]. To classify the received contents each router in the network will use the proposed classification model (Subsection 6.2.2) to assign the contents to the corresponding classes. The nodes outside those networks use the default caching mechanism used in NDN.

6.2.2 Content Classification

As mentioned above most of the proposed methods in the NDN do not provide a robust model for content classification, as they typically use simple parameters to classify their content (e.g., request frequency). One of the most important keys introduced by NDN that can be exploited to classify contents efficiently is the data naming, which adopts the same structure as URL naming. The names in NDN are significant sources of information and knowledge. However, processing names is a difficult task regarding their unstructured format. We use a Naïve Bayes (NB) classifier [152], which has been widely used to analyze and solve many scientific problems due to its simplicity, efficiency, and effectiveness [153, 154]. Naive Bayes is computationally inexpensive and also needs a very low amount of memory [155]. The role of the NB in this study is to classify the received content into two classes, where the first-class represents the consumed data that belongs to the server or domain of the addressed network, and the second class is the consumed data that does not bring any benefit to the addressed network. Before starting to use the BN model, a step concerning the preparation of the dataset for the training and testing of the model should be performed. The dataset used should contain a sufficient number of names of the requested contents in each selected class. Once the preparation of the dataset is done, the conversion of the dataset into a word count matrix is performed, with each row representing a data name, and the columns representing the features that represent the entire vocabulary existing in the dataset. The model uses the n-gram model [111] to extract more features, and also uses the Term-Frequency Inverse Document Frequency (TF-IDF) model [112] to weight the selected features.

The High-Level Description of the Naïve Bayes Classifier

If the number of content names (L) fit into s classes where $s \in \{c1, c2, ..., cs\}$, the predicted output class is $c \in C$. The Naïve Bayes algorithm can be described as follows:

$$P(c \mid l) = \frac{P(l \mid c)P(c)}{P(l)} \tag{6.1}$$

where l is the content name, and c indicates classes.

$$C_{\text{Predicted}} = \arg\max_{c \in C} P(l \mid c)P(c) \tag{6.2}$$

Multinomial Naïve Bayes Classifier

Naive Bayes classifiers differ mainly in the assumptions they make about the probability distribution of $P(l \mid c)$. in text classification like content names of NDN the multinomial distribution performs well compared to other distributions such as the Gaussian distribution or the Bernoulli distribution [123]. The Multinomial Naïve Bayes algorithm can be written as:

$$P(c \mid l) = \frac{P(c)\prod_{w \in l} P(l \mid c)^{n_{\text{wl}}}}{P(l)} \tag{6.3}$$

where n_{wl} denotes the number of times that word w occurs in the name, and $P(w \mid c)$ is the probability of observing word w given class c.

$P(w \mid c)$ is calculated as:

$$P(c \mid w_i) = \frac{\text{count}(w_i, c) + 1}{\text{count}(w, c) + V} \tag{6.4}$$

where $count(w_i, c)$ is the number of times the word appears in class c in the training dataset, $count(w, c)$ is the total number of all words appearing for class c, and V is a constant value that represents the number of features (words) that represent the model.

6.2.3 Resource Allocation

One of these solutions, which is very simple, efficient and suitable for solving problems of allocation of limited resources between a set of independent activities, is that of Lagrange multipliers. The use of this technique does not guarantee that a solution will necessarily be found for all problems, but it is safe in the sense that any solution found by using them is a real solution [165]. To this end, the aim of this work is to optimize the resource allocation of the two selected classes of data by giving each of them an associated space taking into account two factors. The first one is the priority level of the space associated with data that is not important to the targeted network compared to the second space. The second factor is the frequency of requests received in each class. This is done within the constraint of the maximum cache space in each router as formulated by the equation below:

$$P \mid \begin{array}{ll} \max\limits_{C_1, C_2} & f(C_1, C_2) \\ \text{u.c} & g(C_1, C_2) = \text{MaxC} \end{array} \tag{6.5}$$

where C_1 and C_2 represent, respectively, the space associated with the data that belong to the domain of the targeted network, and the second to the other data; *Max*C represents the maximum cache space in the router.

The chosen objective function $f(C_1,C_2)$ and the constrained $g(C_1,C_2)$ that can give the best allocation to the two spaces are formulated according to:

$$P \mid \begin{array}{l} \max\limits_{C_1,C_2} -\alpha C_1 - \beta C_2^k \\ \text{u.c } C_1 + C_2 = \text{MaxC} \end{array} \tag{6.6}$$

where α and β represent, respectively, the frequency of requests that belong to the organization, and the requests that do not belong to the organization, These requests are classified using the Naive Bayes classifier. k represents the priority level of space C_2 with respect to space C_1, where the value of factor k takes the scale by 0.1 to 0.9 inspired from the work done in [80]. The meaning of the scale is presented in Table 6.1 (Note: The values 0.2, 0.4, 0.8 of the factor k mean that the importance of the space C_1 in respect with C_2 is situated between the two adjacent levels shown in Table 6.1).

Table 6.1 Meaning of the scale from 0.1 to 0.9

Scale	**Meaning**
0.1	The data in the C_1 space have an obvious importance in the network compared to the C_2 space.
0.3	The data in the C_1 space have a batter importance in the network compared to the C_2 space.
0.6	The data in the C_1 space have a little importance in the network compared to the C_2 space.
0.9	The data in the C_1 space have the same importance in the network compared to the C_2 space.

The Lagrangian function associated with this program is written as shown below:

$$L(C_1,C_2,\lambda) = -\alpha C_1 - \beta C_2^k - \lambda\,(C_1 + C_2 - \text{Max}\,C) \tag{6.7}$$

Finally we get the two spaces C_1 and C_2 calculating the derivatives of our variable, where C_1 is equal to $\left\lceil \text{Max}\,C - \left(\frac{\alpha}{k\beta}\right)^{\frac{1}{k-1}} \right\rceil$, and C_2 is equal to $\left\lfloor \left(\frac{\alpha}{k\beta}\right)^{\frac{1}{k-1}} \right\rfloor$. We can add the Laplace smoothing to the spaces to avoid zero in the denominator, so C_1 and C_2 will equal,

respectively:

$$\begin{cases} C_1 = \left\lceil \mathrm{Max}\, C - \left(\frac{\alpha+1}{k\beta+1}\right)^{\frac{1}{k-1}} \right\rceil \\ C_2 = \left\lfloor \left(\frac{\alpha+1}{k\beta+1}\right)^{\frac{1}{k-1}} \right\rfloor \end{cases} \tag{6.8}$$

In certain cases when the margin between the frequencies of requests belonging to the two classes is very large, we can find that C_1 is equal to a negative number to satisfy our constraint. Since the Lagrange is limited by the constraints of equality, we can fix this problem by affecting $C_1 = 0$, and $C_2 = MaxC$ when $C_1 \leqslant 0$.

6.2.4 Placement and Replacement of Data

Unlike strategies that share the same space for all content, our strategy allocates the desired space for each content class. It is known that content placement and replacement in NDN is also an important feature that can improve the overall network performance. As shown in Figure 6.2, when the routers in the network receive the content, they classify it using the classification model. Then, the received content is placed in the corresponding space that was allocated to the same class. If the corresponding space is full, a replacement strategy will be applied to insert the received content by evicting another from the same class. We note

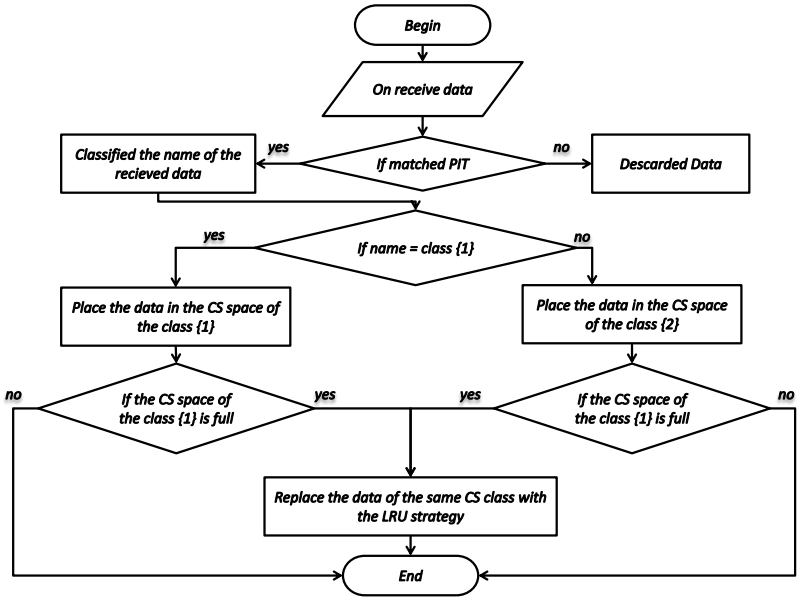


Fig. 6.2 Data placement and replacement process.

that the design of our strategy allows us to adapt any replacement strategy to replace data of

the same class, but for simplicity we will use the Least Recently Used method to replace the received data. In this strategy the received data will replace the least recently used data [160].

6.3 Performance Results of RADC Strategy

In this section, we will present and discuss the different performance results obtained. To this end, we start by detailing the simulation environment used, and next we proceed to show the actual results achieved. Besides, for the sake of reproducibility, the detailed working code of RACD is available at: https://github.com/Herouala/RADC.git (accessed on 12 January 2022).

6.3.1 Simulation Environment

The performance evaluation of the proposed strategy is done in two steps. First, we implement the classification model to classify the dataset used in the simulation. Then, the proposed strategy is evaluated through simulation. Due to the lack of standard dataset in NDN, we constructed our dataset by combining the DMOZ dataset with links extracted from the website of the University Amar Telidji of Laghouat (UATL). The names of the DMOZ dataset are labeled as data belonging to the class $\{0\}$ (unimportant), and the links extracted from the UATL are labeled as data belonging to class $\{1\}$ (important). The dataset used is divided into two parts, one for the training and the other for testing the model. The implemented Multinomial Naïve Bayes classifier model has achieved 98% accuracy in classifying the content names. After that, the RADC strategy is evaluated through ndnSIM [95], which represents the official ns-3-based simulator deployed by the NDN community. ndnSIM enables reliable simulation experiments that can be replicated in real environments without having to modify the source code [164]. To make our simulation as realistic as possible, we chose a real dataset from the University Amar Telidji of Laghouat as a realistic interest consumption. The UATL dataset contains more than one million links consumed during a week at this university.

The strategies are simulated using a non-complete K-ary tree as in [161–163]. This topology is based on two parameters, D and k, where they represent consecutively the depth of the tree and the number of children of each node in the tree. We chose $D \in [3,7]$ and k in the interval $[0,5]$, where k is randomly generated for each node as in [163]. For the simulation scenario, we assume that there are two root nodes, where the first is a server

that belongs to the university, and it contains all its data. The second represents a server containing all other data requests. Furthermore, we assume that all requests are sent by leaf nodes.

For each test run, users are assumed to express interests from the used UATL dataset. For simplicity, we set a similar cache size for all nodes in the network. Each link in our topology has a bandwidth of 10 Gbps, which is greater than the traffic demand, and a propagation delay of 1 ms. Table 6.2 shows the choice of the main parameters for our simulation.

Table 6.2 Parameter settings for the simulation.

Parameter	**Default Value**	**Range**
Tree-ary k	-	[0;5]
Tree depth D	5	[3;7]
node cache capacity	100	[10;100]
Priority level	0.4	[0.1;0.9]
Delay	1 ms	-
Bandwidth	10 Gbps	-

For the evaluation method, we studied the impact of parameters priority level, cache size, and tree depth on the caching performance in the network. We compare the results of our resource allocation mechanism with the mechanism that promotes content sharing in the same cache. The sharing mechanism represents the majority of the proposed strategies, with a difference in the choice of the placement and replacement strategy. Since RADC does not apply any particular technique to manage the cache content, any strategy could easily be adopted. We compare the two strategies in the same environment, applying the same placement and replacement technique. The two strategies represent, consecutively, LCE [23] and LRU [160] for placement and replacement.

We quantify the performance of the in network caching from two metric aspects:

Cache hit ratio, a metric that measures the ratio between the served content requests and the number of received requests. It also reflects the load saving of the servers, and the delay of the data delivery, since the higher the cache hit ratio is, the faster the data delivery becomes before the user's request reaches the data source. In our case, we study the cache hit of the served data that belong to the university, and also the hit of the other data for both

strategies; we can define it as follows:

$$Cachehitratio = \frac{r}{|R|} \tag{6.9}$$

where r represent the number of requests satisfied by in-network caches, and $|R|$ is the total number of requests.

The hop reduction ratio indicates the variation of the data delivery distance in the network. In our study, we investigate the hop reduction ratio of the data belonging to the university against the other data in both tested strategies. The hop reduction ratio is defined as follows:

$$Hop\ reduction\ ratio = 1 - \frac{\sum_{r \in R} h_r}{\sum_{r \in R} H_r} \tag{6.10}$$

where h_r and H_r are the hop counts from the requester of $r \in R$ to the node which serves the request, and to the original content server, respectively.

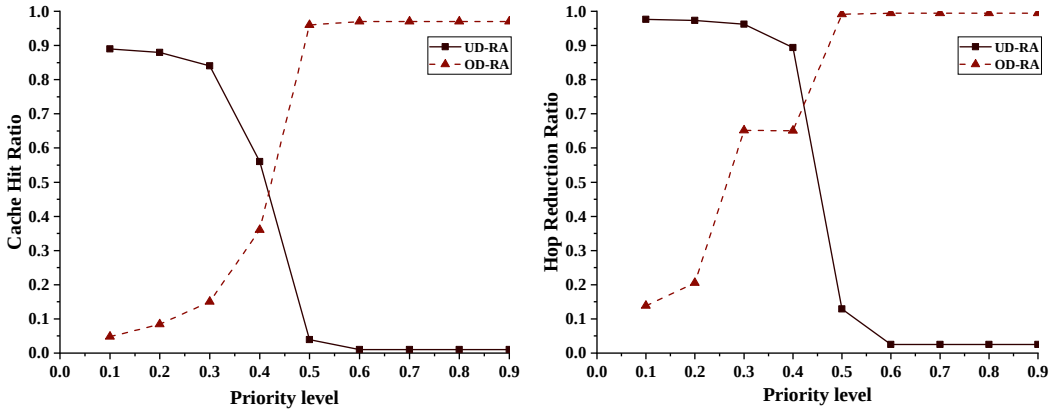
6.3.2 Performance Results

In the comparison of results, we named the results obtained for the two classes with our strategy as UD-RA and OD-RA, where the UD-RA acronym refers to the results obtained for the data belonging to the university using our strategy, and the OD-RA acronym refers to the obtained results for the data outside the university using our strategy. Also, for the sharing strategy, we name the results obtained for the two classes as UD-sharing and OD-sharing.

The Impact of the Priority Level on network performance

Figure 6.3 shows the effect of the priority level (k) on the performance of the resource allocation strategy in terms of hit ratio and hop reduction ratio. We can see that, when the priority level k increases, the results of the university data (UD-RA) decrease, and the results of the other data (OD-RA) increase. Here we observe that, from 0.1 to 0.4, there is an increase in the hit ratio and the hop reduction ratio for university data compared to non-university affiliated data. The reason is that, as the priority coefficient increases, the space given to non-university data increase until it reaches nearly the same size for both types of data when $k = 0.4$. Starting from $k = 0.5$, We can see a clear superiority for OD-RA reaches almost 90% as a difference in the hit ratio, and also for the hop reduction ratio. Notice that, as the k coefficient increases, the space allocated to OD-RA also increases. The

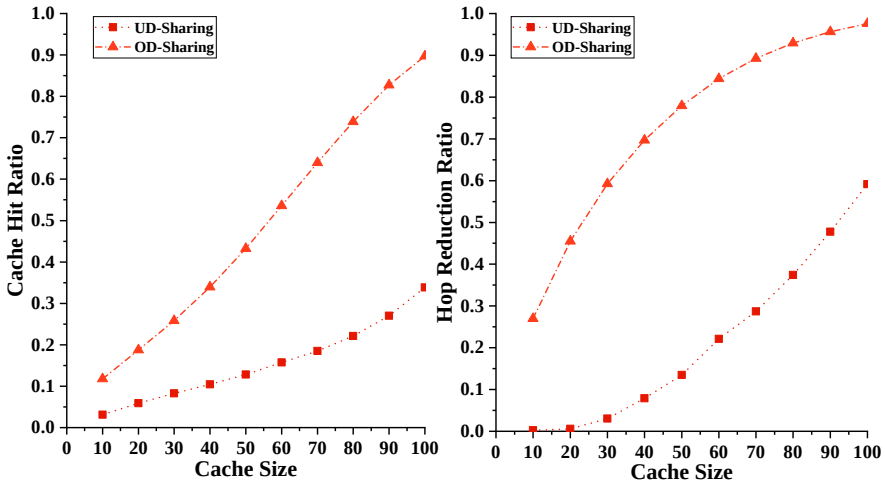
reason why OD-RA takes more storage space compared to UD-RA even with the priority coefficient k in the ranges from 0.5 to 0.9 is because there is a large difference between the amounts of data consumed in the two classes, being that the amount of data that belong to the university represent only 4% of the total resource consumption. This, in turn, has an impact on space allocation.



(a) Priority level (k) vs Cache Hit Ratio (b) Priority level (k) vs Hop Reduction Ratio

Fig. 6.3 Caching performance vs. priority level in terms of (**a**) Cache hit ratio, and (**b**) Hop reduction ratio.

Impact of the Cache Size on the network performance



(a) Cache Size vs Cache Hit Ratio (b) Cache Size vs Hop Reduction Ratio

Fig. 6.4 Caching performance vs. Cache size for sharing strategy in terms of: (**a**) Cache hit ratio, and (**b**) Hop reduction ratio.

Figures 6.4 show the effect of cache size on the performance of the data sharing a in terms of hit ratio and hop reduction ratio. From the figure, we can see that, for the sharing strategy, the hop reduction and hit ratio values increase with the cache size, with a large margin for the OD-Sharing results compared to the UD-Sharing. The improvement in results when cache size increases make sense because now more data are cached, which increases the chances of getting the data searched to the closest possible node. The difference in the results between the two classes is due to the fact that the requests in the OD-Sharing class are more numerous than those in UD-Sharing.

Figures 6.5 and 6.6 show the effect of cache size on the performance of the resource allocation strategy in terms of hit ratio and hop reduction ratio. From the two figures, we can see that also for our strategy the hop reduction and hit ratio values increase with the cache size. In addition when we set k to $k = 0.1$, the university data (UD-RA) achieves a higher hit ratio and hop reduction ratio as the cache size increases. The OD-RA achieves low and constant values compared to UD-RA. This is because, when using a very small value for coefficient k, the cache partition size given to UD-RA is much larger than the size given to OD-RA, which was small and fixed in all sizes tested. When $k = 0.9$, we find that our strategy performs inversely to $k = 0.1$ where, as the cache size increases, OD-RA performs well compared to UD-RA in terms of hit ratio and hop reduction ratio. The reason is that, when the coefficient is $k = 0.9$, the coefficients of the two data classes become almost equivalent. Therefore, the allocations of each partition become related to the amount of

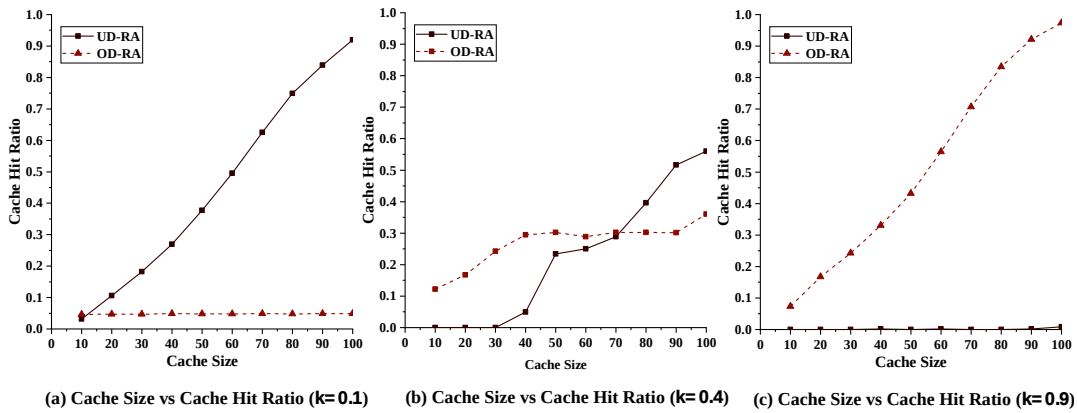


Fig. 6.5 Caching performance vs. Cache size in terms of: Cache hit ratio when varying k.

data received from both classes. Since the consumed university data represents only 4% of

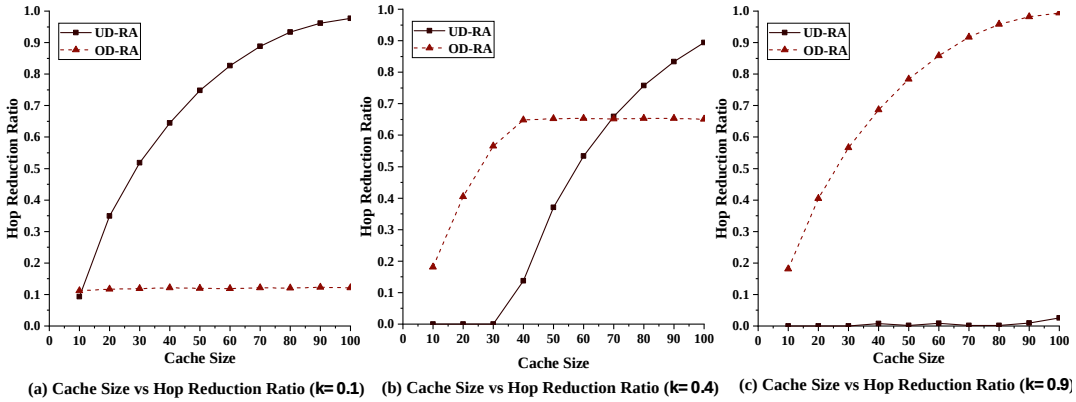


Fig. 6.6 Caching performance vs. Cache size in terms of Hop reduction ratio when varying k.

the total consumption, all the cache is allocated to OD-RA data. We also notice that, when $k = 0.4$, from 10 to 30, the UD-RA obtains a null hit and hop reduction ratio, in contrast to the OD-RA, which increases with the increase in cache size. From 40 to 60 we notice an increase in the results of both classes of data, with their convergence until they are equal in the capacity 70. From 80 to 100, the results show a continuous increase in both classes, with a dominance of UD-RA over OD-RA. The reason for the continuous increase in results for both classes is that both classes get more space when the total cache space increases. The reason why these results are obtained when $k = 0.4$ is that, when the cache size is between 10 and 30, the Lagrange utility function gives the total memory space for the OD-RA. Then, beyond size 40, the function starts to give more memory space for UD-RA, and OD-RA space remains fixed until they reach equal results for size 70. Then, the increase in space continues for UD-RA, which takes more memory space, and thus gets better results than OD-RA.

Impact of Tree Depth on the network performance

Figures 6.7 show the effect of tree depth on the performance of sharing strategy by calculating the hit ratio and hop reduction ratio. For the sharing strategy, we observe a slight increase in the results for both classes of data in terms of hit and hop reduction ratio as depth increases. We also observe a large difference in the results for the two classes, with a 50% difference in hits, and a difference ranging from 60% to 35% in hop reduction results between OD-Sharing

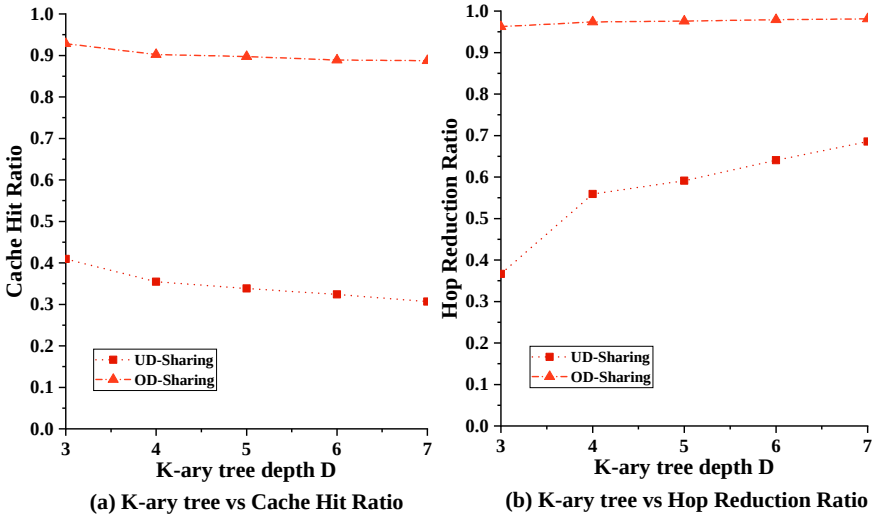


Fig. 6.7 Caching performance vs. Tree depth for sharing strategy in terms of: (**a**) Cache hit ratio, and (**b**) Hop reduction ratio.

and UD-Sharing. The increase in the results obtained is due to the number of nodes that increase with length, allowing more cache space to be allocated. This allows more data to be cached, including low-consumption data, such as university-owned data (UD-Sharing), which results in a higher hit ratio and less hop ratio.

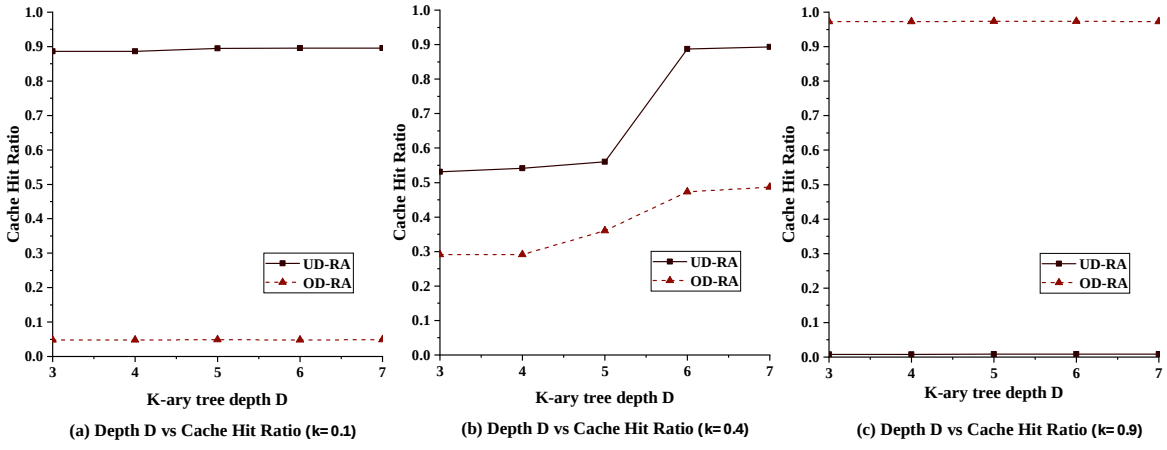


Fig. 6.8 Caching performance vs. Tree depth in terms of Cache hit ratio when varying k.

Figures 6.8 and 6.9 show the effect of tree depth on the performance of resource allocation strategy by calculating the hit ratio and hop reduction ratio. Regarding the resource allocation

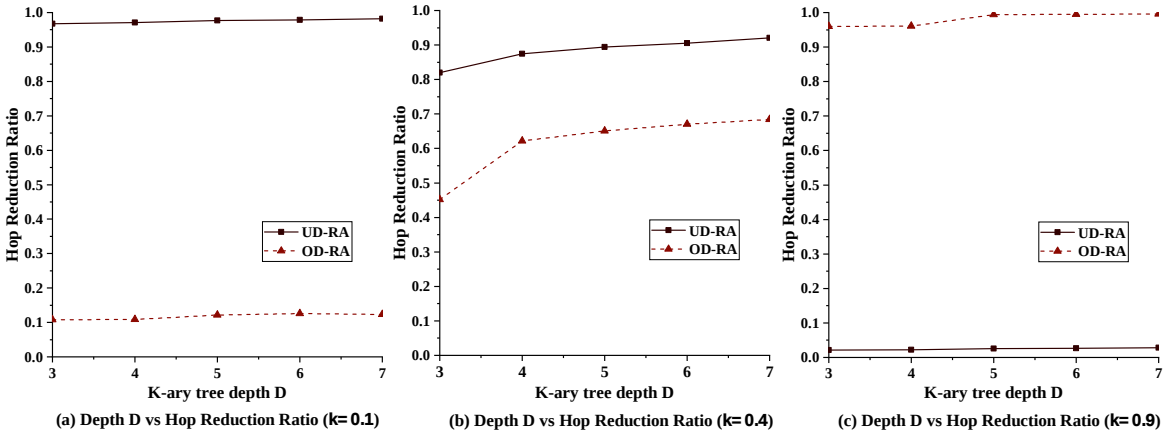


Fig. 6.9 Caching performance vs. Tree depth in terms of Hop reduction ratio when varying k.

strategy, the results differ depending on the k factor. When $k = 0.1$, we notice a very slight increase in the hit and hop reduction results with the increase of depth D. We also notice a big difference between the results obtained for the two classes of data, being that this difference grows up to 85% between the UD-RA and the OD-RA, where OD-RA achieves the lowest percentage. Indeed, the use of a coefficient $k = 0.1$ leads to a very high priority for the data that belong to the university compared to the other data. With the use of $k = 0.9$, we see the opposite of the results given with the use of $k = 0.1$. This is because the coefficients between the two classes of data become closer, and thus the priority becomes dependent on the amount of consumed data for both classes. Since the amount of consumed data that are not affiliated with the university is much larger than the others, OD-RA gained a significant advantage within the given space, leading to the obtained results. At $k = 0.4$, there is an improvement in the results of both classes as the depth D increases. The difference between the two classes of data is between 20% and 60% for the hit ratio, and between 40% and 25% for the hop reduction ratio, with a preference for the UD-RA. The convergence of the results obtained for the two classes is due to the fact that the space given to them is similar, with a bias towards the space reserved for the university data.

Through this study and the results obtained, we can say that the proposed strategy offers the possibility to control the data in an excellent way, even at the levels where there is a big difference between the consumed data classes. Indeed, by using this strategy, we are able

to determine the desired priority level, which in turn can allocate space for the desired data resources.

6.4 Conclusions

The proposed strategies for resource allocation in NDNs are known to be effective in optimizing the overall network performance. However, these strategies may suffer in a limited network environment. That is because a fundamental trade-off problem between resource efficiency and user satisfaction might arise. To this end, a resource allocation scheme based on data classification has been proposed with a weighting factor that is used to decide at which level we want to control the aforementioned trade-off. Simulation results show that our strategy outperforms the sharing strategies since it proves its ability to control any desired trade-off point of their respective efficiency and satisfaction levels.

Chapter 7

Conclusion

7.1 Summary

In-network caching is one of the most important features of named data network to efficiently distribute data and reduce the time to obtain information as well as reduce the pressure on the network. Among the proposals of the NDN is the default LCE caching strategy, which caches a copy of each requested content on the return path to the consumer, despite the importance of this strategy, its simplicity has resulted in wasted memory resources as such resources are limited compared to the huge amount of consumed data. Therefore, many cache strategies have been proposed with the aim of keeping just the important data to increase the performance of this architecture.

In this thesis, many cache management strategies in the NDN domain were reviewed and classified according to the important features of each proposed strategy. By reviewing the proposed strategies, we find that the most proposed strategies are based mainly on the calculation of the content popularity using name space of the content. Although this factor gives good results, the study of data name frequency presents several challenges. In addition, the study show that the most proposed strategies are mainly focuses on the large network scale and don't give an importance the limited network that could have problem of trade-off between resource efficiency and user satisfaction. So, based on these problems the CaDaCa strategy was proposed as a solution, which aims to analyze the categories of the data that are a rich source of knowledge. The classification was done using a machine learning technique that classifies data names based on the Multinomial Naïve Bayes classifier. Then, based on the data classification, two new strategies have been proposed, the first one depends on the

frequency of the consumed data categories to make a placement decision on the received data. As for the second strategy, it depends on the decision to replace the data also based on the calculation of the consumed categories. Overall CaDaCa strategy performed well compared to the other simulated strategies. CaDaCa give best results with higher cache sizes but with low cache sizes it doesn't give the expected results because the predicted probabilities are disturbed with low cache sizes. As a second contribution, the strategy RADC was proposed as resource allocation strategy by classifying the received data based on Multinomial Naïve Bayes classification method. The model gives good results that reach up to 98% in the prediction accuracy of the received data classes. Then, depending on the frequency of data from each class, an adaptive resource allocation strategy has been proposed based on the Lagrange utility function. The obtained results, show that RADC outperforms the sharing mechanism in such network. The RADC strategy has the ability to control any desired trade-off point of their respective efficiency and satisfaction levels compared to the sharing mechanism that almost give the total priority to the most frequently content to be cached. Overall, our proposed caching strategies showed that they perform better compared to the compared strategies. While not all of the results were significant, the overall direction of results showed trends that could help learn about the role of data categorization in implementing new caching mechanisms.

7.2 Future works

Caching strategies in NDN have great importance in developing network performance, so they should get more attention in order to find solutions to the problems and challenges they face.

So, as future work, we see that the idea of classifying data based on its names and integrating it with placement and replacement strategies is an innovative idea that opens up a great opportunity to add this concept into several caching related strategies. As an example, we would like to extend this idea and apply it to several domains such as IoT and VNDN.

In addition, it is also observed that the data names are not limited to the CS, but it is used in other structures such as PIT and FIB, which allows us to include the concept of classification in these structures as future work.

It is mentioned that the proposed strategies in this thesis were based on the Naive Bays classification method, so in the future we will try to use other techniques for data classification

like deep learning and frequent itemset. Finally, it will also be among our objectives to study the performance and impact of the RADC strategy on the overall network.

7.3 Publications

1. Herouala, A. T., Ziani, B., Kerrache, C. A., Karim Tahari, A. E., Lagraa, N., & Mastorakis, S. (2022). CaDaCa: a new caching strategy in NDN using data categorization. Multimedia Systems, 1-16.

2. Herouala, A. T., Kerrache, C. A., Ziani, B., Calafate, C. T., Lagraa, N., & Tahari, A. E. K. (2022). Controlling the Trade-Off between Resource Efficiency and User Satisfaction in NDNs Based on Naïve Bayes Data Classification and Lagrange Method. Future Internet, 14(2), 48.

3. Conference paper in Distributed Simulation and Real Time Applications (DS-RT) 2022 titled "NBCC: a new Caching strategy uses a Naive Bayes Classifier in NDN" (submitted).

References

[1] Muhammad Ali Naeem. *Compound popular content caching strategy to enhance the cache management performance in named data networking*. PhD thesis, Universiti Utara Malaysia, 2020.

[2] Raaid Nasur Kadham Alubady. *An efficient pending interest table control management in named data network*. PhD thesis, Universiti Utara Malaysia, 2017.

[3] Ons Fares, Abdulhalim Dandoush, and Nadjib Aitsaadi. Sdn-based platform enabling intelligent routing within transit autonomous system networks. In *2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC)*, pages 909–912. IEEE, 2022.

[4] Ralph Gross and Jianbo Shi. The cmu motion of body (mobo) database. 2001.

[5] Minyoung Kim, Sanjiv Kumar, Vladimir Pavlovic, and Henry Rowley. Face tracking and recognition with visual constraints in real-world videos. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[6] Yahui Meng, Muhammad Ali Naeem, Rashid Ali, and Byung-Seo Kim. Ehcp: An efficient hybrid content placement strategy in named data network caching. *IEEE Access*, 7:155601–155611, 2019.

[7] https://datareportal.com/essential-facebook-stats, accessed on 1 April 2022.

[8] U Cisco. Cisco annual internet report (2018–2023) white paper. 2020. *Acessado em*, 10(01), 2021.

[9] Theodore Zahariadis, Dimitri Papadimitriou, Hannes Tschofenig, Stephan Haller, Petros Daras, George D Stamoulis, and Manfred Hauswirth. Towards a future internet architecture. In *The Future Internet Assembly*, pages 7–18. Springer, Berlin, Heidelberg, 2011.

[10] Devi Kala, Anurudh Kumar, NB Arunekumar, and K Suresh Joseph. Design and implementation of an efficient scalable forwarding in named data networking (ndn) using huffman coding. In *IOT with Smart Systems*, pages 337–346. Springer, 2022.

[11] Charles E Perkins. Mobile ip. *IEEE communications Magazine*, 35(5):84–99, 1997.

[12] Dingde Jiang, Feng Wang, Zhihan Lv, Shahid Mumtaz, Saba Al-Rubaye, Antonios Tsourdos, and Octavia Dobre. Qoe-aware efficient content distribution scheme for satellite-terrestrial networks. *IEEE Transactions on Mobile Computing*, 2021.

[13] Thomas Karagiannis, Andre Broido, Michalis Faloutsos, and KC Claffy. Transport layer identification of p2p traffic. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 121–134, 2004.

[14] Nikos Fotiou, Konstantinos Katsaros, George C Polyzos, Mikko Särelä, Dirk Trossen, and George Xylomenos. Handling mobility in future publish-subscribe information-centric networks. *Telecommunication Systems*, 53(3):299–314, 2013.

[15] Bengt Ahlgren, Christian Dannewitz, Claudio Imbrenda, Dirk Kutscher, and Borje Ohlman. A survey of information-centric networking. *IEEE Communications Magazine*, 50(7):26–36, 2012.

[16] Mark Handley. Why the internet only just works. *BT Technology Journal*, 24(3):119–129, 2006.

[17] Noor Abani. *Caching Strategies for Private and Efficient Content Retrieval in Information-Centric Networks*. University of California, Los Angeles, 2018.

[18] Muhammad Ali Naeem, Shahrudin Awang Nor, and Suhaidi Hassan. Future internet architectures. In *International Conference of Reliable Information and Communication Technology*, pages 520–532. Springer, 2019.

[19] Teemu Koponen, Mohit Chawla, Byung-Gon Chun, Andrey Ermolinskiy, Kye Hyun Kim, Scott Shenker, and Ion Stoica. A data-oriented (and beyond) network architecture. In *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 181–192, 2007.

[20] Ashok Anand, Fahad Dogar, Dongsu Han, Boyan Li, Hyeontaek Lim, Michel Machado, Wenfei Wu, Aditya Akella, David G Andersen, John W Byers, et al. Xia: An architecture for an evolvable and trustworthy internet. In *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, pages 1–6, 2011.

[21] Arun Venkataramani, James F Kurose, Dipankar Raychaudhuri, Kiran Nagaraja, Morley Mao, and Suman Banerjee. Mobilityfirst: a mobility-centric and trustworthy internet architecture. *ACM SIGCOMM Computer Communication Review*, 44(3):74–80, 2014.

[22] Diego Perino and Matteo Varvello. A reality check for content centric networking. In *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*, pages 44–49, 2011.

[23] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, KC Claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73, 2014.

[24] Jinyang Lv, Xiaobin Tan, Yang Jin, and Jin Zhu. Drl-based forwarding strategy in named data networking. In *2018 37th Chinese Control Conference (CCC)*, pages 6493–6498. IEEE, 2018.

[25] http://www.named-data.net/, accessed on 12 March 2022.

[26] Rasha Salem Abbas. *Performance evaluation of caching placement algorithms in named data network for video on demand service*. PhD thesis, Universiti Utara Malaysia, 2016.

[27] Mohammed Gamal Alsamman. *A hybrid rate control mechanism for forwarding and congestion control in named data network*. PhD thesis, Universiti Utara Malaysia, 2020.

[28] George Xylomenos, Christopher N Ververidis, Vasilios A Siris, Nikos Fotiou, Christos Tsilopoulos, Xenofon Vasilakos, Konstantinos V Katsaros, and George C Polyzos. A survey of information-centric networking research. *IEEE communications surveys & tutorials*, 16(2):1024–1049, 2013.

[29] Muhammad Azfar Yaqub, Syed Hassan Ahmed, Safdar Hussain Bouk, and Dongkyun Kim. Information-centric networks (icn). In *Content-Centric Networks*, pages 19–33. Springer, 2016.

[30] Athanasios V Vasilakos, Zhe Li, Gwendal Simon, and Wei You. Information centric network: Research challenges and opportunities. *Journal of network and computer applications*, 52:1–10, 2015.

[31] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, and Van Jacobson. kc claffy, patrick crowley, christos papadopoulos, lan wang, and beichuan zhang. named data networking. *SIGCOMM Comput. Commun. Rev*, 44(3):66–73, 2014.

[32] Giovanna Carofiglio, Giacomo Morabito, Luca Muscariello, Ignacio Solis, and Matteo Varvello. From content delivery today to information centric networking. *Computer networks*, 57(16):3116–3127, 2013.

[33] George Pallis and Athena Vakali. Insight and perspectives for content delivery networks. *Communications of the ACM*, 49(1):101–106, 2006.

[34] Gabriel M De Brito, Pedro B Velloso, and Igor M Moraes. *Information-Centric Networks: A New Paradigm for the Internet*. John Wiley & Sons, 2013.

[35] Jan Seedorf and Eric Burger. Application-layer traffic optimization (alto) problem statement. Technical report, RFC 5693, October, 2009.

[36] Bruno Magalhães Martins and Antônio Marcos Alberti. Host identification and location decoupling: A comparison of approaches. In *Proceedings of the international workshop on telecommunications*, 2011.

[37] Rapporteurs Mme. *Balkis HAMDANE Réseaux du futur: sécurité et nommage*. PhD thesis, TELECOM SudParis, 2016.

[38] Dmitrij Lagutin et al. Securing the internet with digital signatures. 2010.

[39] Jianli Pan, Subharthi Paul, and Raj Jain. A survey of the research on future internet architectures. *IEEE Communications Magazine*, 49(7):26–36, 2011.

[40] Christian Dannewitz, Jovan Golic, Borje Ohlman, and Bengt Ahlgren. Secure naming for a network of information. In *2010 INFOCOM IEEE conference on computer communications workshops*, pages 1–6. IEEE, 2010.

[41] Ali Ghodsi, Teemu Koponen, Jarno Rajahalme, Pasi Sarolahti, and Scott Shenker. Naming in content-oriented architectures. In *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*, pages 1–6, 2011.

[42] Lixia Zhang, Deborah Estrin, Jeffrey Burke, Van Jacobson, James D Thornton, Diana K Smetters, Beichuan Zhang, Gene Tsudik, Dan Massey, Christos Papadopoulos, et al. Named data networking (ndn) project. *Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC*, 157:158, 2010.

[43] Ibrahim Abdullahi, Suki Arif, and Suhaidi Hassan. Survey on caching approaches in information centric networking. *Journal of Network and Computer Applications*, 56:48–59, 2015.

[44] Md Faizul Bari, Shihabur Rahman Chowdhury, Reaz Ahmed, Raouf Boutaba, and Bertrand Mathieu. A survey of naming and routing in information-centric networks. *IEEE Communications Magazine*, 50(12):44–53, 2012.

[45] Akash Baid and Dipankar Raychaudhuri. Wireless access considerations for the mobilityfirst future internet architecture. In *2012 35th IEEE Sarnoff Symposium*, pages 1–5. IEEE, 2012.

[46] Diana Smetters and Van Jacobson. Securing network content. Technical report, Citeseer, 2009.

[47] Petteri Pöyhönen and O Stranberg. The network of information: Architecture and applications. *SAIL project deliverable*, 2011.

[48] Kostas Pentikousis, Borje Ohlman, Elwyn B Davies, Gennaro Boggia, and Spiros Spirou. Information-centric networking: Evaluation and security considerations. *Internet Engineering Task Force, Internet-Draft draft-irtf-icnrg-evaluation-methodology-05, Apr. 2016, work in Progress*, 2016.

[49] Mark Gritter and David R Cheriton. An architecture for content routing support in the internet. In *3rd USENIX Symposium on Internet Technologies and Systems (USITS 01)*, 2001.

[50] Christian Dannewitz, Dirk Kutscher, Börje Ohlman, Stephen Farrell, Bengt Ahlgren, and Holger Karl. Network of information (netinf)–an information-centric networking architecture. *Computer Communications*, 36(7):721–735, 2013.

[51] Van Jacobson. Congestion avoidance and control. *ACM SIGCOMM computer communication review*, 18(4):314–329, 1988.

[52] Wentao Shang, Adeola Bannis, Teng Liang, Zhehao Wang, Yingdi Yu, Alexander Afanasyev, Jeff Thompson, Jeff Burke, Beichuan Zhang, and Lixia Zhang. Named data networking of things. In *2016 IEEE first international conference on internet-of-things design and implementation (IoTDI)*, pages 117–128. IEEE, 2016.

[53] Divya Saxena, Vaskar Raychoudhury, Neeraj Suri, Christian Becker, and Jiannong Cao. Named data networking: a survey. *Computer Science Review*, 19:15–55, 2016.

[54] Cheng Yi, Alexander Afanasyev, Lan Wang, Beichuan Zhang, and Lixia Zhang. Adaptive forwarding in named data networking. *ACM SIGCOMM computer communication review*, 42(3):62–67, 2012.

[55] Mohamed Hefeeda and Behrooz Noorizadeh. On the benefits of cooperative proxy caching for peer-to-peer traffic. *IEEE transactions on Parallel and Distributed Systems*, 21(7):998–1010, 2009.

[56] Chaker Abdelaziz Kerrche, Farhan Ahmad, Mohamed Elhoseny, Asma Adnane, Zeeshan Ahmad, and Boubakr Nour. Internet of vehicles over named data networking: current status and future challenges. In *Emerging Technologies for Connected Internet of Vehicles and Intelligent Transportation System Networks*, pages 83–99. Springer, 2020.

[57] Leanna Vidya Yovita and Nana Rachmana Syambas. Caching on named data network: a survey and future research. *International Journal of Electrical & Computer Engineering (2088-8708)*, 8, 2018.

[58] Mirothali Chand. A comparative survey on different caching mechanisms in named data networking (ndn) architecture. *International Journal of Emerging Technologies and Innovative Research*, 6(4):264–271, 2019.

[59] Ioannis Psaras, Wei Koong Chai, and George Pavlou. Probabilistic in-network caching for information-centric networks. In *Proceedings of the second edition of the ICN workshop on Information-centric networking*, pages 55–60, 2012.

[60] Ikram Ud Din, Suhaidi Hassan, Muhammad Khurram Khan, Mohsen Guizani, Osman Ghazali, and Adib Habbal. Caching in information-centric networking: Strategies, challenges, and future research directions. *IEEE Communications Surveys & Tutorials*, 20(2):1443–1474, 2017.

[61] Guoyin Zhang, Bin Tang, Xianghui Wang, and Yanxia Wu. An optimal cache placement strategy based on content popularity in content centric network. *JOURNAL OF INFORMATION &COMPUTATIONAL SCIENCE*, 11(8):2759–2769, 2014.

[62] César Bernardini, Thomas Silverston, and Athanasios Vasilakos. Caching strategies for information centric networking: opportunities and challenges. *arXiv preprint arXiv:1606.07630*, 2016.

[63] Nikolaos Laoutaris, Hao Che, and Ioannis Stavrakakis. The lcd interconnection of lru caches and its analysis. *Performance Evaluation*, 63(7):609–634, 2006.

[64] Kideok Cho, Munyoung Lee, Kunwoo Park, Ted Taekyoung Kwon, Yanghee Choi, and Sangheon Pack. Wave: Popularity-based and collaborative in-network caching for content-oriented networks. In *2012 Proceedings IEEE INFOCOM Workshops*, pages 316–321. IEEE, 2012.

[65] Noor Abani, Golnaz Farhadi, Akira Ito, and Mario Gerla. Popularity-based partial caching for information centric networks. In *2016 Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, pages 1–8. IEEE, 2016.

[66] Junaid Ahmed Khan, Cedric Westphal, and Yacine Ghamri-Doudane. A popularity-aware centrality metric for content placement in information centric networks. In *2018 International Conference on Computing, Networking and Communications (ICNC)*, pages 554–560. IEEE, 2018.

[67] Quan Zheng, Yuanzhi Kan, Jiebo Chen, and Song Wang. A cache replication strategy based on betweenness and edge popularity in named data networking. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, pages 1–7. IEEE, 2019.

[68] Yahui Meng, Muhammad Ali Naeem, Rashid Ali, and Byung-Seo Kim. Ehcp: An efficient hybrid content placement strategy in named data network caching. *IEEE Access*, 7:155601–155611, 2019.

[69] Muhammad Ali Naeem, Shahrudin Awang Nor, Suhaidi Hassan, and Byung-Seo Kim. Compound popular content caching strategy in named data networking. *Electronics*, 8(7):771, 2019.

[70] Yiqi Gui and Yongkang Chen. A cache placement strategy based on compound popularity in named data networking. *IEEE Access*, 8:196002–196012, 2020.

[71] Yiqi Gui and Yongkang Chen. A cache placement strategy based on entropy weighting method and topsis in named data networking. *IEEE Access*, 9:56240–56252, 2021.

[72] DaeYoub Kim and Jihoon Lee. An ndn cache management for mec. *Applied Sciences*, 10(3):896, 2020.

[73] Oussama Serhane, Khadidja Yahyaoui, Boubakr Nour, and Hassine Moungla. Cns: A cache and split scheme for 5g-enabled icn networks. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2020.

[74] Boubakr Nour, Hakima Khelifi, Hassine Moungla, Rasheed Hussain, and Nadra Guizani. A distributed cache placement scheme for large-scale information-centric networking. *IEEE Network*, 34(6):126–132, 2020.

[75] Jinfa Yao, Baoqun Yin, and Xiaonong Lu. A novel joint adaptive forwarding and resource allocation strategy for named data networking based on smdp. In *2016 12th IEEE International Conference on Control and Automation (ICCA)*, pages 956–961. IEEE, 2016.

[76] Chengcheng Li, Renchao Xie, Tao Huang, Ru Huo, Jiang Liu, and Yunjie Liu. Joint forwarding strategy and resource allocation in information-centric hwns. In *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pages 1–6. IEEE, 2017.

[77] Dongming Yuan, Yanan Xu, Jing Ran, Hefei Hu, Yuanan Liu, and Xu Li. An optimal fair resource allocation strategy for a lightweight content-centric networking architecture. In *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, pages 573–577. IEEE, 2017.

[78] Yuanzun Zhang, Xiaobin Tan, and Weiping Li. In-network cache size allocation for video streaming on named data networking. In *Proceedings of the 2017 VI International Conference on Network, Communication and Computing*, pages 18–23, 2017.

[79] Mahsa Ehsanpour, Siavash Bayat, and Ali Mohammad Afshin Hemmatyar. On efficient and social-aware object allocation in named data networks using matching theory. In *2018 IEEE Symposium on Computers and Communications (ISCC)*, pages 00298–00303. IEEE, 2018.

[80] Ru Huo, Renchao Xie, Hengyang Zhang, Tao Huang, and Yunjie Liu. What to cache: differentiated caching resource allocation and management in information-centric networking. *China Communications*, 13(12):261–276, 2016.

[81] Leanna Vidya Yovita, Nana Rachmana Syambas, and Ian Yosef Matheus Edward. Capic: Cache based on popularity and class in named data network. In *2018 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC)*, pages 24–29. IEEE, 2018.

[82] Li Zhang and Qiang Zhang. Multi-attribute probability caching algorithm in named data network. In *Journal of Physics: Conference Series*, volume 1570, page 012010. IOP Publishing, 2020.

[83] Sadaq Jebur Taher, Osman Ghazali, and Suhaidi Hassan. A review on cache replacement strategies in named data network. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, 10(2-4):53–57, 2018.

[84] Najla Alzakari, Alanoud Bin Dris, and Saad Alahmadi. Randomized least frequently used cache replacement strategy for named data networking. In *2020 3rd International Conference on Computer Applications & Information Security (ICCAIS)*, pages 1–6. IEEE, 2020.

[85] Jian hua Ran, Na Lv, Ding Zhang, Yuan yuan Ma, and Zhen yong Xie. On performance of cache policies in named data networking. In *2013 International Conference on Advanced Computer Science and Electronics Information (ICACSEI 2013)*. Atlantis Press, 2013.

[86] Huichen Dai, Yi Wang, Hao Wu, Jianyuan Lu, and Bin Liu. Towards line-speed and accurate on-line popularity monitoring on ndn routers. In *2014 IEEE 22nd International Symposium of Quality of Service (IWQoS)*, pages 178–187. IEEE, 2014.

[87] Andrey Silva, Ivanes Araujo, Neiva Linder, and Aldebaro Klautau. Name popularity algorithm: A cache replacement strategy for ndn networks. *Journal of Communication and Information Systems*, 34(1):206–214, 2019.

[88] Salman Rashid, Shukor Abd Razak, and Fuad A Ghaleb. Imu: A content replacement policy for ccn, based on immature content selection. *Applied Sciences*, 12(1):344, 2022.

[89] Anwar Kalghoum, Sonia Mettali Gammar, and Leila Azouz Saidane. Towards a novel cache replacement strategy for named data networking based on software defined networking. *Computers & Electrical Engineering*, 66:98–113, 2018.

[90] Anwar Kalghoum and Leila Azouz Saidane. Fcr-ns: a novel caching and forwarding strategy for named data networking based on software defined networking. *Cluster Computing*, 22(3):981–994, 2019.

[91] Anselme Ndikumana, Nguyen H Tran, Tai Manh Ho, Dusit Niyato, Zhu Han, and Choong Seon Hong. Joint incentive mechanism for paid content caching and price based cache replacement policy in named data networking. *IEEE Access*, 6:33702–33717, 2018.

[92] Mikhail Badov, Anand Seetharam, Jim Kurose, Victor Firoiu, and Soumendra Nanda. Congestion-aware caching and search in information-centric networks. In *Proceedings of the 1st ACM Conference on Information-Centric Networking*, pages 37–46, 2014.

[93] Miki Yamamoto. A survey of caching networks in content oriented networks. *IEICE Transactions on Communications*, 99(5):961–973, 2016.

[94] Giuseppe Rossini. *Design Analysis of Forwarding Strategies for Host and Content Centric Networking*. PhD thesis, telecom-paristech, 2014.

[95] Spyridon Mastorakis, Alexander Afanasyev, and Lixia Zhang. On the evolution of ndnsim: An open-source simulator for ndn experimentation. *ACM SIGCOMM Computer Communication Review*, 47(3):19–33, 2017.

[96] http://www-sop.inria.fr/members/frederic.urbani/ns3dceccnx/getting-started.html, accessed on 12 March 2022.

[97] https://github.com/cn-uofbasel/ccn-lite, accessed on 12 March 2022.

[98] Ignacio Solis, Marc Mosko, Glenn Scott, and Alan Walendowski. Ccnx 1.0 tutorial: Theory and practice. In *Proceedings of the 2nd ACM Conference on Information-Centric Networking*, pages 1–2, 2015.

[99] Carlos MS Cabral, Christian Esteve Rothenberg, and Maurício Ferreira Magalhães. Mini-ccnx: Fast prototyping for named data networking. In *Proceedings of the 3rd ACM SIGCOMM workshop on Information-centric networking*, pages 33–34, 2013.

[100] Ilaria Cianci, L Alfredo Grieco, and Gennaro Boggia. Ccn-java opensource kit emulator for wireless ad hoc networks. In *Proceedings of the 7th International Conference on Future Internet Technologies*, pages 7–12, 2012.

[101] Raffaele Chiocchetti, Dario Rossi, and Giuseppe Rossini. ccnsim: An highly scalable ccn simulator. In *2013 IEEE International Conference on Communications (ICC)*, pages 2309–2314. IEEE, 2013.

[102] Ikram Ud Din, Suhaidi Hassan, and Adib Habbal. Socialccnsim: A simulator for caching strategies in information-centric networking. *Advanced Science Letters*, 21(11):3505–3509, 2015.

[103] Lorenzo Saino, Ioannis Psaras, and George Pavlou. Icarus: a caching simulator for information centric networking (icn). In *SimuTools*, volume 7, pages 66–75. ICST, 2014.

[104] Dario Rossi and Giuseppe Rossini. Caching performance of content centric networks under multi-path routing (and more). *Relatório técnico, Telecom ParisTech*, 2011:1–6, 2011.

[105] Mohammad Alkhazaleh, SA Aljunid, and Naseer Sabri. A comprehensive survey of information-centric network: Content caching strategies perspective. 2016:1–9, 2016.

[106] Anshuman Kalla and Sudhir Kumar Sharma. A constructive review of in-network caching: A core functionality of icn. In *2016 International Conference on Computing, Communication and Automation (ICCCA)*, pages 567–574. IEEE, 2016.

[107] Lada A Adamic and Bernardo A Huberman. Zipf's law and the internet. *Glottometrics*, 3(1):143–150, 2002.

[108] Abdelkader Tayeb Herouala, Benameur Ziani, Chaker Abdelaziz Kerrache, Abdou el Karim Tahari, Nasreddine Lagraa, Spyridon Mastorakis, et al. Cadaca: a new caching strategy in ndn using data categorization. *Multimedia Systems*, pages 1–16, 2022.

[109] Abdelkader Tayeb Herouala, Chaker Abdelaziz Kerrache, Benameur Ziani, Carlos T Calafate, Nasreddine Lagraa, and Abdou el Karim Tahari. Controlling the trade-off between resource efficiency and user satisfaction in ndns based on naïve bayes data classification and lagrange method. *Future Internet*, 14(2):48, 2022.

[110] Ferdinando Fioretto, Enrico Pontelli, and William Yeoh. Distributed constraint optimization problems and applications: A survey. *Journal of Artificial Intelligence Research*, 61:623–698, 2018.

[111] Peter F Brown, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–480, 1992.

[112] Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer, 2003.

[113] Kenneth Ward Church. Word2vec. *Natural Language Engineering*, 23(1):155–162, 2017.

[114] Marco Polignano, Pierpaolo Basile, Marco De Gemmis, Giovanni Semeraro, and Valerio Basile. Alberto: Italian bert language understanding model for nlp challenging tasks based on tweets. In *6th Italian Conference on Computational Linguistics, CLiC-it 2019*, volume 2481, pages 1–6. CEUR, 2019.

[115] Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. Text classification algorithms: A survey. *Information*, 10(4):150, 2019.

[116] Aydin Buluç, Jeremy T Fineman, Matteo Frigo, John R Gilbert, and Charles E Leiserson. Parallel sparse matrix-vector and matrix-transpose-vector multiplication using compressed sparse blocks. In *Proceedings of the twenty-first annual symposium on Parallelism in algorithms and architectures*, pages 233–244, 2009.

[117] FE Harrell. Regression modeling strategies springer. *N Y*, 2001.

[118] Larry M Manevitz and Malik Yousef. One-class svms for document classification. *Journal of machine Learning research*, 2(Dec):139–154, 2001.

[119] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[120] Ashadullah Shawon, Syed Tauhid Zuhori, Firoz Mahmud, and Md Jamil-Ur Rahman. Website classification using word based multiple n-gram models and random search oriented feature parameters. In *2018 21st International Conference of Computer and Information Technology (ICCIT)*, pages 1–6. IEEE, 2018.

[121] Ray R Larson. Introduction to information retrieval, 2010.

[122] Jin Huang and Charles X Ling. Using auc and accuracy in evaluating learning algorithms. *IEEE Transactions on knowledge and Data Engineering*, 17(3):299–310, 2005.

[123] Andrew McCallum, Kamal Nigam, et al. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Citeseer, 1998.

[124] Shuo Xu, Yan Li, and Zheng Wang. Bayesian multinomial naïve bayes classifier to text classification. In *Advanced multimedia and ubiquitous engineering*, pages 347–352. Springer, 2017.

[125] John V Baxley and John C Moorhouse. Lagrange multiplier problems in economics. *The American Mathematical Monthly*, 91(7):404–412, 1984.

[126] Cisco Global Cloud Index. Forecast and methodology, 2015-2020 white paper. *Retrieved 1st June*, page 15, 2016.

[127] Jiachen Yang, Shuai Xiao, Bin Jiang, Houbing Song, Suleman Khan, and Saif Ul Islam. Cache-enabled unmanned aerial vehicles for cooperative cognitive radio networks. *IEEE wireless communications*, 27(2):155–161, 2020.

[128] Mohammad Alhisnawi. Forwarding information base design techniques in content-centric networking: A survey. In *Next Generation of Internet of Things*, pages 157–174. Springer, 2021.

[129] Lixia Zhang, Deborah Estrin, Jeffrey Burke, Van Jacobson, James D Thornton, Diana K Smetters, Beichuan Zhang, Gene Tsudik, Dan Massey, Christos Papadopoulos, et al. Named data networking (ndn) project. *Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC*, 157:158, 2010.

[130] Farhan Ahmad, Chaker Abdelaziz Kerrache, Fatih Kurugollu, and Rasheed Hussain. Realization of blockchain in named data networking-based internet-of-vehicles. *IT Professional*, 21(4):41–47, 2019.

[131] Oussama Serhane, Khadidja Yahyaoui, Boubakr Nour, and Hassine Moungla. A survey of icn content naming and in-network caching in 5g and beyond networks. *IEEE Internet of Things Journal*, 2020.

[132] Cong Wang, Chen Chen, Qingqi Pei, Ning Lv, and Houbing Song. Popularity incentive caching for vehicular named data networking. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[133] Syed Hassan Ahmed, Safdar Hussain Bouk, Muhammad Azfar Yaqub, Dongkyun Kim, Houbing Song, and Jaime Lloret. Codie: Controlled data and interest evaluation in vehicular named data networks. *IEEE Transactions on Vehicular Technology*, 65(6):3954–3963, 2016.

[134] Oussama Serhane, Khadidja Yahyaoui, Boubakr Nour, and Hassine Moungla. Energy-aware cache placement scheme for iot-based icn networks. In *IEEE International Conference on Communications (ICC)*, 2021.

[135] Boubakr Nour, Kashif Sharif, Fan Li, Hassine Moungla, Ahmed E Kamal, and Hossam Afifi. Ncp: A near icn cache placement scheme for iot-based traffic class. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2018.

[136] Ying Xu, Qinghe Du, and Houbing Song. Security-enhanced wireless multicast via adaptive fountain codes over distributed caching network. In *2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pages 2089–2096. IEEE, 2018.

[137] Jiachen Yang, Chaofan Ma, Jiabao Man, Huifang Xu, Gan Zheng, and Houbing Song. Cache-enabled in cooperative cognitive radio networks for transmission performance. *Tsinghua Science and Technology*, 25(1):1–11, 2019.

[138] Divya Saxena, Vaskar Raychoudhury, Neeraj Suri, Christian Becker, and Jiannong Cao. Named data networking: a survey. *Computer Science Review*, 19:15–55, 2016.

[139] Noor Abani. *Caching Strategies for Private and Efficient Content Retrieval in Information-Centric Networks*. PhD thesis, UCLA, 2018.

[140] Junghwan Kim, Myeong-Cheol Ko, Jinsoo Kim, and Moon Sun Shin. Route prefix caching using bloom filters in named data networking. *Applied Sciences*, 10(7):2226, 2020.

[141] http://rdf. dmoz.org/rdf/content.rdf.u8.gz, accessed on 12 March 2017.

[142] Jurica Ševa, Markus Schatten, and Petra Grd. Open directory project based universal taxonomy for personalization of online (re) sources. *Expert Systems with Applications*, 42(17-18):6306–6314, 2015.

[143] Junji Takemasa, Yuki Koizumi, and Toru Hasegawa. Lightweight cache admission algorithm for fast ndn software routers. *Journal of Information Processing*, 27:125–134, 2019.

[144] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479, 1992.

[145] Ashraf M Kibriya, Eibe Frank, Bernhard Pfahringer, and Geoffrey Holmes. Multinomial naive bayes for text categorization revisited. In *Australasian Joint Conference on Artificial Intelligence*, pages 488–499. Springer, 2004.

[146] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(Feb):281–305, 2012.

[147] Ashadullah Shawon, Syed Tauhid Zuhori, Firoz Mahmud, Md Rahman, et al. Web links prediction and category-wise recommendation based on browser history. *arXiv preprint arXiv:1902.08496*, 2019.

[148] Spyridon Mastorakis, Alexander Afanasyev, and Lixia Zhang. On the evolution of ndnsim: An open-source simulator for ndn experimentation. *ACM SIGCOMM Computer Communication Review*, 47(3):19–33, 2017.

[149] Wei Koong Chai, Diliang He, Ioannis Psaras, and George Pavlou. Cache "less for more" in information-centric networks. In *International Conference on Research in Networking*, pages 27–40. Springer, 2012.

[150] Jing Ren, Wen Qi, Cedric Westphal, Jianping Wang, Kejie Lu, Shucheng Liu, and Sheng Wang. Magic: A distributed max-gain in-network caching strategy in information-centric networks. In *2014 IEEE conference on computer communications workshops (INFOCOM WKSHPS)*, pages 470–475. IEEE, 2014.

[151] Xiaoyan Hu, Jian Gong, Guang Cheng, and Chengyu Fan. Enhancing in-network caching by coupling cache placement, replacement and location. In *2015 IEEE International Conference on Communications (ICC)*, pages 5672–5678. IEEE, 2015.

[152] Aura Sukma Andini, Danang Triantoro Murdiansyah, and Kemas Muslim Lhaksmana. Topic classification of islamic question and answer using naïve bayes and tf-idf method. *Computer Engineering and Applications Journal*, 10(3):151–160, 2021.

[153] Pannapa Changpetch, Apasiri Pitpeng, Sasiprapa Hiriote, and Chumpol Yuangyai. Integrating data mining techniques for naïve bayes classification: Applications to medical datasets. *Computation*, 9(9):99, 2021.

[154] Huiliang Wang, Hongfa Wang, Zening Wu, and Yihong Zhou. Using multi-factor analysis to predict urban flood depth based on naive bayes. *Water*, 13(4), 2021.

[155] Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. Text classification algorithms: A survey. *Information*, 10(4), 2019.

[156] Ioannis Psaras, Wei Koong Chai, and George Pavlou. In-network cache management and resource allocation for information-centric networks. *IEEE Transactions on Parallel and Distributed Systems*, 25(11):2920–2931, 2013.

[157] Mingchuan Zhang, Ping Xie, Junlong Zhu, Qingtao Wu, Ruijuan Zheng, and Hongke Zhang. Ncpp-based caching and nur-based resource allocation for information-centric networking. *Journal of Ambient Intelligence and Humanized Computing*, 10(5):1739–1745, 2019.

[158] Muhammad Ali Naeem, Shahrudin Awang Nor, Suhaidi Hassan, and Byung-Seo Kim. Compound popular content caching strategy in named data networking. *Electronics*, 8(7):771, 2019.

[159] Xiong Kexin, Bowen Liang, Aakanksha Rai, and Alex Chan. Url classification with deep learning. 2021.

[160] Hamonangan Situmorang, Nana Rachmana Syambas, Tutun Juhana, and Ian Yosef Matheus Edward. A simulation of cache replacement strategy on named data network. In *2018 12th International Conference on Telecommunication Systems, Services, and Applications (TSSA)*, pages 1–4. IEEE, 2018.

[161] Wei Koong Chai, Diliang He, Ioannis Psaras, and George Pavlou. Cache "less for more" in information-centric networks (extended version). *Computer Communications*, 36(7):758–770, 2013.

[162] Jing Ren, Wen Qi, Cedric Westphal, Jianping Wang, Kejie Lu, Shucheng Liu, and Sheng Wang. Magic: A distributed max-gain in-network caching strategy in information-centric networks. In *2014 IEEE conference on computer communications workshops (INFOCOM WKSHPS)*, pages 470–475. IEEE, 2014.

[163] Xiaoyan Hu, Jian Gong, Guang Cheng, and Chengyu Fan. Enhancing in-network caching by coupling cache placement, replacement and location. In *2015 IEEE International Conference on Communications (ICC)*, pages 5672–5678. IEEE, 2015.

[164] Marica Amadeo, Giuseppe Ruggeri, Claudia Campolo, and Antonella Molinaro. Iot services allocation at the edge via named data networking: From optimal bounds to practical design. *IEEE Transactions on Network and Service Management*, 16(2):661–674, 2019.

[165] Mostafa Dehghan, Laurent Massoulie, Don Towsley, Daniel Sadoc Menasche, and Yong Chiang Tay. A utility optimization approach to network cache design. *IEEE/ACM Transactions on Networking*, 27(3):1013–1027, 2019.

[166] Razieh Nokhbeh Zaeem and K Suzanne Barber. A large publicly available corpus of website privacy policies based on dmoz. In *CODASPY*, pages 143–148, 2021.

[167] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[168] Nidhi Lal, Shishupal Kumar, Garima Kadian, and Vijay Kumar Chaurasiya. Caching methodologies in content centric networking (ccn): A survey. *Computer Science Review*, 31:39–50, 2019.

[169] Safdar Hussain Bouk, Syed Hassan Ahmed, Dongkyun Kim, and Houbing Song. Named-data-networking-based its for smart cities. *IEEE Communications Magazine*, 55(1):105–111, 2017.

[170] Syed Hassan Ahmed, Safdar Hussain Bouk, Dongkyun Kim, Danda B Rawat, and Houbing Song. Named data networking for software defined vehicular networks. *IEEE Communications Magazine*, 55(8):60–66, 2017.

[171] Maroua Meddeb, Amine Dhraief, Abdelfettah Belghith, Thierry Monteil, Khalil Drira, and Hassan Mathkour. Least fresh first cache replacement policy for ndn-based iot networks. *Pervasive and Mobile Computing*, 52:60–70, 2019.

[172] Zhiyi Zhang, Yingdi Yu, Haitao Zhang, Eric Newberry, Spyridon Mastorakis, Yanbiao Li, Alexander Afanasyev, and Lixia Zhang. An overview of security support in named data networking. *IEEE Communications Magazine*, 56(11):62–68, 2018.

[173] Christian Koch, Johannes Pfannmüller, Amr Rizk, David Hausheer, and Ralf Steinmetz. Category-aware hierarchical caching for video-on-demand content on youtube. In *Proceedings of the 9th ACM Multimedia Systems Conference*, pages 89–100, 2018.

[174] Nikos Fotiou, Pekka Nikander, Dirk Trossen, and George C Polyzos. Developing information networking further: From psirp to pursuit. In *International Conference on Broadband Communications, Networks and Systems*, pages 1–13. Springer, 2010.

[175] Xinggang Wang, Baoyuan Wang, Xiang Bai, Wenyu Liu, and Zhuowen Tu. Max-margin multiple-instance dictionary learning. In *International conference on machine learning*, pages 846–854. PMLR, 2013.